SIEMENS

SINUMERIK

SINUMERIK 828D
Turning and Milling

Commissioning Manual

Preface

Scope of delivery and requirements 1

Introduction and use of data classes 2

Operating software settings 3

Commissioning the PLC (Part 1) 4

Commissioning the drive 5

Parameterizing machine data 6

Setting cycle machine data 7

Commissioning the PLC (Part 2) 8

Tool management 9

Easy Archive 10

Lists A

Appendix B

Valid for:

CNC system software Version 4.3

11/2010
6FC5397-3DP40-0BA0

Legal information

Warning notice system

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 product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation for the specific task, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

Proper use of Siemens products

Note the following:

⚠WARNING
Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be adhered to. The information in the relevant documentation must be observed.

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
Industry Sector
Postfach 48 48
90026 NÜRNBERG
GERMANY

order number: 6FC5397-3DP40-0BA0
℗ 09/2010

Copyright © Siemens AG 2010.
Technical data subject to change

Preface

SINUMERIK documentation

The SINUMERIK documentation is organized in three parts:

- General documentation
- User documentation
- Manufacturer/service documentation

Information on the following topics is available under the link (http://www.siemens.com/motioncontrol/docu):

- Ordering documentation:

  Here you can find an up-to-date overview of publications.
- Downloading documentation:

  Links to more information for downloading files from Service & Support.
- Researching documentation online

  Information on DOConCD and direct access to the publications in DOConWEB.
- Customize documentation based on Siemens content using My Documentation Manager (MDM (http://www.siemens.com/mdm)).

  My Documentation Manager provides you with a range of features for generating your own machine documentation.
- Training and FAQs

  Information on our range of training courses and FAQs (frequently asked questions) is available via the page navigation.

Target group

This documentation is intended for commissioning personnel.

The plant or system is readily assembled and wired. For the following steps, e.g. configuring the individual components, the Commissioning Manual contains all necessary information or at least references.

Benefits

The intended target group can use the Commissioning Manual to test and commission the system or the plant correctly and safely.

Utilization phase: Setup and commissioning phase

Standard version

This documentation only describes the functionality of the standard version. 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. This does not, however, represent an obligation to supply such functions with a new control or when 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.

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 2176
E-mail (mailto:docu.motioncontrol@siemens.com)

SINUMERIK Internet address (http://www.siemens.com/sinumerik)

Service & Support

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

	Europe / Africa
Phone	+49 911 895 7222
Fax	+49 911 895 7223
	For connections to German landline numbers dialed from outside Germany, the regional telephone rates and conditions apply.
Internet (http://www.siemens.com/automation/support-request)	

	Americas
Phone	+1 423 262 2522
Fax	+1 423 262 2200
E-mail (mailto:techsupport.sea@siemens.com)	

	Asia / Pacific
Phone	+86 1064 757575
Fax	+86 1064 747474
E-mail (mailto:techsupport.asia@siemens.com)	

Note

Country-specific telephone numbers for technical support are provided on the Internet (http://www.siemens.com/automation/service&support).

EC Declaration of Conformity

The EC Declaration of Conformity for the EMC Directive can be found on the Internet (http://support.automation.siemens.com) under the Product Order No. 15257461 or at a local office of the I DT MC Division of Siemens AG.

CompactFlash cards for users:

- The SINUMERIK CNC supports the file systems FAT16 and FAT32 for CompactFlash cards. You may need to format the memory card if you want to use a memory card from another device or if you want to ensure the compatibility of the memory card with the SINUMERIK. However, formatting the memory card will permanently delete all data on it.
- Do not remove the memory card while it is being accessed. This can lead to damage of the memory card and the SINUMERIK as well as the data on the memory card.
- If you cannot use a memory card with the SINUMERIK, it is probably because the memory card is not formatted for the control system (e.g. Ext3 Linux file system), the memory card file system is faulty or it is the wrong type of memory card.
- Insert the memory card carefully with the correct orientation into the memory card slot (take note of arrows, etc.). This way you avoid mechanical damage to the memory card or the device.
- Only use memory cards that have been approved by Siemens for use with SINUMERIK. Even though the SINUMERIK keeps to the general industry standards for memory cards, it is possible that memory cards from some manufacturers will not function perfectly in this device or are not completely compatible with it (you can obtain information on compatibility from the memory card manufacturer or supplier).
- For SINUMERIK, only the memory card with order number 6FC5313-5AG00-0AA1 is permitted as system CompactFlash Card.

Table of contents

Preface 3

1 Scope of delivery and requirements 13

1.1 System overview 13

1.2 Toolbox CD and other available tools 15

1.3 This is the general sequence for commissioning 16

1.4 Starting up the control 17

1.5 Communication with the control 20
1.5.1 How to communicate with the control using the Programming Tool 20
1.5.2 Example: How to communicate with the control using the NCU Connection Wizard 24
1.5.3 How to communicate with the control using the RCS Commander 26
1.5.4 Communicating with the control via X130 29

2 Introduction and use of data classes 31

2.1 Data classes in the NCK 32

2.2 Data classes in the PLC 35

2.3 Data classes in the operating software 36

3 Operating software settings 39

3.1 Access levels 39

3.2 How to set and change the password 41

3.3 How to set the date and time 42

3.4 Setting the operating software languages 43
3.4.1 Available system languages 43
3.4.2 This is how you install additional system languages 44
3.4.3 Entering Asian characters using the Input Method Editor 47
3.4.4 This is how you enter the Chinese character 49
3.4.5 This is how you enter Korean characters 50

3.5 Checking and entering licenses 51
3.5.1 How to enter a license key 51
3.5.2 This is how you determine missing licenses/options 53

3.6 Configuring user alarms 54
3.6.1 Structure of user PLC alarms 54
3.6.2 How to create user PLC alarms 56
3.6.3 Configuring the alarm log 57
3.6.4 How to configure the log 58

3.7 Creating OEM-specific online help 61
3.7.1 Structure and syntax of the configuration file 61
3.7.2 Structure and syntax of the help book 63
3.7.3 Description of the syntax for the online help 65
3.7.4 Example: This is how you generate an OEM-specific online help book 68
3.7.5 Example: How to create an online help for user PLC alarms 71

3.7.6 Example: This is how you generate an online help for NC/PLC variables ... 74
3.7.7 Example: This is how you generate a programming online help ... 76

4 Commissioning the PLC (Part 1) ... 79

4.1 Activating I/O modules ... 80

4.2 Addressing the I/O modules ... 82

5 Commissioning the drive ... 83

5.1 Configuring the drive ... 83
5.1.1 Example: Configuration with SINAMICS S120 Combi ... 83
5.1.2 Example: This is how you configure a spindle with SMC encoder ... 88
5.1.3 Example: This is how you configure an axis with SMI encoder ... 94
5.1.4 Example: Configuration with SINAMICS S120 Booksize ... 99
5.1.5 Example: How to configure the drive ... 101
5.1.6 Example: How to configure the infeed ... 108
5.1.7 Example: How to configure the external encoder ... 111
5.1.8 Example: How to assign the axes ... 115
5.1.9 Example: Setting machine data for an axis/spindle ... 120
5.1.10 Parameters for the axis / spindle test run ... 122

5.2 Topology rules for DRIVE-CLiQ ... 124
5.2.1 Topology rules for S120 Combi ... 124
5.2.2 Topology rules for S120 Booksize ... 126

5.3 Terminal assignments ... 130
5.3.1 Terminal assignment on X122 ... 130
5.3.2 Terminal assignment on X132 ... 131
5.3.3 Terminal assignment at X242 and X252 ... 132
5.3.4 Terminal assignment on X122 for a Numeric Control Extension ... 133
5.3.5 Example: Circuitry for a CU with line contactor ... 134
5.3.6 Connecting the probes ... 136

6 Parameterizing machine data ... 139

6.1 Classification of machine data ... 139

6.2 Processing part programs from external CNC systems ... 142

6.3 Machining free form surfaces using Advanced Surface ... 143

7 Setting cycle machine data ... 147

7.1 Settings for activating cycles ... 147
7.1.1 Settings to the coordinate system ... 148
7.1.2 How to adapt the manufacturer cycles ... 151
7.1.3 Standard cycle PROG_EVENT.SPF ... 152
7.1.4 Setting the simulation and simultaneous recording (option) ... 153
7.1.5 Simulation for machines with 5 axes ... 155

7.2 Drilling ... 158
7.2.1 Technology cycles for drilling ... 158
7.2.2 ShopTurn: Drilling centered ... 160

7.3 Milling ... 161
7.3.1 Technology cycles for milling ... 161
7.3.2 Cylinder surface transformation (TRACYL) ... 162
7.3.3 Example: Axis configuration for milling machines ... 163
7.3.4 ShopMill: Setting up cycles for milling ... 166

7.4 Turning 169
7.4.1 Technology cycles for turning 169
7.4.2 Example: Residual material machining 172
7.4.3 Example: Axis configuration for lathes 173
7.4.4 Cylinder surface transformation (TRACYL) 175
7.4.5 End face machining (TRANSMIT) 178
7.4.6 Inclined axis (TRAANG) 181
7.4.7 ShopTurn: Setting up cycles for turning 184
7.4.8 ShopTurn: Counterspindle 191
7.4.9 ShopTurn: Cylinder surface transformation (TRACYL) 195
7.4.10 ShopTurn: End face machining (TRANSMIT) 196
7.4.11 ShopTurn: Inclined axis (TRAANG) 197

7.5 Swivel 198
7.5.1 Technology cycles for swiveling 198
7.5.2 Setting the workpiece, tool and rotary table reference 201
7.5.3 ShopMill: Swivel plane and swivel tool 203
7.5.4 CYCLE800 checklist for the identification of the machine kinematics 205
7.5.5 Commissioning of the kinematic chain (swivel data record) 206
7.5.6 Example of the commissioning of swivel head 1 211
7.5.7 Example of the commissioning of swivel head 2 213
7.5.8 Example of the commissioning of a cardanic table 215
7.5.9 Example of the commissioning of a swivel head / rotary table 217
7.5.10 Example of the commissioning of a swivel table 219
7.5.11 Manufacturer cycle CUST_800.SPF 222

7.6 High Speed Settings (Advanced Surface) 228
7.6.1 Configuring the High Speed Settings function (CYCLE832) 228
7.6.2 How to adapt the High Speed Settings function (CYCLE832) 230

7.7 Measuring cycles and measurement functions 231
7.7.1 General settings for measuring 231
7.7.2 Manufacturer cycle CUST_MEACYC.SPF 233
7.7.3 Measuring in the JOG mode 234
7.7.4 JOG: Measure workpiece during milling 236
7.7.5 JOG: Measure tool during milling 239
7.7.6 JOG: Measure tool during turning 242
7.7.7 Measuring in the AUTOMATIC mode 243
7.7.8 AUTO: General settings for the workpiece measurement 245
7.7.9 AUTO: Measure workpiece during milling 247
7.7.10 AUTO: Measure workpiece during turning 249
7.7.11 AUTO: Measure tool during milling 250
7.7.12 AUTO: Measure tool during turning (CYCLE982) 258

8 Commissioning the PLC (Part 2) 261

8.1 Service Planner 261
8.1.1 Interfaces in the PLC user program 263
8.1.2 Interfaces to the operating software 268
8.1.3 This is how you import and export maintenance tasks 270
8.1.4 Acknowledging maintenance tasks 273

8.2 Easy Extend 275
8.2.1 Function of Easy Extend 275
8.2.2 Configuration in the PLC user program 276
8.2.3 Option bits for machine manufacturers and dealers 277
8.2.4 Display on the user interface 280
8.2.5 Generating language-dependent text 280

8.2.6 Examples ... 282
8.2.6.1 Example with control elements ... 282
8.2.6.2 Example with parameters to support the commissioning ... 283
8.2.6.3 User example for a power unit ... 285
8.2.7 Description of the script language ... 287
8.2.7.1 Special characters and operators ... 288
8.2.7.2 Structure of the XML script ... 289
8.2.7.3 CONTROL_RESET ... 290
8.2.7.4 DATA ... 291
8.2.7.5 DATA_ACCESS ... 291
8.2.7.6 DATA_LIST ... 292
8.2.7.7 DRIVE_VERSION ... 293
8.2.7.8 FILE ... 293
8.2.7.9 FUNCTION ... 295
8.2.7.10 FUNCTION_BODY ... 295
8.2.7.11 INCLUDE ... 297
8.2.7.12 LET ... 297
8.2.7.13 MSGBOX ... 299
8.2.7.14 OP ... 300
8.2.7.15 OPTION_MD ... 301
8.2.7.16 PASSWORD ... 302
8.2.7.17 PLC_INTERFACE ... 302
8.2.7.18 POWER_OFF ... 302
8.2.7.19 PRINT ... 303
8.2.7.20 WAITING ... 304
8.2.7.21 ?up ... 304
8.2.7.22 XML identifiers for the dialog ... 305
8.2.7.23 BOX ... 307
8.2.7.24 CONTROL ... 307
8.2.7.25 IMG ... 309
8.2.7.26 PROPERTY ... 310
8.2.7.27 REQUEST ... 311
8.2.7.28 SOFTKEY_OK, SOFTKEY_CANCEL ... 311
8.2.7.29 TEXT ... 312
8.2.7.30 UPDATE_CONTROLS ... 312
8.2.7.31 Addressing the parameters ... 313
8.2.7.32 Addressing the drive objects ... 315
8.2.7.33 XML identifiers for statements ... 317
8.2.8 String functions ... 319
8.2.8.1 string.cmp ... 319
8.2.8.2 string.icmp ... 320
8.2.8.3 string.left ... 321
8.2.8.4 string.right ... 322
8.2.8.5 string.middle ... 322
8.2.8.6 string.length ... 323
8.2.8.7 string.replace ... 324
8.2.8.8 string.remove ... 324
8.2.8.9 string.delete ... 325
8.2.8.10 string.insert ... 326
8.2.8.11 string.find ... 326
8.2.8.12 string.reversefind ... 327
8.2.8.13 string.trimleft ... 328
8.2.8.14 string.trimright ... 328
8.2.9 Trigonometric functions ... 329

9 Tool management 331

9.1 Fundamentals 331
9.1.1 Structure of the tool management 332
9.1.2 Components of the tool management 333
9.1.3 Loading and unloading tools manually 336

9.2 PLC - NCK user interface 337
9.2.1 Relocating, unloading, loading tool, positioning magazine 338
9.2.2 Tool change 344
9.2.3 Transfer-step and acknowledgment-step tables 350

9.3 Machine data for the tool management 353

9.4 PLC Program Blocks 360
9.4.1 Acknowledgment process 360
9.4.2 Types of acknowledgment 361
9.4.3 Acknowledgment states 362
9.4.4 Configuring step tables 367
9.4.5 Configuring acknowledgment steps 370
9.4.6 Adjust the PLC user program 371
9.4.7 Information on magazine location 372
9.4.8 PI service: TMMVTL 375

9.5 Example: Loading/unloading 376

9.6 Example: Change manual tools 378

9.7 Application example for turning machine 382
9.7.1 Example: Turning machine with revolver magazine (MAGKONF_MPF) 382
9.7.2 Example: Acknowledgment steps (turning machine) 387
9.7.3 Example: Tool change cycle for turning machine 388
9.7.4 Example: Turning machine with counterspindle 390
9.7.5 Example: Test for empty buffer 390
9.7.6 Example: Transporting a tool from a buffer into the magazine 391
9.7.7 Example: Repeat "Prepare tool change" order 392

9.8 Application example for milling machine 393
9.8.1 Example: Milling machine with chain magazine and dual gripper (MAGKONF_EXAMPLE_MPF) 393
9.8.2 Flow chart: Tool change 399
9.8.3 Example: Acknowledgment steps (milling machine) 408
9.8.4 Example: Tool change cycle for milling machine 410

10 Easy Archive 411

10.1 Series start-up and archiving 412

10.2 How to create and read in a series start-up archive 414

10.3 This is how you back up just the machine data that have changed 416

10.4 Example: Data archiving "Easy Archive" (use case) 417

10.5 Parameterizing the serial interface 419

A Lists 421
A.1 Table: Language code in file names 421
A.2 List of the alarm number ranges 422
A.3 List of the color codes 423
A.4 List of abbreviations 424
B Appendix 427
B.1 Definitions for license management 427
B.2 Documentation overview 429
Glossary 431
Index 437

Scope of delivery and requirements

1

1.1 System overview

System design

The following configuration shows a typical example with SINAMICS S120 booksize:

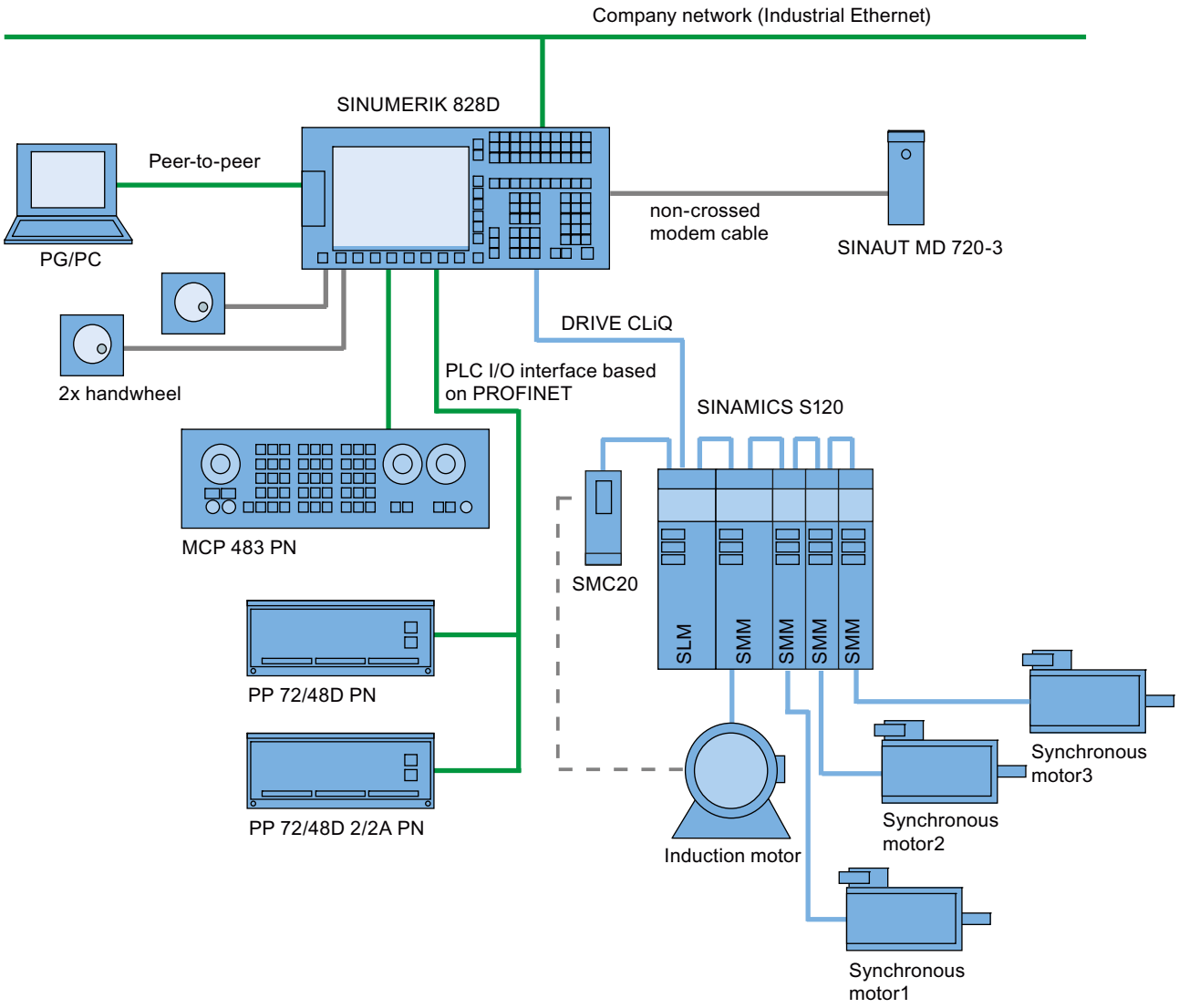


Figure 1-1 Configuration example 1: Basic configuration with 4 axes

The following configuration shows the maximum expansion stage with SINAMICS S120 Combi:

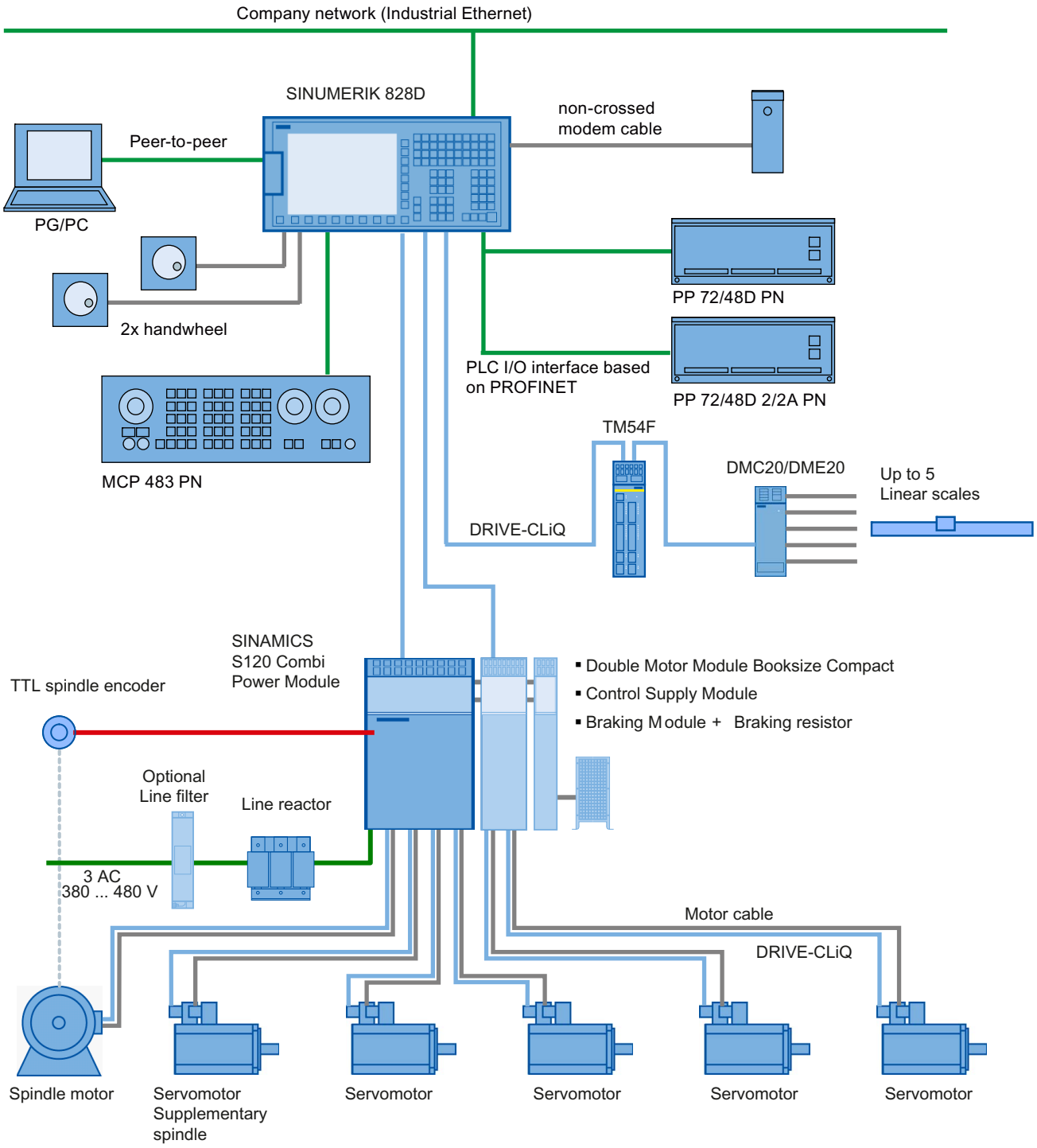


Figure 1-2 Configuration example 2: Maximum expansion stage with 6 axes and with Safety Integrated

1.2 Toolbox CD and other available tools

Toolbox CD

The Toolbox CD for SINUMERIK 828D has the following content:

- PLC Programming Tool for Integrated PLC
- Commissioning software for SINAMICS S120
- PLC Library (example)

PLC Programming Tool for Integrated PLC

The following tool is available for programming the PLC: PLC Programming Tool for Integrated PLC. For the rest of this manual, this will be referred to using the abbreviation "Programming Tool".

Commissioning software for SINAMICS S120

Until the SINAMICS S120 commissioning functionality is completely available via the user interface, drive configuration and optimization is performed using the commissioning software for SINAMICS S120. The PC is connected using the Ethernet interface on the front of the SINUMERIK 828D.

Note

Ordering data

You can find the ordering data for the following tools in Catalog NC 61.

RCS Commander

The RCS Commander (Remote Control System) is a tool the commissioner can use to exchange files between the PC and the control very easily, using drag and drop.

For data transmission, the PC is connected directly to the Ethernet interface on the front of the control. For a point-to-point connection, time-consuming parameterization of the Ethernet interface is not necessary. All settings are made automatically by the RCS Commander. The RCS Commander can also access several NCUs sequentially via a company network.

STARTER drive/commissioning software

Drive commissioning for the SINUMERIK 828D can be performed using the STARTER drive commissioning software. Simple commissioning procedures which are usually performed by field service staff (such as activating direct measuring systems) can be executed directly via the SINUMERIK 828D user interface. Advanced commissioning procedures which are usually performed when the machine is being manufactured (such as drive optimization) can be executed offline via the commissioning software used for SINAMICS S120.

1.3 This is the general sequence for commissioning

Requirements

The mechanical and electrical installation of the system must be completed.

- Check the system visually for:
  - Correct mechanical installation with secure electrical connections
  - Connection of the power supply
  - Connection of shielding and grounding
- Switching on the control and startup in "Normal startup":

  The control has finished booting when the main screen of the operating software is displayed.

Sequence overview

Commissioning of the SINUMERIK 828D is carried out in the following steps:

1. Install the software from Toolbox CD onto PG/PC

   See chapter "Scope of delivery and requirements"
2. Create communication connection with the control

   See chapter "Communication with the control"
3. Addressing the I/O

   See chapter "Addressing the I/O modules"
4. Setting the operating software

   See Chapter "Settings of the operating software"
5. PLC functions

   See Function Manual Basic Functions (P4)
6. Commission drive and connect probes

   See chapter "Configuring drive"
7. Setting NCK machine data

   See chapter "Setting NCK machine data"
8. Configuring cycles

   See chapter "Configuring cycles"
9. Define maintenance tasks and maintenance intervals

   See chapter "Service Planner"
10. Extend machine with additional devices

    See chapter "Easy Extend"
11. Tool management

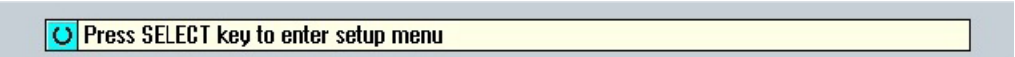
    See chapter "Tool management"

1.4 Starting up the control

Control startup

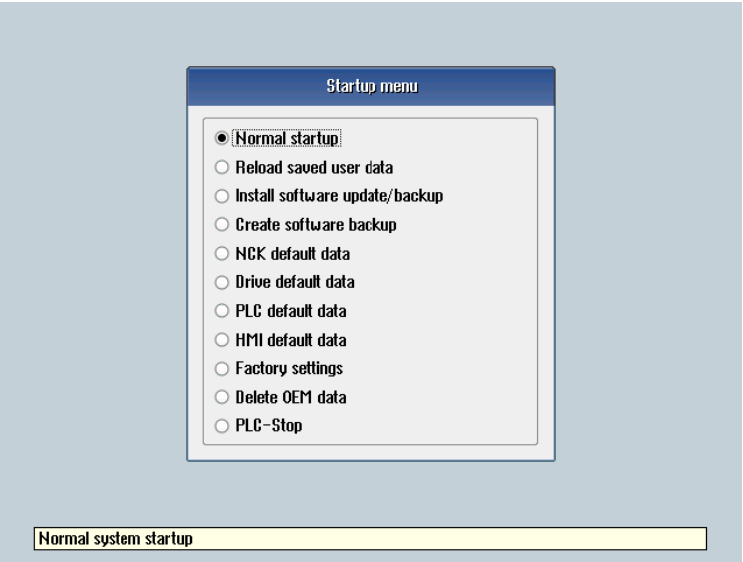
Procedure:

1. Switch the control on. The following display then appears during startup:



2. Press the <SELECT> key within three seconds.
3. Then press the following keys in succession:

   Menu reset key, **HSK2** (horizontal SK2), **VSK2** (vertical SK2)
4. The "Setup menu" is displayed, "Normal startup" is the default setting.



Operating modes for startup

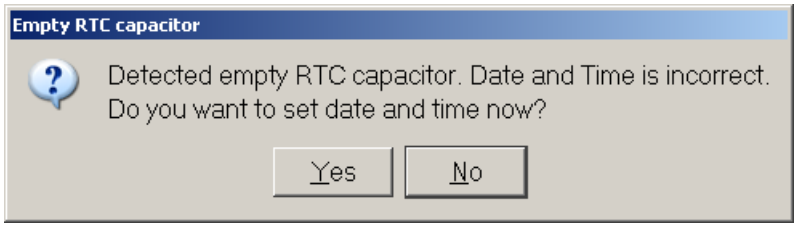
Selection	Function
Normal startup	The system carries out a normal startup.
Reload saved user data	The system loads the stored user data ("Save data" softkey) from the system CompactFlash card.
Install software update/backup	An update is installed on the system CompactFlash card from the user CompactFlash card or USB FlashDrive.
Create software backup	A backup of the system CompactFlash card is saved to the user CompactFlash card or USB FlashDrive.
NCK default data	The system loads the Siemens NCK data default settings and deletes the retentive data on the PLC.
Drive default data	The SINAMICS user data is deleted.
PLC default data	PLC general reset and load default NOP PLC program.

Selection	Function
HMI default data	User data in the directory /user is deleted.
Factory settings	Choice between two cases: **No** [case 1]**/ Yes** [case 2] • **Case 1:** The SINAMICS user data is deleted. Siemens standard NCK data is loaded. PLC general reset and load default NOP PLC program. User data in the directory /user is saved. • **Case 2:** As case 1 and additionally: Deletion of the data in the /oem and /addon directories.
Delete OEM data	All the data under /oem and /addon is deleted: OEM archives; OEM alarm texts; Easy Screen application.
PLC stop	PLC is stopped.

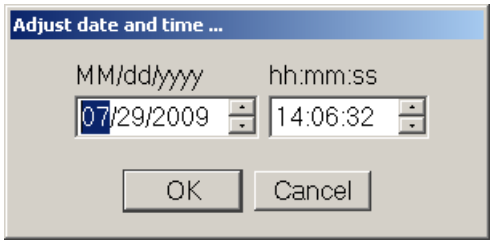
NOTICE
Replacement of the system CompactFlash card between different PPUs Because of the system-related dependency between the CompactFlash card and SRAM for the data storage in the SINUMERIK 828D, the system CompactFlash card should be considered as a permanently installed EEPROM and should not be replaced! If this has to be performed for imperative reasons, the replacement of the system CompactFlash card is detected during startup because of the stored serial number. The reaction of the control is the loading of saved during startup (backup was performed previously with "Save data" softkey). If no stored data is found, a startup is performed automatically with the "NCK default data".

Empty RTC capacitor

If the RTC capacitor is discharged, the following message is issued during startup:



You can then reset the date and time:

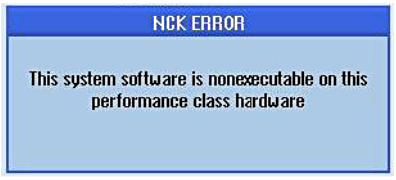


The capacitor is then charged again when the control is switched on during startup.

Checking the hardware

For the PPU 24x.2 hardware a distinction is made between the turning version BASIC T and the milling version BASIC M :

This distinction is made when booting using the MLFB found in the hardware information block, which is then compared with the existing NCK scaling. The following message is output if software that is not suitable for this hardware is found:



1.5 Communication with the control

Creating the connection

An Ethernet cable is needed to connect the control and PG/PC. The following Ethernet interfaces are available on the control:

- **Connection via X127** (behind the flap on the front):

  Cable type: Crossed Ethernet cable

  At interface X127, the control is preset as a DHCP server, delivering the IP address192.168.215.1 for a direct connection (peer-to-peer connection).

- **Connection via X130** (at the back):

  Cable type: Uncrossed Ethernet cable

  The interface X130 is the connection to the company network. The IP address that the PG/PC receives here as a DHCP client is determined by the DHCP server from the company network or fixed IP address is entered manually.

1.5.1 How to communicate with the control using the Programming Tool

Setting up the communications interface in the Programming Tool

Proceed as follows to set up the network connection in the Programming Tool:



1. Start the Programming Tool.
2. In the navigation bar, click the "Communication" icon or select "View" → "Communication" from the menu.
3. In the left column, under "Communications parameters" enter 192.168.215.1as the IP address for X127.
4. Double click on the icon "TCP/IP" at the top right.

5. In the dialog "PG/PC interface" select the TCP/IP protocol of the PG/PC. Normally this is the network card of the PC.

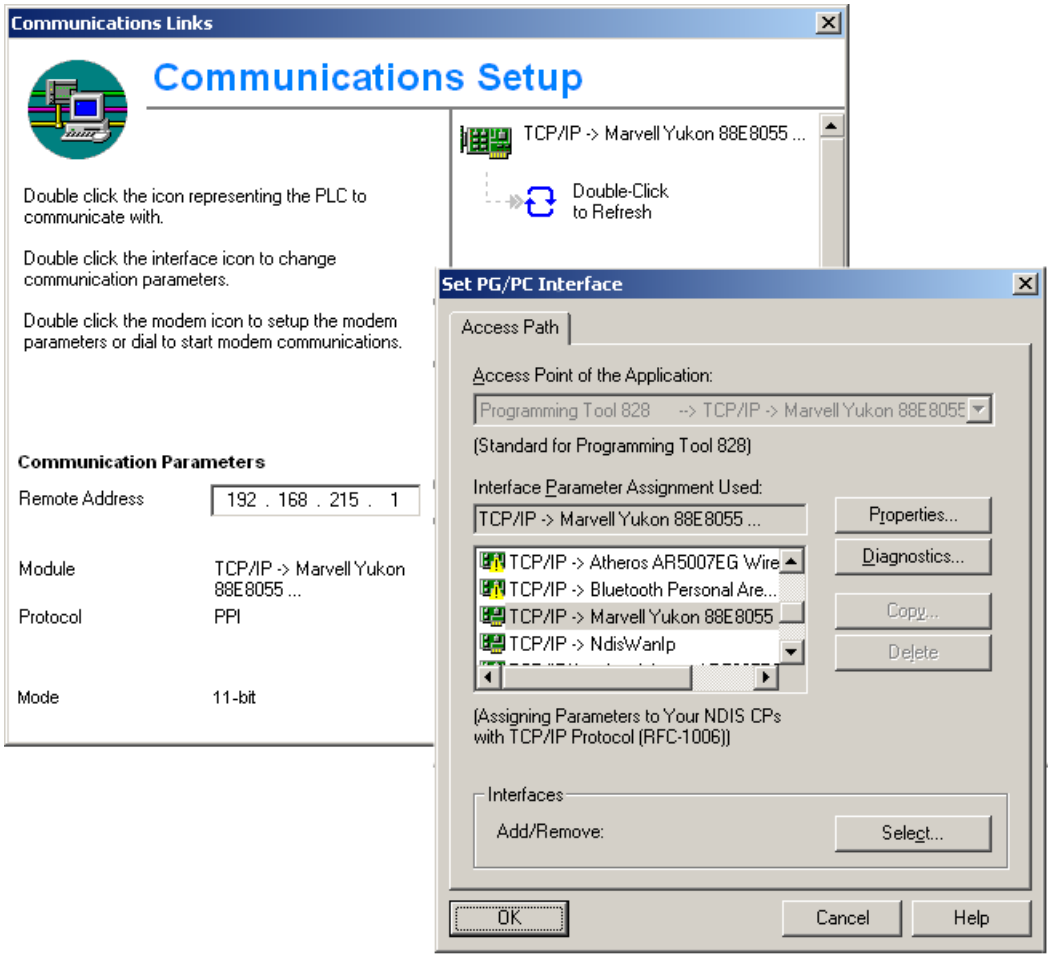


Figure 1-3 TCP/IP communications settings

6. Confirm with "OK".

7. Connect by double clicking on the icon "Double click to update". If the connection is made successfully, the icon will be displayed with a green border:

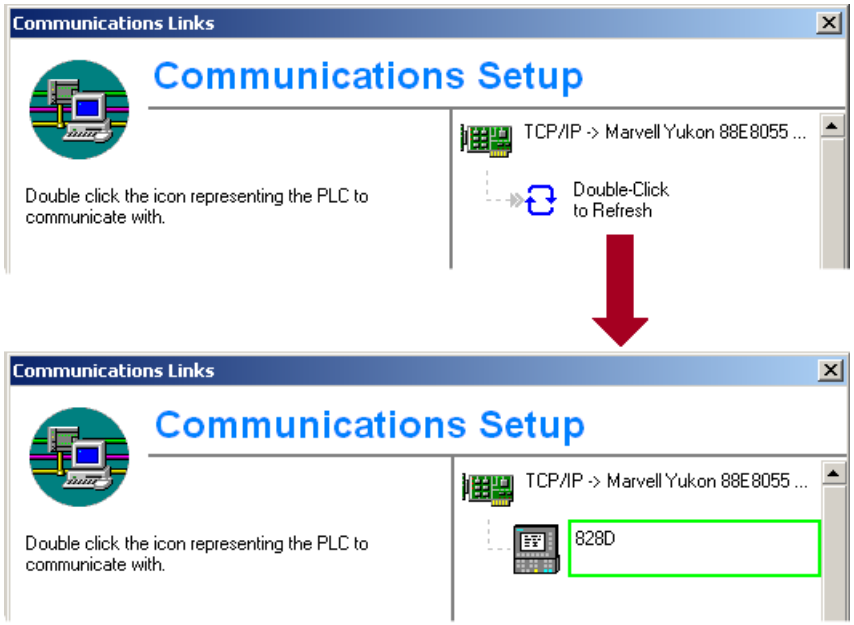


Figure 1-4 Online connection

8. If the connection is unsuccessful, the following setting may have to be deactivated:

   Select "Control Panel" → "Network Connections"→ "Local Area Connection" "Properties"→ "Advanced"→ "Windows Firewall" → "Settings"→ "Advanced": Deactivate the option "Local Area Connection".

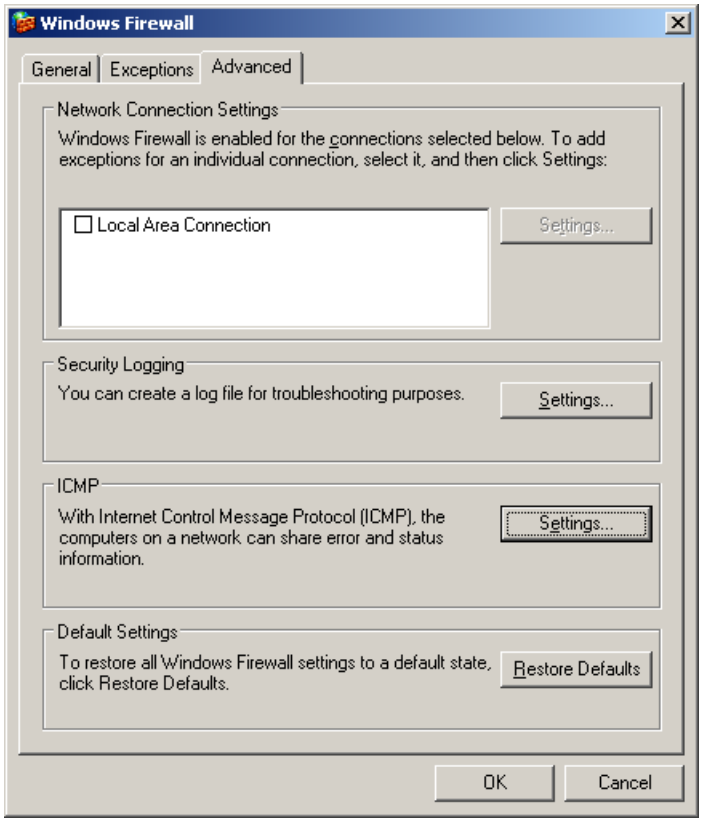


Figure 1-5 Deactivate option

Confirm with "OK" and repeat Step 7.

1.5.2 Example: How to communicate with the control using the NCU Connection Wizard

Requirements

The commissioning software for SINAMICS S120 is installed on the PG/PC. The "NCU Connection Wizard" is part of this software.

The connection to the control has already been set up via the Programming Tool.

Create connection to control

Procedure for the PG/PC:



1. Start the "NCU Connection Wizard" via this link or via the Start menu.
2. In the "Select Control Model" dialog, select "840D solution line" for the NCU type connection to the SINUMERIK 828D.

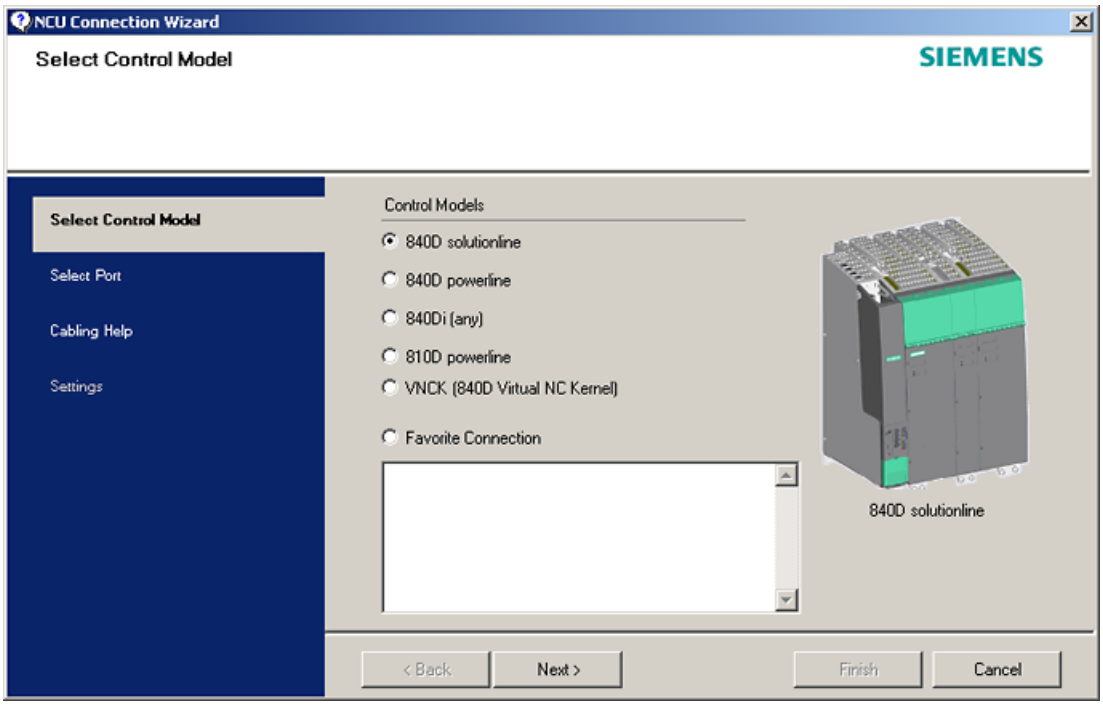


Figure 1-6 Select the NCU type

3. In the "Select Port" dialog, select the connection to the control that you have connected via Ethernet.

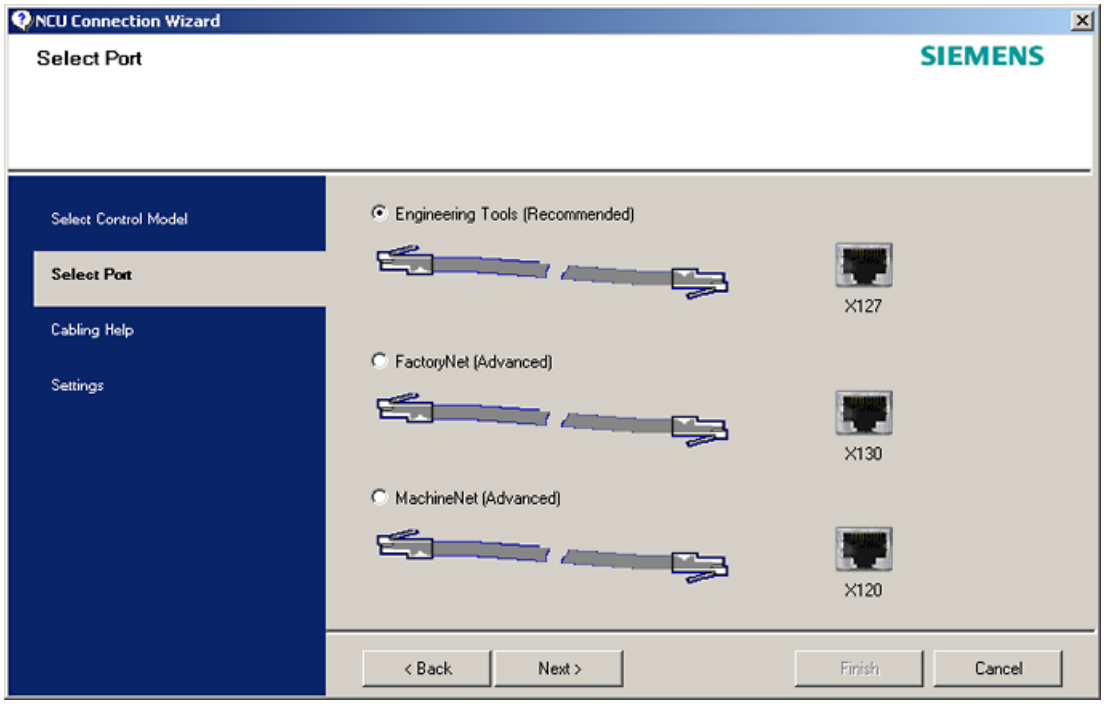


Figure 1-7 Select connection

4. Confirm the cable connection for both devices in the"Cabling Help" dialog.

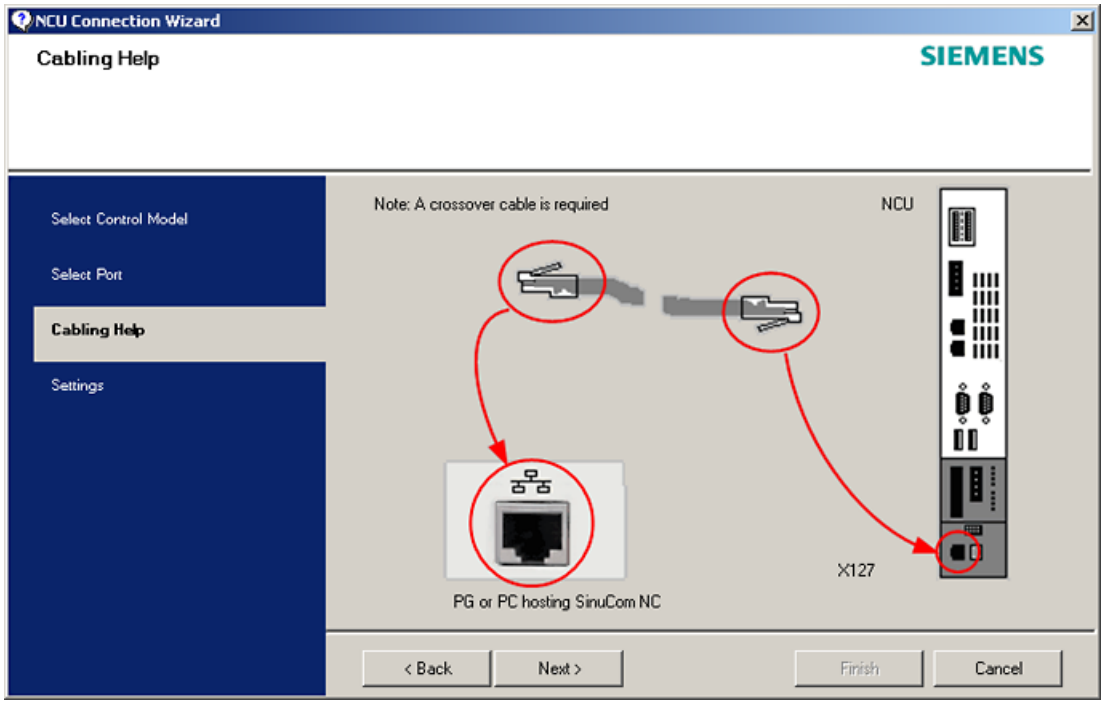


Figure 1-8 Cabling

5. Check the IP address and enter the name for these settings in the "Settings" dialog.

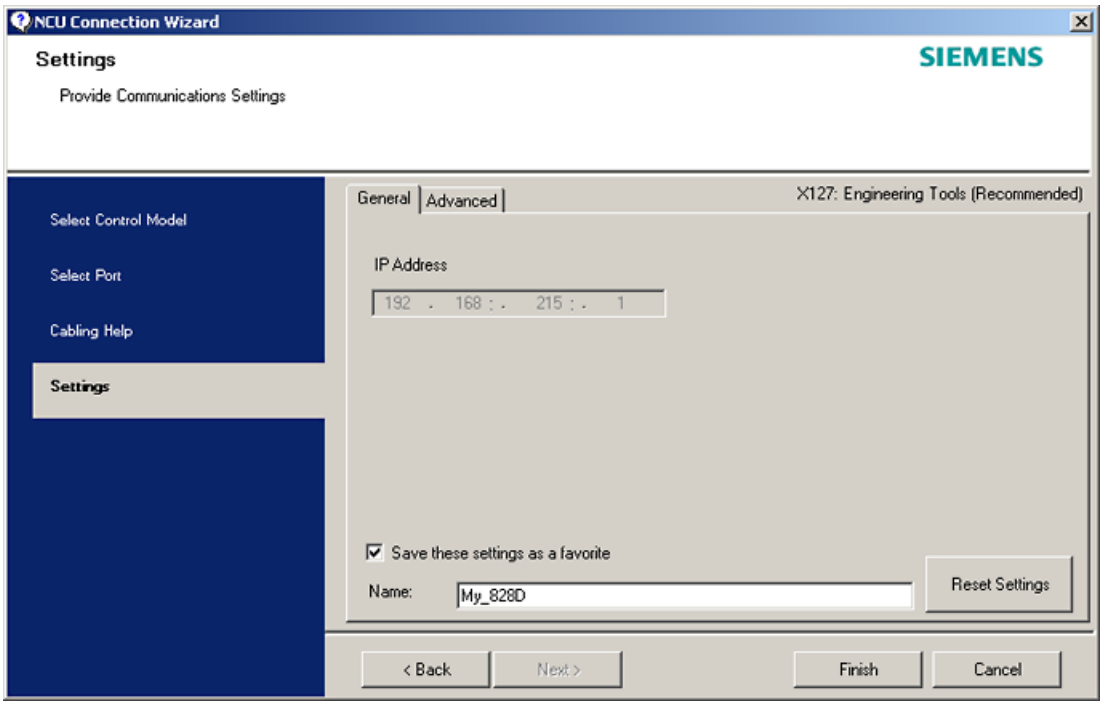


Figure 1-9 Network settings

1.5.3 How to communicate with the control using the RCS Commander

Connection options

The following options are available for the "RCS Commander" to create a connection with the control:

- Direct connection (peer-to-peer)
- Network connection

The current status of the connection is shown at the bottom in the RCS Commander status bar.

Meaning of the buttons:

 Connect

 Disconnect

 Remote control

NOTICE
Generally only **one connection** is permitted, i.e. several simultaneous connections to different controls are not supported: So data exchange between two NCUs using "RCS Commander" is not possible.

Direct connection

To create a direct connection:

1. The login data is entered in the dialog "Settings" → "Connection" → "Direct connection":

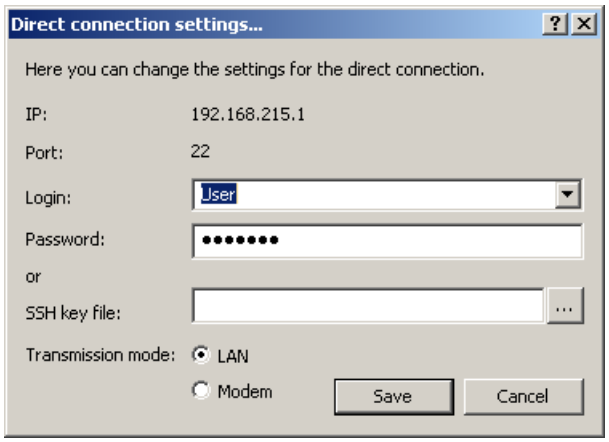


Figure 1-10 Dialog: Login data for direct connection

2. In the menu, select "Connection" → "Connect" → "Direct connection" or click the "Connect" button.

   The following dialog box is displayed:

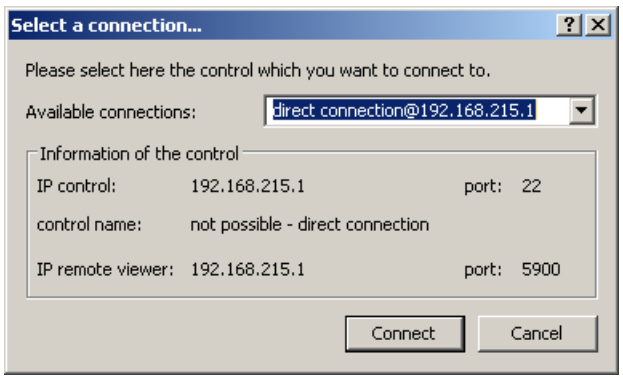


Figure 1-11 Dialog: Direct connection

3. The last selected direct connection is highlighted. Using the "Connect" button, a connection to the IP address196.168.215.1 is created.

   This dialog does not appear when the direct connection is selected using the menu.

Network connection

To create a network connection:

1. In the menu, select "Settings" → "Connection" → "Direct connection" or click the "Connect" button.

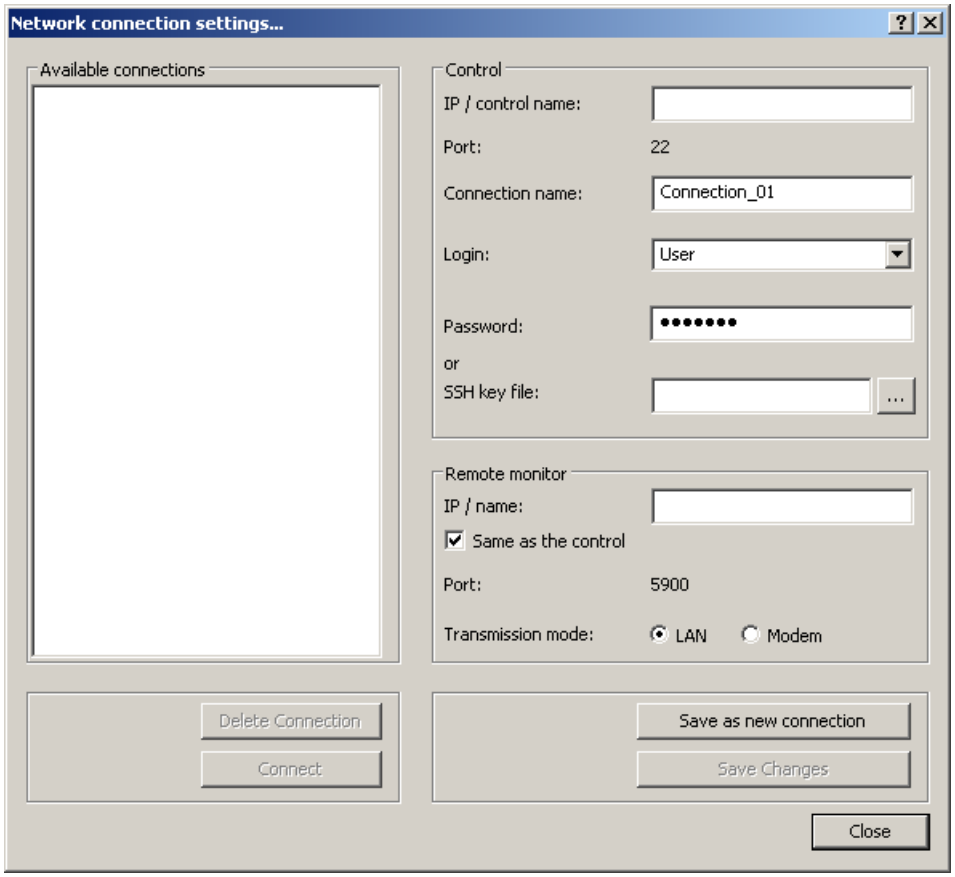


Figure 1-12 Dialog: Network connection

2. In the menu, select "Connection" → "Connect" → "Network connection" or select – if available – one of the previously selected connections.
3. Connection is made to the parameterized control.

Note

SSh key file

As an alternative to entering a password, the user may also use an SSh key for authentication. Please refer to the Online Help for more information on this topic.

1.5.4 Communicating with the control via X130

Connection to the company network

The NCU is connected to the company network via the Ethernet interface X130. The company network is used, for example, to access the network drives.

In the "Diagnostics" operating area select the "Bus TCP/IP " → "TCP/IP Diagnostics" → "Details" softkey with the menu forward key in order to set the parameters for the communication via X130.

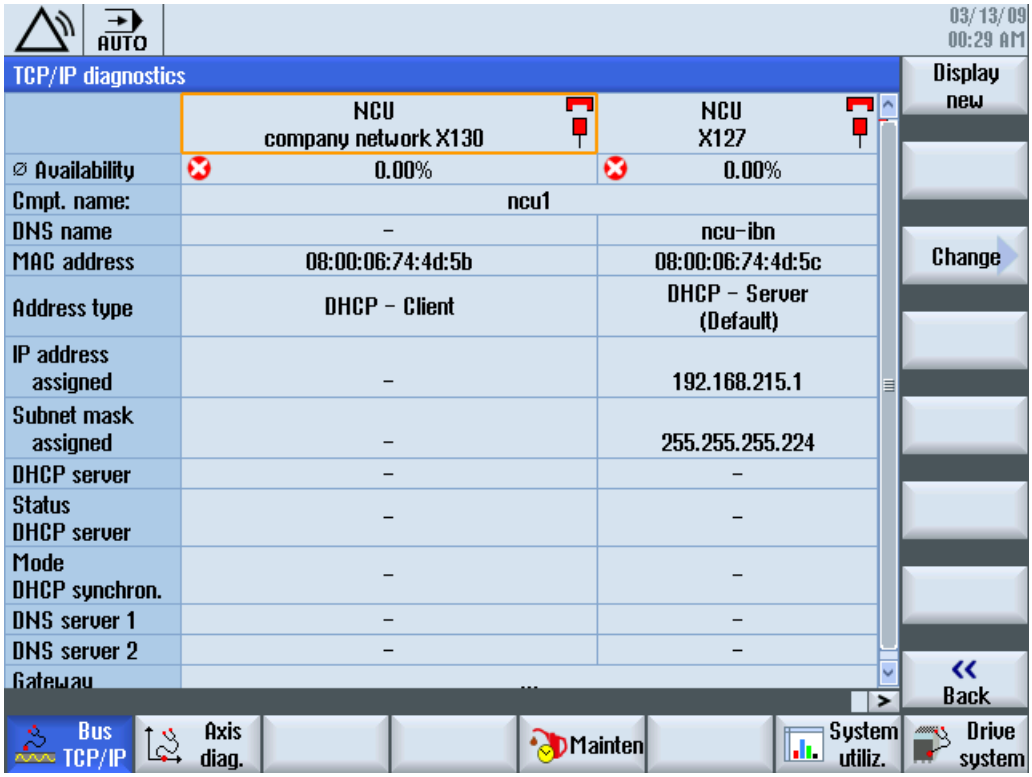


Figure 1-13 Network settings

Connection properties

Company network X130

	white	Network cable inserted
	red	Network cable not inserted

Availability

The availability describes the percentage of faulty data compared to the entire data volume. Problems in the company network (e.g. logical drives that cannot be reached, double IP address, etc.) as well as settling time during power up can result in fluctuations in availability:

	green	Greater than 95%
	yellow	50 - 95 %
	red	Less than 50%

Note

All information that is not available is marked in the relevant table line with a hyphen "-".

Introduction and use of data classes

2

Objective

The principle of data classes creates a clear demarcation between system data, (data that is created when installing the software), OEM data (data that is created when commissioning a prototype machine) and user data (data that is created by the user). OEM data is further classified into data that is valid for a series of machines (standard data for all machines of a series) and individual data, which only applies to a specific machine.

This data classification is realized when creating data class archives, i.e. archives that only include a subset of data (manufacturer data, individual machine data or user data). This simplifies commissioning, series commissioning and upgrading.

2.1 Data classes in the NCK

Classification

All control-relevant data is classified into data classes regarding its later use in commissioning, software upgrade, software update and component replacement. Data is classified according to four data classes: M (Manufacturer), I (Individual), U (User) and S (System)

System (S) data class

This data class covers data in the Siemens and System directories on the CompactFlash Card. For the rest of this document it will be referred to as "S".

Characteristics / properties of data class S

The data is automatically loaded the first time the system is switched on or during initialization and activated as far as is necessary.

"S" class data cannot be backed up and are designated as "read only".

Data type	Example
HMI:	Siemens standard user interface and additional languages available
NCK:	Standard cycles, measurement cycles and JobShop cycles, definitions such as SGUD and SMAC and all NC active data with data class S
PLC:	Hardware configurations
Drives:	Code and data (data descriptions, alarm descriptions, macros)

Manufacturer (M) data class

This data class covers all data defined by the machine manufacturer (OEM) when a machine in a model series is commissioned for the first time. For the rest of this document it will be referred to as "M".

Characteristics / properties of data class M

Class M data is stored in a separate "Manufacturer" archive.

Data type	Example
HMI:	OEM dialogs (Easy Screen) and cycle support, alarm and message texts.
NCK:	Manufacturer cycles, MGUD and MMAC definitions and all NC active data with data class M (no memory normalizing machine data)
PLC:	PLC program, contents of the data blocks in this data class
Drives:	all

Individual (I) data class

This data class covers data concerning a particular machine and is also created during commissioning by the OEM, or later by the dealer. This data class will be referred to as "I" in the rest of this document.

Characteristics / properties of data class I

Class I data is stored in a separate INDIVIDUAL archive.

Data type	Example
HMI:	Easy Extend and Service Planner
NCK:	Reference point dimensions, backlash compensation, compensation data, tool carrier data, all NC active data with data class I
PLC:	Contents of the data blocks in this data class
Drives:	No data available at this time.

User (U) data class

This data class covers all user data as well as data that is created during the machine runtime, for example the maintenance interval timer. This data class will be referred to as "U" in the rest of this document.

Characteristics / properties of data class U

Class U data is stored in a separate "User" archive.

Data type	Example
HMI:	---
NCK:	Tool data, setting data, part programs, subprograms and user cycles, UGUD and UMAC definitions, but no program code (e.g. PLC)
PLC:	Contents of the data blocks in this data class
Drives:	No data available at this time.

GUD and macros

The following table shows how the definition files (GUD, macros) are assigned to a data class:

Definition file	Data class
GUD4	Individual (I)
MGUD	Manufacturer (M)
MMAC	Manufacturer (M)
UGUD	User (U)
UMAC	User (U)

Data class attribute

The data class attribute is set implicitly. The default setting of the data class attribute can be explicitly changed.

- In general, data from the active file system is assigned to a **data class** as it belongs to a certain file.

  Example: Compensation data (data from CEC, EEC or QEC) is assigned to the "Individual" data class. If data are generated as a result of a definition (GUDs and macros) then this data also inherits the data class in which the definition file is located. GUD from UGUD.DEF are assigned the "User" data class.

- However, if the data definition and data content are in **different data classes**, then this must be specified in the data definition.

  Example: The definition of the GUD, which defines a probe, should be located in the "Manufacturer" data class, as it is required to run the manufacturer cycles.

  However, the value of the data should belong to the "Individual" data class as the probe type can differ from machine to machine. This is the reason that the additional variable attribute "DC" (DataClass) has been introduced:

```
MGUD.DEF ("Manufacturer" data class)
DEF CHAN DCI INT CALIPER
```

The definition of the data class of the data content can only be "less than or equal to" the data class of the definition. Any other configurations are rejected with an alarm.

For this purpose, the following weighting of the data classes is assumed: **M > I > U**

2.2 Data classes in the PLC

Data classes in the PLC

Assigning program blocks in data classes is used to efficiently upgrade or commission a SINUMERIK control. Here, a distinction is made between three data classes:

- **Manufacturer (M)** is allocated by the machine manufacturer during construction.
- **Individual (I)** is allocated by the machine manufacturer during the first commissioning (values for this specific machine).
- **User (U)** is set by the end customer (values associated with the particular process). You cannot assign a different data class to the main program or to subroutines.

Block	Default setting	Can be changed
Main program (OB1)	MANUFACTURER	No
Subroutine (SBR, INT, ...)	MANUFACTURER	No
Data block	INDIVIDUAL MANUFACTURER USER	Yes

Example

The data classes are assigned to the block in the dialog "Properties":

1. In the menu,"View" > "Data block", select a data block.
2. In the menu "View" → select "Properties" to change the data class.

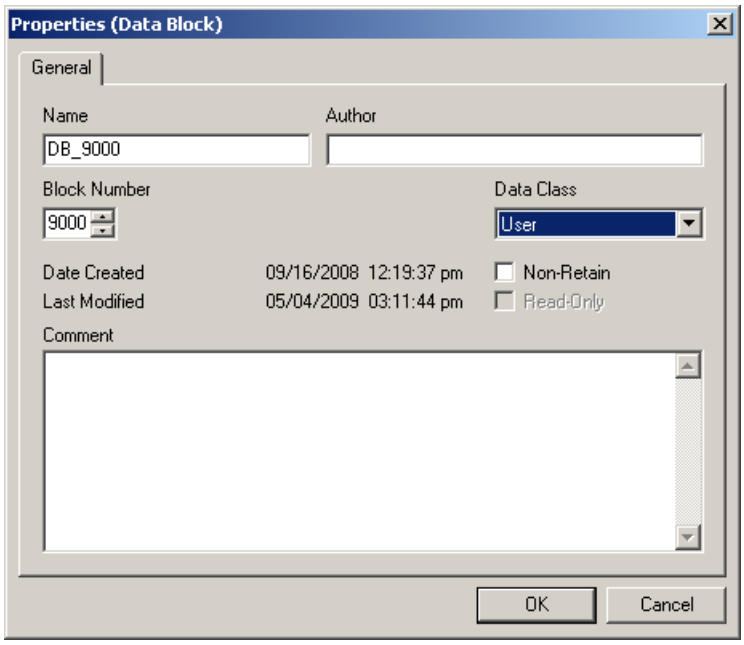


Figure 2-1 Selecting the data class

2.3 Data classes in the operating software

Representing the data classes in the operating software

The arrangement in data classes is essentially already given by the directory structure of the CompactFlash card:

Directory	Data class
system siemens	SYSTEM
addon	MANUFACTURER
oem	MANUFACTURER, INDIVIDUAL
individual	MANUFACTURER, INDIVIDUAL
user	USER

The directories "System" and "Siemens" have no significance for archiving, since they are set up with the installation of the SINUMERIK software and are not changed by installation, configuration or in later use. An update or upgrade of the system will typically be carried out in these directories. The archiving of these directories is for this reason only necessary as a rollback backup in the background.

Further subdividing of these directories into SINUMERIK NCK / PLC / HMI and SINAMICS covers the data areas described above and is echoed in all named directories (data classes).

CompactFlash card	Data area
SINUMERIK	NCK
	PLC
	HMI
SINAMICS	Drives

Structure of the CompactFlash card

The file system comprises the uppermost level of the directories "addon", "individual", "oem", "siemens" and "user": These directories have essentially an identical structure.

The section of the directory structure relevant for the configuration is shown below:

siemens directory		
/siemens/sinumerik		
	/hmi	
	/appl	Applications (operating areas)
	/base	Basic system components
	/cfg	All configuration files

siemens directory			
	/data		Version data
	/hlp		Online help files
	/hlps		Online help files, zipped and version files
	/ico		Icon files
		/ico640	Icons with a resolution 640x480
		/ico800	Icons with a resolution 800x600
		/ico1024	Icons with a resolution 1024x768
		/ico1280	Icons with a resolution 1280x1024
		/ico1600	Icons with a resolution 1600x1240
	/lng		Text files
	/lngs		Text files zipped and versions files
	/osal		
		/ace	ACE/TAO
		/qt	Qt
	/proj		Easy Screen configurations
	/template		Various templates
		/cfg	Templates for configuration files
		/ing	Templates for text files
	/tmpp		Storage, temporary data

addon directory			
/addon/sinumerik			
	/hmi		
	/appl		Applications (operating areas)
	/cfg		Configuration files
	/data		Version data
	/hlp		Online help files, zipped and version files
	/ico		Icon files
		/ico640	Icons with a resolution 640x480
		/ico800	Icons with a resolution 800x600
		/ico1024	Icons with a resolution 1024x768
		/ico1280	Icons with a resolution 1280x1024
		/ico1600	Icons with a resolution 1600x1240
	/lng		Text files
	/lngs		Text files zipped and versions files
	/proj		Easy Screen configurations
	/template		Various templates

oem directory			
/oem/sinumerik			
	/data		Version data
		/archive	Manufacturer archives
	/hmi		
		/appl	Applications (operating areas)
		/cfg	Configuration files
		/data	Version data
		/hlp	Online help files
		/hlps	Online help files, zipped and version files
		/ico	Icon files
		/ico640	Icons with a resolution 640x480
		/ico800	Icons with a resolution 800x600
		/ico1024	Icons with a resolution 1024x768
		/ico1280	Icons with a resolution 1280x1024
		/ico1600	Icons with a resolution 1600x1240
		/lng	Text files
		/lngs	Text files zipped and versions files
		/proj	Easy Screen configurations
		/template	Various templates

user directory			
/user/sinumerik			
	/data		Version data
		/archive	User-specific archives
		/prog	User-specific programs
	/hmi		
		/cfg	Configuration files
		/data	Version data
		/hlp	Online help files
		/ico	Icon files
		/ico640	Icons with a resolution 640x480
		/ico800	Icons with a resolution 800x600
		/ico1024	Icons with a resolution 1024x768
		/ico1280	Icons with a resolution 1280x1024
		/ico1600	Icons with a resolution 1600x1240
		/lng	Text files
		/log	Log files
		/md	Machine data views
		/proj	Easy Screen configurations

Operating software settings 3

3.1 Access levels

Access to functions and machine data

The user only has access to information corresponding to a particular access level and the levels below it. The machine data is assigned different access levels.

The access concept controls access to functions and data areas. Access levels 0 to 7 are available, where 0 represents the highest level and 7 the lowest level. Access levels 0 to 3 are locked using a password and 4 to 7 using the appropriate key-operated switch settings.

Access level	Locked by	Area	Data class
0	---	System (reserved)	System (S)
1	Password: SUNRISE	Manufacturer	Manufacturer (M)
2	Password: EVENING	Service	Individual (I)
3	Password: CUSTOMER	User	User (U)
4	Key-operated switch setting 3	Programmer, machine setter	User (U)
5	Key-operated switch setting 2	Qualified operator	User (U)
6	Key-operated switch setting 1	Trained operator	User (U)
7	Key-operated switch setting 0	Semi-skilled operator	User (U)

The password remains valid until it is reset with the "Delete Password" softkey. The passwords can be changed after activation.

If, for example, the passwords are no longer known, reinitialization (power up with "NCK default data") must be carried out. This resets all passwords to the default (see table). POWER ON does not reset the password.

Note

PI LOGOUT

The password can also be deleted via the PLC (PI services).

See also: Function Manual Basic Functions, Chapter "PLC for SINUMERIK 828D" (P4)

Key-operated switch

Access levels 4 to 7 require a corresponding key-operated switch setting on the machine control panel. Three keys of different colors are provided for this purpose. Each of these keys provides access only to certain areas.

Meaning of the key-operated switch settings:

Access level	Switch setting	Key color
4-7	0 to 3	red
5-7	0 to 2	green
6-7	0 and 1	black
7	0 = Key removal position	No key inserted

The key-operated switch setting must always be edited from the PLC user program and applied to the interface accordingly.

3.2 How to set and change the password

Set password

To change the access level, select the "Start-up" operating area:

1. Press the "Password" softkey.
2. Press the "Set password" softkey to open the following dialog:

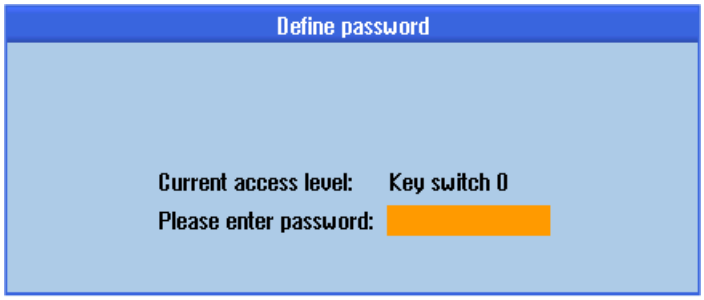


Figure 3-1 Set password

3. Enter a password and confirm this with "OK" or with the <Input> key.

   A valid password is acknowledged as set and the currently applicable access level is displayed. Invalid passwords will be rejected.

4. You must delete the old password before activating a password for a lower access level than the one activated.

   The last valid password is deleted by pressing the "Delete password" softkey. Then the current key-operated switch setting is valid.

Change password

To change the password:

1. Press the "Change password" softkey to open the following dialog:

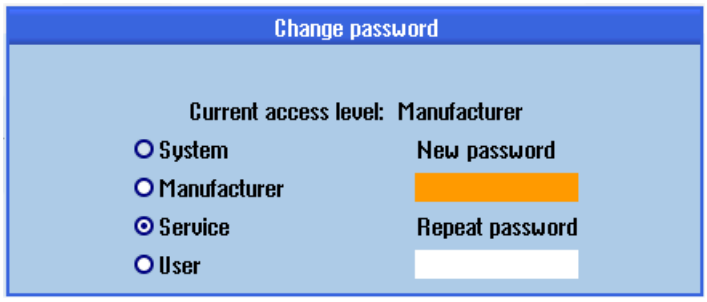


Figure 3-2 Change password

2. Enter the new password in both fields and then confirm with the "OK" softkey. If both passwords match, the new password becomes valid and is adopted by the system.

3.3 How to set the date and time

Requirement

Changes can only be made with the appropriate access authorization (as of "User" and higher).

Setting the date and time

Procedure:

1. Select the "Start-up" operating area.
2. Press the "HMI" softkey.
3. Press the "Date/Time" softkey.

   The "Date/Time" window opens.
4. Select the required formats for the date and time in the "Format" field.
5. Confirm the entry with the "OK" softkey.

   The new date and time details are accepted and output on the first line in the "current" fields.

3.4 Setting the operating software languages

3.4.1 Available system languages

System languages

In the basic configuration, the SINUMERIK 828D is delivered with the following system languages:

- English
- French
- German
- Italian
- Korean
- Portuguese (Brazil)
- Chinese (simplified)
- Spanish
- Chinese (traditional)

All system languages are installed in the SINUMERIK 828D as delivered, so that a change of language can be carried out directly via the user interface, without having to download system language data.

Additional languages

No CNC option needs to be ordered for the installation of additional languages not included in the scope of delivery. Other languages that are available:

Language	Abbreviation in file name
• Czech	csy
• Danish	dan
• Finnish	fin
• Hungarian	hun
• Japanese	jpn
• Dutch	nld
• Polish	plk
• Rumanian	rom
• Russian	rus
• Slovakian	sky
• Swedish	sve
• Turkish	trk

3.4.2 This is how you install additional system languages

Additional system languages

The language extensions are available as tgz file (eSupport: unzip the zip file) and are installed in the control just the same as for a software update.

Preconditions:

1. Copy the tgz file to a storage medium, i.e.CompactFlash Card or USB FlashDrive.
2. Switch-off the control.
3. Insert the storage medium into the slot at the front panel of the control.

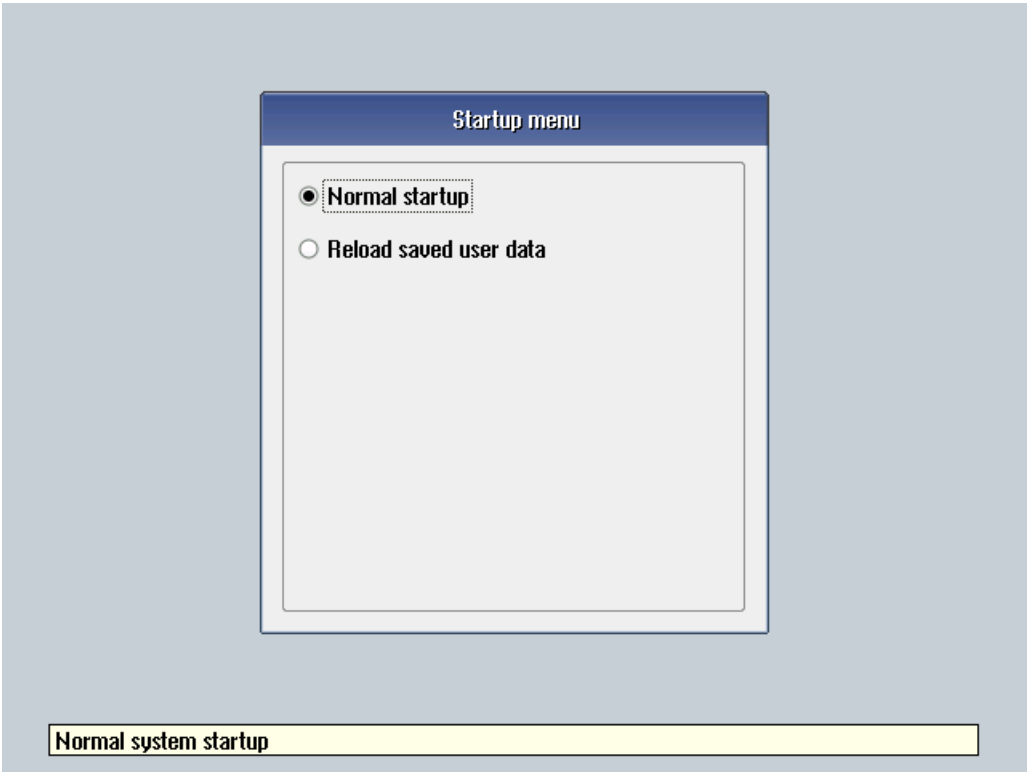
Installing a language

Procedure:

1. Switch-on the control and the following display then appears:



2. Press the <SELECT> key, "Normal startup" is the default setting.

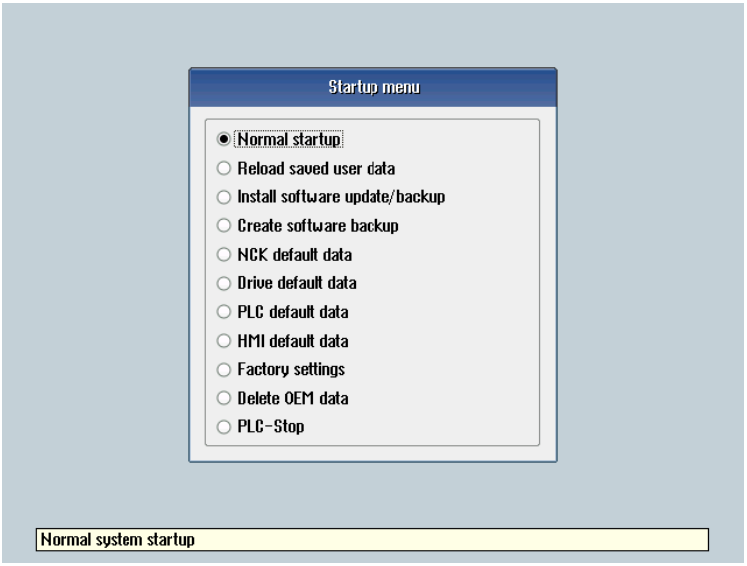


3. Now press the following keys in succession:

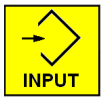
   Menu reset key, **HSK2** (horizontal SK2), **VSK2** (vertical SK2)



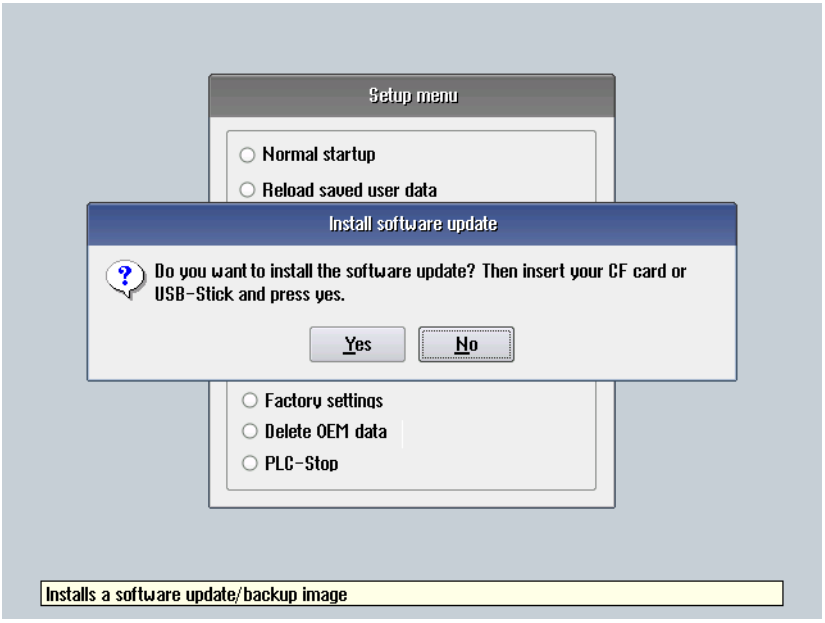
4. The setup menu is displayed:



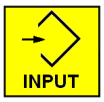
5. Using the cursor keys, select the menu item "Install software update/backup".
6. Press the <INPUT> key to confirm your selection.



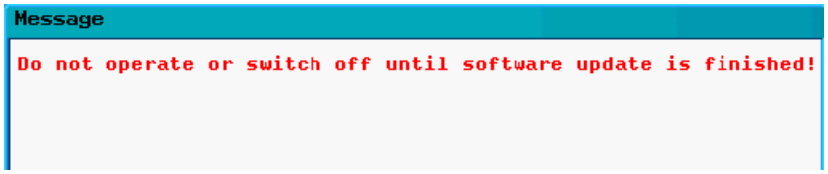
7. Use the cursor keys to select "Yes".



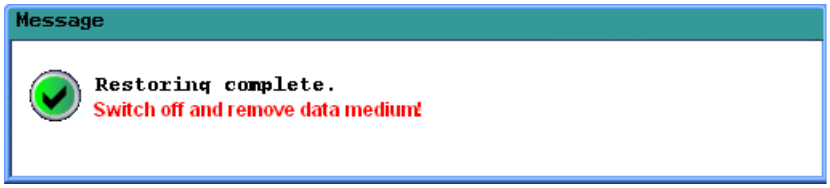
8. Select the update image (*.tgz) on the storage medium and confirm with <INPUT>.



9. The software update is started: The following message appears while the update is running:



10. Wait until the following message is output:



11. Remove the storage medium from the slot.
12. Switch-off the control.
13. Switch-on the control.

Note

After the installation and after switching-off the system for the first time, it is possible that the newly installed language cannot be selected.

In this case, completely switch-off the control again and then switch it on (Power ON).

3.4.3 Entering Asian characters using the Input Method Editor

Input Method Editor (IME)

The program editor and PLC alarm text editor both allow you to edit Asian characters. The Input Method Editor is available as support for the following Asian languages:

- Chinese simplified
- Chinese traditional (Taiwanese)
- Korean

The editor is started using the shortcut key <Alt>+<S>.

Editor with active learning function

For Chinese simplified and Chinese traditional, the system offers the option of working with dictionaries:

- Using a dictionary at the control
- Importing a dictionary into the control

The editor opens a learning function if a phonetic spelling is entered, for which there is no match in the control system. This function allows syllables or words to be combined, which are then permanently available after saving. The editor displays the combination of Chinese characters next to the pinyin phonetic spelling. After being completely compiled, the word should then be saved by pressing the <Input> key. It is simultaneously inserted in the associated text box.

Editing the dictionary

When this function is activated, another line showing the combined characters and sounds will appear. The editor will then offer various characters for this sound, from which you can choose the desired one by entering the appropriate digit (1 ... 9). You can toggle the input cursor between the compound phonetic notations field and the phonetic input field by pressing the <TAB> key.



When the cursor is positioned in the upper field, you can undo the combination by pressing the <backspace> key.



Press <Select> to save the characters currently being displayed.



Press the <Delete> key if you want to delete the currently displayed group of characters from the dictionary.

Importing a dictionary

A dictionary can now be generated using any Unicode editor by attaching the corresponding Chinese characters using the pinyin phonetic spelling. If the phonetic spelling contains several Chinese characters, then the line must not contain any additional match. If there are several matches for one phonetic spelling, then these must be specified in the dictionary line by line. Otherwise, several characters can be specified for each line.

The generated file should be saved in the UTF8 format under the name chs_user.txt (simplified Chinese) or cht_user.txt (traditional Chinese).

Line structure:

Pinyin phonetic spelling <TAB> Chinese character <LF>

OR

Pinyin phonetic spelling <TAB> Chinese character1<TAB> Chinese character2 <TAB> … <LF>

<TAB> - tab key

<LF> - line break

Save the dictionary that has been a generated under the following path:

```
../user/sinumerik/hmi/ime/
```

When the Chinese editor is called the next time, this enters the content of the dictionary into the system dictionary.

Example:

ai	哎　哀　唉　埃　挨
caise	彩色
hongse	紅色
huise	灰色
heli	河裏
zuihaowan	最好玩

3.4.4 This is how you enter the Chinese character

Entering Chinese characters

A character is selected using a phonetic spelling ("pinyin method"): whose sound can be formed by combining Latin letters.

1. Enter the sound by combining Latin letters.

   The editor lists a selection of characters that correspond to that particular sound.

2. Select the desired character using the cursor keys:

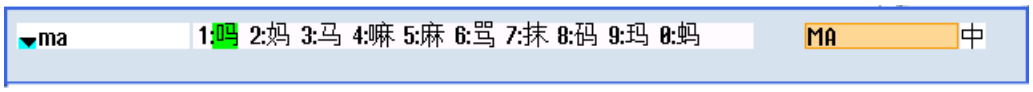


Figure 3-3 Example: Chinese simplified

If you switch over the option button to enter a Latin letters using the <SELECT> key, then the entries are directly transferred into the text box that had the focus before the Chinese editor was opened.

Zhuin input for Chinese traditional

For traditional Chinese, the following input options are available:

1. Use the numerical block of the keyboard to form the individual syllables.

   Each number is assigned a certain number of letters, that can be selected by pressing the numeric key one or several times.

2. In order to accept the selection made, which is displayed in the Zhuyin text box, confirm the selection using the <INPUT> key or enter an additional number.

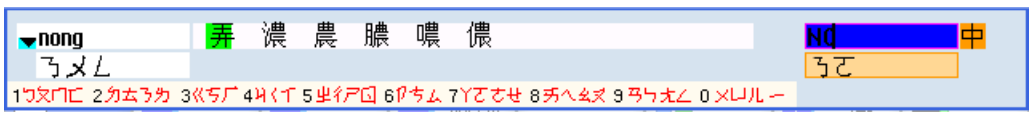


Figure 3-4 Example: Zhuin method

3.4.5 This is how you enter Korean characters

Enter characters using a matrix

If only the keyboard is available at the control, then a matrix technique is used, which only requires the numerical block:

1. Using the first number, select the line: the line is then highlighted in color.
2. Using the second number, select the column: the character will be briefly highlighted in color and then transferred to the "Character" field.

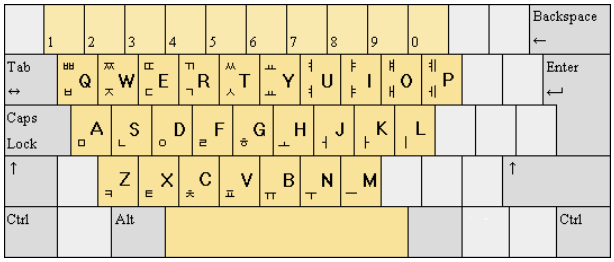
You can toggle between Korean and Latin using the <SELECT> key.



Figure 3-5 Example: Program editor

Using the Korean keyboard

To enter Korean characters, you will need a keyboard with the keyboard assignment shown below. In terms of key layout, this keyboard is the equivalent of an English QWERTY keyboard and individual characters must be grouped together to form syllables.



Example: Korean keyboard

The Korean alphabet (Hangeul) consists of 24 letters: 14 consonants and 10 vowels. Syllables are formed by combining consonants and vowels.

3.5 Checking and entering licenses

Use

The use of the installed system software and the options activated on a SINUMERIK control system require that the licenses purchased for this purpose are assigned to the hardware. In the course of this assignment, a license key is generated from the license numbers of the system software, the options, as well as the hardware serial number. Here, a license database administered by Siemens is accessed via the Internet. Finally, the license information including the license key is transferred to the hardware.

The license database can be accessed using the Web License Manager.

Web License Manager

By using the Web License Manager, you can assign licenses to hardware in a standard Web browser. To conclude the assignment, the license key must be entered manually on the control via the user interface.

Internet address: Web License Manager (http://www.siemens.com/automation/license)

Note

SINUMERIK software products

If a license key has not been activated or does not exist for a SINUMERIK software product, alarm 8081 is output by the control and NC START cannot be executed.

See also

Definitions for license management (Page 427)

3.5.1 How to enter a license key

Requirement

The appropriate licenses are required for the activated options. After licensing the options in the Web License Manager, you receive a "license key" containing all options requiring a license and which is only valid for your system CompactFlash card.

To set or reset options, "Manufacturer" access rights are required.

Reading in or entering a license key

Procedure:

1. Select the "Start-up" operating area.
2. Press the menu forward key.
3. Press the "Licenses" softkey.
4. The "Licensing" window opens
5. Select the input line in order to enter a new license key.

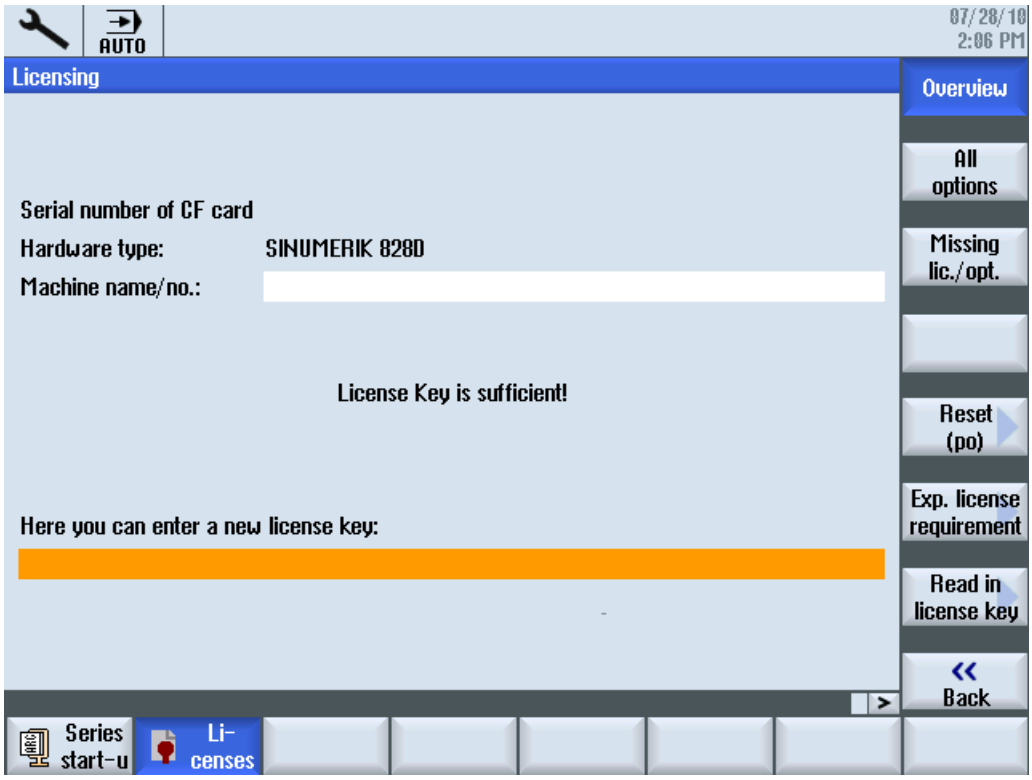


Figure 3-6 Licensing

Other actions include:

- Softkey "All options": All options are displayed.
- Softkey "Missing licenses/options": Missing licenses and options are displayed: This is how you determine missing licenses/options (Page 53)
- Softkey: "Export license requirement": The missing licenses are exported in a file and can be saved on a storage medium.
- Softkey: "Read in the license key": The license key is read in from the license file.

  This file is supplied or can be sourced from the Web License Manager (http://www.siemens.com/automation/license).

3.5.2 This is how you determine missing licenses/options

Determining the license requirement

Procedure:

1. Press the "All options" softkey to list all the options that can be selected for this control.
2. Activate or deactivate the required options in the "Set" column:
   - Mark the checkbox
   - Enter the number of options
3. Press the "Missing lic./opt." softkey in order to display all options that are licensed. In the "Set" column, you can deselect the options that you do not require.

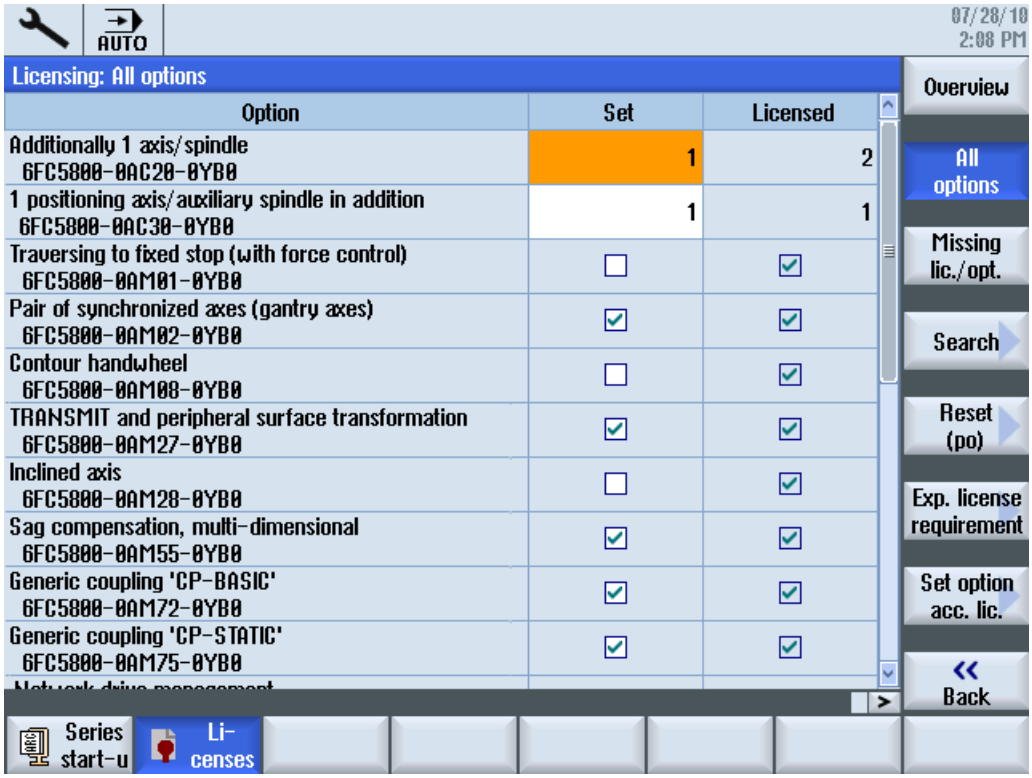


Figure 3-7 Licensing (example)

4. Press the softkey "Set option according to license", to activate all of the options contained in the license key. Confirm the following confirmation prompt with "OK".
5. To activate new selected options, press the "Reset (po)" softkey. A safety prompt appears.
6. Press the "OK" softkey to trigger a warm restart.

   - OR -
7. Press the "Cancel" softkey to cancel the process.

3.6 Configuring user alarms

Creating user PLC alarms

The PLC alarms in the area from **700 000 - 700 247** are configured by the machine manufacturer. The access level "Manufacturer" is required with the appropriate password.

To enter the user PLC alarms via the user interface, select → "HMI"→ "Alarm texts" in the "Start-up" operating area.

Then you receive the following selection:

Alarm texts for	Name of the xml file
User cycle alarms	oem_alarms_cycles
User PLC alarms	oem_alarms_plc
User part program message texts	oem_partprogram_messages

Loading user PLC alarms

The alarm text files are only loaded during startup.

- "Alarm" attribute: red, is shown in the "alarm list".
- "Message" attribute: black, is shown under "Messages".

Select <MENU SELECT>, then the menu forward key and press the "HMI restart" softkey to load the alarm texts.

See also

You can find a detailed description of the alarms with system responses and deleting criteria in: SINUMERIK 828D Diagnostics Manual

3.6.1 Structure of user PLC alarms

Structure of a user PLC alarm

The user PLC alarms have the following structure:

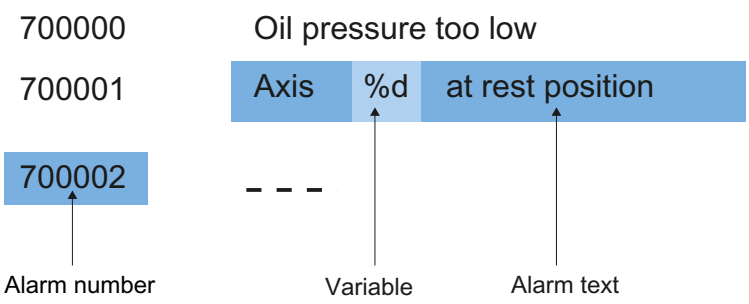


Figure 3-8 Alarm structure

The table below shows the mode of operation of the PLC alarms:

1. The alarm is triggered with the appropriate number and output via the PLC signal.
2. If a variable has been configured to this alarm, the value of this variable is in the specified data word of the PLC variable.
3. The NCK response when the alarm is triggered is defined in the MD14516[x] index (see table below).
4. The alarm text can be freely selected and may be up to 255 characters long.

Alarm number	PLC signal	PLC variable	Alarm response (MD)	Alarm text
700 000	DB1600.DBX0.0	DB1600.DBW1000	14516[0]	Alarm 1
700 001	DB1600.DBX0.1	DB1600.DBW1004	14516[1]	Alarm 2
700 002	DB1600.DBX0.2	DB1600.DBW1008	14516[2]	Alarm 3
700 003	DB1600.DBX0.3	DB1600.DBW1012	14516[3]	Alarm 4
700 004	DB1600.DBX0.4	DB1600.DBW1016	14516[4]	Alarm 5
700 005	DB1600.DBX0.5	DB1600.DBW1020	14516[5]	Alarm 6
700 006	DB1600.DBX0.6	DB1600.DBW1024	14516[6]	Alarm 7

Continuation:				
700 247	DB1600.DBX30.7	DB1600.DBW1988	14516[247]	Alarm 248

Defining the NCK response

The following NCK responses are possible:

MD14516[x]	Meaning
Bit 0	NC start disabled
Bit 1	Read-in disable
Bit 2	Feed hold for all axes
Bit 3	EMERGENCY STOP
Bit 4	PLC in stop
Bit 5	Reserved
Bit 6	Definition for alarm or message Bit 6=1: → alarm, Bit 6=0: → message
Bit 7	POWER ON

Configuring alarm texts with variables

The following data types are permitted for variables in the alarm text:

Variable	Meaning
%b	Binary representation of a 32-bit value
%d	Integer decimal number
%f	4 byte floating point number
%i	Integer decimal number with sign
%o	Integer octal number
%u	Unsigned decimal number
%x	Integer hexadecimal number

3.6.2 How to create user PLC alarms

Notes for processing

The following points should be observed when processing the files:

- The files should be edited externally on a PG/PC with a text editor (e.g. notepad) or with an XML editor. The structure must not be altered.
- The created alarm text files are copied to the the following directory on the CompactFlash card: `oem/sinumerik/hmi/lng`
- To enable the system to recognize the alarm text file, the file name must be written in lower case letters.
- The alarm text file is converted during system startup: The system must be restarted to activate the alarms.

Procedure

To edit a larger number of alarms, first create 2 or 3 alarms directly on the control. Then the file oem_alarms_plc_xxx.ts is created and you have a "document template" with the correct structure, which you can then extend with further alarms. The abbreviation "xxx" stands for the language in which the file has been created.

1. Select the "Start-up" operating area.
2. Press the "HMI" softkey.
3. Press the "Alarm texts" softkey. The "Select file" window appears.
4. Select "oem_alarms_plc" to create user PLC alarm texts.
5. Enter the alarm number in the "Number" field and the desired alarm text in the "Text" field. The alarm numbers and their alarm texts do not have to be consecutive. If an alarm is triggered without a configured text, only the alarm number is specified.

Searching within the alarm texts

To search for a text or a series of characters:

1. Press the "Find >" softkey. The "Find" window opens; and a new menu is displayed on the vertical softkey bar.
2. Enter the search term in the "Text" field.
3. Place the cursor in the "Direction" field and choose the search direction (forward, backward) with the "SELECT" key.
4. Activate the "Case-sensitive" checkbox when a distinction is to be made between upper and lower case in the entered text.
5. Press the "Find + replace" softkey. The "Find and replace" window appears.
6. Press the "OK" softkey to start the search.
7. Press the "Cancel" softkey to cancel the search.

Other navigation options are:

- Softkey "Go to start":

  The cursor jumps to the first entry of the selected alarm text file
- Softkey "Go to end":

  The cursor jumps to the last entry of the selected alarm text file.

See also

Table: Language code in file names (Page 421)

Example: How to create an online help for user PLC alarms (Page 71)

3.6.3 Configuring the alarm log

Logging

Configure the alarm log in the "Diagnostics" operating area.

All alarms and messages are logged in chronological order with their raised and cleared time stamps. The exception are messages of the type "msg" from the NC part program. All alarms and messages that are no longer active when the log is displayed are also retained (historical alarm events).

The alarm log is organized as a ring buffer (default setting). The oldest entries are overwritten with new events in the following cases:

- When the maximum size is exceeded (permissible range: 0 - 32000).
- When the events happened before the last time the system was switched on.

Permanent backup

To save the alarm log permanently, the alarm log is written to the CompactFlash card.

NOTICE
Saving the alarm log For permanent storage, the alarm log is written to the CompactFlash card which only allows a limited number of write cycles. • Therefore, ensure that the backup is only performed when there is a justifiable need! • Make sure you undo the setting "on every event" if you no longer require storage of the alarm log. Default setting: The alarm log is not backed up.

See also

Filtering events: Set up a filter to limit the number of events in the alarm log. You can find more details on this in:

- Commissioning Manual Base Software and Operating Software, chapter "Configuring alarms".
- List of the alarm number ranges (Page 422)

3.6.4 How to configure the log

Configuring the log

Procedure:

1. Select the "Diagnostics" operating area.
2. Press the "Alarm log" softkey.
3. Press the "Settings" softkey.
4. Enter the desired number in the "Number of entries" field to change the maximum number of raised and cleared events.

   Default is 500 events; permissible value range 0 - 32000.
5. Select the type of logging under "File write mode":
   - "**Off**" if the events are not to be written to a file.
   - "**On every event**" if every event is to be written to a file.
   - "**Time controlled**" if the file is to be overwritten after a particular time interval. An additional "Time interval" input field appears in which you can specify the time in seconds.
6. Press the "Save log" softkey to save the alarm log.

The settings become effective only after restarting the system.

Editing the configuration file

Procedure:

1. Copy the configuration file "oem_alarmprot_slaesvcconf.xml" from the `/siemens/sinumerik/hmi/template/cfg` directory.
2. Insert the file into the directory `/oem/sinumerik/hmi/cfg` or `/user/sinumerik/hmi/cfg`
3. Name the file "slaesvcconf.xml".
4. Open the user-specific file "slaesvcconf.xml" in the editor.
5. Enter the number of events to be output in the <Records type .../> identifier.

   The default value is 500. The permissible number is in the range from 0 ... 32000.

OR:

The number of events to be output and the type of logging can also be entered directly via the user interface:

1. Press the "Alarm log" → "Settings >" softkey in the "Diagnostics" operating area.

   As soon as changes are made to the default settings, the "slaesvcconf.xml" file is automatically created in the `/user/sinumerik/hmi/cfg` directory.
2. Enter the mode of the permanent storage in the <DiskCare type="int" value="-1"/> identifier. The following values are possible:

   -1: There is no saving of the alarm log (default setting).

   0: Each alarm event triggers an immediate saving of the alarm log.

   >0: Time for saving the log in seconds:

   When there is a change, the log is saved every n > 0 seconds.
3. You adapt the filter for the entry type in the <Filter> identifier.

   Here the following applies:

   – An alarm event is only entered in the log when it satisfies the filter criteria.
   – When several filters are defined, these should be linked using the logical operators OR or AND.

The settings become effective only after restarting the system.

Note

Number of events

Each incoming or outgoing event of an alarm or message requires a separate entry, even when they belong to the same alarm or message.

Acknowledgement events are also contained in the alarm log. They also require an entry even when they are not recognizable in the alarm log.

Examples

All alarms that fulfill the following conditions are logged:

- CLEARINFO ≠ 15, therefore without part program messages:

```
<CONFIGURATION>
  <Protocol>
    <Filters>
      <Siemens_Filter_01 type="QString" value="CLEARINFO NOT 15" />
    </Filters>
  </Protocol>
</CONFIGURATION>
```

- "SEVERITY larger than 10" and "smaller than 500" :

```
<CONFIGURATION>
  <Protocol>
    <Filters>
      <Filter_01 type="QString" value= "SEVERITY HIGHER 10
 AND SEVERTY LOWER 500" />
    </Filters>
  </Protocol>
</CONFIGURATION>
```

3.7 Creating OEM-specific online help

Overview

In addition to the existing system online help, you also have the option of creating a manufacturer-specific online help and adding this to the operator software.

This online help is generated in the HTML format, i.e. it comprises HTML documents that are linked with one another. The subject being searched for is called in a separate window from a contents or index directory. Similar to a document browser (e.g. Windows Explorer), a list of possible selections is displayed in the left-hand half of the window and when you click the required subject, the explanation is displayed in the right-hand half of the window.

Context-sensitive selection of online help pages is not possible.

General sequence:

1. Generating HTML files
2. Generating a help book
3. Integrating the online help in the operator software
4. Saving help files in the target system

Other application cases

Online help for the following OEM-specific expansions can be generated and used to supplement the SINUMERIK Operate online help system:

- Online help for cycles and/or M functions of the machine manufacturer, which extend the programming options for SINUMERIK control systems. This online help is called in just the same way as the SINUMERIK Operate online help "Programming".

  See also: Example: This is how you generate a programming online help (Page 76)

- Online help for OEM-specific variables of the machine manufacturer. This online help is called from the variable view of SINUMERIK Operate.

  See also: Example: This is how you generate an online help for NC/PLC variables (Page 74)

3.7.1 Structure and syntax of the configuration file

Syntax description of the "slhlp.xml"

You require the configuration file "slhlp.xml" to integrate the help book in the existing online help system of the user interface:

Tag	Number	Meaning
CONFIGURATION	1	Root element of the XML document: Indicates that this involves a configuration file.
OnlineHelpFiles	1	Introduces the section of the help books.
<help_book>	*	Introduces the section of a help book.

Tag	Number	Meaning	
EntriesFile	1	File name of the help book with the list of contents and subject (keyword) entries. Attributes:	
		value	Name of the XML file
		type	Data type of the value (QString)
III-Technology	0,1	Specifies the technology for which the help book applies. "All" applies for all technologies. If the help book applies to several technologies, then the technologies are listed separated by comma. Possible values: All, Universal, Milling, Turning, Grinding, Stroking, Punching Attributes:	
		value	Technology data
		type	Data type of the value (QString)
DisableSearch	0,1	Disable the subject (keyword) search for the help book. Attributes:	
		value	true, false
		type	type, data type of the value (bool)
DisableFullTextSearch	0,1	Disable the full text search for the help book. Attributes:	
		value	true, false
		type	type, data type of the value (bool)
DisableIndex	0,1	Disable the subject index for the help book. Attributes:	
		value	true, false
		type	type, data type of the value (bool)
DisableContent	0,1	Disable the table of contents for the help book. Attributes:	
		value	true, false
		type	type, data type of the value (bool)
DefaultLanguage	0,1	Abbreviation for the language that should be displayed if the actual country language is available for the help book. Attributes:	
		value	chs, cht, deu, eng, esp, fra, ita, kor, ptb ...
		type	Data type of the value (QString)

The following applies to the "Number" column: * means 0 or several

Example of a "slhlp.xml" file

The help book "hmi_myhelp.xml" is configured in the following example; the subject index is not activated:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE CONFIGURATION>
     <CONFIGURATION>
       <OnlineHelpFiles>
         <hmi_myhelp>
           <EntriesFile value="hmi_myhelp.xml" type="QString"/>
           <DisableIndex value="true" type="bool"/>
         </hmi_myhelp>
       </OnlineHelpFiles>
     </CONFIGURATION>
```

3.7.2 Structure and syntax of the help book

Syntax for the help book

The help book is an XML file in which the structure of the online help is defined. The name of the file can be freely selected, e.g. "hmi_myhelp". In this file, you define:

- HTML documents
- Contents and subject index

Tag	Number	Meaning	
HMI_SL_HELP	1	Root element of the XML document	
BOOK	+	Identifies a help book. The name can be freely selected. Attributes:	
		ref	Identifies the HTML document that is displayed as the entry page for the help book.
		titel	Title of the help book that is displayed in the table of contents.
		helpdir	Directory that contains the online help of the help book.
ENTRY	*	Chapter of the online help Attributes:	
		ref	Identifies the HTML document that is displayed as entry page for the chapter.
		titel	Title of the chapter that is displayed in the table of contents.

Tag	Number	Meaning	
INDEX_ENTRY	*	Subject (keyword) to be displayed Attributes:	
		ref	Identifies the HTML document that is jumped to for this subject index entry.
		titel	Title of the subject that is displayed in the subject index.

The following applies to the "Number" column:

* means 0 or several

+ means 1 or several

Formatting the index

You have the following options to format the subject index:

- Single entry: `<INDEX_ENTRY ...title="index"/>`
- Two two-stage entry, whereby each title has a main and a subentry.

  Separate the entries with a comma.

  ```
  <INDEX_ENTRY ...title="mainIndex_1,subIndex_1 with mainIndex_1"/>
  ```

- Two-stage entry, whereby the first title is the main entry and the second title is the subentry.

  Separate the entries with a semicolon.

  ```
  <INDEX_ENTRY ...title="mainIndex_2;subIndex_2
  without mainIndex_1"/>
  ```

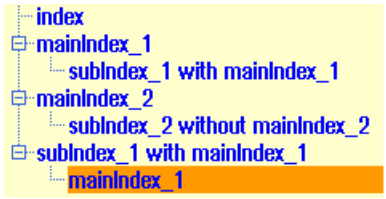


Figure 3-9 Example: Two-stage index

3.7.3 Description of the syntax for the online help

Rules for creating the HTML files

Generating help files in the HTML format. It is possible to save all information in a single HTML file or to distribute the information over several HTML files.

The file names are assigned taking into account the following rules:

- References within HTML files should always be specified with relative paths. Only then can it be ensured that the references function in precisely the same way on both the development computer as well as on the target system.
- If jumps are to be made to certain points within an HTML file per link, then so-called anchors must be defined for this purpose.

  Example of an HTML anchor:

  ```
  <a name="myAnchor">This is an anchor</a>
  ```

  See also: Example: How to create an online help for user PLC alarms (Page 71)
- The contents of HTML documents must be saved with the UTF-8 coding. This guarantees that the HTML documents are correctly displayed in all of the supported country languages.

HTML tags

The following sub-set of the HTML functional scope is supported:

Tag	Description	Comment
a	Anchor or link	Supported attributes: href and name
address	Address	
b	Bold	
big	Larger font	
blockquote	Indented paragraph	
body	Document body	Supported attributes: bgcolor (#RRGGBB)
br	Line break	
center	Centered paragraph	
cite	Inline citation	Same effect as tag i
code	Code	Same effect as tag tt
dd	Definition data	
dfn	Definition	Same effect as tag i
div	Document division	The standard block attributes are supported
dl	Definition list	The standard block attributes are supported
dt	Definition term	The standard block attributes are supported
em	Emphasized	Same effect as tag i
font	Font size, family, color	Supported attributes: size, face, and color (#RRGGBB)
h1	Level 1 heading	The standard block attributes are supported
h2	Level 2 heading	The standard block attributes are supported

Tag	Description	Comment
h3	Level 3 heading	The standard block attributes are supported
h4	Level 4 heading	The standard block attributes are supported
h5	Level 5 heading	The standard block attributes are supported
h6	Level 6 heading	The standard block attributes are supported
head	Document header	
hr	Horizontal line	Supported attributes: width (can be specified as absolute or relative value)
html	HTML document	
i	Italic	
img	Image	Supported attributes: src, width, height
kbd	User-entered text	
meta	Meta-information	
li	List item	
nobr	Non-breakable text	
ol	Ordered list	The standard attributes for lists are supported
p	Paragraph	The standard block attributes are supported (default setting: left-aligned)
pre	Preformated text	
s	Strikethrough	
samp	Sample code	Same effect as tag tt
small	Small font	
span	Grouped elements	
strong	Strong	Same effect as tag b
sub	Subscript	
sup	Superscript	
table	Table	Supported attributes: border, bgcolor (#RRGGBB), cellspacing, cellpadding, width (absolute or relative), height
tbody	Table body	No effect
td	Table data cell	The standard attributes for table cells are supported
tfoot	Table footer	No effect
th	Table header cell	The standard attributes for table cells are supported
thead	Table header	This is used to print tables that extend over several pages
title	Document title	
tr	Table row	Supported attributes: bgcolor (#RRGGBB)
tt	Typewrite font	
u	Underlined	
ul	Unordered list	The standard attributes for lists are supported
var	Variable	Same effect as tag tt

Block attributes

The following attributes are supported by the tags div, dl, dt, h1, h2, h3, h4, h5, h6, p:

- align (left, right, center, justify)
- dir (ltr, rtl)

Standard attributes for lists

The following attributes are supported by tags ol and ul:

- type (1, a, A, square, disc, circle)

Standard attributes for tables

The following attributes are supported by tags td and th:

- width (absolute, relative, no-value)
- bgcolor (#RRGGBB)
- colspan
- rowspan
- align (left, right, center, justify)
- valign (top, middle, bottom)

CSS properties

The following table includes the supported CSS functional scope:

Property	Values	Description
background-color	<color>	Background color for elements
background-image	<uri>	Background image for elements
color	<color>	Foreground color for text
text-indent	<length>px	Indent the first line of a paragraph in pixels
white-space	normal | pre | nowrap | pre-wrap	Defines how whitespace characters are handled in HTML documents
margin-top	<length>px	Width of the upper edge of the paragraph in pixels
margin-bottom	<length>px	Width of the lower edge of the paragraph in pixels
margin-left	<length>px	Length of the left hand edge of the paragraph in pixels
margin-right	<length>px	Width of the righthand edge of the paragraph in pixels
vertical-align	baseline | sub | super | middle | top | bottom	Vertical alignment for text (in tables, only the values middle, top and bottom are supported)
border-color	<color>	Border color for text tables
border-style	none | dotted | dashed | dot-dash | dot-dot-dash | solid | double | groove | ridge | inset | outset	Border style for text tables

Property	Values	Description
background	[<'background-color'> || <'background-image'>]	Short notation for background property
page-break-before	[auto | always]	Page break before a paragraph/table
page-break-after	[auto | always]	Page break after a paragraph/table
background-image	<uri>	Background image for elements

Supported CSS selectors

All CSS 2.1 selector classes are supported with the exception of so-called pseudo selector classes such as :first-child, :visited and :hover.

3.7.4 Example: This is how you generate an OEM-specific online help book

Requirements

Generate the following files:

- Configuration file: "slhlp.xml"

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE CONFIGURATION>
     <CONFIGURATION>
       <OnlineHelpFiles>
         <hmi_myhelp>
           <EntriesFile value="hmi_myhelp.xml" type="QString"/>
           <DisableIndex value="false" type="bool"/>
         </hmi_myhelp>
       </OnlineHelpFiles>
     </CONFIGURATION>
```

- Definition of the help book: "hmi_myhelp.xml"

```
<?xml version="1.0" encoding="UTF-8"?>
<HMI_SL_HELP language="en-US">
     <BOOK ref="index.html" title="Easy Help" helpdir="hmi_myhelp">
       <ENTRY ref="chapter_1.html" title="Chapter 1">
         <INDEX_ENTRY ref="chapter_1.html" title="Keyword 1"/>
         <INDEX_ENTRY ref="chapter_1.html" title="Keyword 2"/>
       </ENTRY>
       <ENTRY ref="chapter_2.html" title="Chapter 2">
         <INDEX_ENTRY ref="chapter_2.html" title="Keyword 2"/>
```

```
      </ENTRY>
      <ENTRY ref="chapter_3.html" title="Chapter 3">
        <INDEX_ENTRY ref="chapter_3.html" title="Keyword 3"/>
        <ENTRY ref="chapter_31.html" title="Chapter 3.1">
          <INDEX_ENTRY ref="chapter_31.html" title="test;Chapter 3.1"/>
        </ENTRY>
        <ENTRY ref="chapter_32.html" title="Chapter 3.2">
          <INDEX_ENTRY ref="chapter_32.html" title="test;Chapter 3.2"/>
        </ENTRY>
      </ENTRY>
    </BOOK>
</HMI_SL_HELP>
```

Saving help files in the target system

In the following example, the structure of a help book with the name "Easy Help" with table of contents and subject index is described.

Procedure:

1. Copy the configuration file "slhlp.xml" to the following directory:
   `/oem/sinumerik/hmi/cfg`
2. Create a directory for the desired language of the online help under the following path: `/oem/sinumerik/him/hlp`

   Use the specified language code from Chapter Table: Language code in file names (Page 421).

   Note

   Notation

   The directory names must be written in lower case.

   For example, if you are integrating a help in English, create an "eng" folder.
3. Place the help book, e.g. "hmi_myhelp.xml" in the "eng" folder.
   `/oem/sinumerik/him/hlp/eng/hmi_myhelp.xml`
4. Copy the help files to the following directory:
   `/oem/sinumerik/him/hlp/eng/hmi_myhelp/`

The settings become effective only after restarting the system.

NOTICE
Updates or changes When displaying the table of contents and index of a help book, for quicker processing, the help files are saved in the binary format in the `/siemens/sinumerik/sys_cache/hmi/hlp` directory: slhlp_<Hilfebuch>_*_<lng>.hmi . In the example: slhlp_hmi_myhelp_*_eng.hmi These files must first be deleted so that the changes can take effect and be displayed in the online help.

Result

The book consists of three chapters with sections:

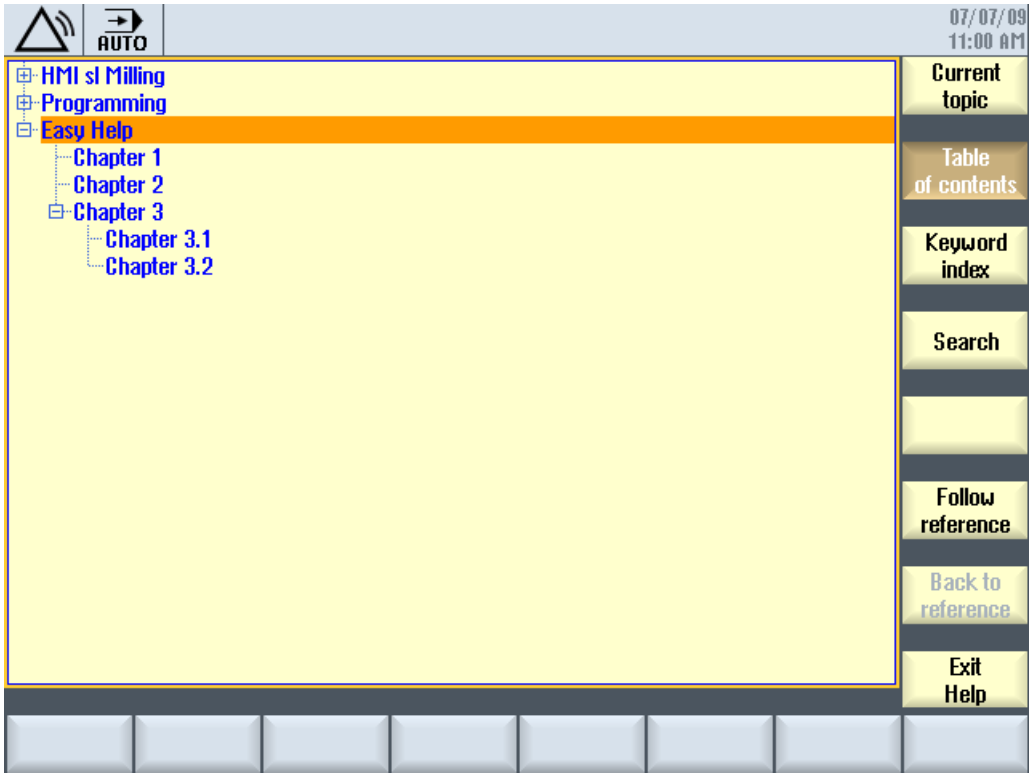


Figure 3-10 Example: OEM online help

Entries in the subject index:

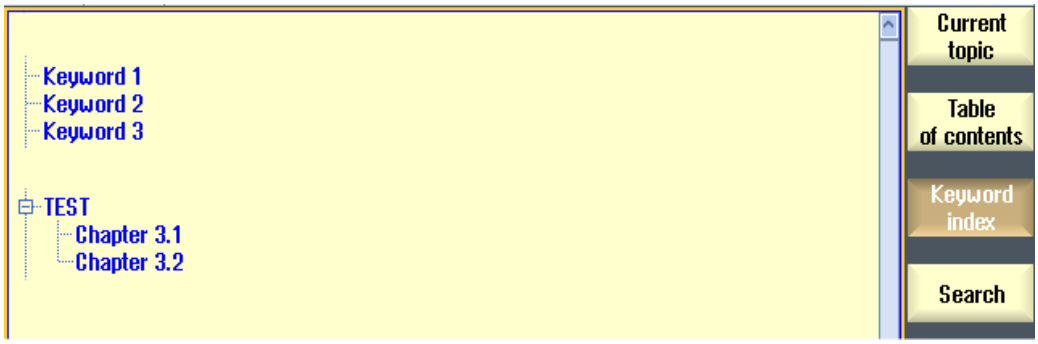


Figure 3-11 Example: Index

3.7.5 Example: How to create an online help for user PLC alarms

Overview

If a user PLC alarm is triggered, a context-sensitive online help can be created for the respective alarm, e.g. with explanation and remedy. The online help texts for the user PLC alarms are managed in the following file: "sinumerik_alarm_oem_plc_pmc.html"

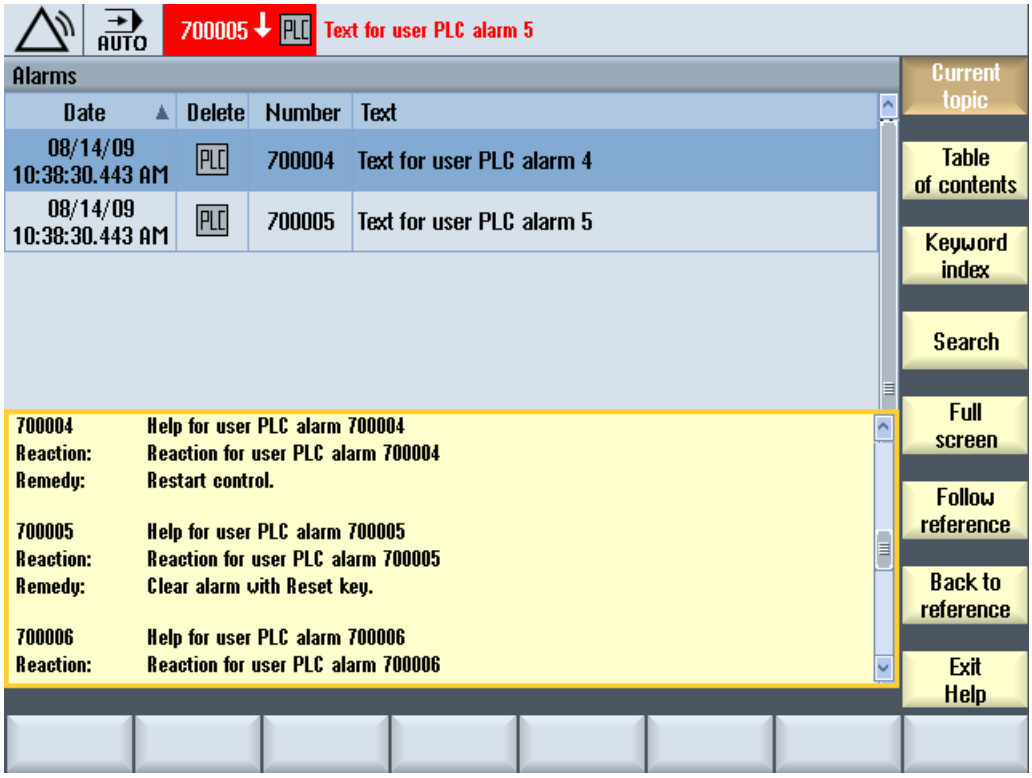


Figure 3-12 Example: Online help of user PLC alarms

Structure of the help file

The following entries are permitted in the help file:

Entry	Meaning
AlarmNo	Hyperlink to the alarm number
.....	Help text for the corresponding alarm
<td width="85%">.....</td>	Text that is displayed after the "Explanation" or "Remedy" field.

Creating a help file

The file name is **language-neutral** and must be:

sinumerik_alarm_oem_plc_pmc.html

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE html PUBLIC>

<html>
<head><title></title></head>
<body>
<table>
...
<tr>
<td width="15%">
<b><a name="700004">700004</a></b></td>
<td width="85%"><b>Help for user PLC alarm 700004 </b></td></tr>
<tr><td valign="top" width="15%"><b>Reaction: </b></td>
<td width="85%">Reaction for user PLC alarm 700004 </td></tr>
<tr><td valign="top" width="15%"><b>Remedy:</b></td>
<td width="85%">Restart control. </td>
</tr>
<br>

<tr>
<td width="15%">
<b><a name="700005">700005</a></b></td>
<td width="85%"><b>Help for user PLC alarm 700005 </b></td></tr>
<tr><td valign="top" width="15%"><b>Reaction: </b></td>
<td width="85%">Reaction for user PLC alarm 700005 </td></tr>
<tr><td valign="top" width="15%"><b>Remedy:</b></td>
<td width="85%">Clear alarm with Reset key. </td>
</tr>
<br>
```

sinumerik_alarm_oem_plc_pmc.html

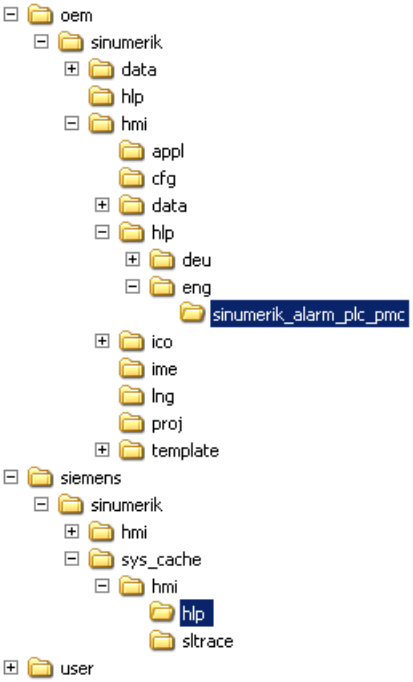
```
...
</table>
<p></p>
</body>
</html>
```

Procedure:

1. Copy the file to one of the following directories:

   `/oem/sinumerik/hmi/hlp/<lng>/sinumerik_alarm_plc_pmc/`
   `/user/sinumerik/hmi/hlp/<lng>/sinumerik_alarm_plc_pmc/`

   <lng> stands for the language code.



2. Delete all files in the directory:

   `/siemens/sinumerik/sys_cache/hmi//hlp`

The settings become effective only after restarting the system.

See also

Table: Language code in file names (Page 421)

3.7.6 Example: This is how you generate an online help for NC/PLC variables

Overview

In order to generate context-sensitive online help for NC/PLC variables or system variables as shown in the following example, the descriptive texts are managed in language-dependent html files.

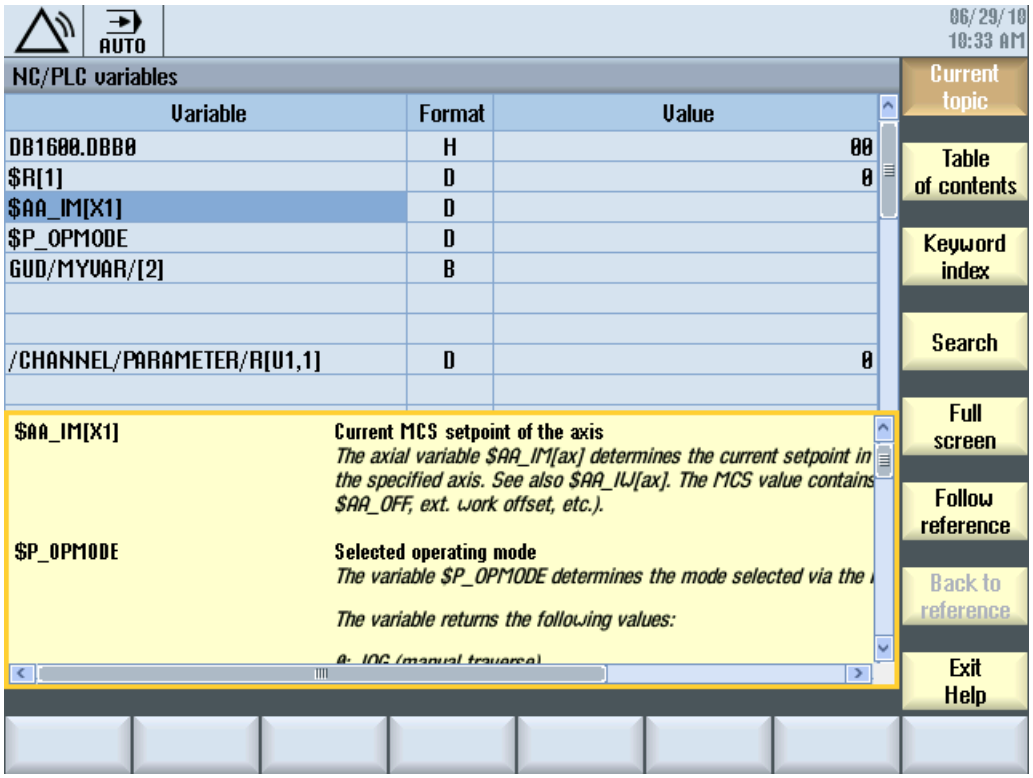


Figure 3-13 Example: Online help for user variables

Structure of the online help

The following files are required for the online help:

File	Meaning
`sldgvarviewhelp.ini`	Configuration file to manage an html file or several html files
`<lng>/<name>1.html`	The contents of all html files of the online help are language-dependent and are saved in the relevant language directory <lng>.
`<lng>/<name>2.html`	
`. . .`	
`<lng>/<name>n.html`	

Structure of the configuration file

The file is **language-neutral** and must be called:

sldgvarviewhelp.ini

```
[HelpBindings]
/BAG/STATE/OPMODE = var1_help.html#var1
$AA_IM[X1] = var1_help.html
$R[1] = var1_help.html#var2
/Channel/Parameter/R[u1,1] = var2_help.html#var2
DB1600.DBB0 = var2_help.html#var1
GUD/MyVar[2] = var2_help.html
```

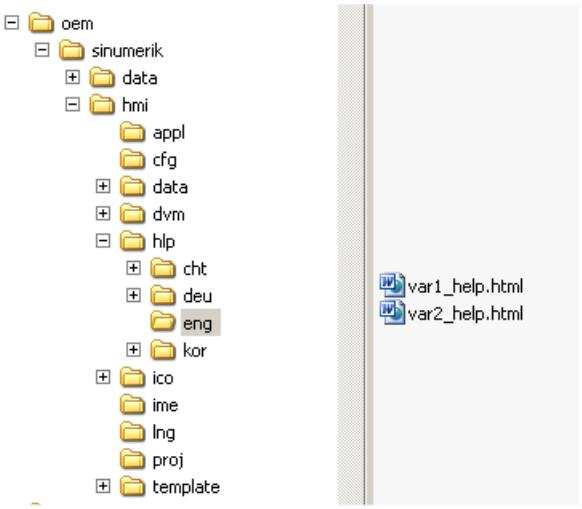
Note

The html files can be generated with any html editor. A definition is provided in the configuration file as to which html files belong to the online help.

The description can comprise one or several html files: For example, one html file for each variable or several identical variables in one file.

Procedure:

1. Copy the configuration file to the following directory:
   `/oem/sinumerik/hmi/cfg/sldgvarviewhelp.ini`
2. Copy the html files to one of the following directories:
   `/oem/sinumerik/hmi/hlp/<lng>/`
   `/user/sinumerik/hmi/hlp/<lng>/`
   <lng> stands for the language code.



3. Delete all files in the directory:
   `/siemens/sinumerik/sys_cache/hmi//hlp`

The settings become effective only after restarting the system.

3.7.7 Example: This is how you generate a programming online help

Requirements

Generate the following files:

- Configuration file "prog_help.ini"

```
[milling]
CYCLE1=cycle1_help.html
CYCLE2=cycle2_help.html#TextAnchor1
CYCLE3=cycle3_help.html
CYCLE4=cycle4_help.html

[turning]
CYCLE3=cycle2_help.html
CYCLE4=cycle3_help.html
```

- Configuration file for the help book "slhlp.xml" (optional)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE CONFIGURATION>
<CONFIGURATION>
  <OnlineHelpFiles>
    <hmi_prog_help>
      <EntriesFile value="hmi_prog_help.xml" type="QString"/>
      <DisableIndex value="true" type="bool"/>
    </hmi_prog_help>
  </OnlineHelpFiles>
</CONFIGURATION>
```

- Configuration file for the help book "hmi_prog_help.xml" (optional)

```
<?xml version="1.0" encoding="UTF-8"?>
<HMI_SL_HELP language="en-US">
  <BOOK ref="index.html" title="OEM_CYCLES" helpdir="hmi_prog_help">
    <ENTRY ref="cycle1_help.html" title="Cycle1"></ENTRY>
    <ENTRY ref="cycle2_help.html" title="Cycle2"></ENTRY>
    <ENTRY ref="cycle3_help.html" title="Cycle3"></ENTRY>
    <ENTRY ref="cycle4_help.html" title="Cycle4"></ENTRY>
    <ENTRY ref="cycle_help.html" title="OEM_Cycles"></ENTRY>
  </BOOK>
</HMI_SL_HELP>
```

- Language-dependent file "<prog_help_eng>.ts": this filename is permanently specified.

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!DOCTYPE TS>
<TS>
     <context>
       <name>oem_cycles</name>
         <message>
           <source>CYCLE1</source>
           <translation>short description for cycle 1</translation>
           <chars>*</chars>
         </message>
         <message>
           <source>CYCLE2</source>
           <translation>short description for cycle 2</translation>
           <chars>*</chars>
         </message>
         <message>
           <source>CYCLE3</source>
           <translation>short description for cycle 3</translation>
           <chars>*</chars>
         </message>
     </context>
</TS>
```

Saving help files in the target system

Procedure:

1. Copy the configuration file "prog.ini" to the following directory:
   `/oem/sinumerik/hmi/cfg`
2. Copy the file "`slhlp.xml`" to the help book in the following directory:
   `/oem/sinumerik/hmi/cfg`
3. Create a directory for the desired language of the online help under the following path: `/oem/sinumerik/hmi/hlp/<lng>` and copy the `hmi_prog_help.xml` file there

   Use the specified language code from the list of language codes for file names (Page 421). The directory names must be written in lower case.
4. Copy the language-dependent `prog_help_<lng>.ts` file for the brief description to the following path: `/oem/sinumerik/hmi/<lng>/prog_help_<lng>.ts`
5. Copy the html files with the description of the OEM cycles to the following directory:
   `/oem/sinumerik/him/hlp/<lng>/hmi_prog_help/cycle<n>_help.html`

The settings become effective only after restarting the system.

See also

Example: This is how you generate an OEM-specific online help book (Page 68)

Commissioning the PLC (Part 1) 4

Creating a PLC program

The PLC is commissioned with the Programming Tool. This is a Windows program and is installed on a PC. A typical Windows easy-to-use online help is available.

If the Programming Tool is called without specifying an existing project, a new project is implicitly created with the name "Project1". This project can be used immediately to create the PLC user program and then saved under an arbitrary name and loaded to the control.

Existing projects can be opened in the usual Windows manner.

See also

A detailed description on the operation of the Programming Tool, the properties and the programming of the PLC as well as further useful functions from the PLC can be found in:

Function Manual Basic Functions: Chapter PLC for SINUMERIK 828D (P4)

4.1 Activating I/O modules

General machine data

I/O modules, machine control panel and PN bus coupler are assigned fixed addresses for the input and output image of the PLC: See table below.

The machine data contains two fields to deactivate the update of the input and output image of the PLC:

Machine data		Value range	
12986[i]	$MN_PLC_DEACT_IMAGE_LADDR_IN	0 ≤ i ≤ 7	Input addresses
12987[i]	$MN_PLC_DEACT_IMAGE_LADDR_OUT	0 ≤ i ≤ 7	Output addresses

The SINUMERIK 828D works with a fixed maximum configuration of the I/O modules. As delivered, the data transfer to the input and output image of the PLC is deactivated for all I/O modules.

Field with logical input addresses:

MD	Logical input address	Data transfer to the PLC deactivated
12986[0]	0	1. PP module inactive
12986[1]	9	2. PP module inactive
12986[2]	18	3. PP module inactive
12986[3]	27	4. PP module inactive
12986[4]	36	5. PP module inactive
12986[5]	96	PN bus coupler inactive
12986[6]	112	Machine control panel inactive

The field of output addresses is empty (default setting): MD12987[i] = -1

If an I/O module is to be activated, its address must not be entered in either MD12986[i] or in MD12987[i]. Instead, the value -1 ("empty") must be entered.

Example

Two PP modules and the machine control panel are activated:

MD	Logical input address	Data transfer to the PLC deactivated
12986[0]	**-1**	**1st PP module active**
12986[1]	**-1**	**2nd PP module active**
12986[2]	18	3. PP module inactive
12986[3]	27	4. PP module inactive
12986[4]	36	5. PP module inactive
12986[5]	96	PN bus coupler inactive
12986[6]	**-1**	**Machine control panel active**

Note

The use of an input/output address of a deactivated module in the PLC user program does not trigger an alarm. The PLC user program always works with the image memory. Whether there is a connection to the physical input/outputs is configured via MD12986[i] and MD12987[i].

Active modules are then monitored cyclically for failure.

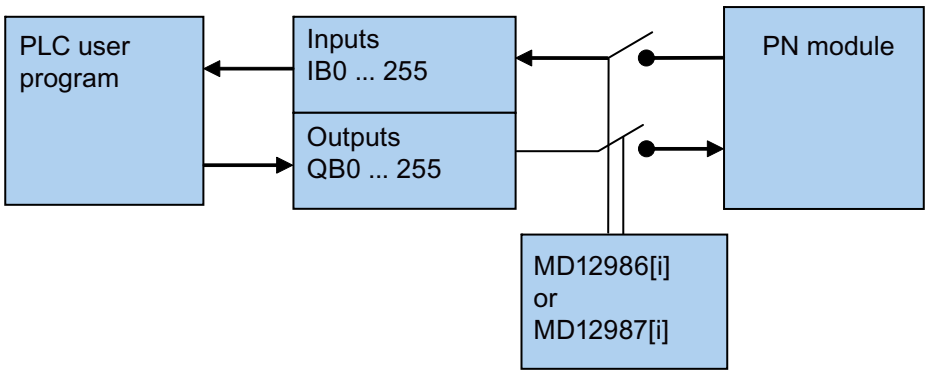


Figure 4-1 I/O switch

4.2 Addressing the I/O modules

IP addresses of the I/O modules

You can find the DIP switch S1 for the IP address of the appropriate I/O module in the following table. In this case, the maximum configuration with I/O modules, bus coupler and machine control panel via the PLC I/O interface based on PROFINET is taken into consideration.

Example: Setting for I/O module with device number "8"

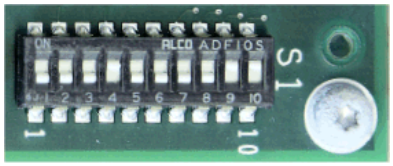


Figure 4-2 DIP Switch S1

I/O module	Bus	Device name	IP address	Input addresses	Output addresses
			192.168.214.	(active with MD12986[x] = -1)	
				Index n:	
1st Digital PP module	PN	pp72x48pn9	9	0 … 8	0 … 5
2nd Digital PP module	PN	pp72x48pn8	8	9 … 17	6 … 11
3rd Digital PP module	PN	pp72x48pn7	7	18 … 26	12 … 17
4th PP module digital	PN	pp72x48pn6	6	27… 35	18 … 23
5th Digital PP module	PN	pp72x48pn5	5	36 … 44	24 … 29
Unassigned				**45**	**30 … 55**
				Index d:	
1st PP module diagnostics	PN	pp72x48pn9	9	46 … 47	--
2nd PP module diagnostics	PN	pp72x48pn8	8	48 … 49	--
3rd PP module diagnostics	PN	pp72x48pn7	7	50 … 51	--
4th PP module diagnostics	PN	pp72x48pn6	6	52 … 53	--
5th PP module diagnostics	PN	pp72x48pn5	5	54 … 55	--
				Index m:	
1st Analog PP module	PN	pp72x48pn9	9	56 … 63	56 … 63
2nd Analog PP module	PN	pp72x48pn8	8	64 … 71	64 … 71
3rd Analog PP module	PN	pp72x48pn7	7	72 … 79	72 … 79
4th Analog PP module	PN	pp72x48pn6	6	80 ... 87	80 ... 87
5th Analog PP module	PN	pp72x48pn5	5	88 … 95	88 … 95
PN bus coupler *	PN	pn-pn-coupler20	20	96 … 111	96 … 111
External machine control panel	PN	mcp-pn64	64	112 … 125	112 … 121
Reserved				126 … 131	122 ... 123

The Index n, m, d is always the start address of the address range.

*) The IP address of the bus coupler is not set using a switch, but is appropriately configured.

Commissioning the drive 5

5.1 Configuring the drive

5.1.1 Example: Configuration with SINAMICS S120 Combi

Overview

The SINAMICS S120 commissioning software is available on the Toolbox CD, free of charge. Until the SINAMICS S120 commissioning functionality is completely available in the user interface, commissioning the drive in two steps is described here.

- The topology is automatically identified and provided using the functionality in the user interface.
- The encoder is configured and parameterized using the commissioning software for SINAMICS S120. The PG/PC is connected via the Ethernet interface on the front of the SINUMERIK 828D.

Sequence

The sequence is divided into the following steps:

- Step 1: Configure the spindle.
- Step 2: Configure the axes.
- Step 3: Assign the axes and backup the data:

  Procedure as described in Chapter "Example: How to assign the axes (Page 115)".

These steps are described in more detail in the following chapters.

Configuring the drive

To configure the drive, the example with SINAMICS S120 Combi and a maximum configuration with 6 axes is selected from Chapter "System overview (Page 13)". The DRIVE-CLiQ connections must correspond to the topology rules in Chapter "Topology rules for S120 Combi (Page 124)".

The sequence of the DRIVE-CLiQ connections corresponds to the sequence of the SINAMICS drive object numbers (= default setting):

Axes	SINAMICS drive object	
	Number	Name
---	1	Control Unit
---	2	Line Module
MSP1	3	SERVO 1
MX1	4	SERVO 2
MY1	5	SERVO 3
MZ1	6	SERVO 4
MA1	7	SERVO 5
MC1	8	SERVO 6
---	11	HUB module
---	9	TM54F master

Initial state

Before you begin:

- Connect the PG/PC to the control: see Chapter "Communication with the control (Page 20)"
- The control is powered-up with "Siemens default data".
- Display in "Start-up" operating area on the control:

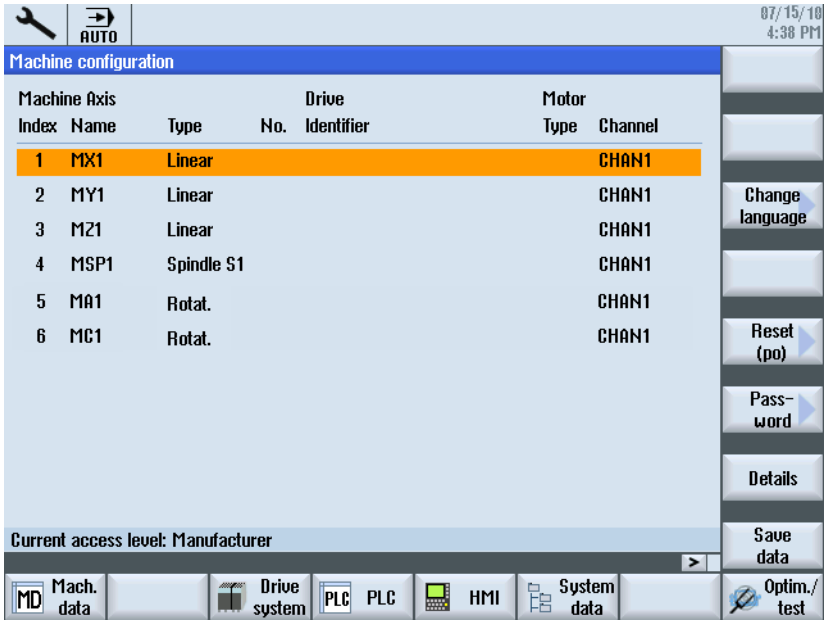


Figure 5-1 Control after powering-up with "Siemens default data"

Procedure

To configure the drive:

1. Start commissioning using the "Drive system" softkey.
2. Confirm the following question with "OK", in order to start the device configuration.

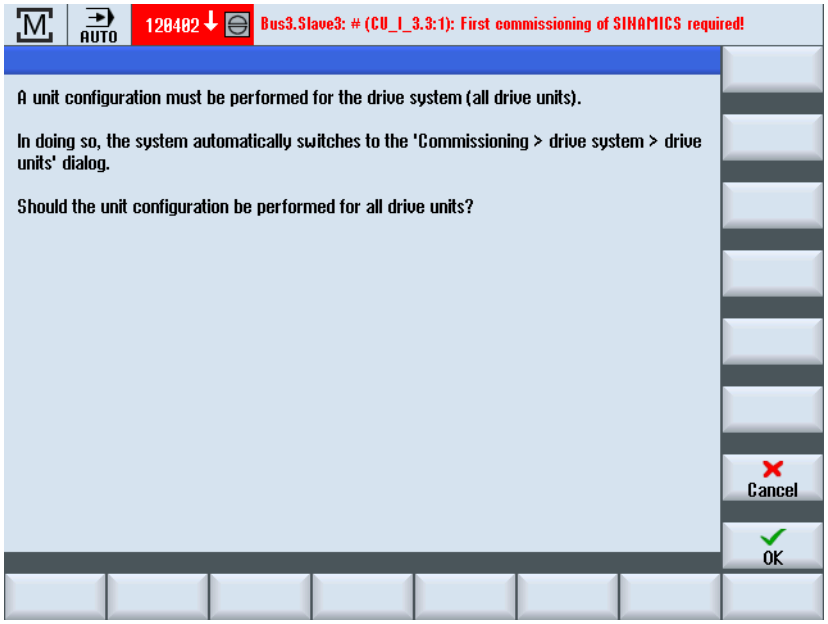


Figure 5-2 Start commissioning with SINUMERIK Operate

The system topology is then automatically read-out.

3. Confirm the following question with "OK", in order to configure the device.

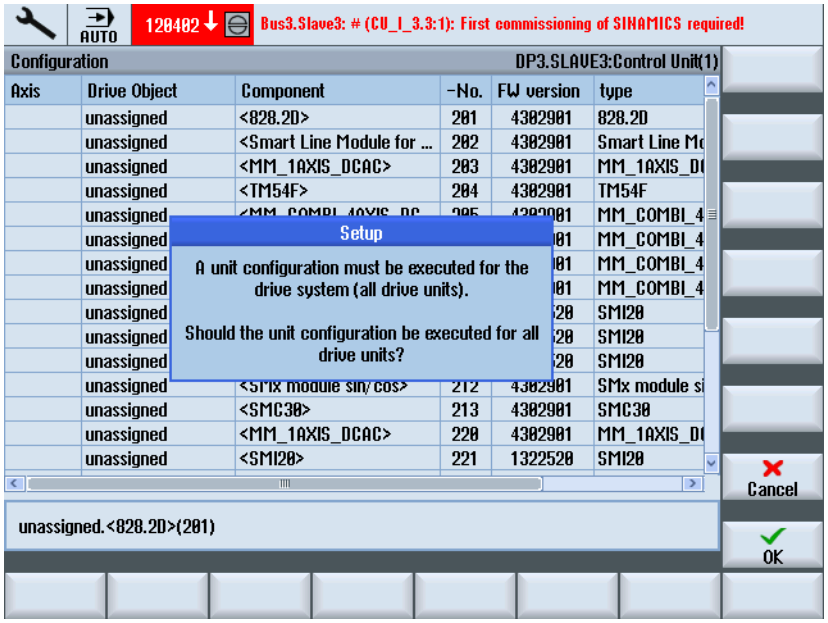


Figure 5-3 Device configuration

4. Confirm the following question with "OK", in order to perform an NCK Power On Reset. This can take several minutes.

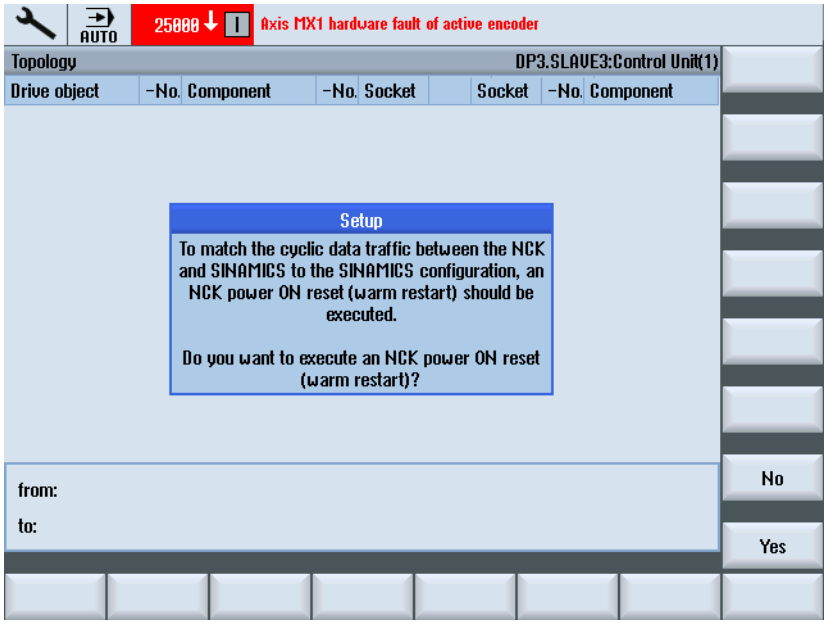


Figure 5-4 Confirming the Power On Reset

5. After the system has been powered-up, you receive the following message:
6. Confirm with "OK".

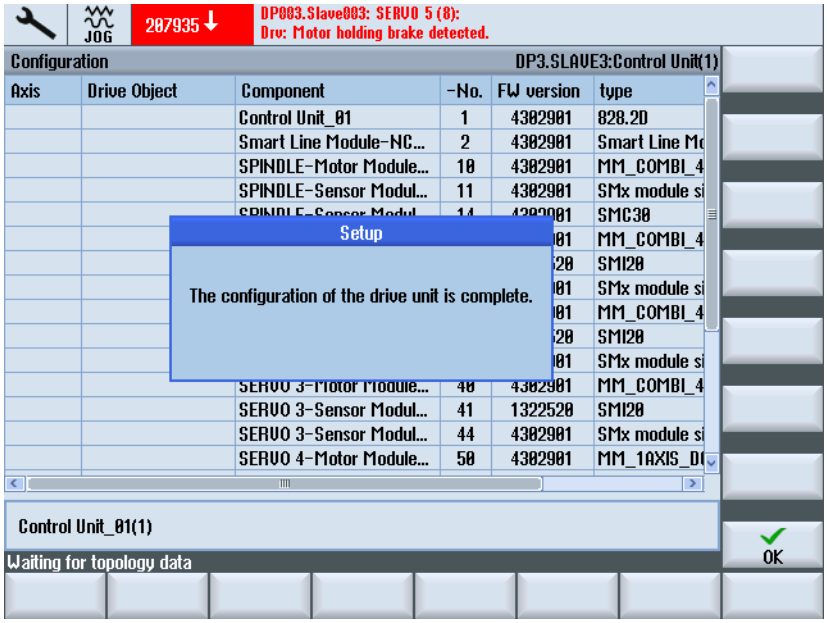


Figure 5-5 Completing the automatic configuration

7. The configuration has been completed once the topology data have been determined.

   The commissioning software for SINAMICS S120 (Page 15) is required on PG/PC for the next commissioning steps.

8. Select the "Start-up" operating area.

9. Set the password to "Manufacturer" access level.
10. Confirm the following question with "OK", in order to start the additional configuration.

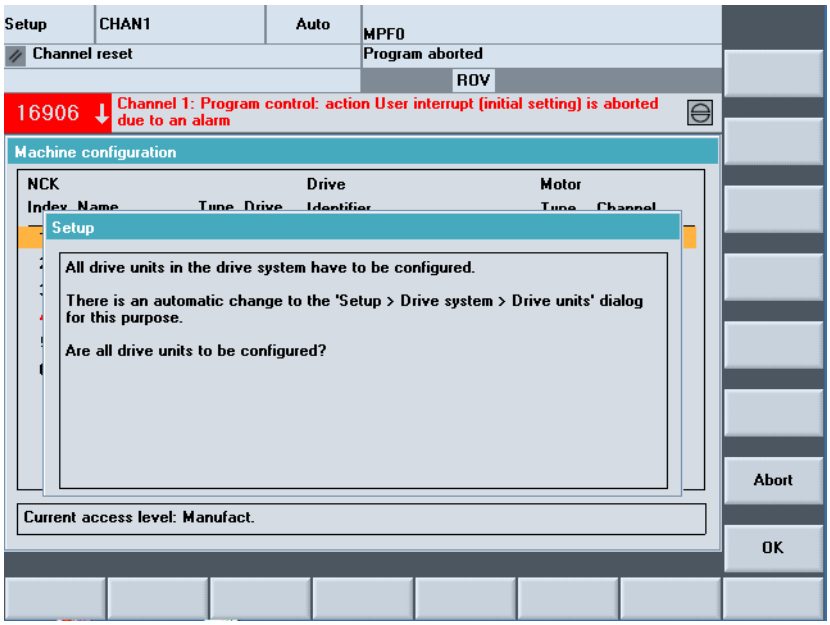


Figure 5-6 Start commissioning

5.1.2 Example: This is how you configure a spindle with SMC encoder

Spindle configuration with SMC20

Procedure:

1. First, select the spindle using the "Drive +" or "Drive -" softkey.
2. Select "Change" to configure the spindle.

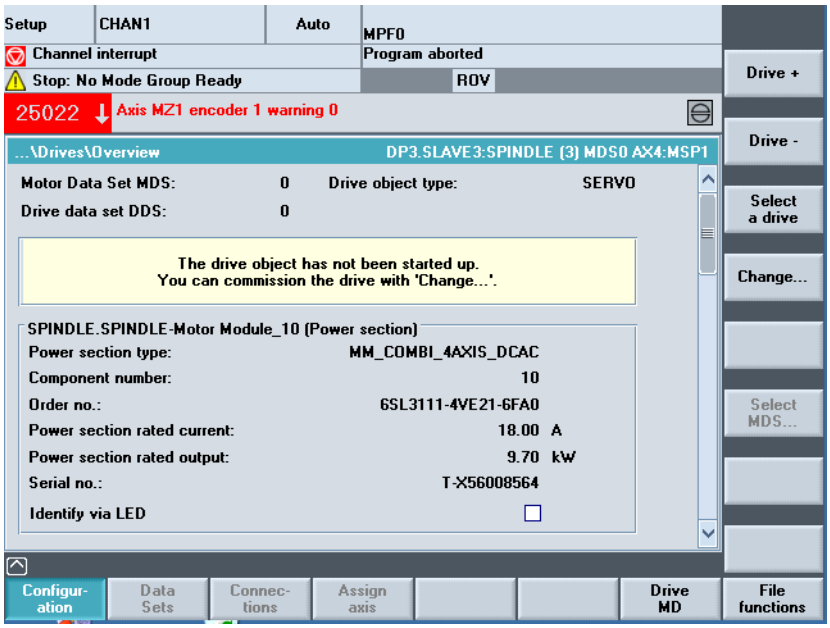


Figure 5-7 Drive – Overview

The following spindle data is displayed:

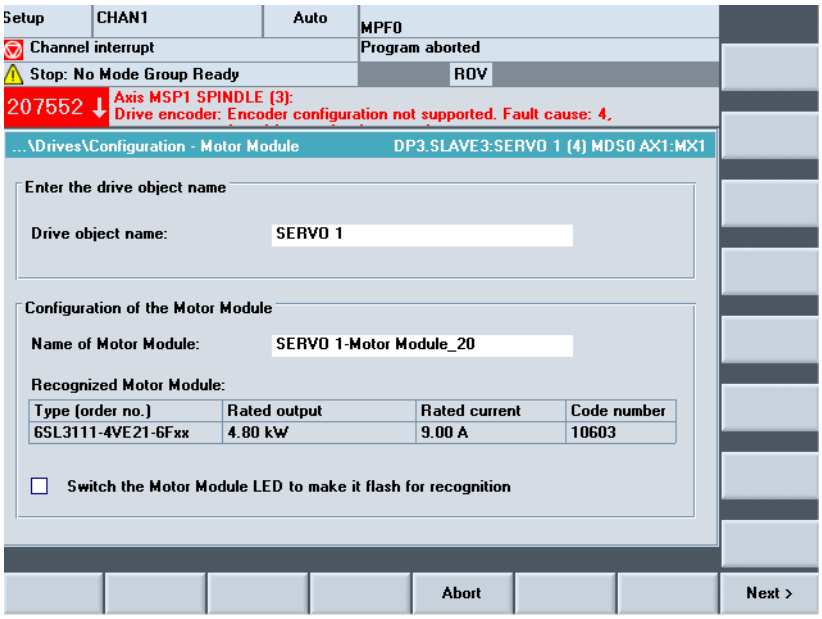


Figure 5-8 Starting the configuration: Spindle

3. Select "Continue >" to assign the associated motor: The motor can either be selected using the "Motor type" or using "Search" to find the code number.

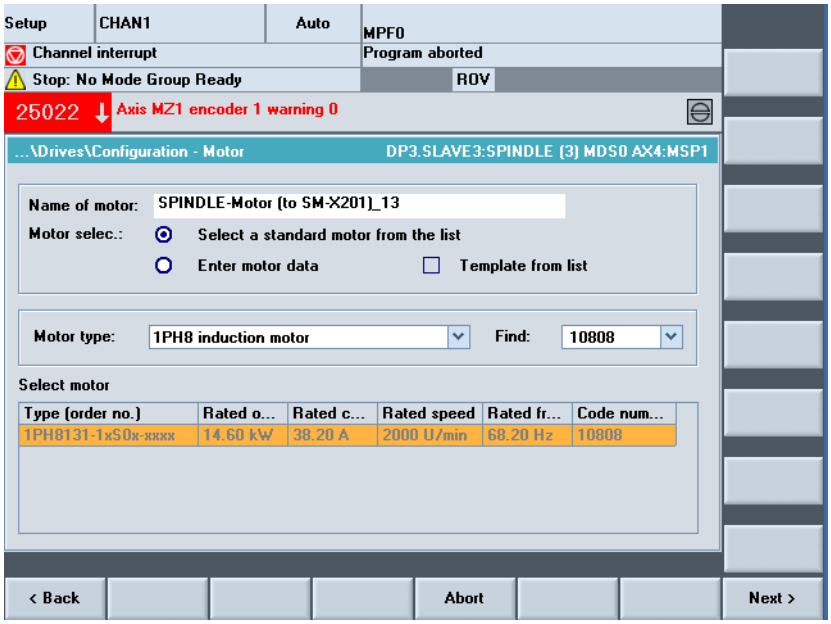


Figure 5-9 Selecting a motor

4. Select "Continue >" to configure the motor holding brake:

   Please observe the note regarding the bearing design.

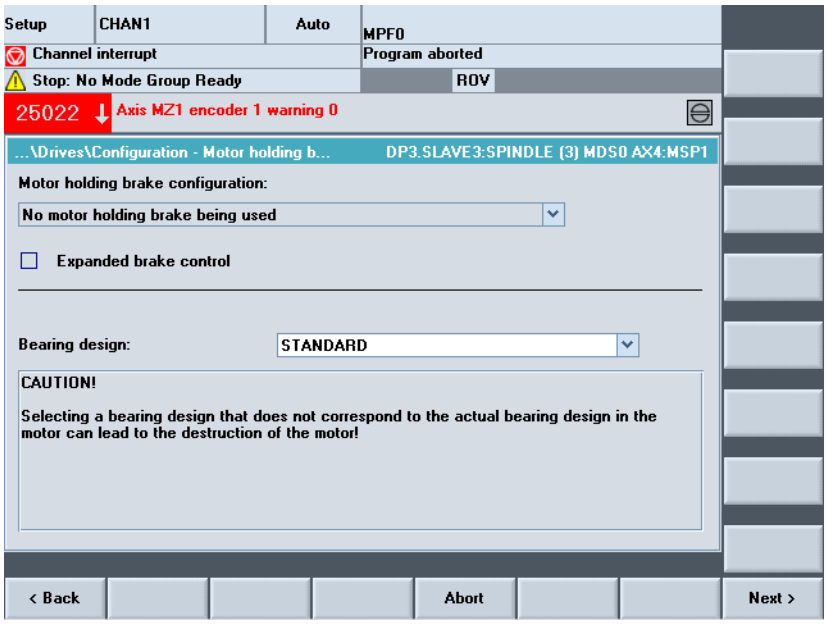


Figure 5-10 Motor holding brake

5. Select "Continue >" to assign the encoder: The two encoders (at the motor measuring system and at the direct measuring system) were already correctly assigned when automatically determining the topology.

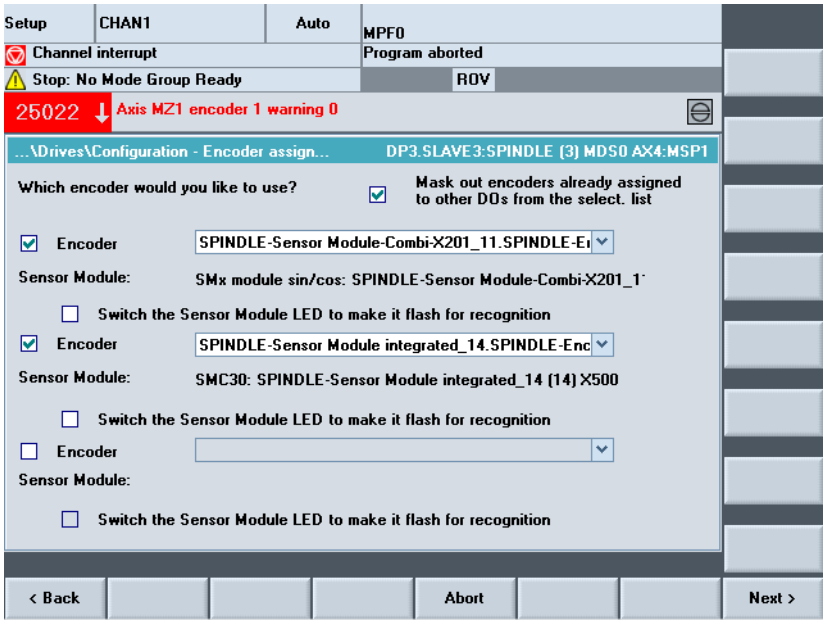


Figure 5-11 Encoder assignment

6. Select "Continue >" to configure the motor encoder.

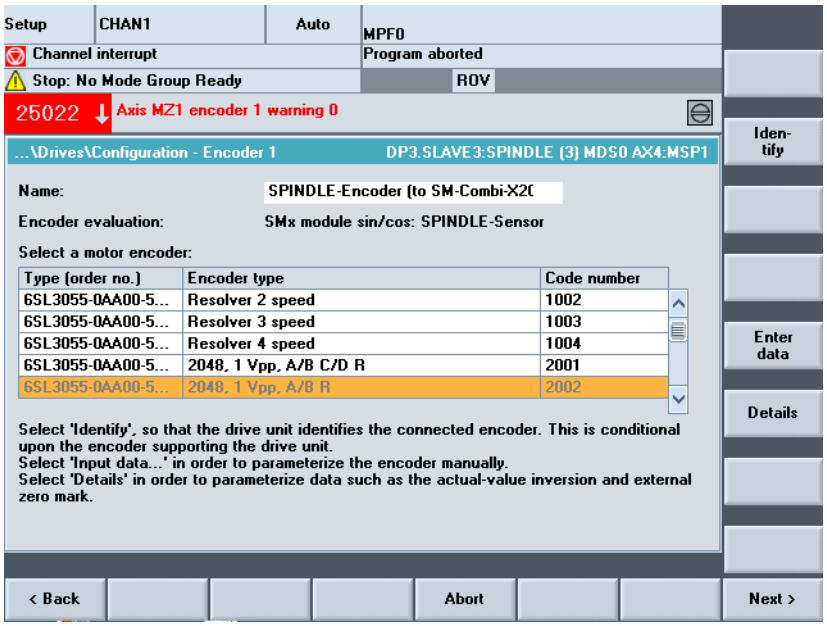


Figure 5-12 Motor encoder 1

7. Select "Input data" to adapt the encoder properties to the value of 4096 revolutions.

8. Confirm this setting with "OK", the encoder is displayed as "user-defined":

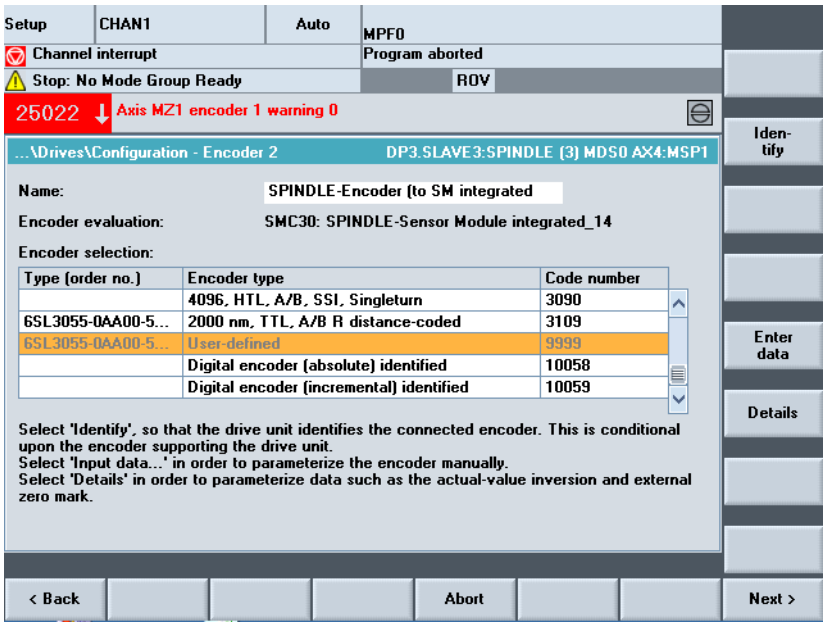


Figure 5-13 Motor encoder: User-defined

9. Select "Continue >" to accept the default setting with telegram 116. This means that for the internal drive variables, additional process data (PZD) is also transferred and saved in system variables.

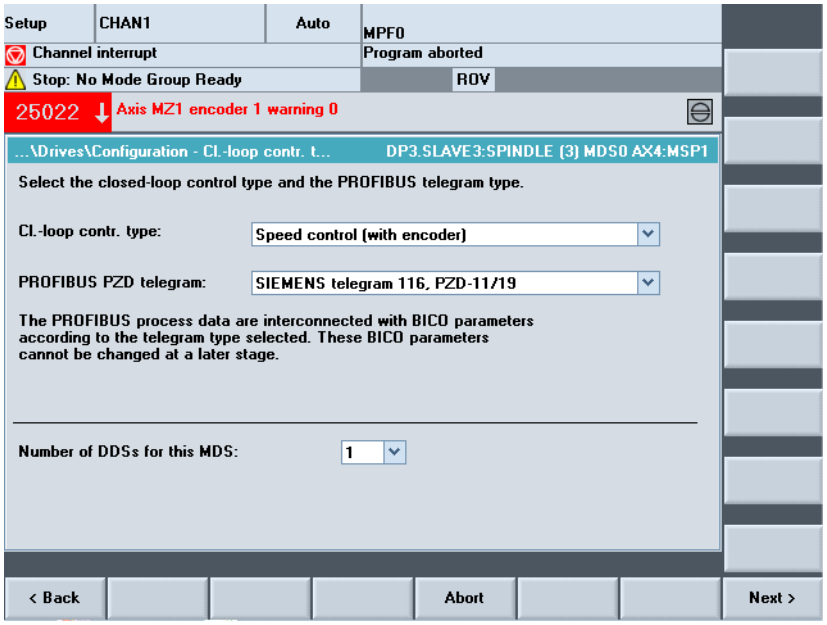


Figure 5-14 Control with speed feed forward control

- OR -

10. Select the setting with telegram 136, in order to transfer additional process data (PZD) for the internal drive variables and save in system variables.

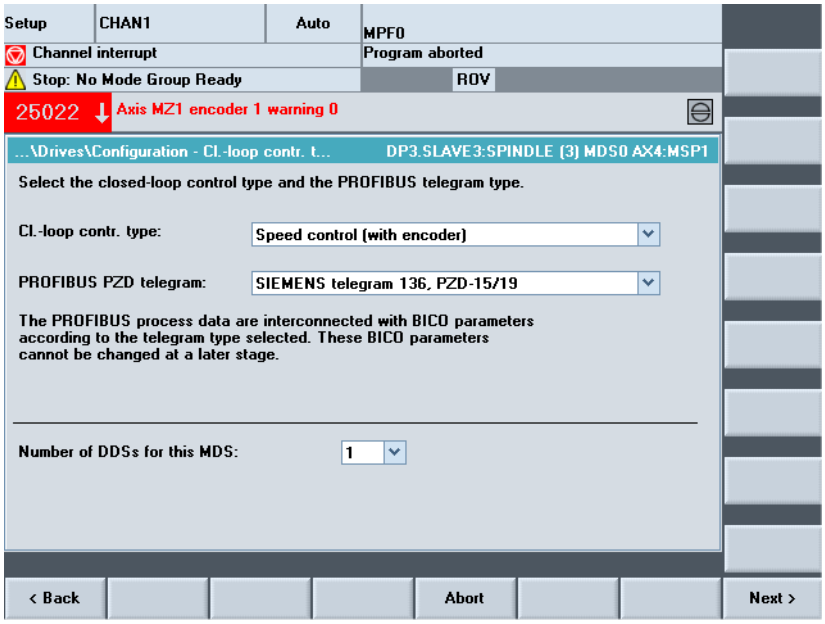


Figure 5-15 Control with torque feed forward control

11. Select "Continue >" to save the selection. This operation can take several minutes.
12. Select "Continue >" to accept the BICO configuration.

    The following summary is then displayed in order to check all of the spindle data:

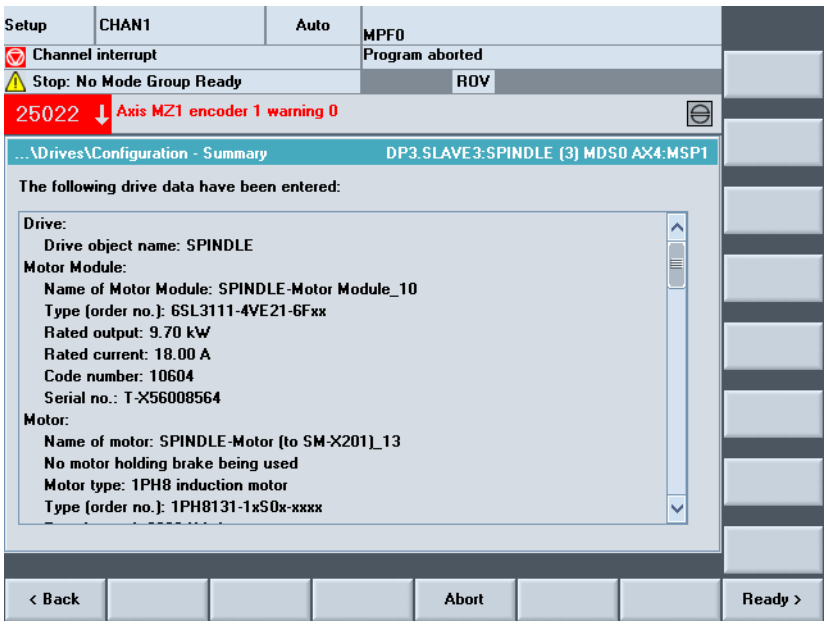


Figure 5-16 Summary, spindle

13. Select "Ready >" to complete the spindle configuration.

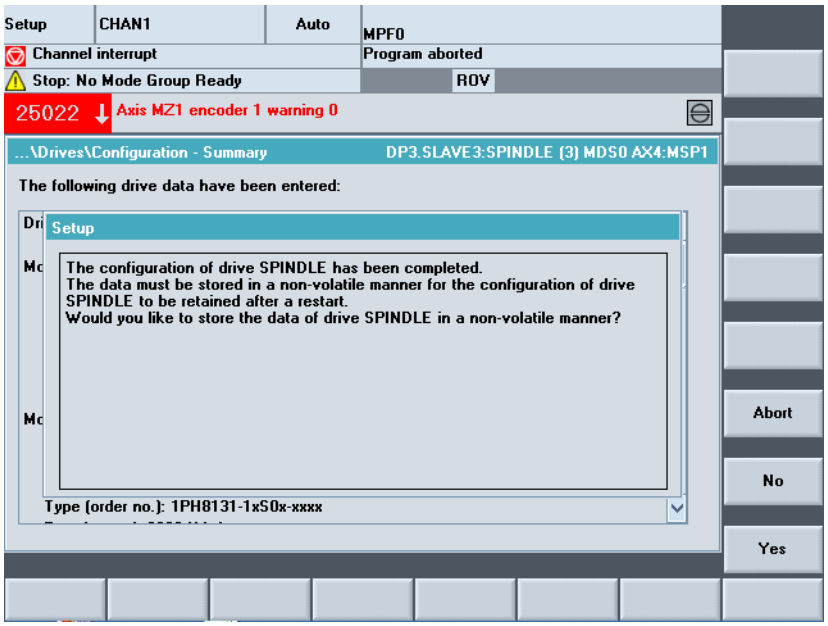


Figure 5-17 Saving a configuration

14. Select "Yes" to save the configuration.

Result

This completes configuration of the spindle.

5.1.3 Example: This is how you configure an axis with SMI encoder

Configuring the axis with SMI encoder

Procedure:

1. First, select the axis using the "Drive +" or "Drive -" softkey.
2. Select "Change" to configure the axis.

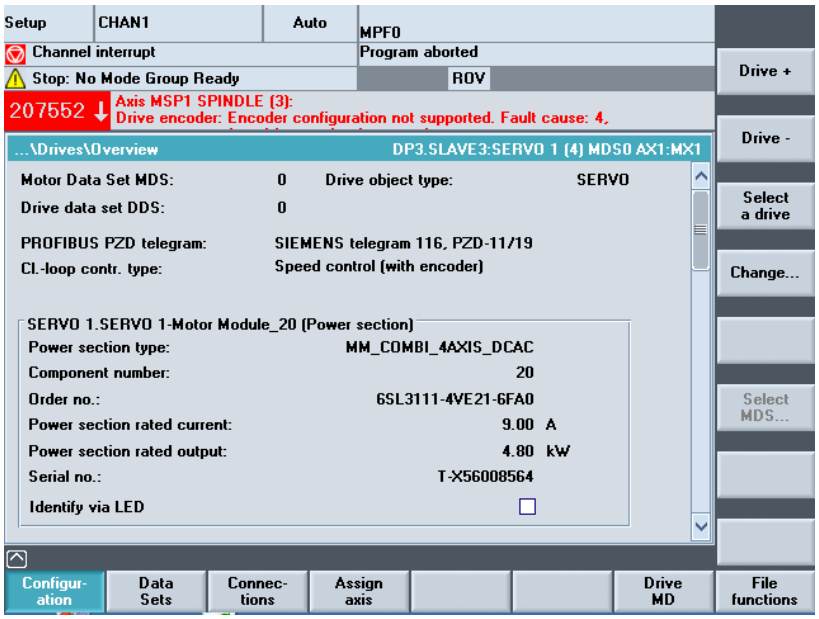


Figure 5-18 Drive – Overview

The following axis data is displayed:

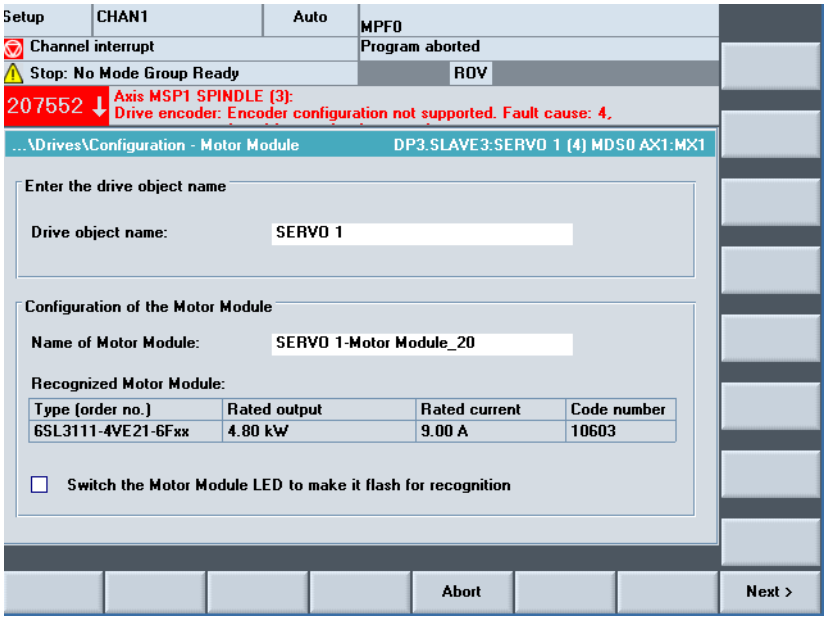


Figure 5-19 Starting the configuration: Axis

3. Select "Continue >" to configure the associated motor: The motor type was already correctly identified when automatically determining the topology.

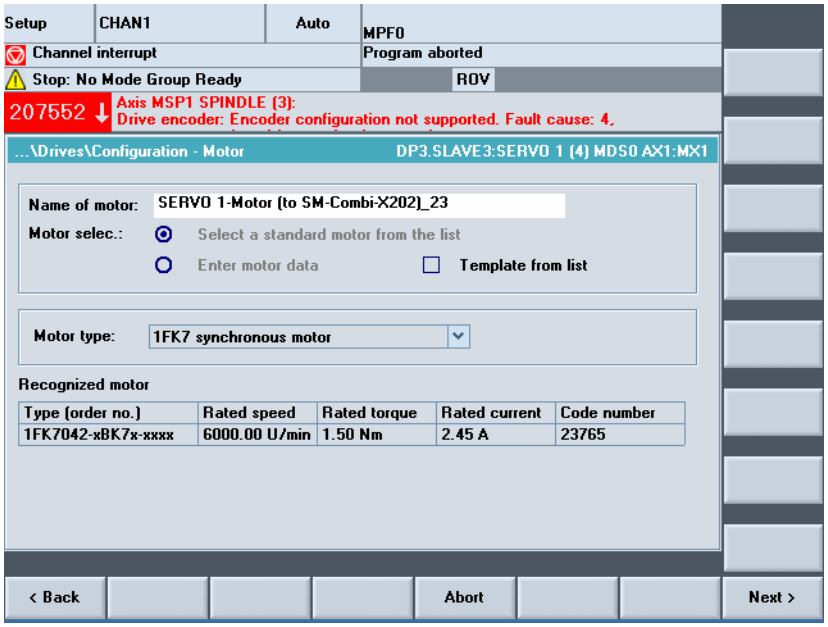


Figure 5-20 Selecting a motor

4. Select "Continue >" to configure the motor holding brake.

   Please observe the note regarding overvoltage protection. The following can be selected:

   – No measure
   – VPM (Voltage Protection Mode)

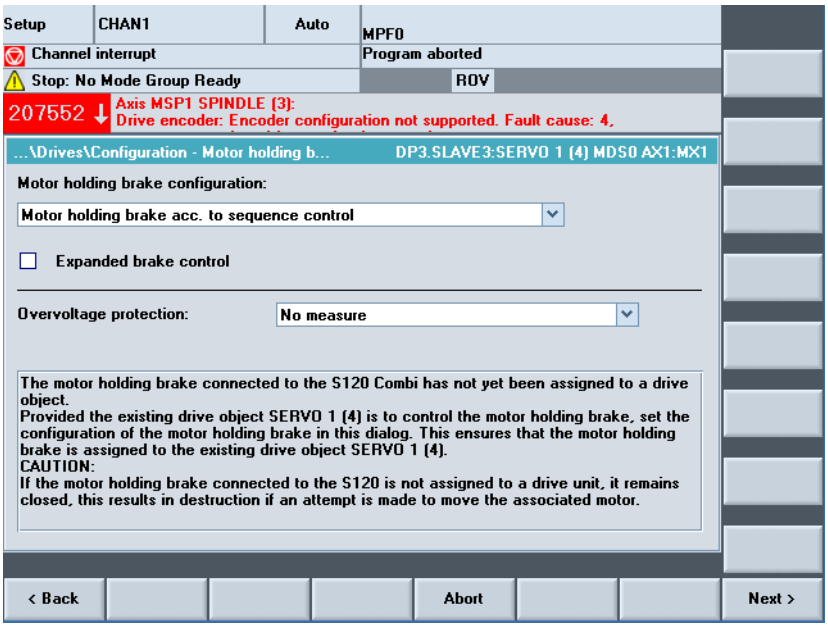


Figure 5-21 Motor holding brake

5. Select "Continue >" to assign the encoder: The two encoders (at the motor measuring system and at the direct measuring system) were already correctly assigned when automatically determining the topology.

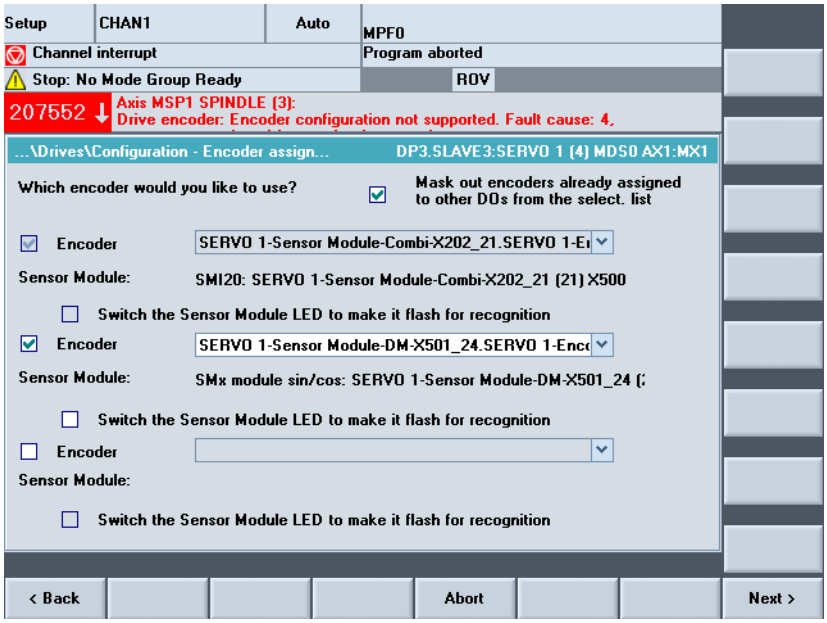


Figure 5-22 Encoder assignment

6. Select "Continue >" to configure the motor encoder.

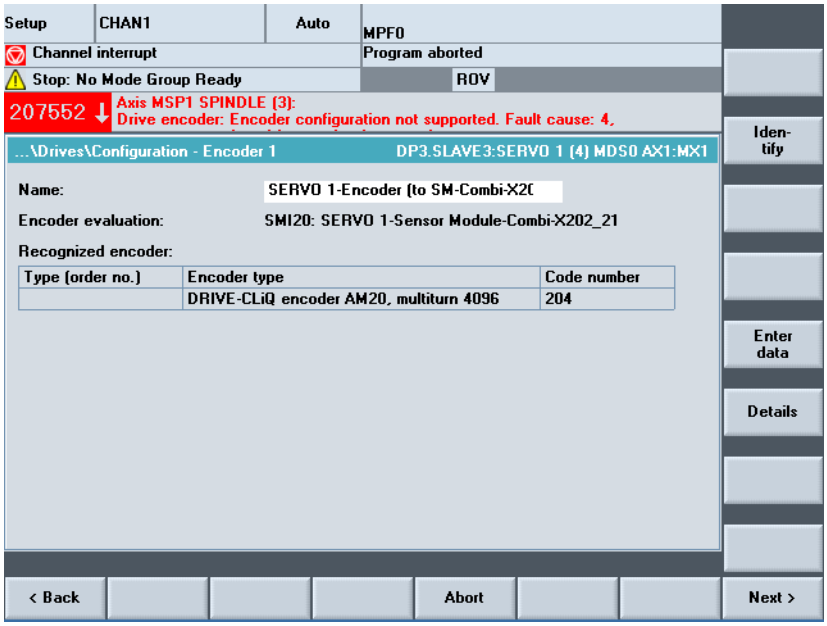


Figure 5-23 Motor encoder

7. Select "Input data" to adapt the encoder properties to "Invert position actual value".

8. Confirm this setting with "OK", the encoder is displayed as "user-defined":

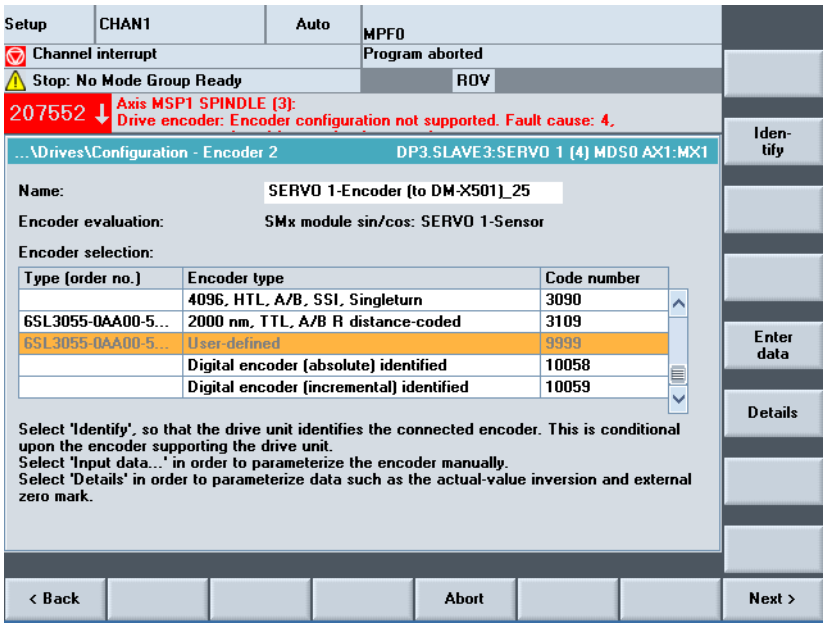


Figure 5-24 Motor encoder: User-defined

9. Select "Continue >" to accept the default setting with telegram 116. This means that for the internal drive variables, additional process data (PZD) is also transferred and saved in system variables.

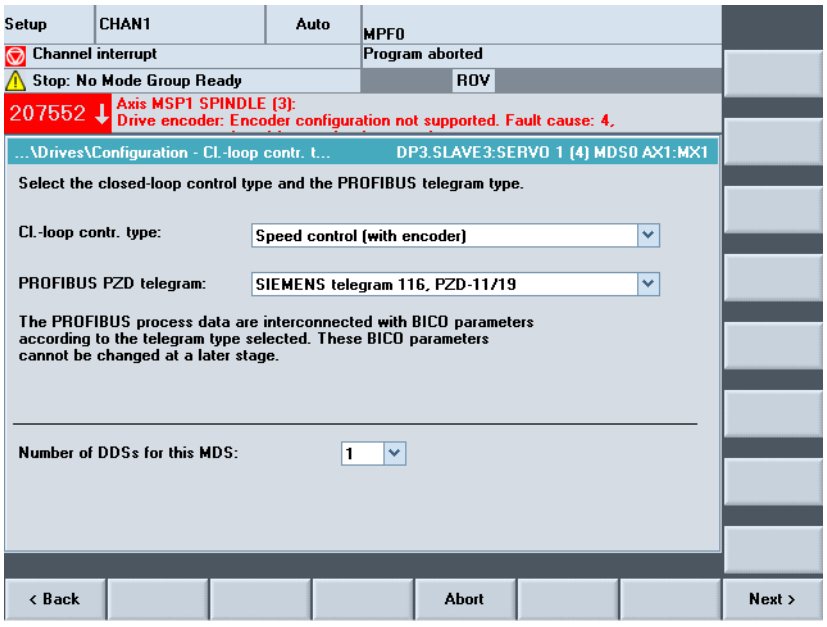


Figure 5-25 Control with speed feed forward control

- OR -

10. Select the setting with telegram 136, in order to transfer additional process data (PZD) for the internal drive variables and save in system variables.

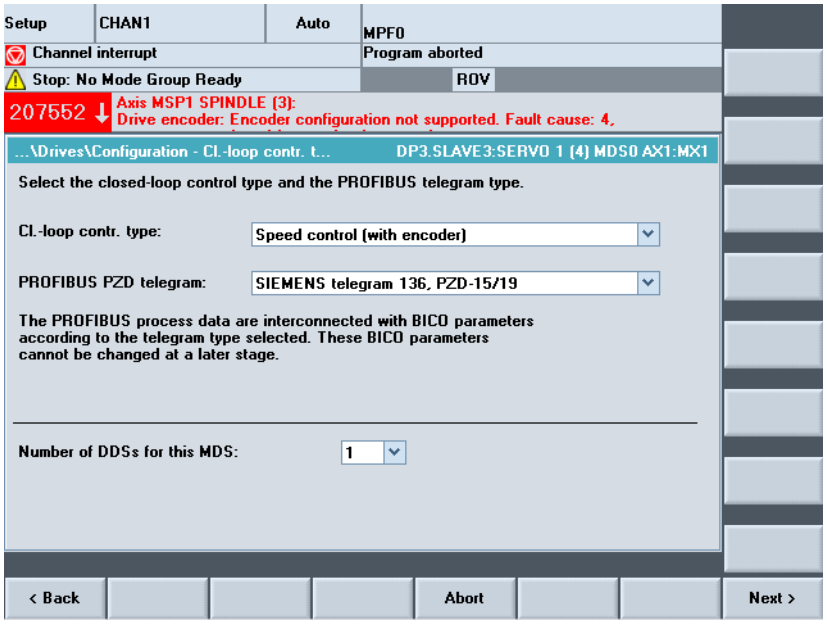


Figure 5-26 Control with torque feed forward control

11. Select "Continue >" to accept the BICO configuration.

The following summary is then displayed in order to check all of the axis data:

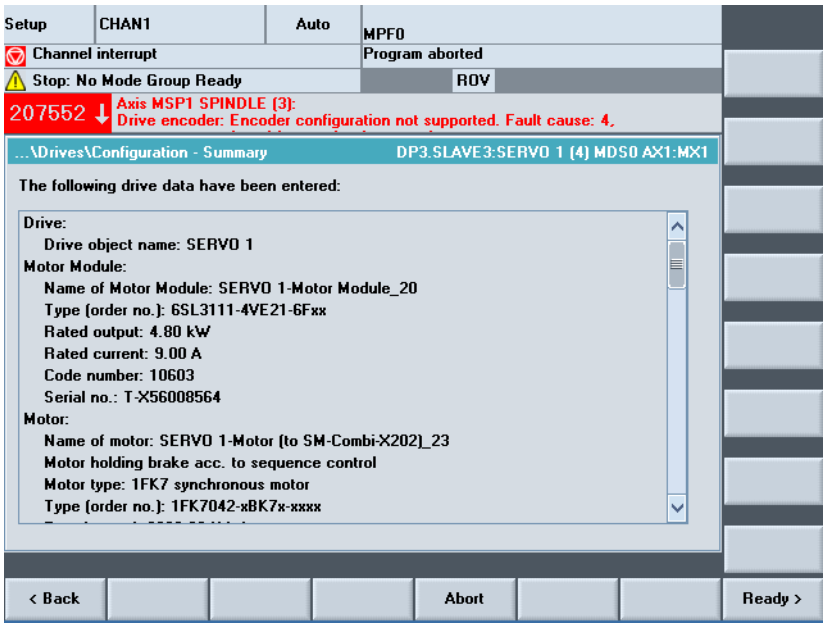


Figure 5-27 Summary, axis

12. Select "Ready >" to complete the axis configuration.

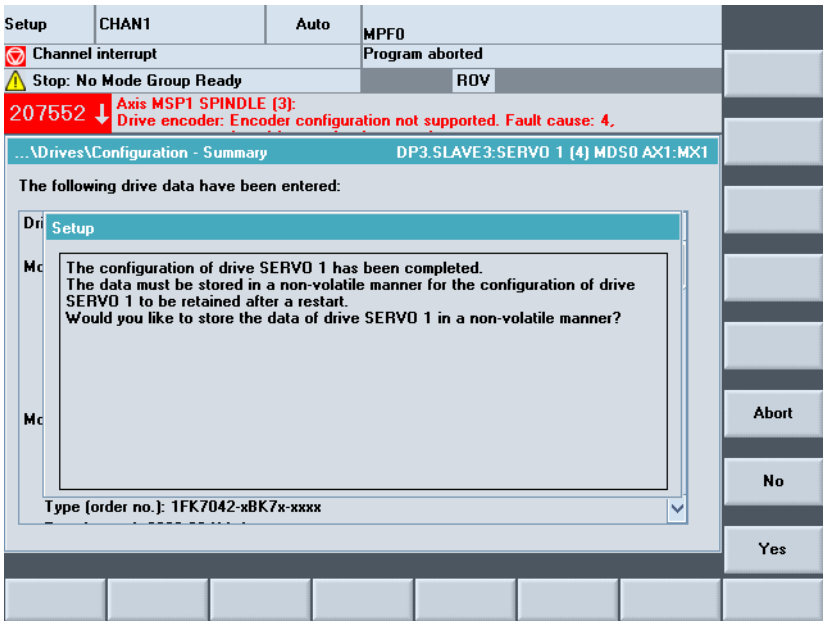


Figure 5-28 Saving a configuration

13. Select "Yes" to save the configuration. This operation can take several minutes.

Result

This completes configuration of the axis.

5.1.4 Example: Configuration with SINAMICS S120 Booksize

Overview

The SINAMICS S120 commissioning software is available on the Toolbox CD, free of charge.

Until the SINAMICS S120 commissioning functionality is completely available via the user interface, drive commissioning is performed using the commissioning software for SINAMICS S120. The PG/PC is connected via the Ethernet interface on the front of the SINUMERIK 828D.

Configuring the drive

To configure the drive, the example with 4 axes and SINAMICS S120 Booksize is selected from Chapter "System overview (Page 13)". The DRIVE-CLiQ connections are connected as in the following diagram:

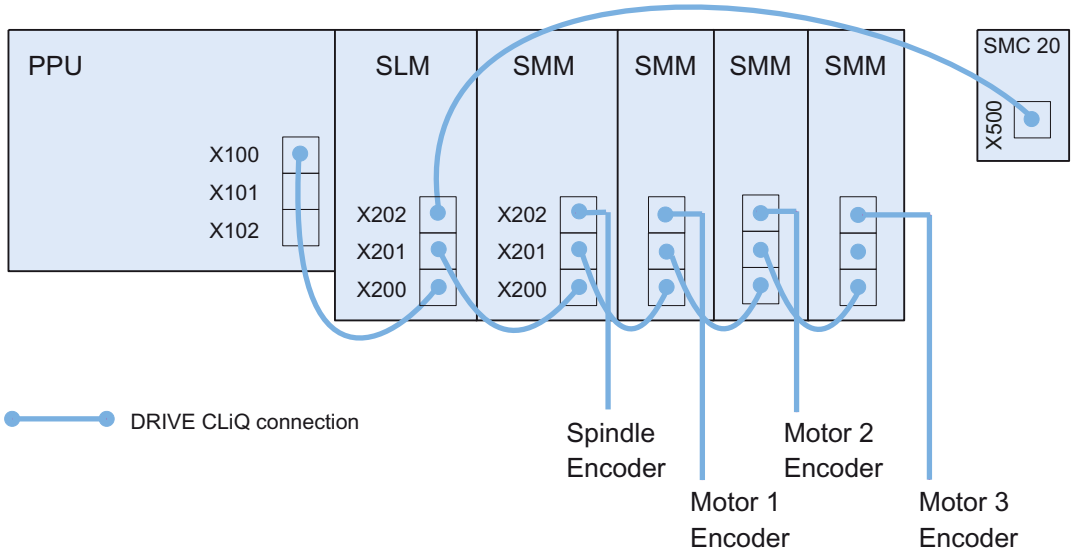


Figure 5-29 DRIVE-CLiQ connections

The following applies to the configuration of 4 axes:

The sequence of the DRIVE-CLiQ connections corresponds to the sequence of the SINAMICS drive object numbers (= default). Adjustments are only necessary if this setting does not fit with the sequence in the drive system.

Index	Axes	SINAMICS drive object	
		Number	Name
		1	CU_I_3.3:1
		2	SLM_3.3:2
4	MSP1	3	SERVO_3.3:3
1	MX1	4	SERVO_3.3:4
2	MY1	5	SERVO_3.3:5
3	MZ1	6	SERVO_3.3:6

Sequence

The sequence is divided into the following steps:

- Create connection to control.
- Step 1: Configure the drive.
- Step 2: Configure the infeed.
- Step 3: Assign the encoders.
- Step 4: Assign the axes.
- Finally: Save the data.

These steps are described in more detail in the following chapters.

See also

Topology rules for S120 Booksize (Page 126)

5.1.5 Example: How to configure the drive

Initial state

Before you begin:

- Connect the PG/PC to the control: See chapterExample: How to communicate with the control using the NCU Connection Wizard (Page 24)
- The control is powered-up with "Siemens default data".
- Display in "Start-up" operating area on the control:

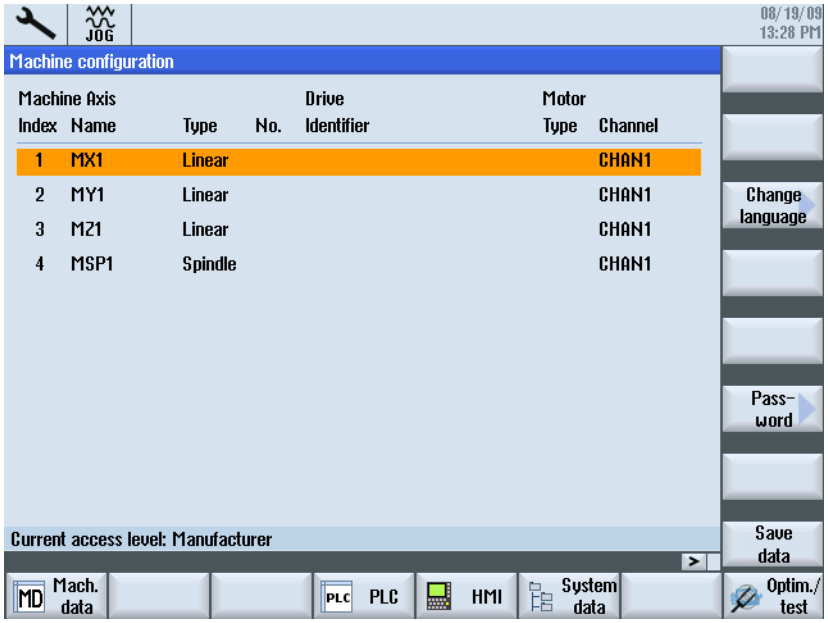


Figure 5-30 Control after power up

Procedure

To configure the drive:

1. Start the commissioning software for SINAMICS S120 on PG/PC:



2. Select the "Start-up" operating area.
3. Set the password to "Manufacturer" access level.

   Then you obtain the following display:

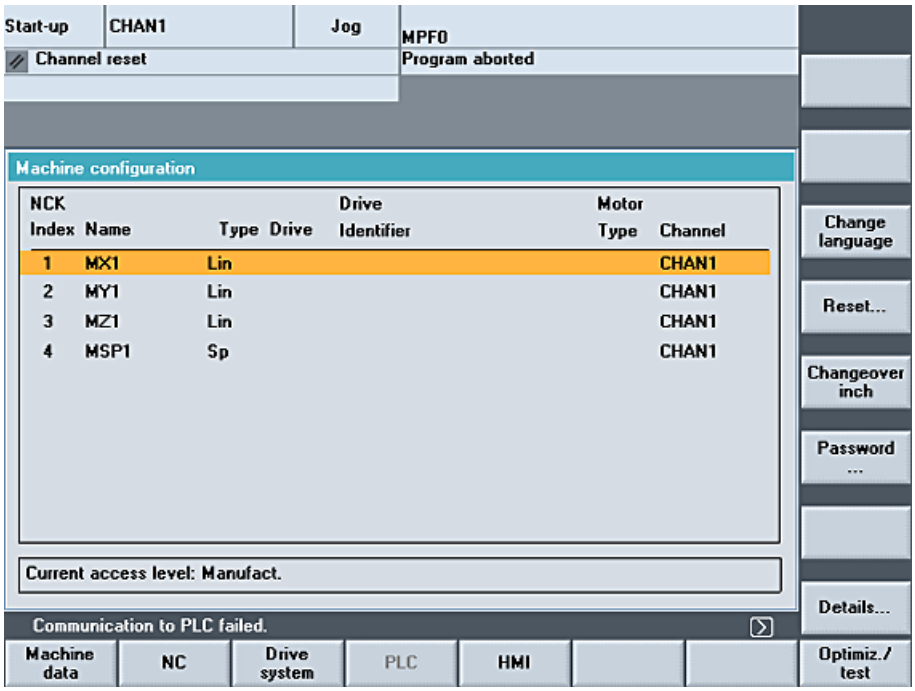


Figure 5-31 Start commissioning

NOTICE
Subsequently inserting SINAMICS components The firmware of the configured SINAMICS components is only correctly updated if the components were inserted in the switched-off state. Subsequently inserting SINAMICS components must **only** be performed in the switched-off state.

Note

Updating firmware

When a drive is connected, the update of the firmware is started automatically.

Alternatively:

The firmware update can also be started via the following softkeys:

4. Start commissioning with the "Drive system" softkey (horizontal bar); the following dialog appears:

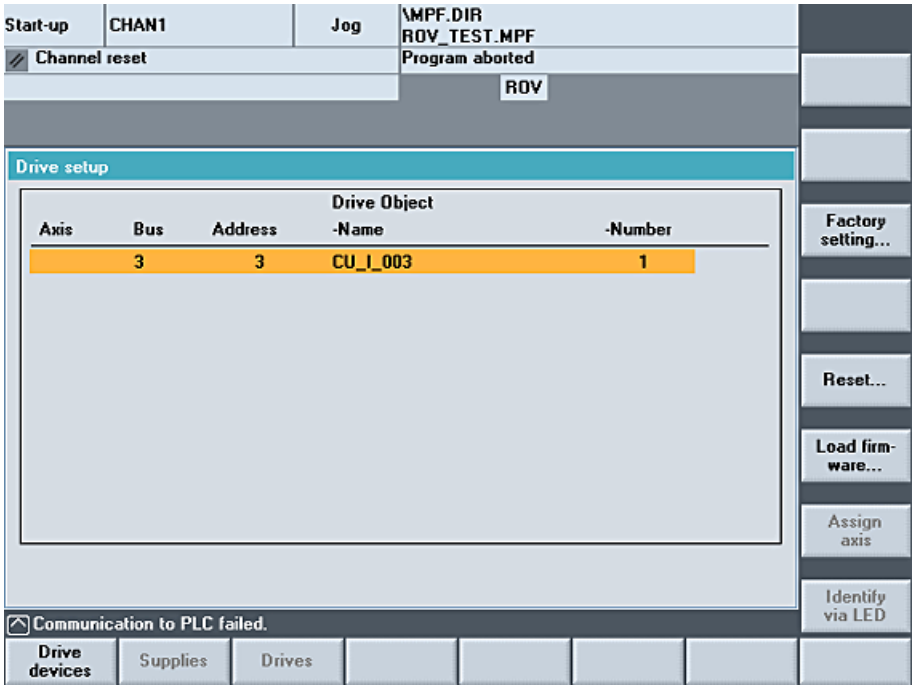


Figure 5-32 Drive commissioning

5. To ensure the drive's firmware matches the software level on the control, press the "Load firmware..." softkey (vertical bar): The current firmware on the control's CompactFlash card is loaded to the drive.

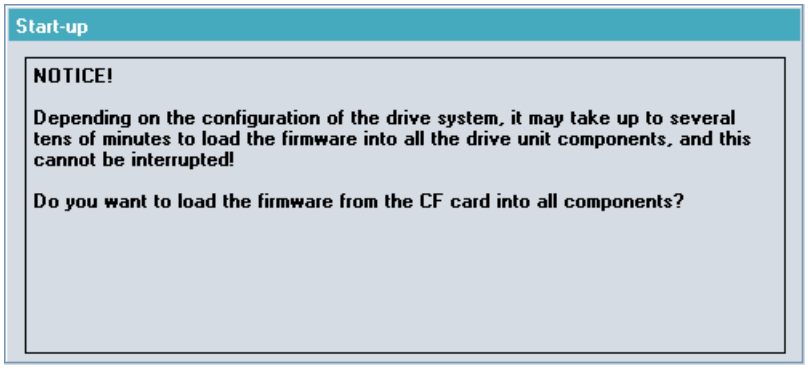


Figure 5-33 Message on time needed

6. Confirm this message with "Yes".
   - During loading, you will be informed of progress with a status display.
   - While the firmware is loading, the "RDY" LED flashes alternately red - green on the appropriate module. When the module is finished, this LED will show a steady green light. The "DC LINK" LED will show a steady orange light.
7. When the firmware has finished loading successfully, confirm this message with "OK":

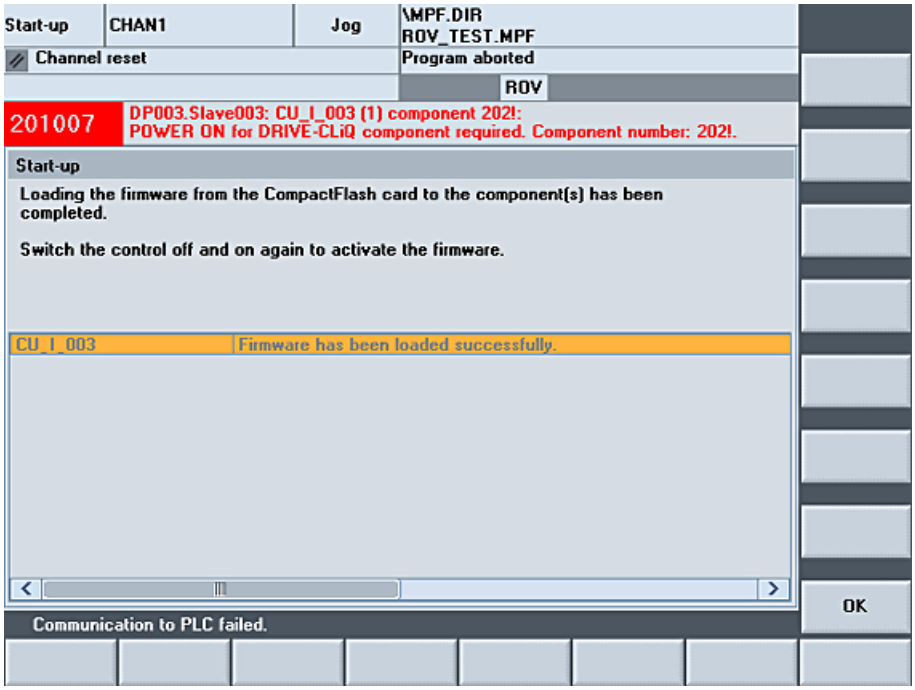


Figure 5-34 Firmware loaded

Note

The following selection depends on the configuration of the target system:

- Without NX expansion module: "Drive unit" softkey
- With NX expansion module: "Drive system" softkey

8. This example is a configuration without NX module. In this example, select the "Drive unit" softkey (vertical bar).

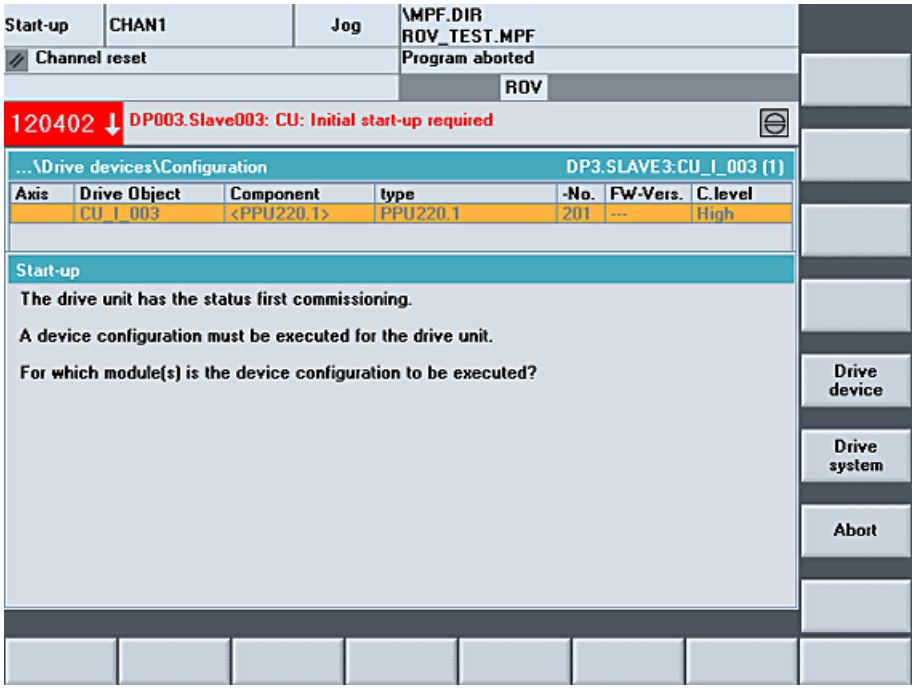


Figure 5-35 Drive unit: Commissioning

9. Then you receive the message that the device configuration process may last several minutes.

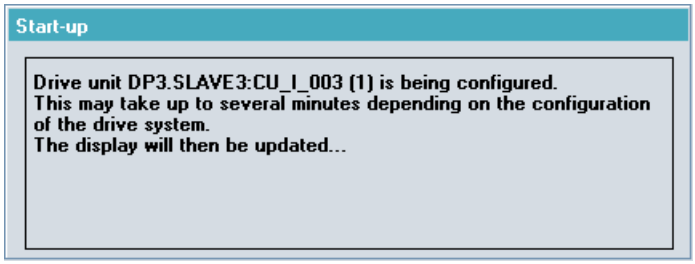


Figure 5-36 Drive unit: Configuration

10. Then you obtain the following display:

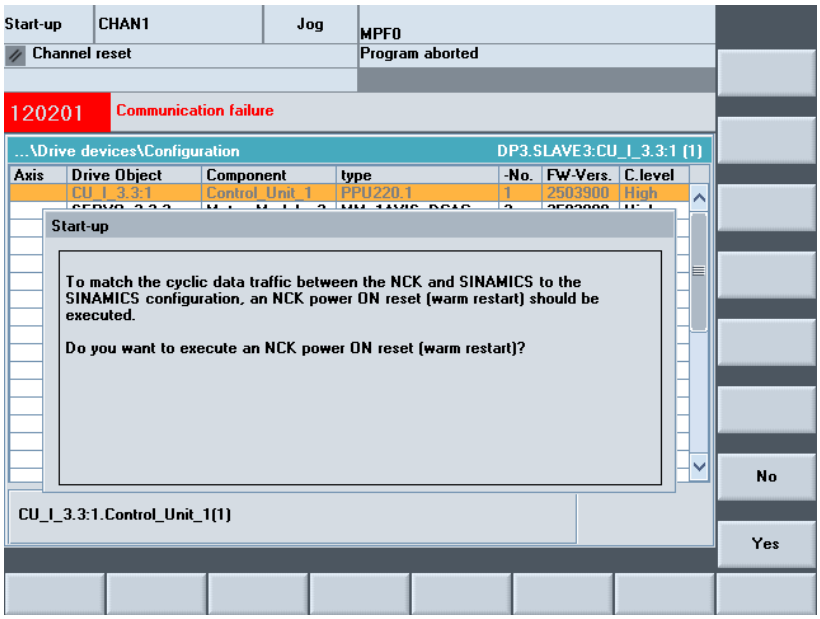


Figure 5-37 NCK Power On Reset

11. Confirm with "Yes". The system carries out a warm restart. This operation can take several minutes.

    After the warm restart you receive the following selection:

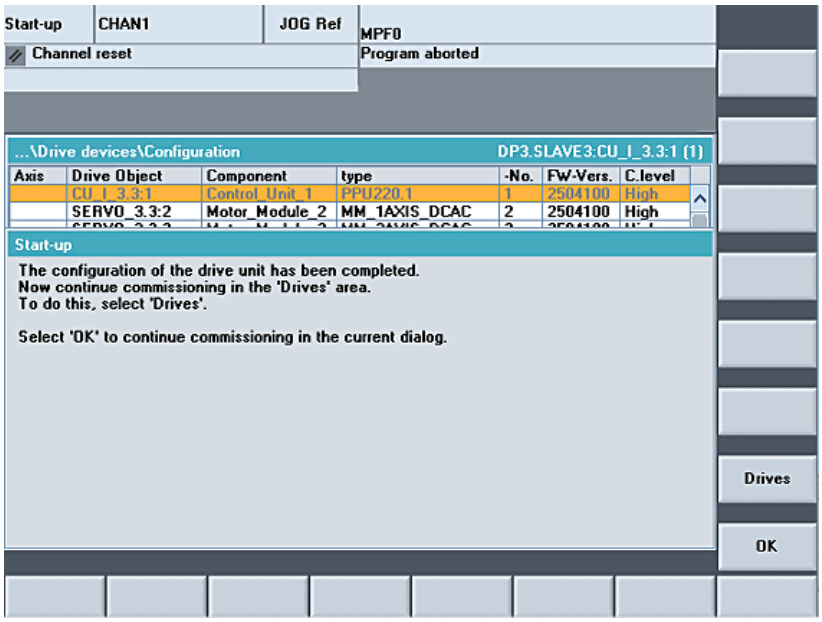


Figure 5-38 Continue configuration

12. To continue commissioning, press the "Drive" softkey.
13. Carry out a POWER ON for the drive – as requested in the message: Switch off completely and then on again. The following will be displayed:

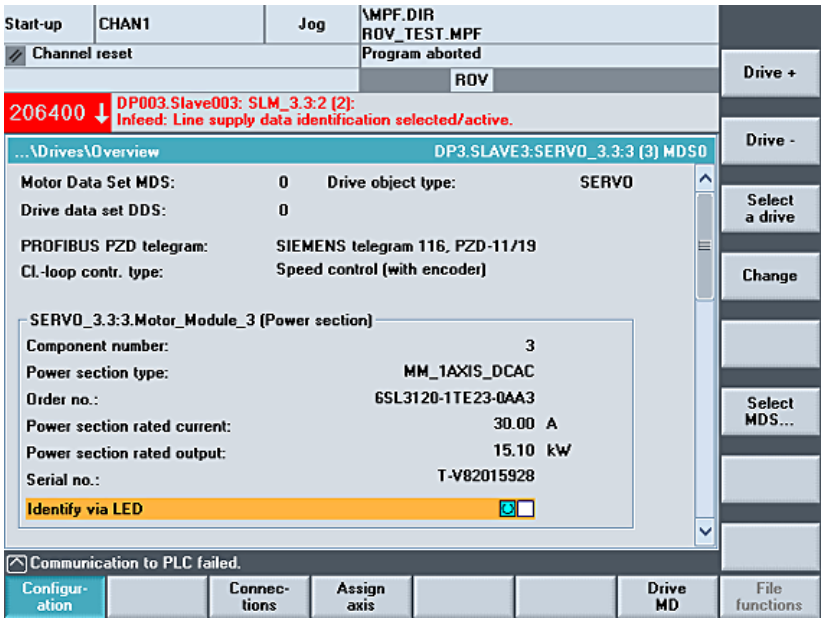


Figure 5-39 Drive data overview

14. Option: To identify the individual Motor Modules, select "Identify via LED": The "RDY" LED flashes alternately red - orange.

    Select the next module with the vertical softkeys "Drive +" and "Drive -".

Result

Step 1 for commissioning of the drives is now complete.

Note

Alternative procedure

If, in the "Configuration" dialog (Fig. Continue configuration), you accidentally press "OK", continue the configuration after POWER ON with the following selection:

On the horizontal softkey bar select: "Drive system" → "Drive devices" → "Configuration".

5.1.6 Example: How to configure the infeed

Configuring the infeed

Note

If there are alarms that require acknowledgment after switching on, these must be acknowledged first. Then commissioning can be continued.

Procedure:

1. In the "Overview" dialog, select the vertical softkey "Change", then the following query appears:

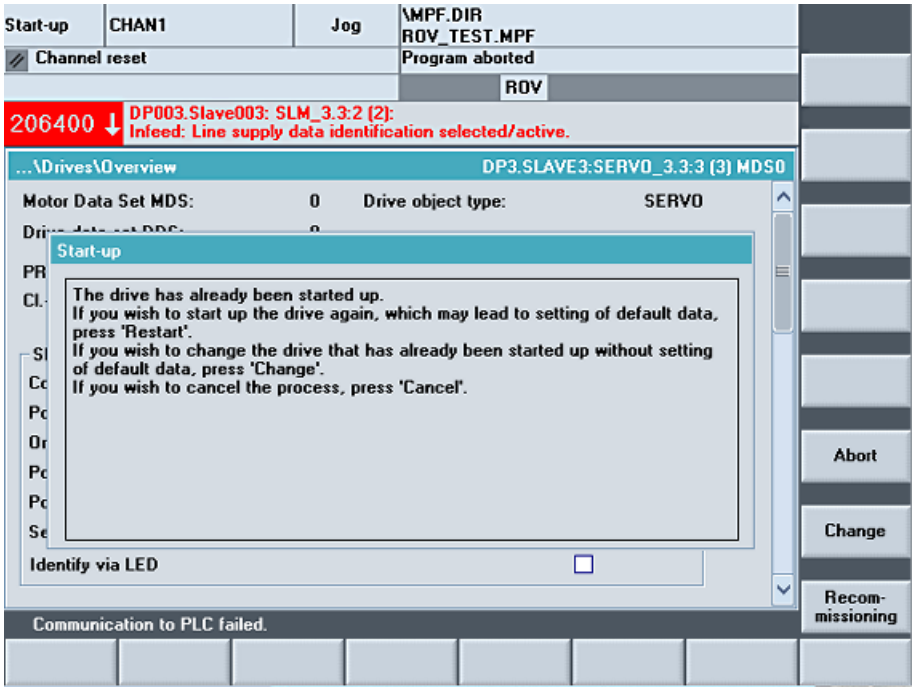


Figure 5-40 Query for further action

2. Here, select the vertical softkey "Change" to continue configuration.

3. Then press the horizontal softkey "Infeed" for the following display:

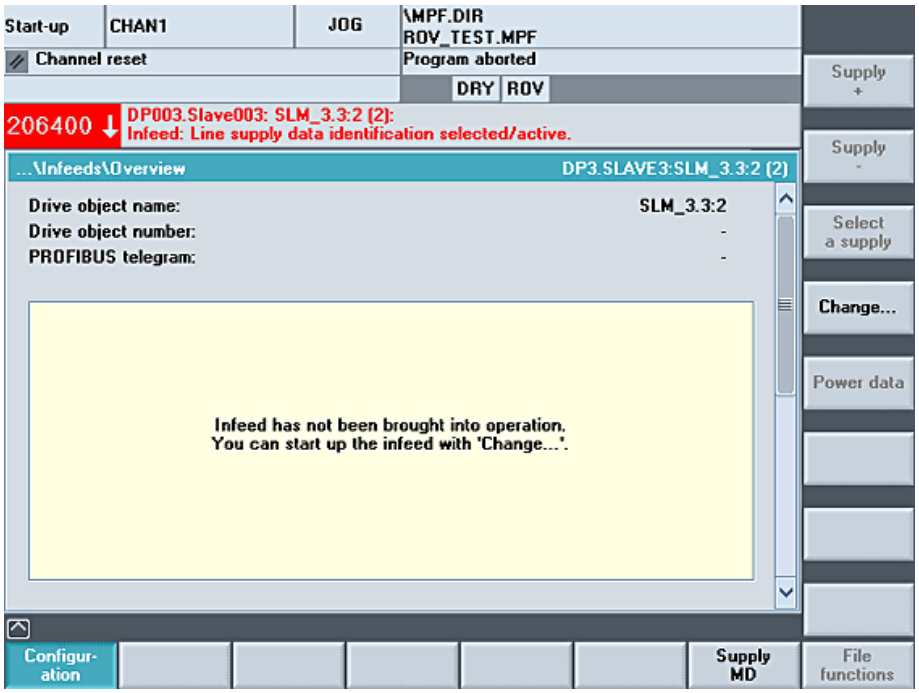


Figure 5-41 Commissioning infeed

4. To configure the infeed, select the vertical softkey "Change".
5. In the following dialog "Configuration - Active Line Module" press "Continue >".
6. In the following dialog "Configuration - Further data" press "Continue >".
7. In the following dialog "Configuration - Terminal wiring" press "Continue >".

8. In the following dialog "Configuration - Summary" press "Finish".

   The infeed has been configured.

9. Confirm the query if you want to save the data with "Yes".

   This process may take several minutes.

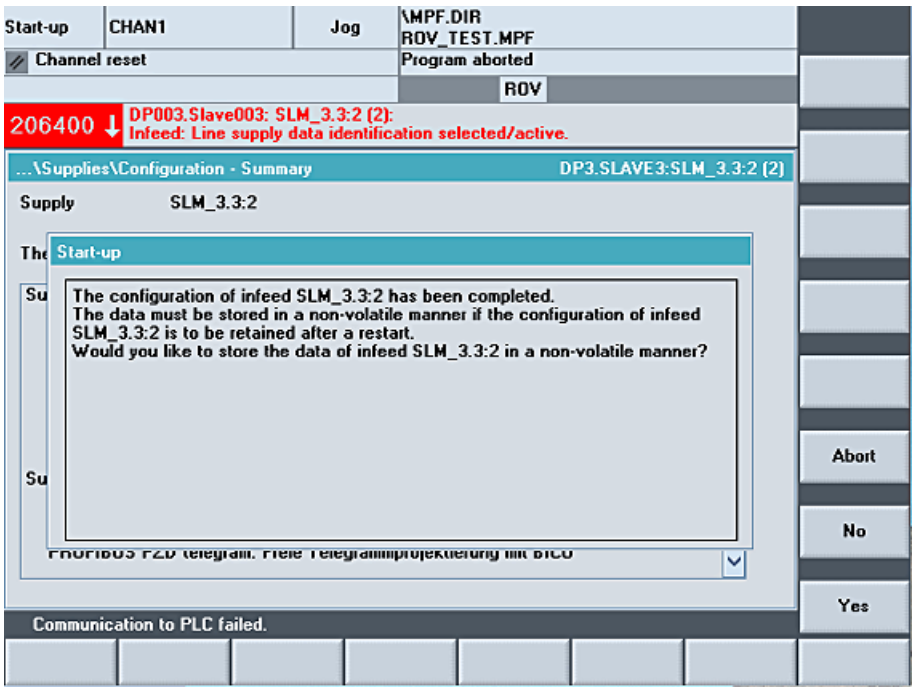


Figure 5-42 Infeed: Non-volatile saving of the data

Result

Step 2 for commissioning of the drives is now complete.

5.1.7 Example: How to configure the external encoder

Connecting a direct measuring system

In addition, a direct measuring system is connected for the spindle (see chapter Example: Configuration with SINAMICS S120 Booksize (Page 99)). The following section describes the configuration.

Procedure

To change the configuration of the drive:

1. You pressed the vertical softkey "Change".

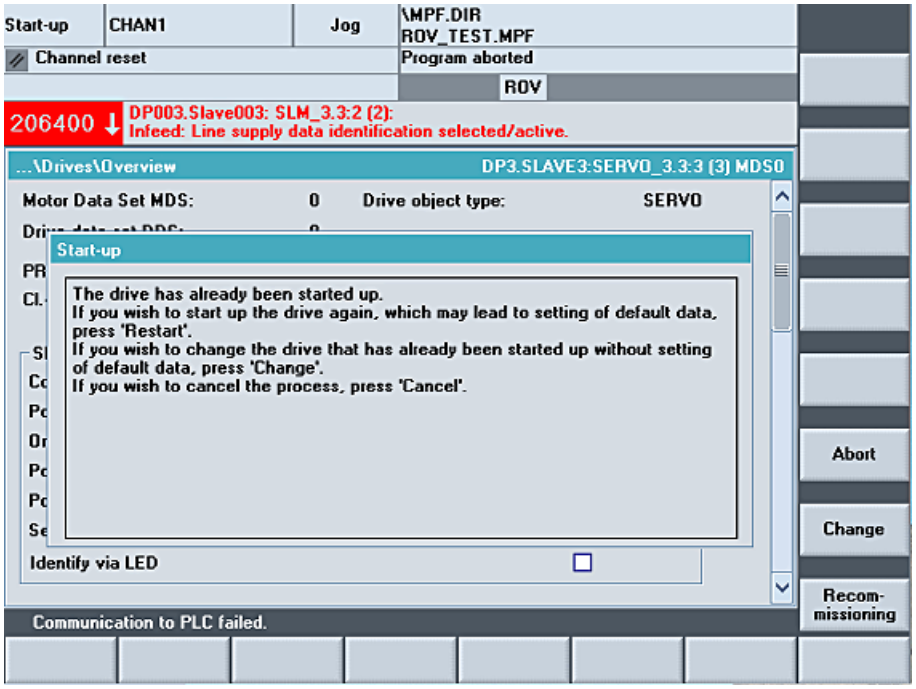


Figure 5-43 Change drive

2. Confirm the query with the "Change" softkey.

   The dialog "Configuration - Motor" is opened and the drive object 3 = Spindle is selected. Press "Next >".

3. The next dialog tells you which motor is assigned to the drive.

   Press "Next >".

4. The next dialog tells you about the motor assignment. Press "Next >".

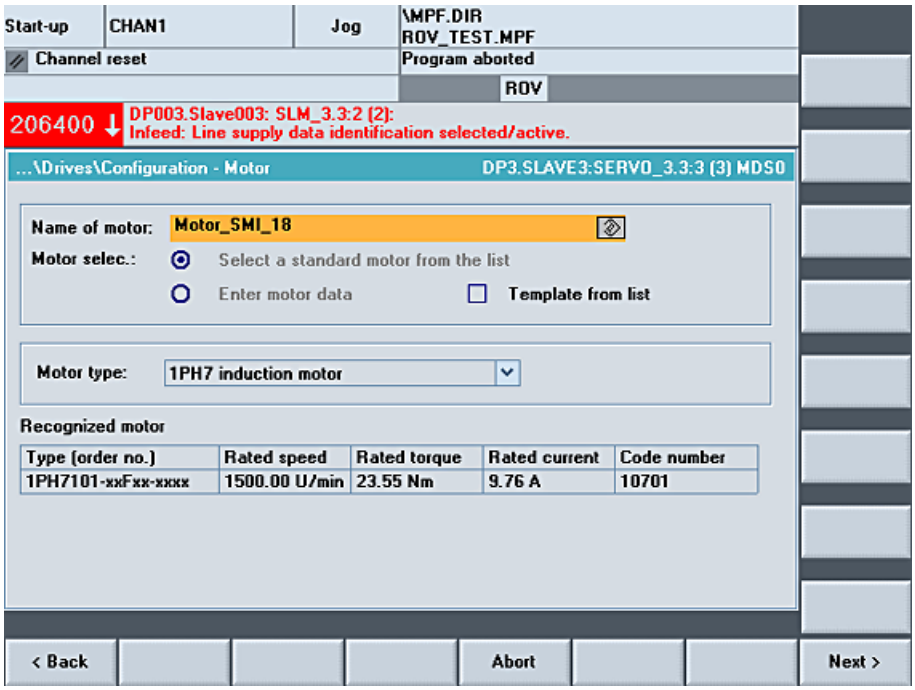


Figure 5-44 Configuring the motor

5. This dialog tells you the exact data of the motor recognized.
   Press "Next >".
6. The next dialog tells you about the configuration of the motor brake.
   Press "Next >".

7. The next dialog tells you which encoder has already been assigned to this drive object (= spindle):

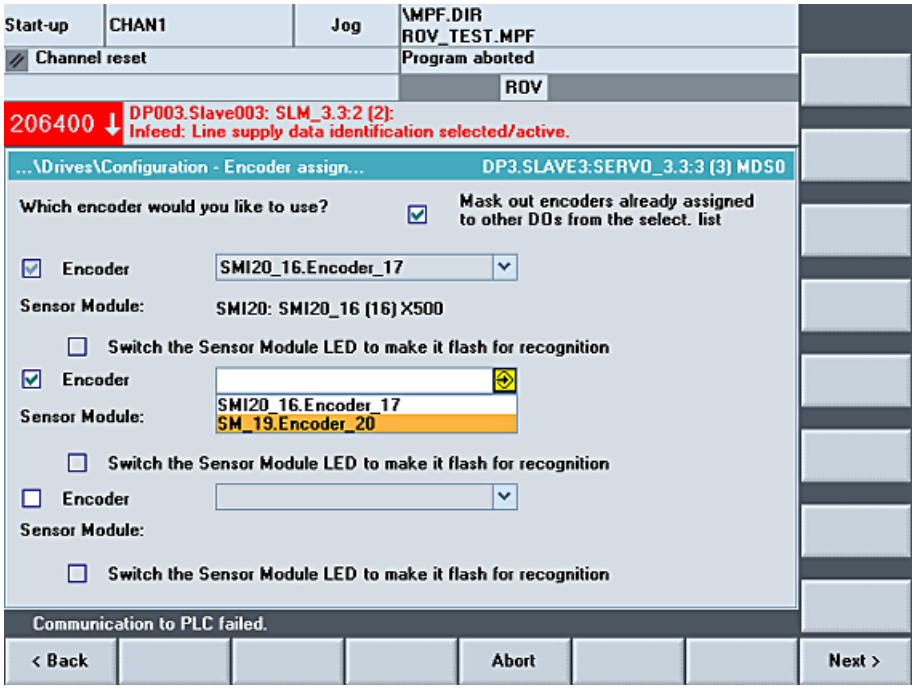


Figure 5-45 Assigning the encoder

8. Activate the "Encoder 2" option.
9. Select the encoder "SM_19.Encoder_20" from the selection list.
10. Confirm with the <INPUT> key.
11. Press "Next >".
12. Confirm the query with "OK".

    This may take several minutes.
13. While the data is being saved, you receive the following status display:

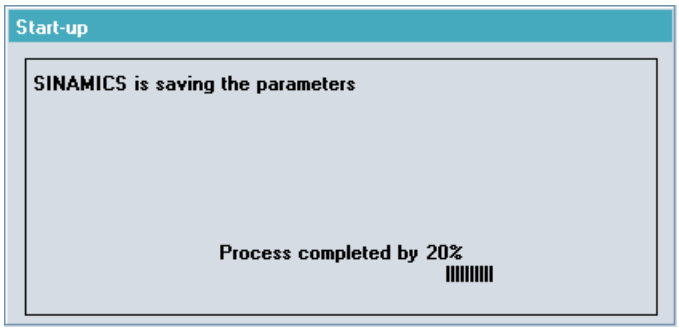


Figure 5-46 Encoder 2: Saving data

14. The next dialog tells you about the configuration of the control mode.
15. Press "Next >".
16. The next dialog tells you about the BICO connection.

17. Press "Next >".
18. At the end you receive a summary showing all the data.
19. If you press "Finish >" you receive the following information:

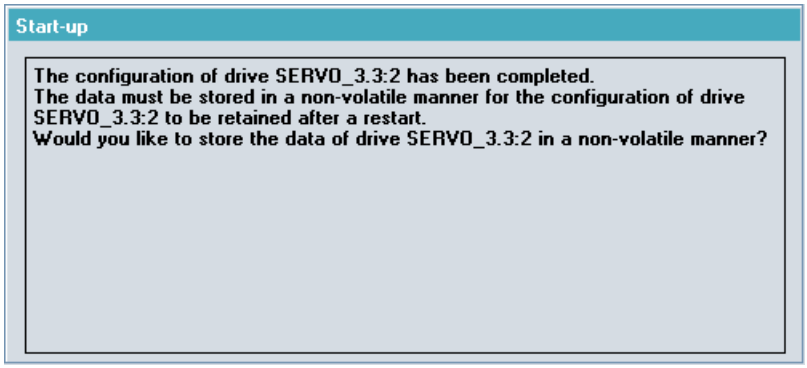


Figure 5-47 Confirm saving

20. Confirm the saving of the configuration data with "Yes".
    Saving can take several minutes.

Result

Step 3 for commissioning of the drives is now complete.

5.1.8 Example: How to assign the axes

Assigning axes

After the final save of the Encoder 2 configuration data, the following overview is shown:

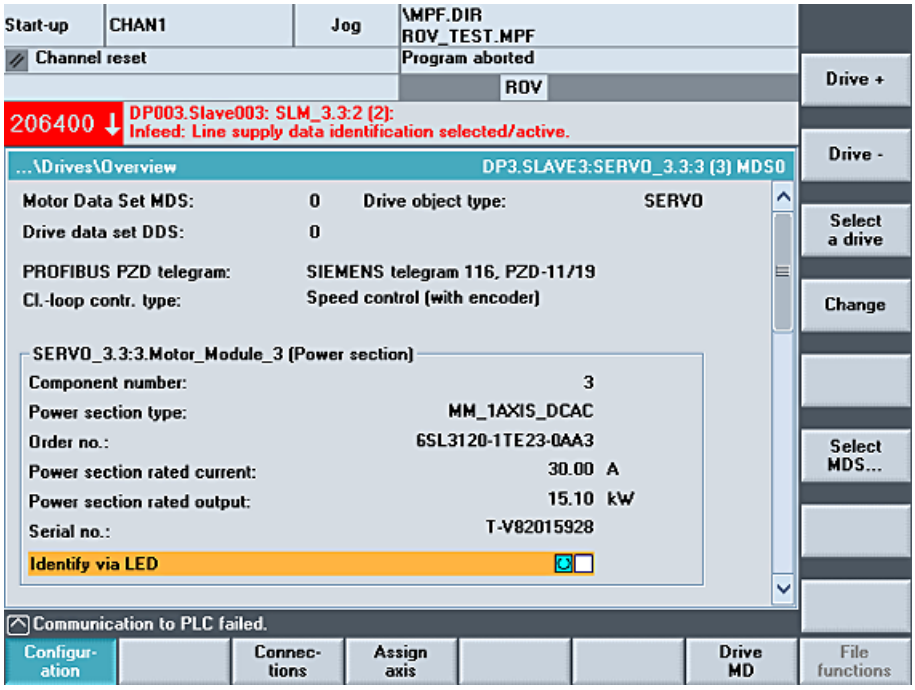


Figure 5-48 Drive data overview

1. To assign every logical drive a real axis, select the horizontal softkey "Assign axis". The "Axis assignment" dialog is opened:

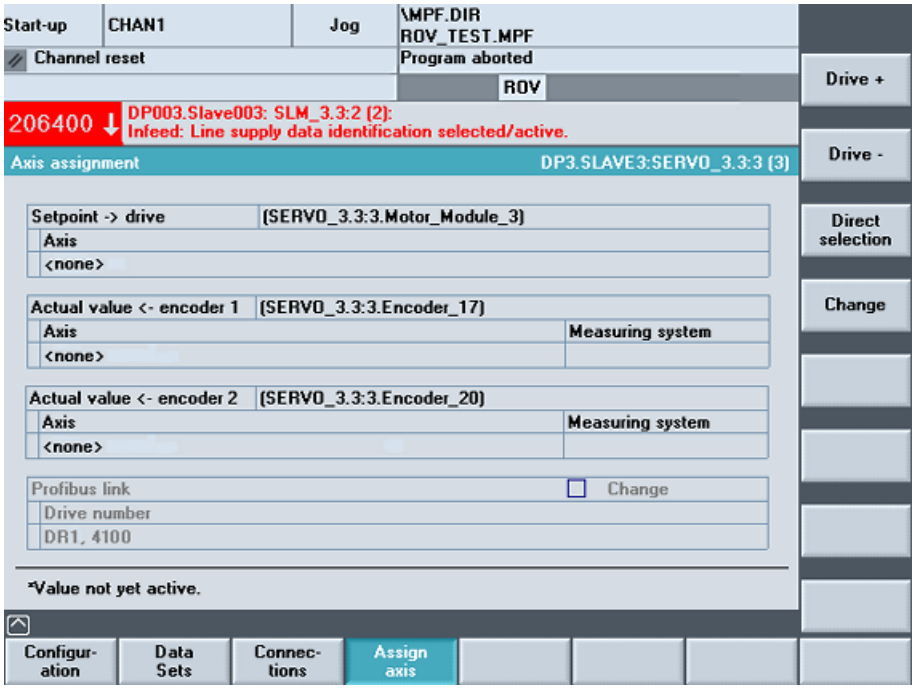


Figure 5-49 Assigning axis

Note

Both the following actions are each carried out **twice** so the softkeys "Change" and "Accept" must be pressed repeatedly.

2. Press the "Change" softkey to assign an axis to DO number 2.
3. From the selection list choose "MSP1" and press the “Accept” softkey.
4. An NCK Power On Reset is needed to make the assignment effective in the system.

5. Press the "Cancel" softkey to assign the other axes first.

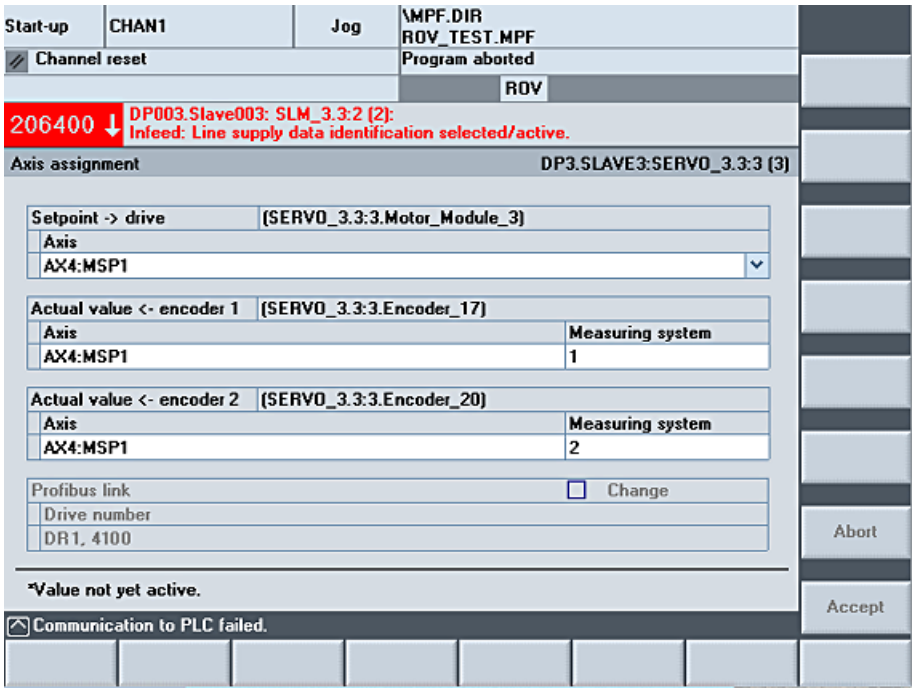


Figure 5-50 Assigning axis: Spindle

6. Select the next module with the vertical softkeys "Drive +" and "Drive -".
7. Assign all axes one after another:

Axis	Drive
MSP1	SERVO_3.3:3
MX1	SERVO_3.3:4
MY1	SERVO_3.3:5
MZ1	SERVO_3.3:6

Accept settings

Finally, carry out the NCK Power On Reset and check the following settings:

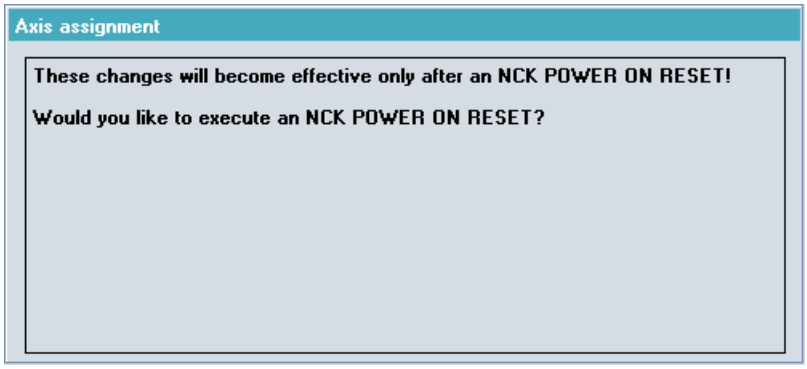


Figure 5-51 Confirming Power On Reset

Confirm with "OK" and carry out a restart; not only for the drives and but also for the control. The following assignment is shown on the PG/PC:

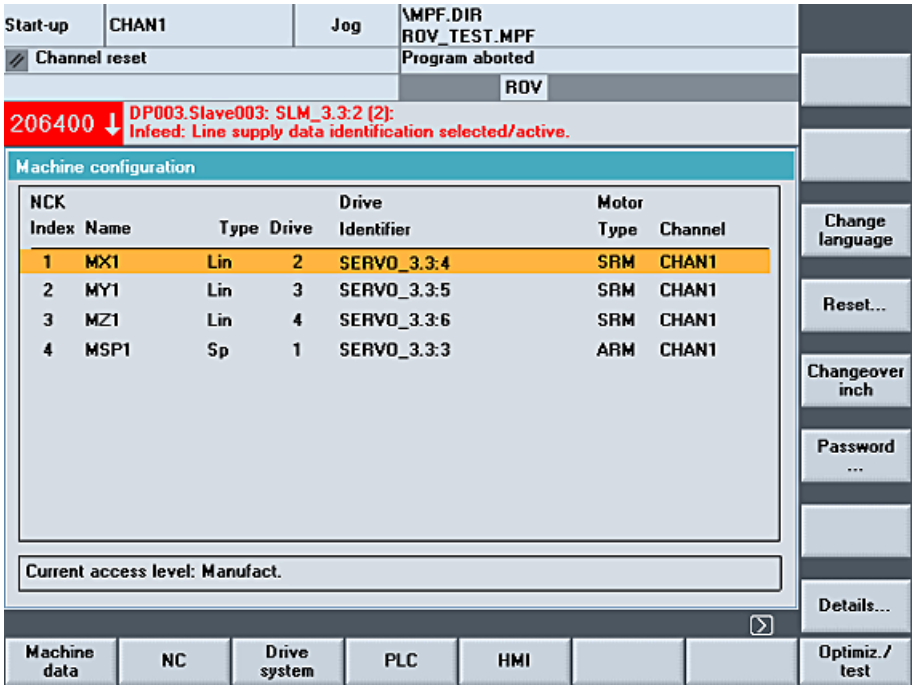


Figure 5-52 Assignment of all axes

After the restart, the following machine configuration is displayed on the control:

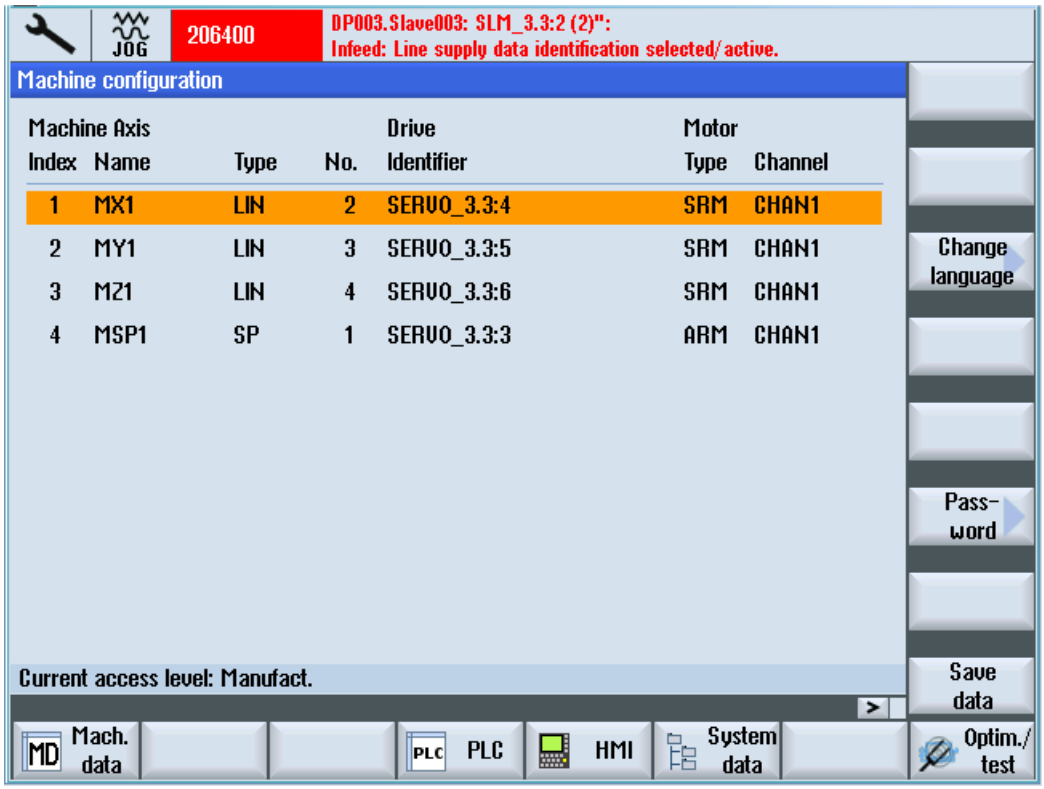


Figure 5-53 Control: Configuration of all axes

Result

Step 4 for commissioning of the drives is now complete.

Data backup

The configuration data is backed up after commissioning in the non-volatile memory with the vertical softkey "Save data":

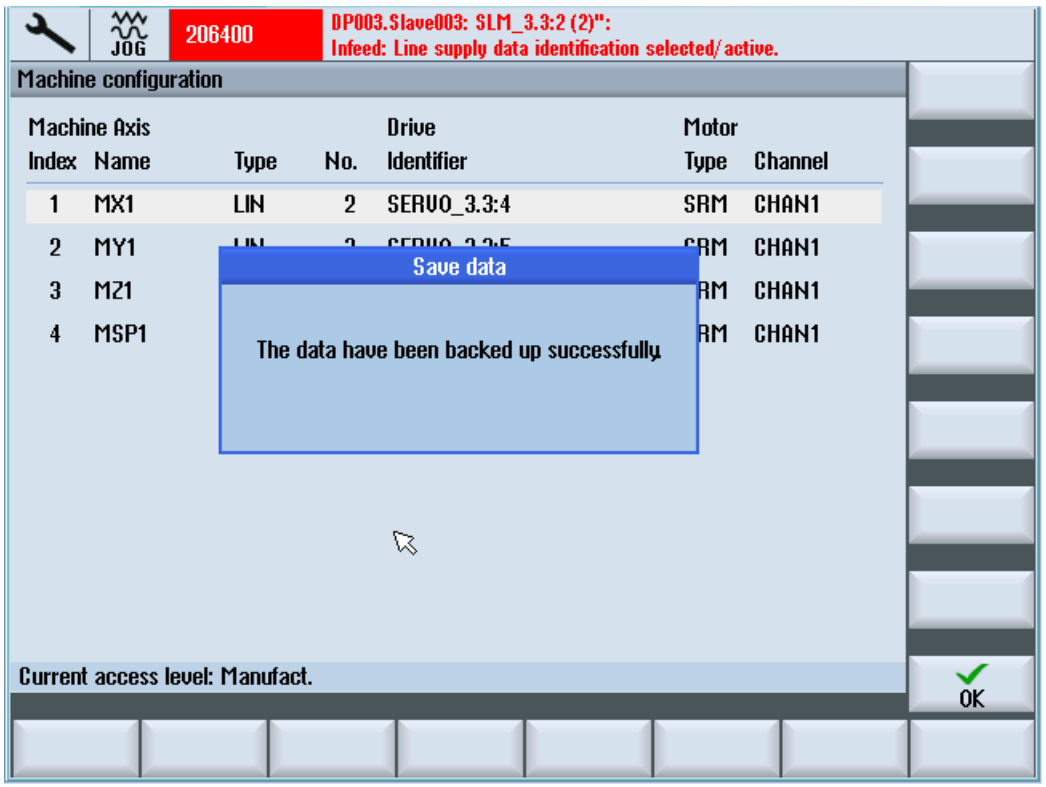


Figure 5-54 Data backup

5.1.9 Example: Setting machine data for an axis/spindle

Axis machine data

After the commissioning in the previous example, the following values are set for the axis machine data:

Axis machine data		X	Y	Z	SP	A
30130	$MA_GEAR_STEP_MAX_VELO	1	1	1	1	1
30230	$MA_ENC_INPUT_NR	1	1	1	2	1
30240	$MA_ENC_TYPE	1	1	1	1	4
31020	$MA_ENC_RESOL	2048	2048	2048	1024	512
34200	$MA_ENC_REFP_MODE	1	1	1	1	0

To operate the axes in the previous example in JOG mode **after** commissioning the drive, enter the new values for the following machine data:

Spindle machine data		Default setting	New value
32000	$MA_MAX_AX_VELO	10000	3000
32010	$MA_JOG_VELO_RAPID	10000	100
32020	$MA_JOG_VELO	2000	50
35100	$MA_SPIND_VELO_LIMIT	10000	3000
35110[0]	$MA_GEAR_STEP_MAX_VELO[0]	500	3000
35110[1]	$MA_GEAR_STEP_MAX_VELO[1]	500	3000
35130[0]	$MA_GEAR_STEP_MAX_VELO_LIMIT[0]	500	3150
35130[1]	$MA_GEAR_STEP_MAX_VELO_LIMIT[1]	500	3150
36200[0]	$MA_AX_VELO_LIMIT[0]	11500	3300
36200[1]	$MA_AX_VELO_LIMIT[1]	11500	3300

Further information on the status of axes and spindle is available in the "Diagnostics" operating area with the menu forward key:

- The "Axis diagnostics" softkey opens the "Service overview".
- The "Service axis" softkey opens the "Service Axis/Spindle" dialog.

The drive parameters are set under: Operating area "Start-up" → Softkey "Drive units" → "Inputs/outputs".

NOTICE

DSC operation

Parameterization of the combination STIFFNESS_CONTROL_ENABLE=1 and ENC_FEEDBACK_POL= -(encoder inversion) is not permitted.

Encoder inversion must be performed in the drive parameter p0410 bit 1 of the SINAMICS.

- DSC operation is preset for motor measuring systems.
- DSC operation must be explicitly activated for external measuring systems.
  Requirement: Telegram ≥ 116
  SINAMICS drive parameters:
  p1192[0] encoder selection
  p1193[0] encoder adaptation factor

Logical I/O address of drive

The following data must be assigned parameters in the NC for the cyclic process data exchange between the NC and the drives:

- I/O address of the drives
- I/O address of the Control Units
- Telegram

The I/O address tells the NC the data areas via which data is cyclically exchanged between the drive and the Control Unit.

The I/O addresses parameterized in the PLC project are entered in the following machine data:

Machine data		without NX module
13120[0]	$MN_CONTROL_UNIT_LOGIC_ADRESS	= 6500
13120[1]	$MN_CONTROL_UNIT_LOGIC_ADRESS	= 0

Machine data		with NX module
13120[0]	$MN_CONTROL_UNIT_LOGIC_ADRESS	= 6500
13120[1]	$MN_CONTROL_UNIT_LOGIC_ADRESS	= 6516

5.1.10 Parameters for the axis / spindle test run

Relevant parameters and terminals

Drive:

Parameter/terminal	Meaning
p0840	ON/OFF1
p0844	1. OFF2
p0845	2. OFF2
p0848	1. OFF3
p0849	2. OFF3
p0852	Enable operation
X21.3 (+24 V) and X21.4 (ground)	EP terminals enable (pulse enable)
p0864	Infeed enable
p1140	Ramp-function generator enable
p1141	Ramp-function generator start
p1142	Setpoint enable

Infeed:

Parameter/terminal	Meaning
p0840	ON/OFF1
p0844	1. OFF2
p0845	2. OFF2
p0852	Enable operation
X21.3 (+24 V) and X21.4 (ground)	EP terminals enable (pulse enable)

The drive parameters are set under: Operating area "Start-up" → Softkey "Drive units" → "Inputs/outputs".

See also

Further references on the drive:

- SINAMICS S120 Commissioning Manual
- Manual Booksize power units

5.2 Topology rules for DRIVE-CLiQ

5.2.1 Topology rules for S120 Combi

Topology rules for DRIVE-CLiQ

There are fixed DRIVE-CLiQ topology rules for the S120 Combi. These rules must be observed. If these rules are violated, then a corresponding alarm is displayed.

Assigning the DRIVE-CLiQ interfaces

Table 5- 1 Assigning the DRIVE-CLiQ interfaces on the S120 Combi

DRIVE-CLiQ interface	Connection with
X200	X100 of the PPU
X201	Motor encoder, spindle
X202	Motor encoder, feedrate 1
X203	Motor encoder, feedrate 2
X204	Motor encoder feed 3 → only for 4 axis Power Modules remains empty for 3-axis Power Modules
X205	Optional: 2. direct sin/cos encoder for spindle (via SMx20) [1] Remains empty when connecting a direct TTL spindle encoder via X220

[1] In this case, the TTL encoder interface X220 remains free

Table 5- 2 Assigning the DRIVE-CLiQ interfaces to the SINUMERIK 828D (PPU)

DRIVE-CLiQ interface	Connection with
X100	X200 of the S120 Combi
X101	X200 of a Single Motor Module or Double Motor Module
X102	X500 of the Terminal Module TM54F X500 of the Hub Modules (DMx20) [1]

[1] when using a TM54F, the DMx20 is connected in series at the TM54F via the DRIVE-CLiQ interface X501

Table 5- 3 Assigning the DRIVE-CLiQ interfaces of the expansion axes

DRIVE-CLiQ interface	Connection with
First Single Motor Module	
X200	X102 of the PPU
X201 [1]	X200 of the second Single Motor Module
X202	Motor encoder for feedrate 1st expansion axis (via Sensor Module)

DRIVE-CLiQ interface	Connection with
Second Single Motor Module	
X200	X201 of the first Single Motor Module
X201	Remains empty
X202	Motor encoder for feedrate 2nd expansion axis (via Sensor Module)
Double Motor Module	
X200	X205 of the S120 Combi
X201	Remains empty
X202	Motor encoder for feedrate 1st expansion axis
X203	Motor encoder for feedrate 2nd expansion axis

1) remains empty if only one Single Motor Module is used

Table 5- 4 Assigning the DRIVE-CLiQ interfaces at the TM54F

DRIVE-CLiQ interface	
X500	X102 of the control (PPU)
X501	X500 of the DMx20 If a DMx20 is not being used, then this interface remains empty.

Table 5- 5 Assigning the DRIVE-CLiQ interfaces at the DMx20 to assign a direct measuring system to the feed axes.

DRIVE-CLiQ interface	Feed axis
X500	X501 of the TM54F X102 of the PPU (if a TM54F is not being used)
X501	Feedrate 1 at the S120 Combi
X502	Feedrate 2 at the S120 Combi
X503	Feedrate 3 at the S120 Combi (4 axes Power Modules)
X504	Feedrate 1st expansion axis at the Motor Module
X505	Feedrate 2nd expansion axis at the Motor Module

References

Additional examples relating to connecting up can be found in: SINAMICS S120 Combi Manual

5.2.2 Topology rules for S120 Booksize

Introduction

The following rules apply for wiring components with DRIVE-CLiQ. A distinction is made between **DRIVE-CLiQ rules**, which must always be observed, and **recommended rules**, which, when observed, do not require any subsequent changes to the topology.

The maximum number of DRIVE-CLiQ components and the possible wiring form depend on the following points:

- The binding DRIVE-CLiQ wiring rules
- The number and type of activated drives and functions on the respective control unit
- The computing power of the respective control unit
- The set processing and communication cycles

In addition to the binding wiring rules that must be observed, some additional recommendations as well as topology examples for DRIVE-CLiQ wiring are provided in the PPU manual.

The components used in these examples can be removed, replaced with others or supplemented. If components are replaced by another type or additional components are added, the SIZER tool should be used to check the topology. Every topology that SIZER permits can also run and is therefore correct (SINAMICS S120 D Function Manual /FH1/).

DRIVE-CLiQ rules

The wiring rules below apply to standard cycle times (servo 125 µs, vector 400 µs). For cycle times that are shorter than the corresponding standard cycle times, additional restrictions apply due to the computing power of the Control Unit.

The rules below apply on a general basis, unless limited, as a function of the firmware version.

- A maximum of 8 DRIVE-CLiQ nodes can be connected in one row. A row is always seen from the perspective of the Control Unit.
- A maximum of 14 nodes can be connected to one DRIVE-CLiQ line on a Control Unit.
- Ring wiring is not permitted.
- Components must not be double-wired.

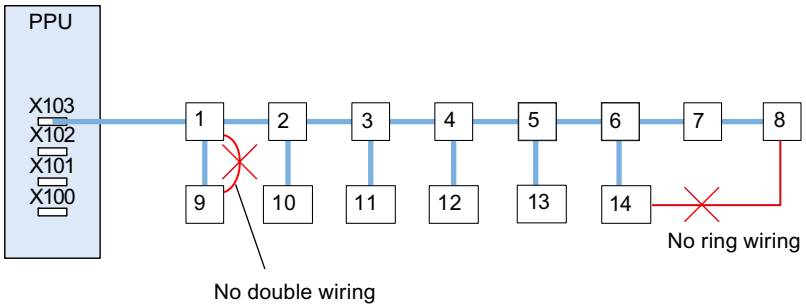


Figure 5-55 Example: DRIVE-CLiQ line at terminal X103 of a Control Unit

- The Terminal Modules TM15, TM17 and TM41 have faster sample cycles than the TM31 and TM54F. For this reason, the two Terminal Module groups must be connected to separate DRIVE-CLiQ lines.
- Only one Line Module can be connected to a Control Unit. Further Line Modules may be connected in parallel to this Line Module.
- If using chassis design components, no more than one Smart Line Module and one Basic Line Module may be jointly operated on one Control Unit (mixed operation on a DRIVE-CLiQ line).
- The default sampling times may be changed.
- Mixed operation of servo and vector is not permitted.
- Mixed operation (servo with vector V/f control) is possible.
- During mixed operation of servo and vector V/f control, separate DRIVE-CLiQ lines must be used for Motor Modules (mixed operation is not permissible on Double Motor Modules).
- With vector V/f control, more than four nodes can only be connected to one DRIVE-CLiQ line on the Control Unit.
- A maximum of nine encoders can be connected.
- A maximum of eight Terminal Modules can be connected to one Control Unit.
- The Active Line Module booksize and Motor Modules booksize
  - can be connected to one DRIVE-CLiQ line in **servo** mode.
  - must be connected to separate DRIVE-CLiQ lines in **vector** mode.
- The Line Module (chassis) (Active Line, Basic Line, Smart Line) and the Motor Modules (chassis) must be connected to separate DRIVE-CLiQ lines.
- Motor Modules in chassis format with different current controller cycles must be connected to separate DRIVE-CLiQ lines. For this reason, chassis Motor Modules and booksize Motor Modules must be connected to separate DRIVE-CLiQ lines.
- The Voltage Sensing Module (VSM) should be connected to a free DRIVE-CLiQ port of the corresponding Active Line Module / Motor Module (due to automatic assignment of the VSM).

- The sampling times (p0115[0] and p4099) of all components that are connected to a DRIVE-CLiQ line must be divisible by one another with an integer result. If the current controller sampling time on a DO has to be changed to another pattern that does not match the other DOs on the DRIVE-CLiQ line, the following options are available:
  - Reconnect the DO to a separate DRIVE-CLiQ line.
  - Also change the current controller sampling time and the sampling time of the inputs/outputs of the DO not involved so that they again fit into the time grid.

Note

A Double Motor Module, a DMC20, and a TM54F each correspond to two DRIVE-CLiQ nodes. This also applies to Double Motor Modules, of which just one drive is configured.

To enable the function "Automatic configuration" to assign the encoders to the drives, the recommended rules below must be observed.

Recommended rules

- The DRIVE-CLiQ cable from the Control Unit must be connected as follows:
  - To X200 of the first booksize power unit
  - To X400 of the first chassis power unit
- The DRIVE-CLiQ connections between the power units must each be connected from interface X201 to X200 or from X401 to X400 on the follow-on component.

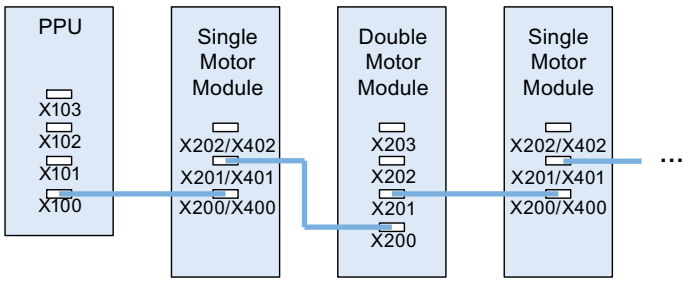


Figure 5-56 Example: DRIVE-CLiQ line

- The motor encoder must be connected to the associated power unit.

Component	Connecting the motor encoder via DRIVE-CLiQ
Single Motor Module booksize	X202
Double Motor Module booksize	• Motor connection X1: Encoder at X202 • Motor connection X2: Encoder at X203
Single Motor Module chassis	X402
Power Module chassis	X402

Note

If an additional encoder is connected to a Motor Module, it is assigned to this drive as encoder 2 in the automatic configuration.

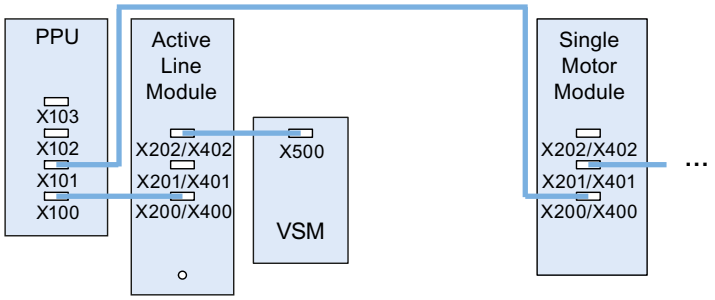


Figure 5-57 Example: Topology with VSM for booksize and chassis components

Component	VSM connection
Active Line Module booksize	X202
Active Line Module (chassis)	X402
Power Modules	The VSM is not supported.

5.3 Terminal assignments

5.3.1 Terminal assignment on X122

Terminal assignment X122 of the Control Unit (PPU)

Pin no.	Function	Assignment	BICO source/sink	
1	Input[1)]	ON/OFF1 infeed for: Line Module **with** DRIVE-CLiQ connection	CU: r0722.0	Infeed p0840
		"Infeed ready signal" for: Line Module **without** DRIVE-CLiQ connection	SLM X21.1	Drive p0864
2	Input	"OFF3 – quick stop"	CU: r0722.1	Each drive 2. OFF3, p0849
3	Input	SH/SBC 1 - Group 1 SINAMICS Safety Integrated (SH enable = p9601)	CU: r0722.2	p9620 (all drives in the group)
4	Input	SH/SBC 1 - Group 2 SINAMICS Safety Integrated (SH enable = p9601)	CU: r0722.3	p9620 (all drives in the group)
5	Input	No default setting	--	--
6	Input	No default setting	--	--
7	MEXT0: Ground for pins 1...6			
8	+24 V power supply			
9	Output	SH/SBC 1 - Group 1 SINAMICS Safety Integrated	CU: p0738	p9774 bit 1 BICO from CU after the 1st drive in the group
10	Output	SH/SBC 1 - Group 2 SINAMICS Safety Integrated	CU: p0739	p9774 bit 1 BICO from CU after the 1st drive in the group
11	MEXT1: Ground for pins 9, 10, 12, 13			
12	Input	BERO 1 - zero mark substitute	CU: r0722.10	Drive p0495 = 2
13	Input	Probe 1: Distributed measurement (MD13210=1)	CU: p0680[0]=0	Every drive p0488 Index=encoders 1,2,3=3
14	MEXT1: Ground for pins 9, 10, 12, 13			
[1)] Low-high edge required				

5.3.2 Terminal assignment on X132

Terminal assignment X132 of the Control Unit (PPU)

Pin No.	Function	Assignment	BICO source/sink	
1	Input	No default setting	--	--
2	Input	No default setting	--	--
3	Input	No default setting	--	--
4	Input	Line contactor, feedback signal	CU: r0722.7	LM: p0860
5	Input	2. OFF 2	CU: r0722.14	Drive p0845
6	Input	No default setting	--	--
7	MEXT0: Ground for pins 1...6			
8	+24 V power supply			
9	Output	Infeed: in operation (LM with DRIVE-CLiQ connection)	LM: r0863.0	CU: p0742
10	Output	Infeed: Ready for switching on (LM with DRIVE-CLiQ connection)	LM: r0899.0	CU: p0743
11	MEXT1: Ground for pins 9, 10, 12, 13			
12	Output	Line contactor control	LM: r0863.1	CU: p0744
	Input	BERO 2 - zero mark substitute	CU: r0722.14	Drive p0495 = 5
13	Input	Probe 2: Distributed measurement (MD13210=1)	CU: p0680[1]=0 CU: p0728 bit 15=0	Every drive p0489 Index=encoders 1,2,3=6
14	MEXT1: Ground for pins 9, 10, 12, 13			

5.3.3 Terminal assignment at X242 and X252

Terminal assignment X242 of the Control Unit (PPU)

Pin No.	Function	Assignment
1	Not connected	
2	Not connected	
3	Input	Digital input $A_IN[1]
4	Input	Digital input $A_IN[2]
5	Input	Digital input $A_IN[3]
6	Input	Digital input $A_IN[4]
7	MEXT2 Ground for pins 3...6	
8	+24 V power supply	
9	Output	Digital output $A_OUT[1]
10	Output	Digital output $A_OUT[2]
11	MEXT3: Ground for pins 9, 10, 12, 13	
12	Output	Digital output $A_OUT[3]
13	Output	Digital output $A_OUT[4]
14	MEXT3: Ground for pins 9, 10, 12, 13	

Terminal assignment X252 of the Control Unit (PPU)

Pin No.	Function	Assignment
1	AOUT	Analog output (voltage for analog spindle)
2	AGND	Analog ground
3	Input	Digital input $A_IN[9]
4	Input	Digital input $A_IN[10]
5	Input	Digital input $A_IN[11]
6	Input	Digital input $A_IN[12]
7	MEXT2 Ground for pins 3...6	
8	+24 V power supply	
9	Output	Digital output $A_OUT[9]
10	Output	Digital output $A_OUT[10]
11	MEXT3: Ground for pins 9, 10, 12, 13	
12	Output	Digital output $A_OUT[11]
13	Output	Digital output $A_OUT[12]
14	MEXT3: Ground for pins 9, 10, 12, 13	

5.3.4 Terminal assignment on X122 for a Numeric Control Extension

Terminal assignment X122 on the NX

Pin No.	Function	Assignment	BICO source/sink	
1	Input	No default setting	--	--
2	Input	No default setting	--	--
3	Input	SH/SBC 1 - Group 1 SINAMICS Safety Integrated (SH enable = p9601)	NX: r0722.2	p9620 (all drives in the group)
4	Input	SH/SBC 1 - Group 2 SINAMICS Safety Integrated (SH enable = p9601)	NX: r0722.3	p9620 (all drives in the group)
5	Input	2. OFF 2	NX: r0722.11	Drive p0485
6	Input	No default setting	--	--
7	Ground for pins 1...6			
8	Ground for pins 9, 10, 12, 13			
9	Output	SH/SBC 1 - Group 1 SINAMICS Safety Integrated	NX: p0738	p9774.1 BICO from CU after the 1st drive in the group
10	Output	SH/SBC 1 - Group 2 SINAMICS Safety Integrated	NX: p0739	p9774.1 BICO from CU after the 1st drive in the group
11	Ground for pins 9, 10, 12, 13			
12	Input	BERO 1 - zero mark substitute	NX: r0722.10	Drive p0489 = 2
	Input	Probe 2: Distributed measurement	NX: p0680[1]=0 p0728 bit 10=0	Every drive on NX: p0489 Index=encoders 1,2,3=2
13	Input	BERO 2 - zero mark substitute	NX: r0722.11	Drive p0488 = 3
	Input	Probe 1: Distributed measurement	NX: p0680[0]=0 p0728 bit 11=0	Every drive on NX: p0488 Index=encoders 1,2,3=3
14	Ground for pins 9, 10, 12, 13			

Note

The ON/OFF 1 and OFF 3 enable signals are only required at terminal X122 of the CU.

5.3.5 Example: Circuitry for a CU with line contactor

Example

The circuitry shown refers to the assignment of the terminals in the previous chapters.

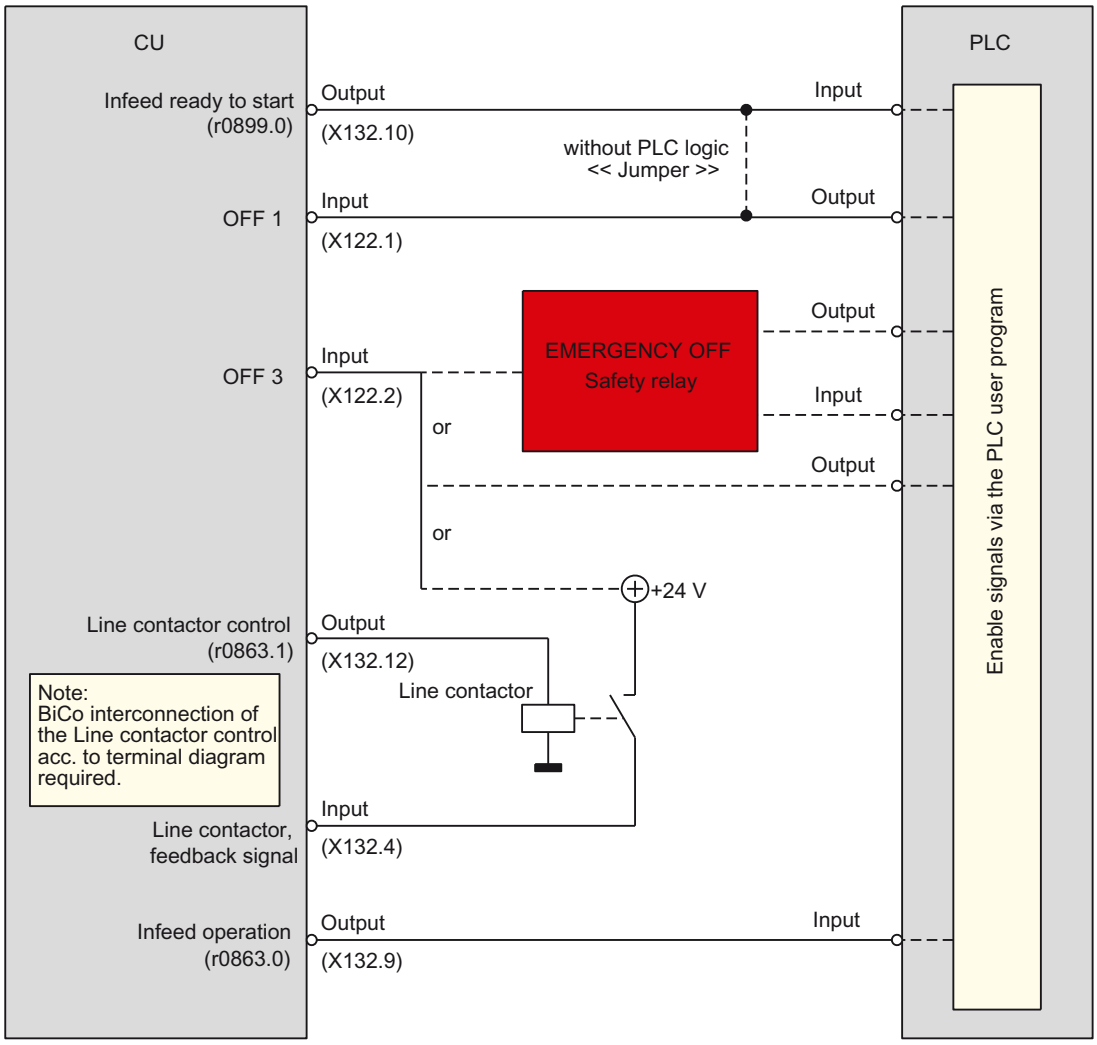
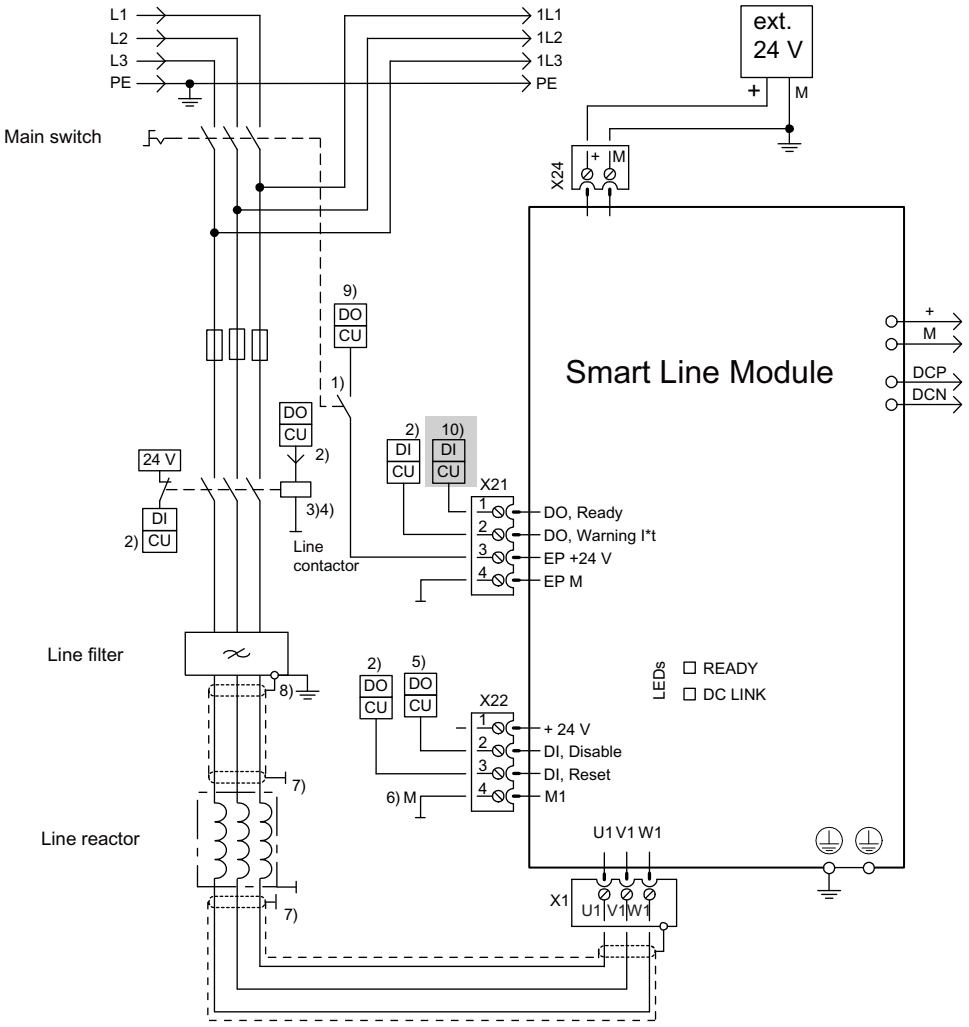


Figure 5-58 Circuitry for a Control Unit with line contactor

Smart Line Module connection

The connections for the digital in/outputs X122 and X132 are on the rear side of the control.



① Early opening contact t > 10 ms, 24 VDC and ground must be applied for operation

② DI/DO controlled by the Control Unit

③ No additional load permitted downstream of line contactor

④ The current carrying capacity of the DO must be observed; an output interface must be used if required.

⑤ DO high, feedback deactivated (a jumper can be inserted between X22 pin 1 and pin 2 for permanent deactivation).

⑥ X22 pin 4 must be connected to ground (external 24 V).

⑦ Contacting via rear mounting panel or shielding buses in accordance with EMC installation guideline

⑧ 5 kW and 10 kW line filters via shield connection

⑨ Signal output of the control, to avoid interference of the 24 VDC supply on the EP terminal.

⑩ **Connect via BICO to parameter p0864 → X122.1**

Figure 5-59 Example: SLM connection

ON/OFF1 enable: Connection of Smart Line Module pin X21.1 → X122.1 SINUMERIK 828D

Connect further input and output signals to the PLC I/O:

- DI → PLC inputs
- DO → PLC outputs

See also

Additional information can be found in:

- SINUMERIK 828D Manual PPU
- SINAMICS S120 Manual Booksize Power Units

5.3.6 Connecting the probes

Connecting the probes

The probes are connected to both the SINUMERIK 828D CU and the NX:

1. Probe at terminal X122 pin 13 / X122 pin 13 of the NX
2. Probe at terminal X132 pin 13 / X122 pin 12 of the NX

Note

A precondition for measuring with the SINUMERIK 828D is that the distributed (local) measurement function has been parameterized.

Central measurement is not possible with SINUMERIK 828D.

Machine data

The following machine data should be checked and adjusted if necessary:

- **General machine data:**

  MD13200[0] $MN_MEAS_PROBE_LOW_ACTIVE = 0 or 1

  MD13200[1] $MN_MEAS_PROBE_LOW_ACTIVE = 0 or 1

  Value 0 = deflected state 24 V (default)

  Value 1 = deflected state 0 V

  MD13210 $MN_MEAS_TYPE = 1 distributed measurement

- **Axis-specific machine data:**

  MD30244[0] $MA_ENC_MEAS_TYPE = 1 for all axes

  MD30244[1] $MA_ENC_MEAS_TYPE = 1 for all axes

Note

The machine data MD13210 and MD30244 are preset to the value 1 and cannot be changed! (Data class: SYSTEM)

- **Control Unit parameters:**

  p0680[0] Central probe input terminal = 0

  p0680[1] Central probe input terminal = 0

  p0680[2] Central probe input terminal = 0

  The 1st probe is connected to terminal X122 pin 13, the 2nd probe to terminal X132 pin 13 of the SINUMERIK 828D and - if present - on the NX10 module to terminal X122 pin 12.

  A precondition is the switchover of pin X132.13 from output to input.

  Set CU input or output = set p0728 bit 15 to 0 (DI/DO X132.13)

- **Drive parameters:**

  p0488[0] Probe 1 input terminal: Encoder 1 = 3 → connector X 122.13

  p0488[1] Probe 1 input terminal: Encoder 2 = 3 → connector X 122.13

  p0488[2] Probe 1 input terminal: Encoder 3 = 0 → not used

  p0489[0] Probe 2 input terminal: Encoder 1 = 6 → connector X 132.13

  p0489[1] Probe 2 input terminal: Encoder 2 = 6 → connector X 132.13

  p0489[2] Probe 2 input terminal: Encoder 3 = 0 → not used

  The 2nd probe at terminal X122.12 must be parameterized for all axes that are parameterized on the NX module:

  p0489[0] Probe 2 input terminal: Encoder 1 = 6 → connector X 122.12

  p0489[1] Probe 2 input terminal: Encoder 2 = 6 → connector X 122.12

  p0489[2] Probe 2 input terminal: Encoder 3 = 0 → not used

Note

All drives must be parameterized.

Probe status

DB2700	General signals from NCK [r] NCK → PLC interface							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBB0							EMERGENCY STOP active	
DBB1	Inch dimension system						Probe actuated	
							Probe 2	Probe 1

See also

Measuring cycles and measurement functions (Page 231)

Parameterizing machine data 6

6.1 Classification of machine data

Authorization for machine data

At least the Manufacturer password is necessary to enter or change machine data.

⚠ DANGER
Changes in the machine data have a considerable influence on the machine. Incorrect configuration of the parameters can endanger human life and cause damage to the machine.

Classification of machine data

The machine data is divided into the following areas:

- General machine data ($MN)
- Channel-specific machine data ($MC)
- Axis-specific machine data ($MA)
- SINAMICS machine data (Control Unit and drive machine data):

  r0001 ... r9999 (read-only)

  r0001 ... r9999 (read/write)
- General setting data ($MNS)
- Channel-specific setting data ($MCS)
- Axis-specific setting data ($MAS)
- Display machine data ($MM)

Note

Machine data for turning and milling technologies is already set up in such a way that an adjustment of machine data is only necessary in exceptional cases.

A separate list image is provided for each of these areas in which you can view and edit machine data:

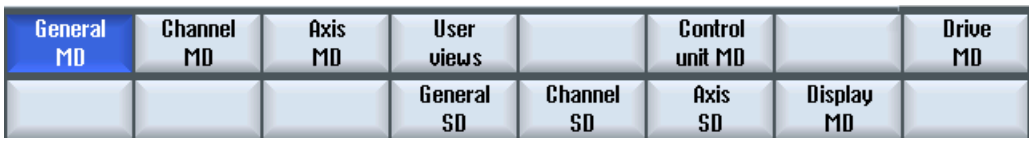


Figure 6-1 Softkey bar

The following properties of the machine data are displayed from left to right:

- Number of the machine data, if applicable with array index in square brackets
- Name of the machine data
- Value of the machine data
- Unit of the machine data
- Effectiveness of the machine data
- Data class

See also

Description of the data classes: Function Manual Basic Functions (P4)

A detailed description of the machine data and interface signals is given in the Parameter Manual with cross-references to the appropriate section in the Function Manual.

Physical units of the machine data

The physical units of machine data are displayed on the right-hand side of the input field:

Display	Unit	Measured quantity
m/s**2	m/s²	Acceleration
rev/s**3	rev/s³	Acceleration change for the rotating axis
kg/m**2	kgm²	Moment of inertia
mH	mH (millihenry):	Inductance
Nm	Nm (Newton meters):	Torque
us	µs (microseconds):	Time
µA	µA (microamperes):	Electric current
µVs	µVs (microvolt-seconds):	Magnetic flux
userdef	user-defined:	The unit is defined by the user.

If the machine data does not use units, no units are displayed.

If the data is not available, the "#" symbol is displayed instead of the value. If the value ends in an "H", it is a hexadecimal value.

Effectiveness of the machine data

The right-hand column indicates when a machine data becomes effective:

cf =	with confirmation using the "Activate MD" softkey
po =	NCK Power On Reset
re =	Reset
so =	immediately effective

User views

User views are user-specific groups of machine data. They are used to call all relevant machine data in a certain operating state from various areas for processing.

The user views are stored on the CompactFlash card with the following path:

`user/sinumerik/hmi/template/user_views`

The following user views are already available as template:

- Electrical_Startup
- Mechanical_Startup
- Optimizing_Axis

6.2 Processing part programs from external CNC systems

NOTICE
Default setting For SINUMERIK 828D, the following machine data are already suitably preset for the particular technology – either turning or milling. These machine data cannot be changed and are not displayed.

Activate the ISO Dialect function

The MD18800 $MN_EXTERN_LANGUAGE machine data is used to activate the external language. The ISO dialect M or T language types can be selected using MD10880 $MN_EXTERN_CNC_SYSTEM.

The two G commands of group 47 are used to switch from Siemens mode to ISO dialect mode:

- G290: Siemens NC programming language active
- G291: ISO dialect NC programming language active

The active tool, tool offsets and work offsets are retained. G290 and G291 must be contained just by themselves in an NC program block.

The switchover to an external programming language is contained in the scope of delivery for SINUMERIK 828D. MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB is only valid for NC language commands in the Siemens mode.

See also

Function Manual ISO Dialects

6.3 Machining free form surfaces using Advanced Surface

Machining with Advanced Surface

When executing CAM programs in the high-speed cutting (HSC) range, the control has to process high feedrates with the shortest NC blocks. The result should be a good surface quality of the workpiece with high precision in the µm range at extremely high machining feedrates of > 10 m/min.

Using CYCLE832, the NC program is very precisely harmonized across the various machining strategies:

- When roughing, using smoothing of the contour the emphasis is on speed.
- When finishing, the emphasis is on accuracy.

In both cases, specifying a tolerance ensures that the machining contour is maintained in order to achieve the desired surface quality.

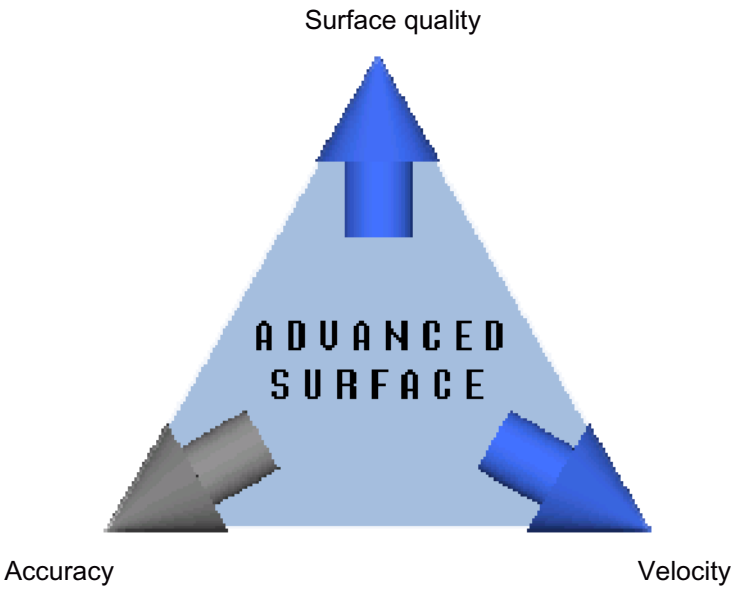


Figure 6-2 Machining a workpiece surface

Machine data for Advanced Surface

The following setpoints are recommended for the listed machine data in order to achieve the optimum workpiece surface when machining mold workpieces:

General machine data:

MD number	Designation	Default setting	Recommen-dation	Data class	Parameter
10200	$MN_INT_INCR_PER_MM	100000	100000	M	--
10210	$MN_INT_INCR_PER_DEG	100000	100000	M	--
Default setting - is not displayed:					
18360	$MN_MM_EXT_PROG_BUFFER_SIZE	500 250	Milling Turning	S	Permanently set depending on the technology.

Channel-specific machine data:

MD number	Designation	Default setting	Recommen-dation	Data class	Parameter
20150[3]	$MC_GCODE_RESET_VALUES	3	3	M	FIFOCTRL
20150[44]	$MC_GCODE_RESET_VALUES	2	2	M	UPATH
20170	$MC_COMPRESS_BLOCK_PATH_LIMIT	20	20	M	--
20172	$MC_ COMPRESS_VELO_TOL	60000	1000	M	--
20443[0]	$MC_LOOKAH_FFORM	0	0	M	DYNNORM
20443[1]	$MC_LOOKAH_FFORM	0	0	M	DYNPOS
20443[2]	$MC_LOOKAH_FFORM	0	1	M	DYNROUGH
20443[3]	$MC_LOOKAH_FFORM	0	1	M	DYNSEMIFIN
20443[4]	$MC_LOOKAH_FFORM	0	1	M	DYNFINISH
20482	$MC_COMPRESSOR_MODE	100	100	M	--
20490	$MC_IGNORE_OVL_FACTOR_FOR_ADIS	1	1	M	--
20550	$MC_EXACT_POS_MODE	3	3	M	--
20600[0]	$MC_MAX_PATH_JERK	10000	10000	M	DYNNORM
20600[1]	$MC_MAX_PATH_JERK	10000	10000	M	DYNPOS
20600[2]	$MC_MAX_PATH_JERK	10000	10000	M	DYNROUGH
20600[3]	$MC_MAX_PATH_JERK	10000	10000	M	DYNSEMIFIN
20600[4]	$MC_MAX_PATH_JERK	10000	10000	M	DYNFINISH
20602[0]	$MC_CURV_EFFECT_ON_PATH_ACCEL	0	0	M	--
20602[1]	$MC_CURV_EFFECT_ON_PATH_ACCEL	0	0	M	--
20602[2]	$MC_CURV_EFFECT_ON_PATH_ACCEL	0,6	0,65	M	--
20602[3]	$MC_CURV_EFFECT_ON_PATH_ACCEL	0,6	0,6	M	--
20602[4]	$MC_CURV_EFFECT_ON_PATH_ACCEL	0,6	0,5	M	--
20603[0]	$MC_CURV_EFFECT_ON_PATH_JERK	0	0	M	--
20603[1]	$MC_CURV_EFFECT_ON_PATH_JERK	0	0	M	--
20603[2]	$MC_CURV_EFFECT_ON_PATH_JERK	0	0	M	--
20603[3]	$MC_CURV_EFFECT_ON_PATH_JERK	0	0	M	--

MD number	Designation	Default setting	Recommen-dation	Data class	Parameter
20603[4]	$MC_CURV_EFFECT_ON_PATH_JERK	0	0	M	--
20605[0]	$MC_PREPDYN_SMOOTHING_FACTOR	0	1	M	--
20605[1]	$MC_PREPDYN_SMOOTHING_FACTOR	0	1	M	--
20605[2]	$MC_PREPDYN_SMOOTHING_FACTOR	0	1	M	--
20605[3]	$MC_PREPDYN_SMOOTHING_FACTOR	0	1	M	--
20605[4]	$MC_PREPDYN_SMOOTHING_FACTOR	0	1	M	--
20606[0]	$MC_PREPDYN_SMOOTHING_ON	0	0	M	DYNNORM
20606[1]	$MC_PREPDYN_SMOOTHING_ON	0	0	M	DYNPOS
20606[2]	$MC_PREPDYN_SMOOTHING_ON	0	1	M	DYNROUGH
20606[3]	$MC_PREPDYN_SMOOTHING_ON	0	1	M	DYNSEMIFIN
20606[4]	$MC_PREPDYN_SMOOTHING_ON	0	1	M	DYNFINISH
20607[i]	$MC_PREPDYN_MAX_FILT_LENGTH_GEO	2	2	S	DYNNORM
20608[i]	$MC_PREPDYN_MAX_FILT_LENGTH_RD	5	5	S	--
28060	$MC_MM_IPO_BUFFER_SIZE	150	150	M	--
28070	$MC_MM_NUM_BLOCKS_IN_PREP	80	80	M	--
28302[0]	$MC_MM_PROTOC_NUM_ETP_STD_TYP	28	28	M	--
28530	$MC_MM_PATH_VELO_SEGMENTS	5	5	M	--
28533	$MC_MM_LOOKAH_FFORM_UNITS	18	18	M	--
28540	$MC_MM_ARCLENGTH_SEGMENTS	10	10	M	--
28610	$MC_MM_PREPDYN_BLOCKS	10	10	M	--
29000	$OC_LOOKAH_NUM_CHECKED_BLOCKS	150	150	M	--

Axis machine data:

MD number	Designation	Default setting	Recommen-dation	Data class	Parameter
32310[0]	$MA_MAX_ACCEL_OVL_FACTOR	1.2	1.2	M	--
32310[1]	$MA_MAX_ACCEL_OVL_FACTOR	1.2	1.2	M	--
32310[2]	$MA_MAX_ACCEL_OVL_FACTOR	1.2	1.2	M	--
32310[3]	$MA_MAX_ACCEL_OVL_FACTOR	1.2	1.2	M	--
32310[4]	$MA_MAX_ACCEL_OVL_FACTOR	1.2	1.2	M	--
32402	$MA_AX_JERK_MODE	2	2	M	--
32431[0]	$MA_MAX_AX_JERK	1000000	1000000	I	DYNNORM
32431[1]	$MA_MAX_AX_JERK	1000000	1000000	I	DYNPOS
32431[2]	$MA_MAX_AX_JERK	1000000	1000000	I	DYNROUGH
32431[3]	$MA_MAX_AX_JERK	20	20	I	DYNSEMIFIN
32431[4]	$MA_MAX_AX_JERK	20	20	I	DYNFINISH

MD number	Designation	Default setting	Recommen-dation	Data class	Parameter
32432[0]	$MA_PATH_TRANS_JERK_LIM	1000000	1000000	I	DYNNORM
32432[1]	$MA_PATH_TRANS_JERK_LIM	1000000	1000000	I	DYNPOS
32432[2]	$MA_PATH_TRANS_JERK_LIM	1000000	1000000	I	DYNROUGH
32432[3]	$MA_PATH_TRANS_JERK_LIM	20	20	I	DYNSEMIFIN
32432[4]	$MA_PATH_TRANS_JERK_LIM	20	20	I	DYNFINISH
32620	$MA_FFW_MODE	3	3	M	DYNNORM

Setting data:

MD number	Designation	Default setting	Recommen-dation	Data class	Parameter
42471	$SC_MIN_CURV_RADIUS	1	1	M	--

Setting cycle machine data

7

7.1 Settings for activating cycles

Configuring the cycles

The cycles are configured via the following machine and setting data:

- General machine data
- Channel-specific machine data
- Axis-specific machine data
- General setting data
- Channel-specific setting data
- Axis-specific setting data



Software option

For the "ShopMill" and "ShopTurn" functions, you require the software option: "ShopMill/ShopTurn"



Software option

For a SINUMERIK 828D BASIC T and BASIC M, you require the software option "Extended technology functions" in order to enable the following cycles:

Cycle	Function (sequence of the softkeys)
Drilling patterns	Drilling → Positions
CYCLE60	Milling → Engraving
CYCLE63	Milling → Contour milling → Pocket Milling → Contour milling → Spigot
CYCLE64	Milling → Contour milling → Pre-drilling
CYCLE70	Milling → Thread milling
CYCLE78	Milling → Drilling → Thread→ Thread milling
CYCLE79	Milling → Spigot, polygon
CYCLE930	Turning → Groove → Groove 3
CYCLE940	Turning → Undercut → Undercut thread DIN
CYCLE951	Turning → Groove → Stock removal 3
CYCLE952	Turning → Contour turning → Stock removal, plunging, plunge turning, ...

7.1.1 Settings to the coordinate system

Setting the planes

MD52005 $MCS_DISP_PLANE_MILL	
Plane selection G17, G18, G19	
= 0	Milling: Plane selection in the cycle support when programming under "programGUIDE G code"
= 17	G17 plane (default setting)
= 18	G18 plane
= 19	G19 plane

MD52006 $MCS_DISP_PLANE_TURN	
Plane selection G17, G18, G19	
= 0	Turning: Plane selection in the cycle support when programming under "programGUIDE G code"
= 17	G17 plane
= 18	G18 plane (default setting)
= 19	G19 plane

Selecting a coordinate system

Set the coordinate system at the machine using the following channel-specific machine data:

MD52000 $MCS_DISP_COORDINATE_SYSTEM	
Position of the coordinate system	
= 0 ... 47	

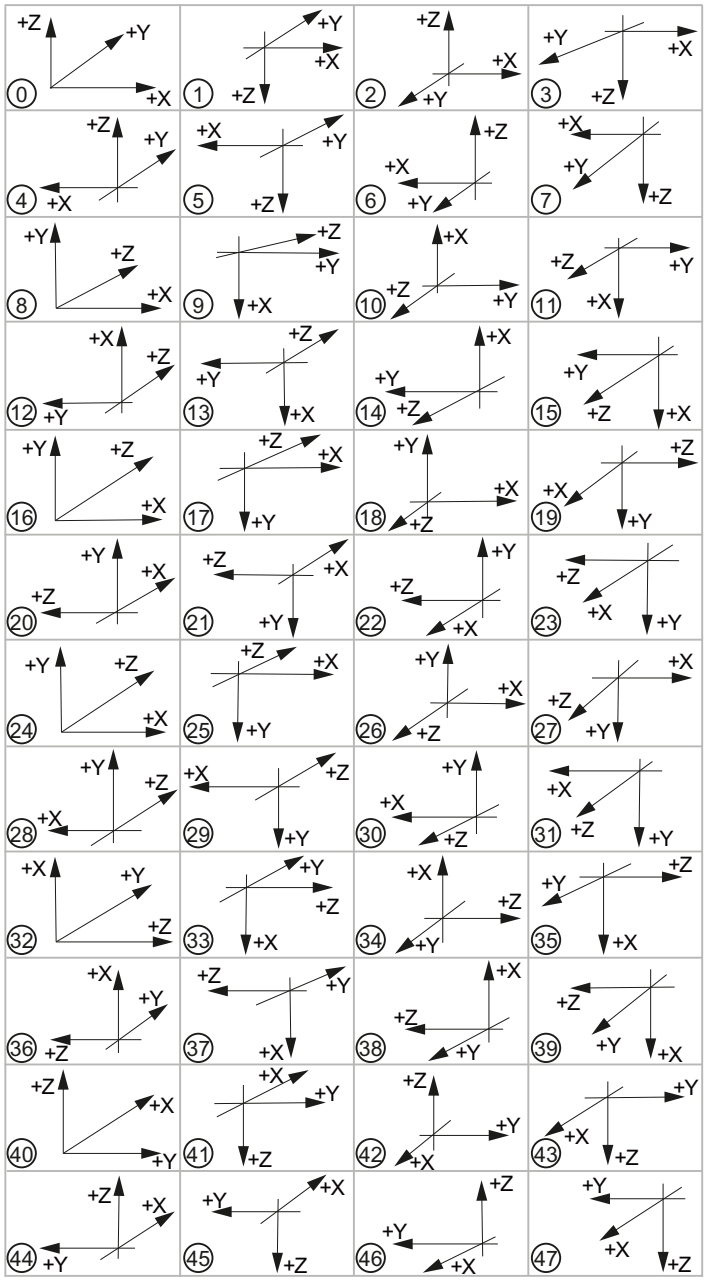


Figure 7-1 Position of the coordinate system

Technology extensions

MD52201 $MCS_TECHNOLOGY_EXTENSION	
Technology extension for combined machines with several technologies	
= 1	Additional settings for turning
= 2	Additional settings for the milling technology, e.g. lathe with milling technology MD52200 $MCS_TECHNOLOGY = 1 MD52201 $MCS_TECHNOLOGY_EXTENSION = 2

MD52212 $MCS_FUNCTION_MASK_TECH	
Cross-technology function mask	
Bit 0	Enable swivel
= 0	Swivel plane, swivel tool not enabled
= 1	Swivel plane, swivel tool enabled
Bit 1	No optimized travel along the software limit switches
= 0	No optimized travel along the software limit switches
= 1	Optimized travel along the software limit switches
Bit 2	Approach logic for stepped drill (ShopTurn)
= 0	
= 1	
Bit 3	Call block search cycle
= 0	The E_S_ASUP and F_S_ASUP cycles are not called in the block search cycle PROG_EVENT.SPF.
= 1	The E_S_ASUP and F_S_ASUP cycles are called in the block search cycle PROG_EVENT.SPF (default setting).
Bit 4	Approach logic using the cycle (ShopTurn)
= 0	
= 1	

MD52240 $MCS_NAME_TOOL_CHANGE_PROG	
Tool change program for G code steps	
= Program name	The associated program is called for the tool change.

Meaning of the axes

MD52206 $MCS_AXIS_USAGE[i]	
Meaning of the axes in the channel	
= 0	No special meaning
= 1	Tool spindle (driven tool)
= 2	Auxiliary spindle (driven tool)
= 3	Main spindle (turning)

MD52206 $MCS_AXIS_USAGE[i]	
= 4	Separate C axis of the main spindle (turning)
= 5	Counterspindle (turning)
= 6	Separate C axis of the counterspindle (turning)
= 7	Linear axis of the counterspindle (turning)
= 8	Tailstock (turning)
= 9	Back rest (turning)

Enter the direction of rotation for the rotary axes that are not configured in a tool carrier or a 5-axis transformation via the following channel-specific machine data:

MD52207 $MCS_AXIS_USAGE_ATTRIB[i]	
Attributes of the axes	
Bit 0	Rotation around 1st geometry axis (for rotary axes)
Bit 1	Rotation around 2nd geometry axis (for rotary axes)
Bit 2	Rotation around 3rd geometry axis (for rotary axes)
Bit 3	Displayed positive direction of rotation is counter-clockwise (for rotary axes C axis)
Bit 4	Displayed direction of rotation for M3 is counter-clockwise (for spindles)
Bit 5	Direction of rotation for M3 corresponds to minus rotary axis (for spindles)

Bit 5 must be set so that it matches the PLC signal for the spindle: DB380x.DBX2001.6

7.1.2 How to adapt the manufacturer cycles

Overview of cycles

The following cycles are also available in the cycle package for individual adaptation:

Designation	Used for	Description
CUST_800.SPF	Manufacturer cycle for the adaptation of the "Swivel plane" and the "Swivel tool" functions.	Manufacturer cycle CUST_800.SPF (Page 222)
CUST_832.SPF	Manufacturer cycle for the adaptation of the "High Speed Settings" function.	High Speed Settings (Advanced Surface) (Page 228)
CUST_MEACYC.SPF	Manufacturer cycle for the adaptation of measuring functions.	Manufacturer cycle CUST_MEACYC.SPF (Page 233)
CUST_TECHCYC.SPF	Manufacturer cycle for the adaptation of technology cycles.	ShopTurn: Setting up cycles for turning (Page 184)

Generating manufacturer cycles

General procedure:

1. Select the "Start-up" operating area.
2. Press the "System data" softkey.
3. Open the following directory:
   `NC data / Cycles / Standard cycles`
4. Select the manufacturer cycles: CUST_*.SPF
5. Press the "Copy" softkey.
6. Open the following directory:
   `NC data / Cycles / Manufacturer cycles`
7. Press the "Paste" softkey.

The cycles copied to the "Manufacturer cycles" directory can be modified.

See also

Standard cycle PROG_EVENT.SPF (Page 152)

7.1.3 Standard cycle PROG_EVENT.SPF

Rules for PROG_EVENT

PROG_EVENT.SPF is a standard cycle and cannot be changed (not accessible).

- Storage of a PROG_EVENT.SPF cycle that you have written under manufacturer of user cycles is **not** switched active.
- Manufacturer-specific "prog_events" must be reproduced by the following cycles, which, when available, are automatically called by the PROG_EVENT.
  - CYCPE1MA.SPF is called at the start of the internal prog_event.
  - CYCPE_MA.SPF is called at the end of the internal prog_event.

    Remark: CYCPE_MA.SPF should preferably be used.
- Block search:

  Default setting: The correction of tool change, spindle direction of rotation / speed, swivel axes is performed by the internal PROG_EVENT, the necessary machine data is appropriately preset. The manufacturer does not have to use a CYCPE_MA.SPF for the actions described above.

  Special features of the block search in conjunction with PROG_EVENT:

Machine data		Bit
MD11450	$MN_SEARCH_RUN_MODE	Bit 1=1
MD52212	$MCS_FUNCTION_MASK_TECH	Bit 3=1

Note

If $P_PROG_EVENT==5 is used in the cycpe_ma, no REPOSA may be programmed as this is already present in the prog_event, otherwise an error occurs for SSL with calculation.

7.1.4 Setting the simulation and simultaneous recording (option)

Machining technologies

Milling technology:

- Swivel head change

Milling on turning machines:

- Milling with geometry axes: TRANSMIT, TRACYL, TRAANG

Turning technology:

- Conventional turning with two geometry axes
- Three spindles: Main spindle, counterspindle, tool spindle
- Counterspindle slides, tailstock as NC axis
- B axis: Aligning turning tools in the tool spindle

Options



Software option

For the "Simultaneous recording" function, you require the option:
"Simultaneous recording (real-time simulation)"



Software option

You require the following option for further simulation settings:
"3D simulation 1 (finished part)"

Setting the coordinate system for simulation

Set the appropriate coordinate system in the channel via the following channel-specific machine data:

MD52000 $MCS_DISP_COORDINATE_SYSTEM	
Position of coordinate system (0 ... 47)	
= 0	For milling (example)
= 34	For turning (example)

See also: SectionSettings for activating cycles (Page 147)

Deactivating simulation:

1. Copy the file "slsimconfigsettings.xml" from the following directory:

   /**siemens**/sinumerik/hmi/appl

2. Store the file in the following directory:

   /**user**/sinumerik/hmi/cfg or /**oem**/sinumerik/hmi/cfg

   If the "slsimconfigsettings.xml" file already exists in the directory, add the entries from the Siemens file "slsimconfigsettings.xml".

The settings become effective only after restarting the system.

Simultaneous recording: Activating the runtime

Timers are provided as system variables in the Program runtime function. While the NCK-specific timers are always activated (for time measurements since the last control power-up), the channel-specific timers must be started via the following channel-specific machine data.

MD27860 $MC_PROCESS_TIMERMODE	
Activating the runtime measurement of the program	
Bit 0 = 1	Measurement of the total runtime for all part programs $AC_OPERATING_TIME
Bit 1 = 1	Measurement of the current program runtime
Bit 2 = 1	Measurement of machining time
Bit 3	n. a.
Bit 4 = 1	Measurement during active dry run feedrate
Bit 5 = 1	Measurement during program test
Bit 6 = 1	Delete condition $AC_CYCLE_TIME
Bit 7 = 1	Count condition $AC_CUTTING_TIME
Bit 8 = 1	Delete $AC_CYCLE_TIME with GOTOS
Bit 9 = 1	Measurement during override = 0%

7.1.5 Simulation for machines with 5 axes

Simulation

Notes for users:

- When simulating at milling machines the workpiece is always stationary in space and only the tools move - this always applies. As a consequence, simulation is independent of the kinematics used.
- For lathes, the only workpiece movement is the rotation around the main spindle axis.
- Before the blank definition in the G code program, by specifying the zero point and the swivel plane, a standard initial condition can be established for simulation.

  Example:

```
G54 G17 G90¶
CYCLE800(0,"TISCH",100000,57,0,0,0,0,0,0,0,0,0,-1,100,1)¶
WORKPIECE(,,,"BOX",112,0,-50,-80,0,0,155,100)¶
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$¶
T="NC-ANBOHRER_D16"¶
```

- All movements that are caused during swiveling by CYCLE800 are not displayed, i.e. at the start of the cycle, the tool is hidden and at the end of the cycle it is unhidden (displayed) in the new plane. This is also the reason that collisions cannot be detected, which for example are caused by a tool that has not been retracted.
- Programming in the machine coordinate system refers to the currently set zero point. "D0" should be programmed in order to hide the tool. In the simulation, these movements are only displayed as lines.

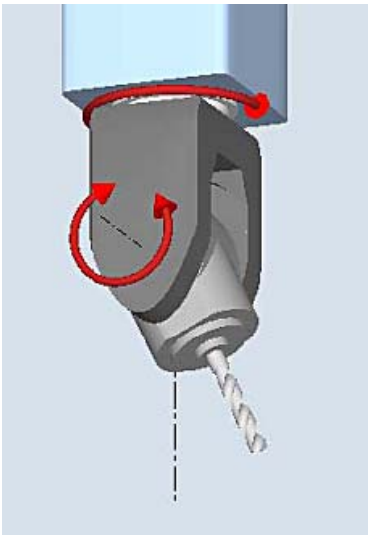
Boundary conditions

The following boundary conditions must be observed:

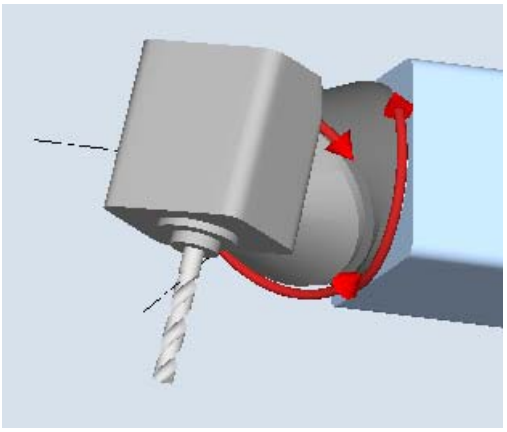
- All of the existing data records (tool carrier) are evaluated and must be correctly commissioned for correct simulation.
- Changes to the tool carrier or transformation data only become effective after Power On.
- The following rules apply for a kinematic change (tool carrier, reselection/deselection):
  - Rotary axes, which are no longer required, are set to the value 0, but are not removed.
  - Newly selected rotary axes are supported corresponding to the new kinematics.
- The simulation of mold building programs with extremely short block change times can take longer than machining itself, as the computation time distribution for this application is selected in favor of the machining and to the detriment of simulation.

Examples

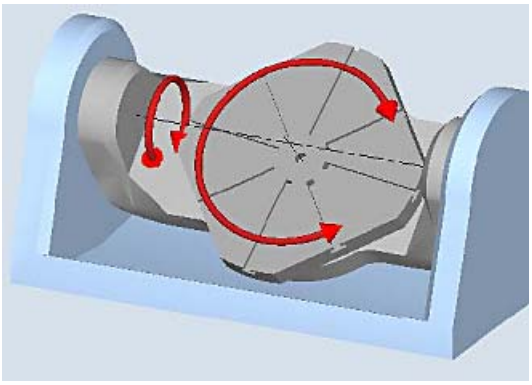
The following kinematics are supported:



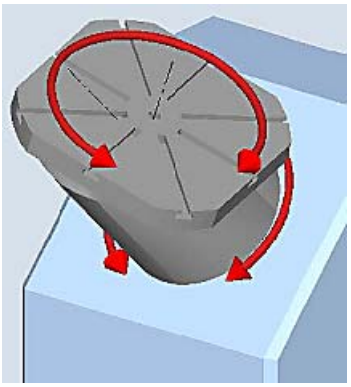
Swivel head 90°/90°



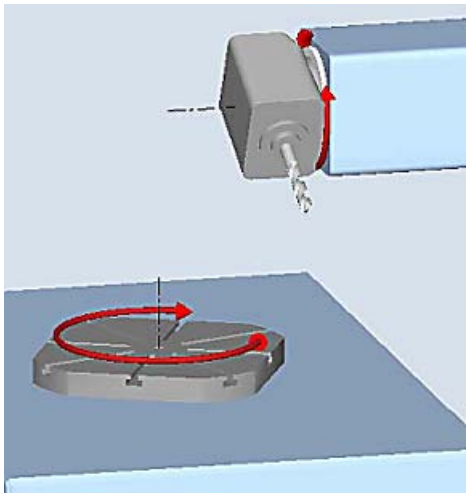
Swivel head 90°/45°



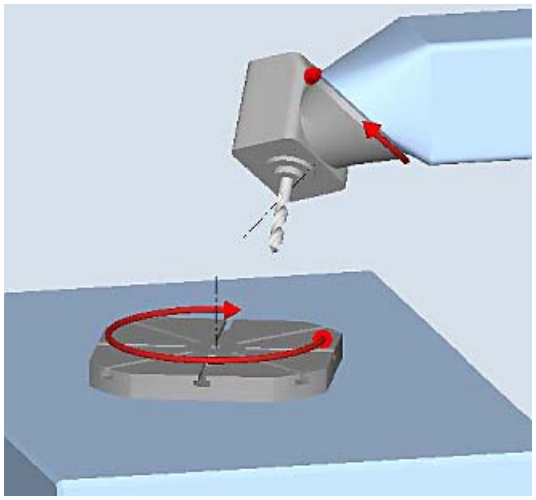
Swivel table 90°/90°



Swivel table 90°/45°



Swivel combination 90°/90°



Swivel combination 90°/45°

7.2 Drilling

7.2.1 Technology cycles for drilling

Drilling technology

You can set the drilling technology via the following channel-specific machine data and channel-specific setting data.

MD52216 $MCS_FUNCTION_MASK_DRILL	
Drilling function mask	
	Tapping cycle CYCLE84, technology input fields
= 0	Hide input fields
= 1	Display input fields
Bit 1	Tapping cycle CYCLE840, technology input fields
= 0	Hide input fields
= 1	Display input fields

SD55216 $SCS_FUNCTION_MASK_DRILL_SET	
Drilling function mask	
Bit 0	Tapping CYCLE84, determine spindle direction of rotation in the cycle
= 0	Do not reverse spindle direction of rotation
= 1	Reverse spindle direction of rotation
Bit 4	Tapping CYCLE840, monitoring of the machine data: MD31050 $MA_DRIVE_AX_RATIO_DENOM MD31060 $MA_DRIVE_AX_RATIO_NUMERA
= 0	No monitoring
= 1	Monitoring

Tapping (CYCLE84 and CYCLE840)

If the technology masks are hidden using the channel-specific machine data MD52216 $MCS_FUNCTION_MASK_DRILL, then the settings in the following channel-specific setting data are effective:

SD55481 $SCS_DRILL_TAPPING_SET_GG12[0]	
Exact stop response	
= 0	Exact stop response as before the cycle call (default value).
= 1	G601
= 2	G602
= 3	G603

SD55482 $SCS_DRILL_TAPPING_SET_GG21[0]	
Acceleration behavior	
= 0	Acceleration behavior as before the cycle call (default value).
= 1	SOFT
= 2	BRISK
= 3	DRIVE

SD55483 $SCS_DRILL_TAPPING_SET_GG24[0]	
Precontrol	
= 0	Precontrol as before the cycle call (default value).
= 1	FFWON
= 2	FFWOF

Tapping (CYCLE84)

SD55484 $SCS_DRILL_TAPPING_SET_MC[0]	
Spindle operation for MCALL	
= 0	For MCALL, reactivate spindle operation (default value).
= 1	For MCALL, remain in position-controlled spindle operation.

7.2.2 ShopTurn: Drilling centered

Requirement



Software option
You require the following software option in order to use the ShopTurn function:
"ShopMill/ShopTurn"

Tapping centered (CYCLE84)

If the technology masks are hidden using the channel-specific machine data MD52216 $MCS_FUNCTION_MASK_DRILL, then the settings in the following channel-specific setting data are effective:

SD55481 $SCS_DRILL_TAPPING_SET_GG12[1]	
Exact stop response	
= 0	Exact stop response as before the cycle call (default value).
= 1	G601
= 2	G602
= 3	G603

SD55482 $SCS_DRILL_TAPPING_SET_GG21[1]	
Acceleration behavior	
= 0	Acceleration behavior as before the cycle call (default value).
= 1	SOFT
= 2	BRISK
= 3	DRIVE

SD55483 $SCS_DRILL_TAPPING_SET_GG24[1]	
Precontrol	
= 0	Precontrol as before the cycle call (default value).
= 1	FFWON
= 2	FFWOF

7.3 Milling

7.3.1 Technology cycles for milling

Contour milling (CYCLE63)

Channel-specific setting data:

SD55214 $SCS_FUNCTION_MASK_MILL_SET	
Milling function mask	
Bit 0	Basic setting, milling in synchronous operation.
Bit 1	Angle of rotation referred to the center or corner of a rectangular pocket (POCKET3).
= 0	When dimensioning rectangular pockets using the corner point, the angle of rotation refers to this reference point.
= 1	When dimensioning rectangular pockets using the corner point, the angle of rotation refers to the center point of the pocket.
Bit 2	Depth calculation of the milling cycles, with or without safety clearance.
= 0	Depth calculation of the milling cycles is performed between the reference plane + safety clearance and the depth.
= 1	Depth calculation is performed without including the safety clearance. Bit 2 is effective in the following milling cycles: CYCLE61, CYCLE71, CYCLE76, CYCLE77, CYCLE79, CYCLE899, LONGHOLE, SLOT1, SLOT2, POCKET3, POCKET4.

SD55460 $SCS_MILL_CONT_INITIAL_RAD_FIN	
Finishing approach circle radius The radius of the approach circle during the finishing of contour pockets is affected.	
= –1	The radius is selected so that at the starting point the safety clearance to the finishing allowance is maintained (default setting).
= >0	The radius is selected so that at the starting point the value of this channel-specific setting data to the finishing allowance is maintained.

SD55461 $SCS_MILL_CONT_DIFF_TOOLRAD_MIN	
Contour pocket milling	
= 5	Smallest possible cutter radius deviation (default setting).

SD55462 $SCS_MILL_CONT_DIFF_TOOLRAD_MAX	
Contour pocket milling	
= 0.01	Largest possible cutter radius deviation in mm (default setting).

Channel-specific setting data for:

- Multi-edge (CYCLE79)
- Circle position pattern (HOLES2)
- Circumferential groove (SLOT2)

SD55230 $SCS_CIRCLE_RAPID_FEED	
Rapid feed in mm/min for positioning on a circular path between the circumferential grooves or the contour elements.	
= 100000	(default setting)

7.3.2 Cylinder surface transformation (TRACYL)

Requirement



Software option

You require the following software option in order to use this function:
"TRANSMIT and peripheral surface transformation".

Requirements on the machine:

- There must be at least one rotary axis at the machine.
- The milling tool must be radially oriented to the cylinder to be machined.

Cylinder surface transformation

The following groove machining operations can be performed with the Cylinder surface transformation function:

- Longitudinal grooves on cylindrical bodies
- Transverse grooves on cylindrical objects
- Grooves with any path on cylindrical bodies

The path of the grooves is programmed with reference to the unwrapped, level surface of the cylinder. The programming can be performed using straight line / circle, drilling or milling cycles or contour milling (free contour programming).

There are two variants of cylinder surface transformation, i.e.

1. with groove side offset (ON)
2. without groove side offset (OFF)

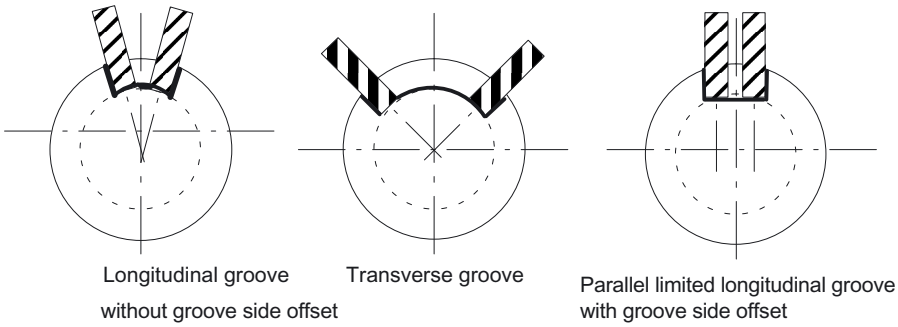
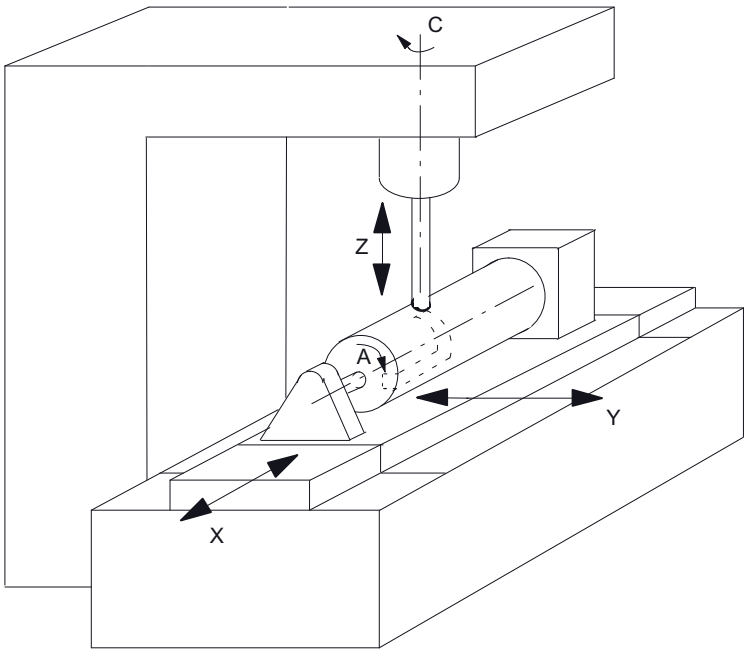


Figure 7-2 Grooves with and without groove side offset

7.3.3 Example: Axis configuration for milling machines

XYZ-AC axis configuration on a milling machine



X 1st axis of the machining plane parallel to the rotary axis
Y 2nd axis of the machining plane
Z Infeed axis (tool axis) perpendicular (radial) to the rotary axis
A Rotary axis
C Working spindle

Figure 7-3 Machining slots on a cylinder surface with X-A-Z kinematics

Two data sets with the following machine data are configured for the machine illustrated above:

MD20070 $MC_AXCONF_MACHAX_USED[4]	
Machine axis number valid in channel	
= 5	Number of axes in the channel

MD20080 $MC_AXCONF_CHANAX_NAME_TAB[i]	
Name of the axis in the channel	
[0] = X1	Default setting (geometry axes)
[1] = Y1	
[2] = Z1	
[3] = SP1	Default setting
[4] = A1	Default setting
[5] = C1	Default setting

General settings for the transformation:

MD10602 $MN_FRAME_GEOAX_CHANGE_MODE	
Frames when switching over geometry masks	
= 1	The current total frame (work offsets) is recalculated when switching over geometry axes (selecting/deselecting TRACYL).

Data set for the 1st transformation in the channel

MD24100 $MC_TRAFO_TYPE_1	
Definition of transformation 1 in the channel: 0: No transformation ... As of 256: TRANSMIT transformation As of 512: TRACYL transformation	
= 512	TRACYL transformation **without** groove side offset
= 514	TRACYL with additional linear axis and **with** groove side offset

MD24110 $MC_TRAFO_AXES_IN_1		
Axis assignment for the 1st transformation in the channel		
[0] = 3	Channel axis: Infeed axis (tool axis) perpendicular (radial) to the rotary axis	Z
[1] = 4	Channel axis: Rotary axis	A
[2] = 1	Channel axis: 1st axis of the machining plane parallel to the rotary axis	X
[3] = 2	Channel axis: 2nd axis of the machining plane	Y

MD24120 $MC_TRAFO_GEOAX_ASSIGN_TAB_1		
Assignment of the geometry axes to channel axes for transformation 1.		
[0] = 1	Channel axis: 1st geometry axis	X
[1] = 4	Channel axis: 2nd geometry axis	A
[2] = 3	Channel axis: 3rd geometry axis	Z

MD24800 $MC_TRACYL_ROT_AX_OFFSET_1	
= 0	Offset of rotary axis for the 1st TRACYL transformation.

MD24805 $MC_TRACYL_ROT_AX_FRAME_1	
= 1	Axial offset of rotary axis for the 1st TRACYL transformation.

MD24810 $MC_TRACYL_ROT_SIGN_IS_PLUS_1	
= 1	Sign of the rotary axis for the 1st TRACYL transformation.

MD24820 $MC_TRACYL_BASE_TOOL_1[i]	
Vector of base tool for the 1st TRACYL transformation in XYZ	
[0] = 0	
[1] = 0	
[2] = 0	

Data set for the 2nd transformation in the channel

MD24100 $MC_TRAFO_TYPE_1	
Definition of transformation 1 in channel	
= 513	TRACYL transformation **with** groove side offset

MD24210 $MC_TRAFO_AXES_IN_2		
[0] = 3	Channel axis: Infeed axis perpendicular (radial) to rotary axis	Z
[1] = 4	Channel axis: Rotary axis	A
[2] = 1	Channel axis: 1st axis of the machining plane parallel to the rotary axis	X
[3] = 2	Channel axis: 2nd axis of the machining plane	Y

MD24220 $MC_TRAFO_GEOAX_ASSIGN_TAB_1		
[0] = 1	Channel axis: 1st geometry axis	X
[1] = 4	Channel axis: 2nd geometry axis	A
[2] = 3	Channel axis: 3rd geometry axis	Z

MD24850 $MC_TRACYL_ROT_AX_OFFSET_2	
= 0	Offset of rotary axis for the 2nd TRACYL transformation

MD24855 $MC_TRACYL_ROT_SIGN_IS_FRAME_2	
= 1	Axial offset for the 1st TRACYL transformation

MD24860 $MC_TRACYL_ROT_SIGN_IS_PLUS_2	
= 1	Sign of rotary axis for the 2nd TRACYL transformation

MD24870 $MC_TRACYL_BASE_TOOL[i]	
Vector of base tool for the 2nd TRACYL transformation in XYZ	
[1] = 0	
[2] = 0	
[3] = 0	

Note

For both data sets, you can use any transformations from all available transformations (MD24100 $MC_TRAFO_TYPE_1, MD24200 $MC_TRAFO_TYPE_2, etc.).

The two data sets need not be directly next to each other.

The 1st data set must always be used for "Cylinder surface transformation **without** groove side offset" (= 512) and the 2nd data set for "Cylinder surface transformation **with** groove side offset" (=513).

7.3.4 ShopMill: Setting up cycles for milling

Requirement



Software option

You require the following software option in order to use the ShopMill function:

"ShopMill/ShopTurn"

In order that simulation and simultaneous recording are displayed without any errors, set the machine data as described in the following section:

Setting the simulation and simultaneous recording (option) (Page 153)

Milling under ShopMill

In order that the direction of rotation is correctly displayed in the ShopMill user interface, and when programming ShopMill functions, the correct direction of rotation is executed, you must make some settings that are coordinated with one another.

You must align these settings to the actual direction of rotation of the axis at the machine.

MD52229 $MCS_ENABLE_QUICK_M_CODES	
Enable quick M commands	
= 0H (default setting)	
Bit 0	Coolant OFF.
Bit 1	Coolant 1 ON.
Bit 2	Coolant 2 ON.
Bit 3	Coolant 1 and 2 ON.

MD52230 $MCS_M_CODE_ALL_COOLANTS_OFF	
M code for all coolants OFF.	
= 9	You define the M function to switch off the coolant that is output when the tool is changed.

MD52231 $MCS_M_CODE_COOLANT_1_ON	
M code for coolant 1 ON.	
= 8	You define the M function for coolant 1 that is output when the tool is changed.

MD52232 $MCS_M_CODE_COOLANT_2_ON	
M code for coolant 2 ON.	
= 7	You define the M function for coolant 2 that is output when the tool is changed.

MD52233 $MCS_M_CODE_COOLANT_1_AND_2_ON	
M code for both coolants ON.	
= -1	You define the M function for coolant 1 and 2 that is output when the tool is changed.

MD52281 $MCS_TOOL_MCODE_FUNC_ON[i]	
M code for tool-specific function ON.	
= -1	M function is output (default setting): If both M commands of a function "= -1", the corresponding field is not displayed on the interface.
[0]	M code for tool-specific function 1 ON.
[1]	M code for tool-specific function 2 ON.
[2]	M code for tool-specific function 3 ON.
[3]	M code for tool-specific function 4 ON.

MD52282 $MCS_TOOL_MCODE_FUNC_OFF[i]	
M code for tool-specific function OFF.	
= -1	M function is output (default setting): If both M commands of a function "= -1", the corresponding field is displayed on the interface.
[0]	M code for tool-specific function 1 OFF.
[1]	M code for tool-specific function 2 OFF.
[2]	M code for tool-specific function 3 OFF.
[3]	M code for tool-specific function 4 OFF.

Channel-specific setting data:

SD55212 $SCS_FUNCTION_MASK_TECH_SET	
Cross-technology function mask.	
= 6H	Default setting
Bit 0	Tool preselection active. The next tool is prepared directly after a tool change. **Note:** For a turret, the setting data must be set to "0".
Bit 1	Automatic calculation of the thread depth for metric threads.
Bit 2	Take the thread diameter and thread depth from the table.

7.4 Turning

7.4.1 Technology cycles for turning

Thread-cutting (CYCLE99)

During program runtime, the master spindle can be the main spindle or the counterspindle. Bit 3 must be set accordingly in the array index [i=channel axis number]:

MD52207 $MCS_AXIS_USAGE[i]	
Direction of rotation of the master spindle [channel axis number]	
Bit 3	Direction of rotation of the C axis normal or in the opposite direction
= 0	Normal (M3 is +C)
= 1	Opposite (M3 is -C)

Set setting data for:

- Contour grooving CYCLE930
- Contour turning CYCLE950, CYCLE952
- Stock removal, corner CYCLE951

SD55500 $SCS_TURN_FIN_FEED_PERCENT	
Enter the finishing feedrate for complete machining, roughing and finishing. The percentage of the value corresponds to that entered under parameter F (feedrate).	
= 100	100 % finishing feedrate

SD55510 $SCS_TURN_GROOVE_DWELL_TIME	
Dwell time, which is necessary between grooving and retracting for grooving technology. Tool clearance time during grooving at the base.	
= > 0	Dwell time in seconds
= < 0	Dwell time in spindle revolutions

SD55580 $SCS_TURN_CONT_RELEASE_ANGLE	
Angle, through which the tool is lifted from the contour for contour turning, roughing.	
= 45	Retraction angle of 45 degrees

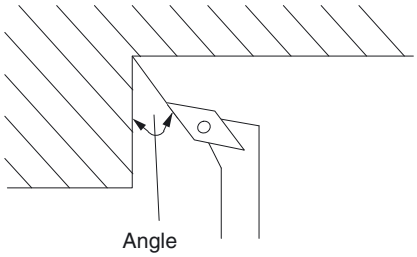


Figure 7-4 Angle of retraction

SD55581 $SCS_TURN_CONT_RELEASE_DIST	
Amount, by which the tool is lifted in both axes when roughing a contour. This applies for stock removal, plunge cutting and plunge turning.	
= 1	1 mm or 1 inch retraction distance

SD55582 $SCS_TURN_CONT_TRACE_ANGLE	
The angle between cutting edge and contour as of which rounding is performed on the contour during contour turning in order to remove residual material. If the angle of the residual material is greater than that specified in the setting data, the tool will round the contour.	
= 5	5 degree angle

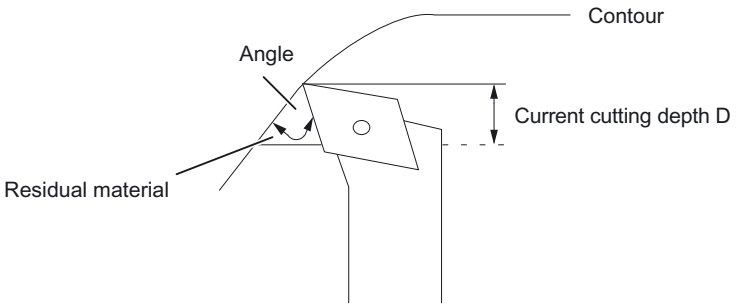


Figure 7-5 Angle of the residual material

SD55583 $SCS_TURN_CONT_VARIABLE_DEPTH	
Percentage of the changing cutting depth when turning a contour You can select changing the cutting depth for stock removal and for removing residual material.	
= 20	20 % variable cutting depth

SD55584 $SCS_TURN_CONT_BLANK_OFFSET	
Distance from the blank at which G0 is switched over to G1 during contour turning to compensate for any blank allowances. This applies for stock removal, plunge cutting and plunge turning.	
= 1	1 mm or 1 inch blank allowance

SD55585 $SCS_TURN_CONT_INTERRUPT_TIME	
Time for the feedrate interruption for contour turning. This applies for stock removal, plunge cutting and plunge turning.	
= > 0	Interruption time in seconds
= < 0	Interruption time in revolutions
= 0	No interruption

Note

The channel-specific setting data SD55585 is only evaluated if SD55586 $SCS_TURN_CONT_INTER_RETRACTION = 0.

SD55586 $SCS_TURN_CONT_INTER_RETRACTION	
Retraction distance for contour turning for feedrate interruption. This applies for stock removal, plunge cutting and plunge turning.	
= > 0	Retraction distance at feed interruption SD55585 $SCS_TURN_CONT_INTERRUPT_TIME has no effect.
= 0	No retraction distance

SD55587 $SCS_TURN_CONT_MIN_REST_MAT_AX1	
Limit for removal of residual material in the direction of axis 1 (for G18 Z) min. This applies for stock removal, plunge cutting and plunge turning.	
50	50 % min. differential dimension for the residual material machining, axis 1

7.4.2 Example: Residual material machining

Requirement



Software option

You require the following software option in order to use this function:

"Residual material identification and machining".

Residual material machining, axis 1

If the limit is set to 50% and the final machining allowance is 0.5 mm, any residual material thinner than 0.25 mm is not machined in a separate machining step but is removed during finishing.

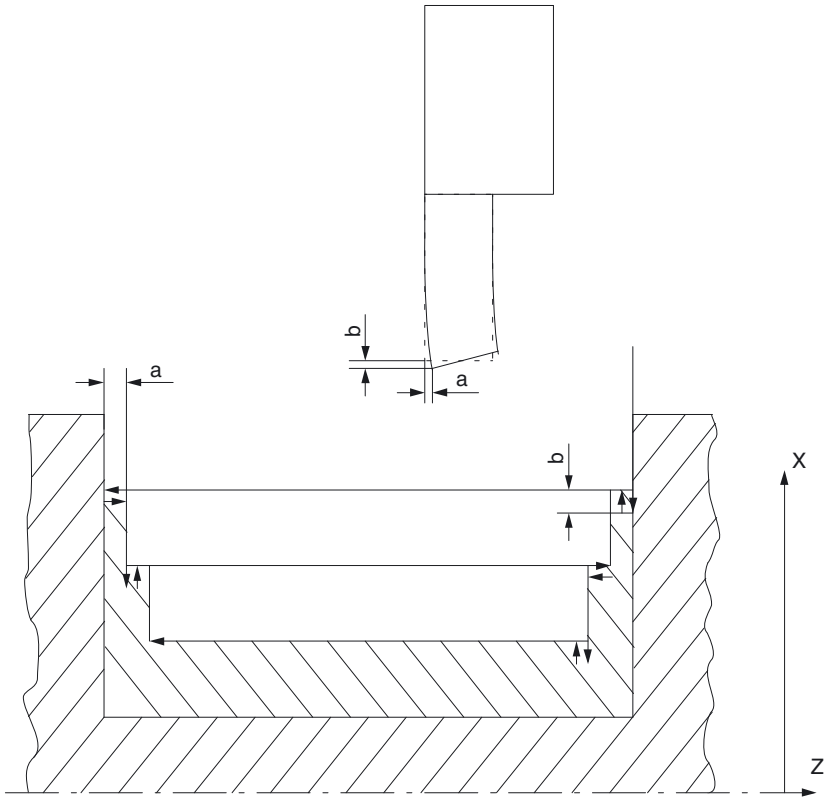
SD55588 $SCS_TURN_CONT_MIN_REST_MAT_AX2	
Limit value for removing residual material in the direction of axis 2 (for G18 X). This applies for stock removal, plunge cutting and plunge turning.	
50	50 % min. differential dimension for the residual material machining, axis 2

Residual material machining, axis 2

If the limit is set to 50% and the final machining allowance is 0.5 mm, any residual material thinner than 0.25 mm is not machined in a separate machining step but is removed during finishing.

As the tool bends during plunge turning, the tool cannot travel right up to the contour during stock removal. The lateral distance to the last cut by which the next cut is shortened is specified in the following channel-specific setting data.

SD55595 $SCS_TURN_CONT_TOOL_BEND_RETR	
Retraction distance because of tool bending	
0.1	0.1 mm or 0.1 inch retraction distance



a SD55595: Distance to the last cut

b SD55596: Retraction between plunge cutting and stock removal

Figure 7-6 Set SD55595 and SD55596

As the tool bends during plunge turning, the tool would make an excessively deep cut during stock removal. The retraction distance of the tool between plunge cutting and stock removal is specified in the following channel-specific setting data:

SD55596 $SCS_TURN_CONT_TURN_RETRACTION	
Retraction depth before turning	
= 0.1	0.1 mm or 0.1 inch retraction depth

7.4.3 Example: Axis configuration for lathes

Lathe with milling tools

If driven milling tools are available on a lathe, then the following functions can also be set-up on this machine:

- Cylinder surface transformation (TRACYL) (Page 175)
- End face machining (TRANSMIT) (Page 178)

Lathe with X and Z axes, main and tool spindle

For example, for a lathe with X and Z axes, main spindle (C1) and tool spindle (SP1), you can configure the following channel-specific machine data:

MD20080 $MC_AXCONF_CHANAX_NAME_TAB[i]	
Name of the axis in the channel	
[0] = X1	Channel axis X1
[1] = Z1	Channel axis Z1
[2] = C1	Main spindle C1
[3] = SP1	Tool spindle SP1

Lathe with X and Z axes, main and tool spindle and counterspindle

For example, for a lathe with X and Z axes, main spindle (C1), tool spindle (SP1) and counterspindle (C2), you can configure the following machine data:

MD20080 $MC_AXCONF_CHANAX_NAME_TAB[i]	
Name of the axis in the channel	
[0] = X1	Channel axis X1
[1] = Z1	Channel axis Z1
[2] = C1	Main spindle C1
[3] = SP1	Tool spindle SP1
[5] = C2	Counterspindle C2

Lathe with X and Z axes, main and tool spindle and Y axis

For example, for a lathe with X, Z and Y axes, main spindle (C1) and tool spindle (SP1), you can configure the following machine data:

MD20080 $MC_AXCONF_CHANAX_NAME_TAB[i]	
Name of the axis in the channel	
[0] = X1	Channel axis X1
[1] = Z1	Channel axis Z1
[2] = C1	Main spindle C1
[3] = SP1	Tool spindle SP1
[5] = Y1	Channel axis Y1

7.4.4 Cylinder surface transformation (TRACYL)

Requirement



Software option

You require the following software option in order to use this function:

"TRANSMIT and peripheral surface transformation"

Function

Using the cylinder surface transformation function (TRACYL), you can machine the peripheral surface of a turned part.

General settings for the cylinder surface transformation

MD10602 $MN_FRAME_GEOAX_CHANGE_MODE	
Frames when switching over geometry axes.	
= 1	The current total frame (work offsets) is recalculated when switching over geometry axes (selecting and deselecting TRACYL).

MD24040 $MC_FRAME_ADAPT_MODE	
Adapting the active frames	
7H	Default setting
Bit 0 = 1	Rotations in active frames that rotate coordinate axes for which there are no geometry axes, are deleted from the active frames.
Bit 1 = 1	Shearing angles in the active frames are made orthogonal.
Bit 2 = 1	Scaling factors of all geometry axes in the active frames are set to 1.

MD28082 $MC_MM_SYSTEM_FRAME_MASK	
Bit 6 = 1	Configuration of channel-specific system frames, which are included in the channel calculation.

When setting up the functions, you can take the following channel-specific machine data into account:

MD24300 $MC_TRAFO_TYPE_3	
Cylinder surface transformation, main spindle: Transformation 3	
= 512	Without groove side offset (without Y axis)
= 513	With groove side offset (with Y axis):
= 514	With groove side offset and Y offset

MD24400 $MC_TRAFO_TYPE_4	
Cylinder surface transformation, counterspindle: Transformation 4	
= 512	Without groove side offset (without Y axis)
= 513	With groove side offset (with Y axis)
= 514	With groove side offset and Y offset

Note

You must also set up further machine data for the individual transformations.

Cylinder surface transformation without groove side offset

MD24300 $MC_TRAFO_TYPE_3	
Definition of the 3rd transformation in the channel	
= 512	TRACYL main spindle.

MD24310 $MC_TRAFO_AXES_IN_3[i]	
Axis assignment for transformation 3	
[0] = 1	Perpendicular to the rotary axis X1
[1] = 3	Rotary axis (main spindle) C1
[2] = 2	Parallel to the rotary axis Z1

MD24320 $MC_TRAFO_GEOAX_ASSIGN_TAB_3[i]	
Assignment of geometry axes to channel axes for transformation 3.	
[0] = 1	1st channel axis X
[1] = 3	2nd channel axis Y
[2] = 2	3rd channel axis Z

MD24800 $MC_TRACYL_ROT_AX_OFFSET_1	
Offset of the rotary axis for the 1st TRACYL transformation.	
= 0	

MD24805 $MC_TRACYL_ROT_AX_FRAME_1	
Axial offset of the rotary axis is taken into account during TRACYL.	
= 2	

MD24810 $MC_TRACYL_ROT_SIGN_IS_PLUS_1	
Sign of the rotary axis for the 1st TRACYL transformation.	
= 1	

MD24820 $MC_TRACYL_BASE_TOOL_1[i]	
Vector of the base tool for the 1st TRACYL transformation.	
[0] = 0	Index i assumes values 0, 1, 2 for the 1st to 3rd geometry axis. Programmed tool length offsets are added to the base tool.
[1] = 0	
[2] = 0	

Cylinder surface transformation with groove side offset

MD24300 $MC_TRAFO_TYPE_3	
Definition of the 3rd transformation in the channel	
= 513	TRACYL main spindle

MD24310 $MC_TRAFO_AXES_IN_3[i]	
Axis assignment for transformation 3	
[0] = 1	Perpendicular to the rotary axis X1
[1] = 3	Rotary axis (main spindle) C1
[2] = 2	Parallel to the rotary axis Z1
[3] = 6	Parallel to the cylinder surface and perpendicular to the rotary axis ZC

MD24320 $MC_TRAFO_GEOAX_ASSIGN_TAB_3[i]	
Assignment of geometry axes to channel axes for transformation 3.	
[0] = 1	1st channel axis X
[1] = 3	2nd channel axis Y
[2] = 2	3rd channel axis Z

MD24800 $MC_TRACYL_ROT_AX_OFFSET_1	
Offset of the rotary axis for the 1st TRACYL transformation.	
= 0	

MD24805 $MC_TRACYL_ROT_AX_FRAME_1	
Axial offset of the rotary axis is taken into account during TRACYL.	
= 2	

MD24810 $MC_TRACYL_ROT_SIGN_IS_PLUS_1	
Sign of the rotary axis for the 1st TRACYL transformation.	
= 1	

MD24820 $MC_TRACYL_BASE_TOOL_1[i]	
Vector of the base tool for the 1st TRACYL transformation.	
[0] = 0	
[1] = 0	
[2] = 0	

7.4.5 End face machining (TRANSMIT)

Requirement



Software option

You require the following software option in order to use this function:
"TRANSMIT and peripheral surface transformation"

For general settings for transformations, please refer to Chapter:

Cylinder surface transformation (TRACYL) (Page 175)

Function

The end faces of a turned part are machined with the end face machining function (TRANSMIT).

You can make additional settings in the following channel-specific machine data:

MD24100 $MC_TRAFO_TYPE_1	
End face machining, main spindle: Transformation 1	
= 256	Machining without Y axis
= 257	Machining with Y axis

MD24200 $MC_TRAFO_TYPE_2	
End face machining, counterspindle: Transformation 2	
= 256	Machining without Y axis
= 257	Machining with Y axis

MD24110 $MC_TRAFO_AXES_IN_1[i]	
Axis assignment for the 1st transformation in the channel	
[0] = 1	Perpendicular to the rotary axis X1
[1] = 3	Rotary axis (main spindle) C1
[2] = 2	Parallel to the rotary axis Z1

MD24120 $MC_TRAFO_GEOAX_ASSIGN_TAB_1[i]	
Assignment of the geometry axes to channel axes for transformation 1	
[0] = 1	1st channel axis X
[1] = 3	2nd channel axis Y
[2] = 2	3rd channel axis Z

MD24900 $MC_TRANSMIT_ROT_AX_OFFSET_1	
= 0	Offset of the rotary axis for the 1st TRANSMIT transformation.

MD24905 $MC_TRANSMIT_ROT_AX_FRAME_1	
= 2	Axial offset of the rotary axis is taken into account during TRANSMIT 1.

MD24910 $MC_TRANSMIT_ROT_SIGN_IS_PLUS_1	
= 0	Sign of the rotary axis for the 1st TRANSMIT transformation.

MD24911 $MC_TRANSMIT_POLE_SIDE_FIX_1	
= 1	Limitation of working range in front of / behind the pole, 1st TRANSMIT.

MD24920 $MC_TRANSMIT_BASE_TOOL_1[i]	
Vector of the base tool for the 1st TRANSMIT transformation.	
[0] = 0	Index i assumes values 0, 1, 2 for the 1st to 3rd geometry axis. Programmed tool length offsets are added to the base tool.
[1] = 0	
[2] = 0	

TRANSMIT with real Y axis

MD24100 $MC_TRAFO_TYPE_1	
= 257	Definition of transformation 1 in the channel: TRANSMIT main spindle.

MD24110 $MC_TRAFO_AXES_IN_1[i]	
Axis assignment for the 1st transformation in the channel.	
[0] = 1	Perpendicular to the rotary axis X1
[1] = 3	Rotary axis C1
[2] = 2	Parallel to the rotary axis Z1

MD24120 $MC_TRAFO_GEOAX_ASSIGN_TAB_1[i]	
Assignment of the geometry axes to channel axes for transformation 1.	
[0] = 1	1st channel axis X
[1] = 3	2nd channel axis Y
[2] = 2	3rd channel axis Z

MD24900 $MC_TRANSMIT_ROT_AX_OFFSET_1	
= 0	Offset of the rotary axis for the 1st TRANSMIT transformation

MD24905 $MC_TRANSMIT_ROT_AX_FRAME_1	
= 2	Axial offset of the rotary axis is taken into account during TRANSMIT 1.

MD24910 $MC_TRANSMIT_ROT_SIGN_IS_PLUS_1	
= 0	Sign of the rotary axis for the 1st TRANSMIT transformation.

MD24911 $MC_TRANSMIT_POLE_SIDE_FIX_1	
= 1	Limitation of working range in front of / behind the pole, 1st TRANSMIT.

MD24920 $MC_TRANSMIT_BASE_TOOL_1[i]	
Vector of the base tool for the 1st TRANSMIT transformation.	
[0] = 0	
[1] = 0	
[2] = 0	

References

Function Manual, Extended Functions; Kinematic Transformation (M1): TRANSMIT

7.4.6 Inclined axis (TRAANG)

Requirement



Software option

You require the following software option in order to use this function:

"Inclined axis"

Function

If the lathe has an inclined Y axis (i.e. this axis is not perpendicular to axes X and Z), you can still completely program machining operations in Cartesian coordinates. The control uses the "Inclined axis function (TRAANG)" to transform the Cartesian coordinates to the motion of the inclined axis.

You still have to set up the "Inclined axis function (TRAANG)" via machine data.

See also

Function Manual, Extended Functions; Kinematic Transformations (M1)

Example for a turning machine

For example, for a lathe with X and Z axes and inclined Y axis, main spindle (C1) and tool spindle (SP1), you must configure the following machine data:

MD20050 $MC_AXCONF_GEOAX_ASSIGN_TAB[i]	
Assignment of geometry axis to channel axis.	
[0] = 1	1st real geometry axis X axis.
[1] = 0	2nd real geometry axis Y axis not available.
[2] = 2	3rd real geometry axis, Z axis.

MD20110 $MC_RESET_MODE_MASK	
Bit 0 = 1	TRAANG is retained after ramp-up.
Bit 7 = 0	

MD20112 $MC_START_MODE_MASK	
Bit 7 = 1	TRAANG is retained after "Cycle start".

MD20118 $MC_GEOAX_CHANGE_RESET	
= 1	Permit automatic change of the geometry axes.

MD20140 $MC_TRAFO_RESET_VALUE	
= 5	TRAANG always active after reset.

MD20144 $MC_TRAFO_MODE_MASK	
Bit 0 = 1	TRAANG runs in the background (persistent) and is not shown on the user interface.

MD20070 $MC_AXCONF_MACHAX_USED[4]	
= 5	Channel axis Y1 = 5th machine axis.

MD20080 $MC_AXCONF_CHANAX_NAME_TAB[i]	
Name of the axis in the channel	
[0] = X1	1st channel axis X1 (default setting)
[1] = Z1	2nd channel axis Z1 (default setting)
[2] = C1	3rd channel axis C1 (main spindle)
[3] = SP1	4th channel axis SP1 (tool spindle)
[4] = Q1	5th channel axis Q1 (default setting)
[5] = Y1	6th channel axis Y1 (default setting)
[6] = Z2	7th channel axis Z2 (default setting)
[7] = C2	8th channel axis C2 (default setting)

Data set for inclined axis:

MD24430 $MC_TRAFO_TYPE_5	
= 1024	Transformation 5: TRAANG

MD24432 $MC_TRAFO_AXES_IN_5[i]	
Axis assignment for transformation 5.	
[0] = 5	1st transformation axis = channel axis Y1
[1] = 1	2nd transformation axis = channel axis X1
[2] = 2	3rd transformation axis = channel axis Z1

MD24434 $MC_TRAFO_GEOAX_ASSIGN_TAB_5[i]	
Assignment of geometry axes to channel axes for transformation 5.	
[0] = 1	1st axis = channel axis X1
[1] = 5	2nd axis = channel axis Y1
[2] = 2	3rd axis = channel axis Z1

MD24436 $MC_TRAFO_INCLUDES_TOOL_5	
= 0	Tool handling with active transformation 5.

MD24700 $MC_TRAANG_ANGLE_1	
= 55	Angle between 1st and 2nd transformation axis. Data set for linking (TRACON) of end face machining on main spindle (TRANSMIT) and inclined axis (TRAANG).

Data set for linking (TRACON) of end face machining on main spindle (TRANSMIT) and inclined axis (TRAANG):

MD24440 $MC_TRAFO_TYPE_6	
= 8192	Type of transformation that is available as sixth in the channel.

MD24444 $MC_TRAFO_GEOAX_ASSIGN_TAB_6[i]	
Assignment of geometry axes to channel axes for transformation 6.	
[0] = 1	1st axis = channel axis X1
[1] = 3	2nd axis = channel axis Y1
[2] = 2	3rd axis = channel axis Z1

MD24995 $MC_TRACON_CHAIN_1[i]	
Transformation linking	
[0] = 1	Number of the transformation: TRANSMIT (main spindle) for linking.
[1] = 5	Number of the transformation: TRAANG for linking. Data set with linking (TRACON) of cylinder surface transformation on main spindle (TRACYL) and inclined axis (TRAANG).

Data set for linking (TRACON) of cylinder surface transformation on main spindle (TRACYL) and inclined axis (TRAANG):

MD24450 $MC_TRAFO_TYPE_7	
= 8192	Type of transformation 7 in the TRACON channel.

MD24454 $MC_TRAFO_GEOAX_ASSIGN_TAB_7[i]	
Assignment of geometry axes to channel axes for transformation 7.	
[0] = 1	1st axis = channel axis X1
[1] = 3	2nd axis = channel axis Y1
[2] = 2	3rd axis = channel axis Z1

MD24996 $MC_TRACON_CHAIN_2[i]	
Transformation linking	
[0] = 3	Number of the TRACYL transformation (main spindle) for linking.
[1] = 5	Number of the TRAANG transformation for linking.

7.4.7 ShopTurn: Setting up cycles for turning

Manufacturer cycle CUST_TECHCYC.SPF

In cycle CUST_TECHCYC.SPF, the function markers (_M1: to _M142) are prepared and documented. The manufacturer cycle CYC_TECHCUST.SPF is called by the ShopTurn cycles.

Requirement



Software option

You require the following software option in order to use the ShopTurn function:

"ShopMill/ShopTurn"

Making adaptations

Adapt the cycle if you want to perform one of the following actions:

- Switch between spindle and C axis mode for the main spindle or counterspindle.

  Markers _M1, _M2, _M21, _M22
- Clamp or release rotary axes (main spindle / counterspindle).

  Markers _M3, _M4, _M23, _M24
- Open, close, flush chuck (main spindle / counterspindle).

  Markers _M5 to _M8, _M25 to _M29
- Engage/disengage driven tool (make/break connection to drive).

  Markers _M41, _M42
- Configure special functions for switchover between machining planes.

  Markers _M61 to _M68

  (You do not have to make any settings for cylinder surface transformation or end face machining with the C axis.)
- Position receptacle for cut-off, move out or in.

  Markers _M100, _M101, _M102

- Configure special functions for tool changing.

  Markers _M110, _M111, _M112

  (These special functions are called after the T command is output.)
- Change default settings for the coupling of the main spindle and counterspindle.

  Marker _M120
- Set special features for the program start or program end.

  Markers _M131, _M135, _M13

Direction of rotation of spindle

The direction of rotation is set in the following machine data:

MD52207 $MCS_AXIS_USAGE_ATTRIB[5]

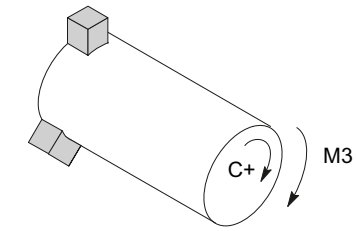
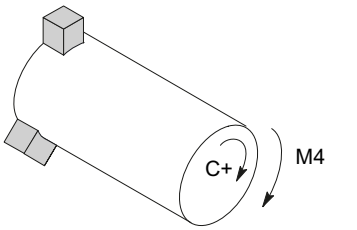
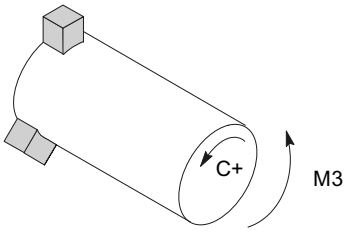
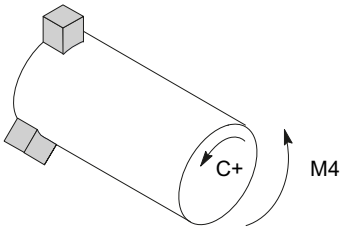
The direction of spindle rotation (M3/M4) is assigned to the positive rotational direction of the C axis via interface signal DB380x.DBX2001.6 (where n = index of the relevant C axis). Bit 4 defines whether M3 and C+ rotate in the same direction (=0) or in opposite directions (=1).

The direction of rotation of an NC rotary axis is set via the following machine data:

MD32100 $MA_AX_MOTION_DIR	
Travel direction	
-1	Direction reversal
0, 1	No direction reversal

As a result, the following setting options are obtained for the main spindle. However, the settings for the machine data depend on the direction from which the coordinate axis is viewed. The settings for the direction of rotation as well as the interface signal DB380x.DBX2001.6 must be observed under all circumstances.

NOTICE
Under all circumstances, ensure that bit 5 in MD52207 is identical to DB380x.DBX2001.6!

Main spindle direction of rotation	52207[C-Ax] Bit 3 =	52207[Spnd] Bit 4 =	52207[Spnd] Bit 5 =	DB380x. DBX2001.6
	0	0	0	
	1	0	1	
	1	1	0	
	0	1	1	

The setting of MD52207 assumes that the viewing direction is toward the negative coordinate axis. If, on the other hand, you are looking toward the positive coordinate axis, you need to reverse the values, i.e. swap "0" and "1".

Note

The setting of MD52207[i] is only relevant for the display in the ShopTurn user interface, not for correct machining on the machine.

Further settings for ShopTurn

MD52241 $MCS_SPINDLE_CHUCK_TYPES[ ]	
Spindle jaw type.	
[0]	Main spindle
[1]	Counterspindle
= 0	Clamping, outer
= 1	Clamping, inner

MD52242 $MCS_MAIN_SPINDLE_PARAMETER[ ]	
Main spindle parameters.	
[0]	Chuck dimensions
[1]	Stop dimensions
[2]	Jaw dimensions

MD52243 $MCS_SUB_SPINDLE_PARAMETER[ ]	
Parameters for counterspindle.	
[0]	Chuck dimensions
[1]	Stop dimensions
[2]	Jaw dimensions

MD52246 $MCS_TAILSTOCK_DIAMETER	
Tailstock diameter	
= 0	

MD52247 $MCS_TAILSTOCK_LEGTH	
Tailstock length	
= 0	

The M code, e.g. M34 or M1 = 34 for the spindle chuck is defined in the following machine data. The manufacturer cycle CUST_TECHCYC.SPF takes the M functions from the following machine data:

MD52250 $MCS_M_CODE_CHUCK_OPEN[i]	
M code for open chuck with stationary spindle.	
[0] = 0	Main spindle
[1] = 0	Counterspindle

MD52251 $MCS_M_CODE_CHUCK_CLOSE_OPEN_ROT[i]	
M code for open chuck with spindle rotating.	
[0] = 0	Main spindle
[1] = 0	Counterspindle

MD52252 $MCS_M_CODE_CHUCK_CLOSE[i]	
M code for close chuck.	
[0] = 0	Main spindle
[1] = 0	Counterspindle

MD52214 $MCS_FUNCTION_MASK_MILL	
Milling function mask.	
Bit 3	"Inner/rear" machining is enabled in the ShopTurn masks which themselves define the machining plane.
Bit 4	If you have implemented the "Clamp/release spindle" function using the manufacturer cycle CUST_TECHCYC.SPF, then using this machine data, you can activate the "Clamp/release spindle" parameter in the drilling and milling masks.
= 0	The "Clamp/release spindle" parameter is not displayed in the drilling and milling masks. ShopTurn automatically clamps the spindle if it makes sense for the particular machining operation.
= 1	The "Clamp/release spindle" parameter is displayed in the drilling and milling masks: Specification for which machining operation the spindle should be clamped.

Enable various functions under the Turning function mask in the following channel-specific machine data.

MD52218 $MCS_FUNCTION_MASK_TURN	
Turning function mask	
Bit 0	Enable zoom under manual for tool measurement.
Bit 1	Enable part catcher for cut-off.
Bit 2	Enable tailstock.
Bit 3	Reserved.
Bit 4	Enable spindle control of main spindle via user interface.
Bit 5	Enable spindle control of tool spindle via user interface.
Bit 6	Enable taper angle mask.

MD52229 $MCS_ENABLE_QUICK_M_CODES = 0H (default setting)	
Enable quick M commands	
Bit 0	Coolant OFF.
Bit 1	Coolant 1 ON.
Bit 2	Coolant 2 ON.
Bit 3	Coolant 1 and 2 ON.

MD52230 $MCS_M_CODE_ALL_COOLANTS_OFF	
M code for all coolants OFF	
= 9	This machine data is used to define the M function for switching off the coolant; which is output when the tool is changed.

MD52231 $MCS_M_CODE_COOLANT_1_ON	
M code for coolant 1 ON	
= 8	This machine data is used to define the M function for coolant 1, which is output when the tool is changed.

MD52232 $MCS_M_CODE_COOLANT_2_ON	
M code for coolant 2 ON	
= 7	This machine data is used to define the M function for coolant 2, which is output when the tool is changed.

MD52233 $MCS_M_CODE_COOLANT_1_AND_2_ON	
M code for both coolants ON	
= -1	This machine data is used to define the M function for coolant 1 and 2, which is output when the tool is changed.

MD52210 $MCS_FUNCTION_MASK_DISP	
Function mask display	
Bit 0	Dimension system for programs always in the basic system.
Bit 1	For G17, use the training coordinate system.

MD52281 $MCS_TOOL_MCODE_FUNC_ON[ ]	
M code for tool-specific function ON	
= -1	M function is not output: If both M commands of a function "= -1", the corresponding field is not displayed on the user interface.
[0]	M code for tool-specific function 1 ON.
[1]	M code for tool-specific function 2 ON.
[2]	M code for tool-specific function 3 ON.
[3]	M code for tool-specific function 4 ON.

MD52282 $MCS_TOOL_MCODE_FUNC_OFF[i]	
M code for tool-specific function OFF	
= -1	M function is output: If both M commands of a function "= -1", the corresponding field is displayed on the user interface.
[0]	M code for tool-specific function 1 OFF.

MD52282 $MCS_TOOL_MCODE_FUNC_OFF[i]	
[1]	M code for tool-specific function 2 OFF.
[2]	M code for tool-specific function 3 OFF.
[3]	M code for tool-specific function 4 OFF.

Setting data for rounding on the contour:

SD55582 $SCS_TURN_CONT_TRACE_ANGLE	
Contour turning: Minimum angle for rounding the contour	
= 5	Specifies the angle between the cutting edge and contour, above which for contour turning, the contour is rounded in order to remove residual material (default setting).

SD55505 $SCS_TURN_ROUGH_O_RELEASE_DIST	
Retraction distance for stock removal during external machining	
= 1	Specifies the distance, by which the tool is retracted from the contour when removing stock from an outer corner. This does not apply to stock removal at a contour (default setting).
= -1	The distance is internally defined.

SD55506 $SCS_TURN_ROUGH_I_RELEASE_DIST	
Retraction distance for stock removal during internal machining	
= 0.5	Specifies the distance, by which the tool is retracted from the contour when removing stock from an internal corner. This does not apply to stock removal at a contour (default setting).
= -1	The distance is internally defined.

SD55515 $SCS_TURN_THREAD_RELEASE_DIST	
Retraction distance when cutting threads.	
= 2	Specifies the distance to the workpiece that is retracted between the infeeds when cutting threads (default setting).

7.4.8 ShopTurn: Counterspindle

Requirement



Software option

You require the following software option in order to use the Counterspindle function:

"Travel to fixed stop (with force control)"

Function

If your turning machine has a counterspindle, you can machine workpieces using turning, drilling and milling functions on the front and rear faces without reclamping the workpiece manually. Before machining the rear face, the counterspindle must grip the workpiece, pull it out of the main spindle, and position it at the new machining position.

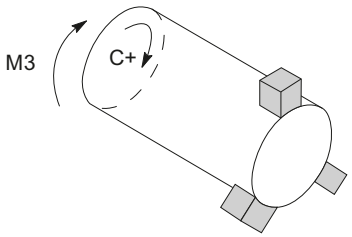
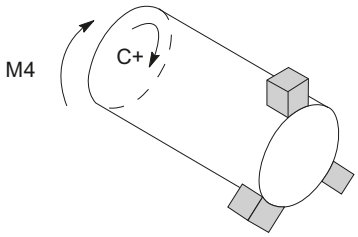
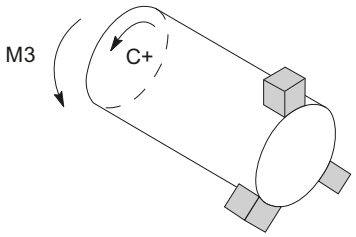
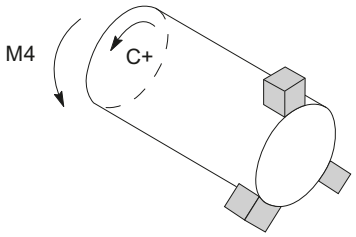
The direction of rotation of an NC rotary axis is set via **MD32100 $MA_AX_MOTION_DIR**. The PLC signal DB380x.DBX2001.6 (nn = 31 + machine axis index) is used to specify whether M3 has the same direction as rotary axis + (bit = 0).

The directions of rotation are set in the following machine data:

MD52207 $MCS_AXIS_USAGE_ATTRIB[i]

Settings required for the counterspindle

As a result, the following settings are required for the counterspindle. However, the settings in the machine data depend on the direction from which the coordinate axis is viewed. The settings for the direction of rotation as well as the interface signal DB380x.DBX2001.6 must be observed under all circumstances.

Counter-spindle direction of rotation	52207[C-Ax] Bit 3 =	52207[Spnd] Bit 4 =	52207[Spnd] Bit 5 =	DB380x. DBX2001.6
	1	0	0	
	0	0	1	
	0	1	0	
	1	1	1	

The setting of MD52207 assumes that the viewing direction is toward the negative coordinate axis. If, on the other hand, you are looking towards the positive coordinate axis, you need to reverse the values, i.e. swap "0" and "1".

The position to which the counterspindle travels when the program starts is defined in the following channel-specific setting data:

SD55232 $SCS_SUB_SPINDLE_REL_POS	Retraction position Z for counterspindle.

The following channel-specific setting data become effective when traveling to the fixed stop:

SD55550 $SCS_TURN_FIXED_STOP_DIST	Distance for travel to fixed stop.
SD55551 $SCS_TURN_FIXED_STOP_FEED	Feedrate for travel to fixed stop.
SD55552 $SCS_TURN_FIXED_STOP_FORCE	Force for travel to fixed stop in %.

Between traveling to the fixed stop and gripping, the counterspindle can retract a short distance to counteract compressive stress in the workpiece.

SD55553 $SCS_TURN_FIXED_STOP_RETRACTION	Retraction distance before clamping after fixed stop.

After gripping you can cut off the workpiece. Before doing so, the counterspindle can retract a short distance with the workpiece to exert tensile stress on the workpiece. This relieves pressure on the tool when cutting off.

SD55543 $SCS_TURN_PART_OFF_RETRACTION	Retraction distance before parting.

After the part, you can carry out a cutting-off check and for turning, use the "Travel to fixed stop" function. You can activate or deactivate the cut-off check via the following channel-specific setting data:

SD55540 $SCS_TURN_PART_OFF_CTRL_DIST	Distance for cut-off check.
SD55541 $SCS_TURN_PART_OFF_CTRL_FEED	Feedrate for cut-off check.
SD55542 $SCS_TURN_PART_OFF_CTRL_FORCE	Force for cut-off check in %.

The cut-off is successful when travel to fixed stop fails. The following alarms are output:

Alarm	Alarm text
20091	Axis %1 has not reached the fixed stop.
20094	Axis %1 end stop has been aborted.

You can switch off the alarm display using the following machine data:

MD37050 $MA_FIXED_STOP_ALARM_MASK	
Enabling the fixed stop alarms.	
= 2	Suppressing alarms 20091 and 20094.

You can set this machine data axis-specifically in the "Machine Data" window in the "Tool zero" operating area.

If, however, the specified force is reached during the cut-off check (i.e. travel to fixed stop is successful), alarm 61255 "Error during cut-off: Tool break?" is issued.

Note

The "Travel to fixed stop" function can also be used when gripping the spindle (see above). If travel to fixed stop does not succeed when gripping, an alarm will of course also be issued. Instead of alarms 20091 and 20094, the alarm 61254 "Error during travel to fixed stop" will be issued.

Dimensions of the counterspindle

To define the reference point for moving the counterspindle, you must first announce the dimensions of the counterspindle. You can either enter the dimensions in the following channel-specific machine data or in the menu "Tools - zero offset" → ">" → "Spindles". Changes to the machine data are automatically accepted in the menu and vice versa.

MD52241 $MCS_SPINDLE_CHUCK_TYPES[ ]	
Spindle jaw type.	
[0]	Main spindle
[1]	Counterspindle
= 0	Clamping, outer
= 1	Clamping, inner

MD52242 $MCS_MM_MAIN_SPINDLE_PARAMETER[ ]	
Main spindle parameters.	
[0]	Chuck dimensions
[1]	Stop dimensions
[2]	Jaw dimensions

MD52243 $MCS_MM_SUB_SPINDLE_PARAMETER[ ]	
Parameters for counterspindle.	
[0]	Chuck dimensions
[1]	Stop dimensions
[2]	Jaw dimensions

MD52244 $MCS_SUB_SPINDLE_PARK_POS_Y	
Parking position of Y axis for counterspindle.	
= 0	

Manufacturer cycle CUST_TECHCYC.SPF

If you want to perform one of the following actions, you must adapt the manufacturer cycle CUST_TECHCYC.SPF.

- Switch between spindle and C axis mode for the main spindle or counterspindle.
- Open, close, flush chuck (main spindle / counterspindle).
- Change default settings for the coupling of the main spindle and counterspindle.

See also

ShopTurn: Setting up cycles for turning (Page 184)

7.4.9 ShopTurn: Cylinder surface transformation (TRACYL)

Requirement



Software option

You require the following software option in order to use this function:

"TRANSMIT and peripheral surface transformation"

Function

If you wish to use the function cylinder surface transformation (TRACYL) under ShopTurn, please take the settings from Chapter:

Cylinder surface transformation (TRACYL) (Page 175)

In addition, set the following channel-specific machine data:

MD52214 $MCS_FUNCTION_MASK_MILL	
Milling function masks.	
Bit 3	Enable "internal / rear" machining in ShopTurn masks that define the machining plane themselves.
Bit 4	If you have implemented the "Clamp/release spindle" function using the machine manufacturer cycle CUST_TECHCYC.SPF, then using this machine data, you can activate the "Clamp/release spindle" parameter in the drilling and milling masks.
= 0	The "Clamp/release spindle" parameter is not displayed in the drilling and milling masks. ShopTurn automatically clamps the spindle if it makes sense for the particular machining operation.
= 1	The "Clamp/release spindle" parameter is displayed in the drilling and milling masks. The operator decides for which machining operation the spindle should be clamped.

References

Programming Manual Job Preparation: Cylinder surface transformation (TRACYL)

7.4.10 ShopTurn: End face machining (TRANSMIT)

Requirement



Software option

You require the following software option in order to use this function:

"TRANSMIT and peripheral surface transformation"

Function

If you wish to use the end face machining function under ShopTurn, please proceed as described in the following Chapter:

End face machining (TRANSMIT) (Page 178)

In addition, set the following channel-specific machine data:

MD52214 $MCS_FUNCTION_MASK_MILL	
Function masks, ShopTurn.	
Bit 3	Enable "internal / rear" machining in ShopTurn masks that define the machining plane themselves.
Bit 4	If you have implemented the "Clamp/release spindle" function using the machine manufacturer cycle CUST_TECHCYC.SPF, then using this machine data, you can activate the "Clamp/release spindle" parameter in the drilling and milling masks.
= 0	The "Clamp/release spindle" parameter is not displayed in the drilling and milling masks. ShopTurn automatically clamps the spindle if it makes sense for the particular machining operation.
= 1	The "Clamp/release spindle" parameter is displayed in the drilling and milling masks. The operator decides for which machining operation the spindle should be clamped.

Note

The end face machining is automatically integrated in the cycles, with the exception of the straight line and circle. You can select the function for these two cycles in the "Program" operating area at "Straight line" and "Circle".

References

Function Manual, Extended Functions; Kinematic Transformation (M1): TRANSMIT

7.4.11 ShopTurn: Inclined axis (TRAANG)

Requirement



Software option

You require the following software option in order to use this function:

"Inclined axis"

Function

If you wish to use the "Inclined axis" function under ShopTurn, please take the settings from Chapter:

Inclined axis (TRAANG) (Page 181)

Note

Once the "Inclined axis" function has been set up in the user interface, it is automatically integrated in the cycles. This means that for machining with inclined axis, you can select the "Face Y" or "Surface Y" machining plane and enter Cartesian coordinates.

References

Programming Manual Job Preparation: Inclined axis (TRAANG)

7.5 Swivel

7.5.1 Technology cycles for swiveling

Requirement

The commissioning of the kinematic chain of the machine is a mandatory requirement for the swivel function (CYCLE800). The kinematic chain is stored in the tool parameters $TC_CARR1 to $TC_CARR65.

Note

For the activation of the swivel function, **one** tool holder that can be oriented (swivel data set) and **the system frames** Setting the workpiece, tool and rotary table reference (Page 201) are already activated in the NCK (default setting).

References

Function Manual Basic Functions: Tool offset (W1),
inclined surface machining with 3 + 2 axes

Activating the swivel function

The swivel function is activated via the following channel-specific machine data:

MD52212 $MCS_FUNCTION_MASK_TECH	
Cross-technology function mask	
Bit 0 = 1	Enable swivel

Configuring the input dialog

You can configure the input dialog for swivel via the following channel-specific setting data. The setting data is effective for all of the declared swivel data sets.

SD55221 $SCS_FUNKTION_MASK_SWIVEL_SET	
Function mask, swivel CYCLE800	
Bit 0	Input field "No swivel"
= 0	Hide
= 1	Display
Bit 1	Text displayed for retract the tool axis
= 0	Display text Z = "Z", display text Z, XY = "Z,XY"

SD55221 $SCS_FUNKTION_MASK_SWIVEL_SET	
= 1	Display text Z = "Fixed point 1", Display text Z, XY = "Fixed point 2". If you wish to modify the retraction version "Z" or "Z, XY" via the manufacturer cycle CUST_800.SPF, the neutral text "Fixed point 1" and "Fixed point 2" can be displayed when retracting in this case.
Bit 2	Deselecting the active swivel set
= 0	Deselection not permitted If deselection is not permitted, the "Swivel data set" (TC) selection field is not displayed in the "Swivel" input dialog.
= 1	Deselection permitted Parameter TC_CARR37[n] HUNDRED MILLIONS position

Additional settings

To use the "Swivel" function, set the following machine data:

MD10602 $MN_FRAME_GEOAX_CHANGE_MODE	
= 1	The current total frame (work offsets) is recalculated when switching over geometry axes (selecting/deselecting TRAORI).

MD11450 $MN_SEARCH_RUN_MODE	
Settings, block search	
Bit 1 = 1	Activate PROG_EVENT.SPF after block search. This means that for a block search, the rotary axes of the active swivel data set are pre-positioned.

MD11602 $MN_ASUP_START_MASK	
Ignore stop conditions for ASUB	
Bit 0 = 1	ASUB, modal; used for the swivel function in JOG.

MD11604 $MN_ASUP_START_PRIO_LEVEL	
Priorities $MN_ASUP_START	
= 64	Corresponds to 100; used for the swivel function in JOG.

MD20196 $MC_TOCARR_ROTAX_MODE	
Rotary axis mode for tool holders with orientation capability	
Bit 0 = 1	Swivel data set with one rotary axis, used for a rotary table with C axis.
Bit 1 = 1	Swivel data set with two rotary axes, used with swivel default setting.

MD20360 $MC_TOOL_PARAMETER_DEF_MASK	
Tool parameter setting	
Bit 10 = 1	Orientation vector is retained for T0 and D0 (no tool). Used for machine kinematics, types "T" and "M". See also: Parameter $TC_CARR34

MD21186 $MC_TOCARR_ROT_OFFSET_FROM_FR	
Offset of the rotary axes for a tool holder with orientation capability from the work offset of the rotary axis.	
= 0	In CYCLE800, the WCS is recalculated when there is a value in the work offset (WO) of the rotary axes.
= 1	A value in the WO of the rotary axes acts as offset of the tool holder that has orientation capability. The WCS remains unchanged.

MD21186 may not be rewritten in a program with call CYCLE800.

If several swivel data sets are declared per channel, and if machine functions need to be activated on changeover between swivel heads or tables, an M command can be issued in the PLC user program on switchover to another swivel data set.

MD22530 $MC_TOCARR_CHANGE_M_CODE	
M code for swivel data set change.	
= 0	No swivel data set change
< 0	M code + number of the swivel data set for the swivel data set change.

Example

MD22530 $MC_TOCARR_CHANGE_M_CODE	= -800
Number of swivel data sets in channel 1	= 2
Programming swivel data set 1 (TCARR=1)	= M801
Programming swivel data set 2 (TCARR=2)	= M802

With the output of the M commands, the PLC can, for example, limit or invert the spindle speed or clamp or release the rotary axes.

Kinematics with Hirth tooth system (swivel head / mixed kinematics)

Depending on the active plane (G17, G18, G19), the TOROT command or TOROTX, TOROTY is programmed in the NCU (G group 53) to calculate the compensating frame for the Hirth tooth system in CYCLE800. If the Hirth tooth system causes the programmed rotation to deviate from the possible positions of the rotary axes, a $P_TOOLFRAME compensating frame is created for swivel head and mixed kinematics.

If the compensating frame must be retained after RESET or end of part program, enter the following value in the channel-specific machine data:

MD20150 $MC_GCODE_RESET_VALUES[52]	
Reset behavior of the G groups	
= 2	For G17 (TOROT)
= 3	For G18 (TOROTY)
= 4	For G19 (TOROTX)

7.5.2 Setting the workpiece, tool and rotary table reference

Using the system frames

System frames can be active after Reset or Power On, in order, e.g. to retract a drill from a swiveled position without causing a collision.

Using the following machine data, you can set the workpiece, tool and rotary table reference system frames or you can influence the behavior of the system frames.

MD24006 $MC_CHSFRAME_RESET_MASK	
Active system frames after reset	
Bit 4	System frame workpiece reference
= 0	Not active
= 1	Remains active

MD24007 $MC_CHSFRAME_RESET_CLEAR_MASK	
Deletion of system frames after reset	
Bit 4	System frame workpiece reference
= 0	Do not delete
= 1	Delete

When used for measuring or swiveling in JOG, the workpiece reference must be active on reset and not deleted (cascaded measuring).

MD24006 $MC_CHSFRAME_RESET_MASK	
Active system frames after reset	
Bit 4 = 1	System frame for workpiece reference remains active after reset

MD24007 $MC_CHSFRAME_RESET_CLEAR_MASK	
Deletion of system frames after reset	
Bit 4 = 0	Do not delete system frame workpiece reference after reset

MD24008 $MC_CHSFRAME_POWERON_MASK	
Reset system frames after power on	
Bit 2	System frame rotary table reference (PAROT).
= 0	Do not reset
= 1	Reset
Bit 3	System frame tool reference (TOROT, ...)
= 0	Do not reset
= 1	Reset
Bit 4	System frame workpiece reference
= 0	Do not reset
= 1	Reset

MD24080 $MC_USER_FRAME_POWERON_MASK	
Settings for settable frames.	
Bit 0	
= 0	Settable work offset via power on not active.
=1	Last active settable work offset remains active after power on if MD20152 $MC_GCODE_RESET_MODE[7] = 1.

Use: If work offset G5xx, including all rotations, is to remain active after power on.

Axial machine data for the modulo rotary axes of the swivel data set:

MD30455 $MA_MISC_FUNCTION_MASK	
Axis functions	
Bit 0	Modulo rotary axis programming
= 0	No modulo rotary axis programming (e.g. 0 to 359.999 degrees)
= 1	Modulo rotary axis programming (e.g. -180 to 180 degrees)
Bit 2	Positioning, rotary axis
= 0	As programmed.
= 1	Along the shortest path Use: With the setting, bit 2=1 then e.g. for G90 with DC, rotary axis C travels along the shortest path. See also: Manufacturer cycle CUST_800.SPF.

MD32010 $MA_JOG_VELO_RAPID[AX] AX = axis name	
Rapid traverse in JOG, rotary and machine axes that are traversed for swivel in JOG.	
= 10000	Rapid traverse in JOG mode for swivel in JOG.

SD42980 $SC_TOFRAME_MODE	
Setting, frame definition for TOROT, PAROT.	
= 2000	Swivel (default setting)

SD42974 $SC_TOCARR_FINE_CORRECTION	
Fine offset TCARR (swivel data set).	
= 0	No fine offset of the swivel data set vectors.
= 1	Fine offset of the swivel data set vectors. The parameters of the swivel data set as of $TC_CARR41[n] apply.

n : Number of swivel data set

Swivel in the JOG mode

Cycle alarms 62186 and 62187 can be hidden or displayed via the following machine data:

MD55410 $MC_MILL_SWIVEL_ALARM_MASK	
Activate fault evaluation CYCLE800	
Bit 0	Activates fault 61186
= 0	Hide fault 61186 "Active work offset G%4 and base (base reference) contains rotations" (default setting).
= 1	Display fault 61186
Bit 1	Activate fault 61187
= 0	Hide fault 61187 "Active base and base reference (G500) contain rotations" (default setting).
= 1	Display fault 61187

7.5.3 ShopMill: Swivel plane and swivel tool

Requirement



Software option

You require the following software option in order to use the ShopMill function:
"ShopMill/ShopTurn"

Refer to the following chapter for a description and setting the function:

Technology cycles for swiveling (Page 198)

Swivel under ShopMill

To enable the swivel function under ShopMill, set the following additional channel-specific machine data:

MD52212 $MCS_FUNCTION_MASK_TECH	
Cross-technology function mask	
Bit 0 = 1	Enable swivel

You must create a swivel data set for every swivel head, swivel table or combination of both. A swivel data set consists of the following parameters:

$TC_CARR1[n] to $TC_CARR65[n] where n = number of the swivel data set

The parameters of the swivel data set can be read-in and read-out in the startup operating area.

System variable $TC_CARR37[n]

Programming with appropriate value assignment is also possible in an NC program (manufacturer cycle). The parameters of the swivel data set are immediately effective after the program has started. You can adapt the swivel function to specific user requirements in the CUST_800.spf manufacturer cycle.

$TC_CARR37[n]	Display versions of the input dialog for CYCLE800.
HUNDRED MILLION	Swivel data set enabled Swivel data set change - tool change
= 0	Swivel data set not enabled
= 4	Swivel data set enabled Automatic swivel data set and tool change
= 5	Swivel data set enabled Automatic swivel data set change and manual tool change.
= 6	Swivel data set enabled Manual swivel data set change and automatic tool change
= 7	Swivel data set enabled Manual swivel data set and tool change

7.5.4 CYCLE800 checklist for the identification of the machine kinematics

Identification of the machine kinematics (kinematic chain) according to DIN 66217 or ISO 841-2001

Note

This checklist does not claim to be complete.

- **Do the three linear axes of the machine that are active for the transformation form an orthogonal coordinate system?**

  Geometry axes XYZ
- **How many swivel kinematics does the machine have?**

  Combinations of two (or one) rotary axis and the three linear axes are always formed.
- **Which kinematics type is it?**

  Swivel head, swivel table or mixed kinematics of swivel head and swivel table.
- **What are the names of the rotary axes of the kinematics?**

  Manual rotary axes are permitted and do not have to be declared in the NC.
- **What is the 1st or 2nd rotary axis of a swivel data set?**

  Rule: Rotary axis 2 is based on rotary axis 1. With mixed kinematics, rotary axis 1 is always the axis for the tool orientation.
- **Is the traversing direction of the linear axes and the rotary axes correct?**

  Right-hand rule:

  If the linear axis or the rotary axis moves the workpiece, the direction of motion of the axis and also the sign of the rotary axis vector change.
- **What is the initial setting of the kinematics?**

  This defines the tool orientation and the plane G17, G18, G19.
- **Which rotary axis rotates around which axis of the coordinate system or the machine axis (axes)?**

  This defines the rotary axis vectors of the kinematics.

  Example 1:

  Head kinematics. Rotary axis 2 rotates around axis Y → rotary axis vector V2xyz = 0,1,0

  Example 2:

  Table kinematics. Rotary axis 1 rotates around axis X → rotary axis vector V1xyz = -1,0,0

7.5.5 Commissioning of the kinematic chain (swivel data record)

Swivel data set (SDS)

You must create a swivel data set for every swivel head, swivel table and combination of both.

A swivel data set consists of the parameters:

$TC_CARR1[n] to $TC_CARR65[n] where n = number of the swivel data set

The parameters of the swivel data set ($TC_CARR1[n] to $TC_CARR65[n]) can be read-in and read-out in the startup operating area. Programming with appropriate value assignment is also possible in an NC program (manufacturer cycle). The parameters of the swivel data set are immediately effective after the program has started.

References

Function Manual Basic Functions: Tool offset (W1), tool holders with orientation capability

Offset vectors I1 to I4

The vectors always contain three components, which represent the reference to the X, Y and Z machine axes. The position in the kinematic chain is measured by the machine manufacturer and is always relevant for a swivel head / swivel table (swivel data set).

Offset vectors I1 to I4 refer to the non-swiveled state of the rotary axes (machine kinematics initial setting).

The machine kinematics used do not need to be fully implemented. However, be aware that the traversing range in the swivel planes may be restricted. If machine kinematics are to be implemented with just one rotary axis, this must always be declared as the 1st rotary axis.

$TC_CARR1[n],	$TC_CARR2[n],	$TC_CARR3[n]	Offset vector I1xyz
$TC_CARR4[n],	$TC_CARR5[n],	$TC_CARR6[n]	Offset vector I2xyz
$TC_CARR15[n],	$TC_CARR16[n],	$TC_CARR17[n]	Offset vector I3xyz
$TC_CARR18[n],	$TC_CARR19[n],	$TC_CARR20[n]	Offset vector I4xyz

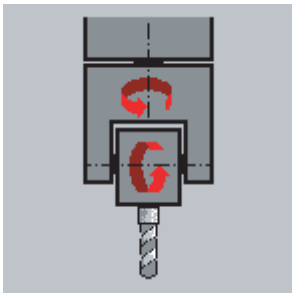
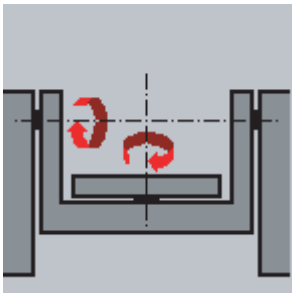
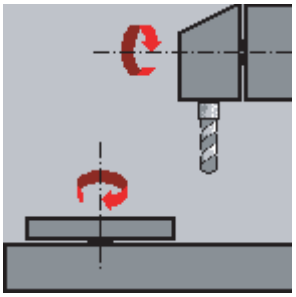
Rotary axis vectors V1 and V2

$TC_CARR7[n],	$TC_CARR8[n],	$TC_CARR9[n]	Rotary axis vector V1xyz
$TC_CARR10[n],	$TC_CARR11[n],	$TC_CARR12[n]	Rotary axis vector V2xyz

Kinematic types $TC_CARR23[n]

Selection:

- Swivel head (type T)
- Swivel table (type P)
- Swivel head + swivel table (type M)

Swivel head (type T)	Swivel table (type P)	Swivel head + swivel table (type M)
		
Offset vector I1	Offset vector I2	Offset vector I1
Rotary axis vector V1	Rotary axis vector V1	Rotary axis vector V1
Offset vector I2	Offset vector I3	Offset vector I2
Rotary axis vector V2	Rotary axis vector V2	Offset vector I3
Offset vector I3	Offset vector I4	Rotary axis vector V2
		Offset vector I4

Offset parameters

An offset value is entered for rotary axis 1 or 2 when the position of the rotary axes is not equal to 0 in the initial setting of the kinematics. In the initial setting of the kinematics, the tool orientation to a geometry axis (X, Y, Z) must be parallel.

$TC_CARR24[n]	Kinematics offset of rotary axis 1.
$TC_CARR25[n]	Kinematics offset of rotary axis 2.
$TC_CARR26[n]	Angular offset of the Hirth tooth system at the start of gearing of rotary axis 1.
$TC_CARR27[n]	Angular offset of the Hirth tooth system at the start of gearing of rotary axis 2.
$TC_CARR28[n]	Angle grid of the Hirth tooth system of rotary axis 1.
$TC_CARR29[n]	Angle grid of the Hirth tooth system of rotary axis 2.

A valid angle range (e.g. -90 to +90 degrees) must be assigned to each rotary axis. This does not have to be the software end position range of the relevant rotary axis.

An angle range of 0 degrees and 360 degrees can be entered for modulo axes.

$TC_CARR30[n]	Angle range of rotary axis 1 (minimum value).
$TC_CARR31[n]	Angle range of rotary axis 2 (minimum value).
$TC_CARR32[n]	Angle range of rotary axis 1 (maximum value).
$TC_CARR33[n]	Angle range of rotary axis 2 (maximum value).

Name of swivel data set, name of rotary axis

If several swivel data sets are declared in each NC channel, then a name is assigned to each swivel data set. No name needs to be specified if the swivel-mounted tool carrier is not exchangeable.

$TC_CARR34[n]	Name of the swivel data set.
$TC_CARR35[n]	Name of rotary axis 1.
$TC_CARR36[n]	Name of rotary axis 2.

Note

The name of the swivel data set may only contain characters that are also permitted in the NC programming: A...Z, 0...9 and _!

The following identifiers should be chosen for the rotary axes' names, where possible:

- Rotary axis rotates around machine axis X → A
- Rotary axis rotates around machine axis Y → B
- Rotary axis rotates around machine axis Z → C

For automatic rotary axes, the channel names of the corresponding NC rotary axes must be entered (see $TC_CARR37[n] TENS and HUNDREDS position: Automatic mode).

For manual (manually adjustable) and semi-automatic rotary axes, you can use any axis identifier (up to six letters or numbers).

System variable $TC_CARR37[n]

In the "Program" → "Miscellaneous" operating area, the following softkeys are assigned to the swivel function:

Milling technology	Turning technology	
"Swivel plane"	"Swivel plane"	
"Advance milling tool"	"Swivel tool"	→ "Align turning tool"
		→ "Align milling tool"
		→ "Advance milling tool"

The "Align turning tool" and "Align milling tool"softkeys are only displayed if the "B-axis kinematics" function was activated (refer to the following table).

Significance of the decimal places:

In order that a value can be displayed in the input / selection boxes of the dialog for swiveling, the following display versions can be set.

$TC_CARR37[n]	Display variants of the input screens for CYCLE800	
Decimal place	**Meaning**	
ONES	Selects the swivel mode	
	0 =	Axis by axis
	1 =	Axis-by-axis + projection angle
	2 =	Axis-by-axis + projection angle + solid angle
	3 =	Axis-by-axis + direct
	4 =	Axis-by-axis + projection angle + direct
	5 =	Axis-by-axis + projection angle + solid angle + direct
TENS	Rotary axis 1	
	0 =	Automatic
	1 =	Manual
	2 =	Semi-automatic
HUNDREDS	Rotary axis 2	
	0 =	Automatic
	1 =	Manual
	2 =	Semi-automatic
THOUSANDS	Selection field, direction: Direction selection of the rotary axes	
	0 =	No, no display of the direction reference for kinematics that only have one solution. Direction selection (_DIR) Minus is generated in the cycle call CYCLE800.
	3 =	Direction reference, rotary axis 1 optimized, direction selection Minus in the basic setting of the kinematics.
	4 =	Direction reference, rotary axis 2 optimized, direction selection Minus in the basic setting of the kinematics.
	5 =	No, no display of the direction reference for kinematics that only have one solution. Direction selection (_DIR) Plus is generated in the cycle call CYCLE800.
	8 =	Direction reference, rotary axis 1 optimized, direction selection Plus in the basic setting of the kinematics.
	9 =	Direction reference, rotary axis 2 optimized, direction selection Plus in the basic setting of the kinematics.
		The values 1, 2, 6 and 7 are not permitted.
TEN THOUSANDS	Selection field, correction of the tool tip or B axis kinematics	
	0 =	No, no display of the Correction of tool tip input field
	1 =	Yes, correction of tool tip by means of TRAORI.
	2 =	No tracking of tool tip + B axis kinematics turning technology.
	3 =	Correction of tool tip + B axis kinematics turning technology. The Correction of tool tip function requires the "5-axis transformation (TRAORI)" option.
HUNDRED THOUSANDS	Reserved	

$TC_CARR37[n]	Display variants of the input screens for CYCLE800		
Decimal place	**Meaning**		
ONE MILLION TEN MILLION	Selection field, retraction		
	00 =	No retraction	
	01 =	Retraction Z	
	02 =	Retraction Z, XY	
	03 =	Retraction Z or Z, XY	
	04 =	Maximum retraction in tool direction	
	...		
	08 =	Incremental retraction in tool direction	
	...		
	15 =	Retraction Z or Z, XY or in maximum tool direction or in incremental tool direction	
		$TC_CARR38[n]	Retraction position X
		$TC_CARR39[n]	Retraction position Y
		$TC_CARR40[n]	Retraction position Z
HUNDRED MILLION	Swivel data set enabled Swivel data set change - tool change		
	0 =	Swivel data set not enabled	
	4 =	Swivel data set enabled automatic swivel data set and tool change	
	5 =	Swivel data set enabled automatic swivel data set change and manual tool change	
	6 =	Swivel data set enabled manual swivel data set change and automatic tool change	
	7 =	Swivel data set enabled manual swivel data set and tool change	

Retracting the geometry axes before swiveling

The ONE MILLION and TEN MILLION positions of the system variable $TC_CARR37[n] define which retraction versions are displayed in the swivel input mask:

- Retraction of axis Z
- Retract axes Z, XY
- Retract in the tool direction, maximum or incremental

Retracting axis Z or retracting axes Z, XY is realized as an absolute machine position at the values of parameters $TC_CARR38[n] to $TC_CARR40[n].

$TC_CARR38[n]	Retraction position X
$TC_CARR39[n]	Retraction position Y
$TC_CARR40[n]	Retraction position Z

The type of retraction is modified in the manufacturer cycle CUST_800.SPF.

NOTICE
Traversing the tool axes Retract the tool axis in such a way that the tool and workpiece cannot collide when swiveled.

Fine offsets of the the offset vectors

$TC_CARR41[n]	to	$TC_CARR60[n]

Assignment of base vectors to the fine offset vectors:

I1 $TC_CARR1..3[n]	to	$TC_CARR41..43[n]
I2 $TC_CARR4..6[n]	to	$TC_CARR44..46[n]
I3 $TC_CARR15..17[n]	to	$TC_CARR55..57[n]
I4 $TC_CARR18..20[n]	to	$TC_CARR58..60[n]

The fine offsets are activated by the following setting data:

SD42974 $SC_TOCARR_FINE_CORRECTION = 1.

The fine offsets act in addition to the corresponding base vectors when the Swivel function CYCLE800 or the NC function TCARR=n is called.

See also

Manufacturer cycle CUST_800.SPF (Page 222)

7.5.6 Example of the commissioning of swivel head 1

Swivel head 1 "HEAD_1"

The vectors relate to the kinematics initial setting (not true-to-scale in the drawing):

Rotary axis 1(C)	(Manual) around Z
Rotary axis 2(A)	(Manual) around X
Changeable swivel head	Manually adjustable

Changeable swivel head with steep taper to hold the spindle

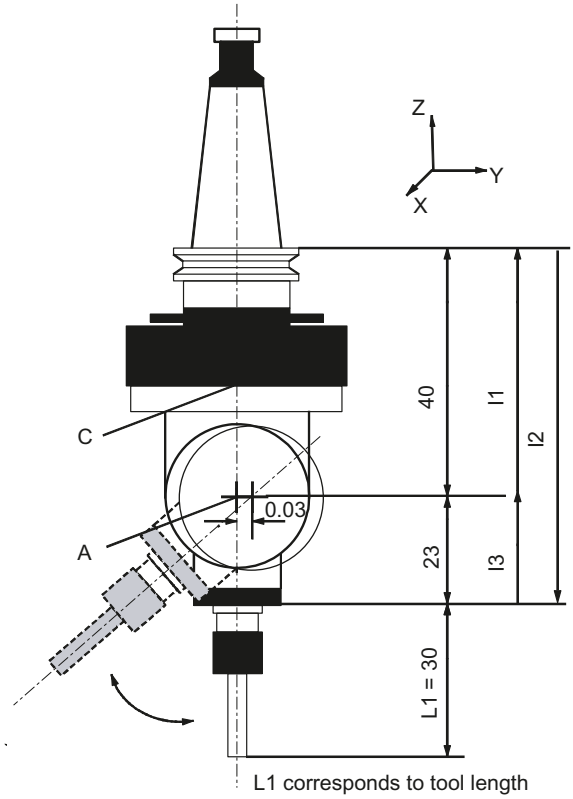


Figure 7-7 Swivel head 1 "HEAD_1"

Commissioning softkey "Swivel", kinematics (example 1):

Kinematics	**Swivel head**		**HEAD_1**
Retract	Z		
	X	Y	Z
			200.000
Offset vector I1	0.000	0.030	-63.000
Rotary axis vector V1	0.000	0.000	1.000
Offset vector I2	0.000	0.000	40.000
Rotary axis vector V2	1.000	0.000	0.000
Offset vector I3	0.000	-0.030	23.000
Display version			
Swivel mode	Axis by axis		
Direction reference	Rotary axis 2		
Correct tool	No		

Kinematics	**Swivel head**		**HEAD_1**
Rotary axes			
Rotary axis 1	C	Mode	Manual
Angular range	0.000		360.000
Rotary axis 2	A	Mode	Manual
Angular range	-15.000		100.000

7.5.7 Example of the commissioning of swivel head 2

Swivel head 2 "HEAD_2"

The vectors relate to the initial setting of the kinematics:

Rotary axis vector V1:	Rotary axis B rotates around Y
Rotary axis vector V2:	Rotary axis C rotates around Y **and** around Z
Offset vector I1:	Closure of vector chain with fixed-mounted swivel head I1=-(I2+I3)
Offset vector I2:	Distance between pivot point of **rotary axis 1** and pivot point of **rotary axis 2**
Offset vector I3:	Distance between reference point of tool and pivot point of **rotary axis 2**
If the swivel head is fixed-mounted, the vector chain is closed (see I1)	

Cardanic swivel head (manually adjustable) with Hirth tooth system:

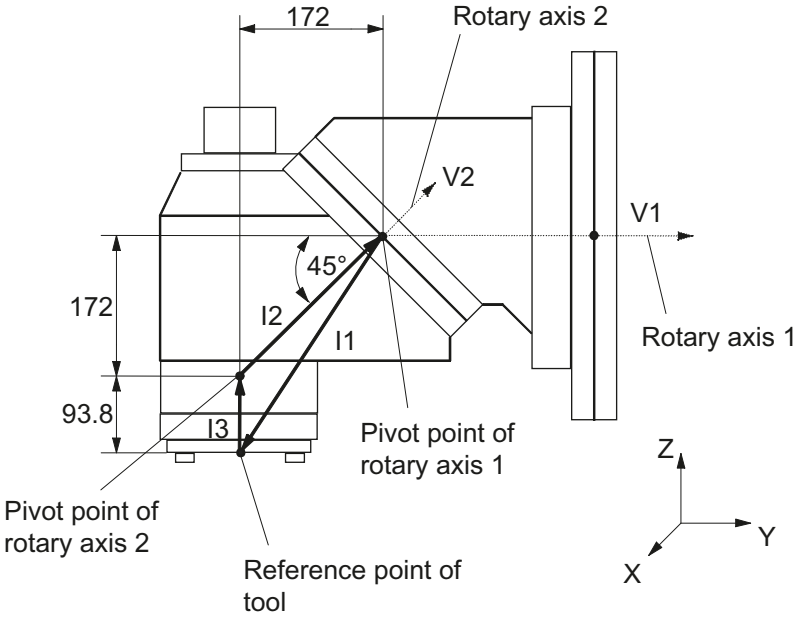


Figure 7-8 Swivel head 2 "HEAD_2"

Commissioning softkey "Swivel", kinematics (example 2):

Kinematics	**Swivel head**		**HEAD_2**
Retract	Z	Tool direction	Max.+inc.
	X	Y	Z
			200.000
Offset vector I1	0.000	-172.000	-265.800
Rotary axis vector V1	0.000	1.000	0.000
Offset vector I2	0.000	172.000	172.000
Rotary axis vector V2	0.000	1.000[1)]	1.000 *)
Offset vector I3	0.000	0.000	93.800
Display version			
Swivel mode	Axis by axis		
Direction reference	Rotary axis 2		

Kinematics	**Swivel head**		**HEAD_2**
Rotary axes			
Rotary axis 1	B	Mode	Manual
Angular range	0.000		360.000
Kinematics offset	0.000		
Hirth tooth system	yes	Angular grid	1.000
Rotary axis 2	C	Mode	Manual
Angular range	0.000		180.000
Kinematics offset	0.000		

*) Calculation of rotary axis vector V2: 45 degree angle

V2Y= sin(45)= 0.7071

V2z= cos(45)= 0.7071

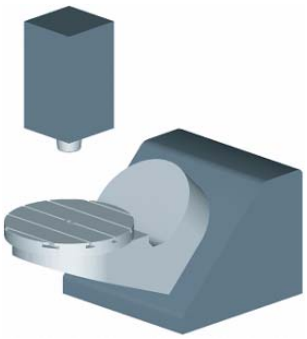
V2Y and V2z can be normalized to 1.

The reference point of the pivot point of rotary axes 1 and 2 can be offset on the line of rotation and does not have to coincide with the mechanical pivot point.

7.5.8 Example of the commissioning of a cardanic table

Cardanic table "TABLE_45"

The vectors relate to the initial setting of the kinematics:



Rotary axis vector V1:	Rotary axis B rotates around Y **and** around Z.
Rotary axis vector V2	Rotary axis C rotates around Z.
Offset vector I2:	Distance from the reference point of the machine to the pivot point / intersection of **rotary axis 1**
Offset vector I3:	Distance between pivot point / intersection of **rotary axis 1** and pivot point / intersection of **rotary axis 2**
Offset vector I4:	Closure of vector chain I4=-(I2+I3)

Side view of the machine:

Spindle (tool adapter) is positioned on a block dimension above the top edge of the table (rotary axis C) or the center of the table. A measuring rod in the spindle is used to determine the turning center of rotary axis C.

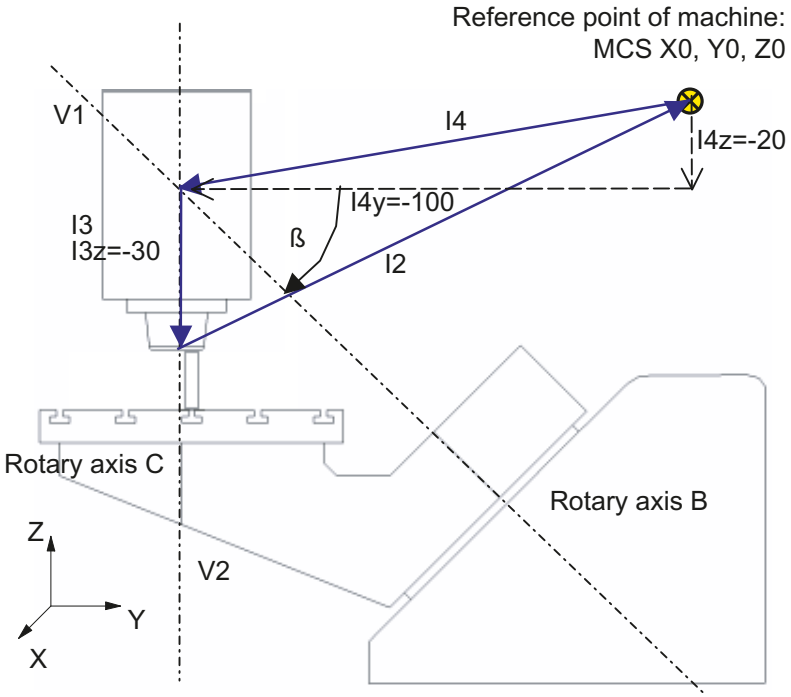


Figure 7-9 Cardanic table "TABLE_45"

Commissioning softkey "Swivel", kinematics (example 3):

Kinematics	**Swivel table**		**TABLE_45**
	X	Y	Z
Offset vector I2	0.000	100.000	50.000
Rotary axis vector V1	0.000	-1.000[1)]	1.000 *)
Offset vector I3	0.000	0.000	-30.000
Rotary axis vector V2	0.000	0.000	-1.000
Offset vector I4	0.000	-100.000	-20.000
Display version			
Swivel mode	Axis by axis		
Direction reference	Rotary axis 2		
Correct tool	No		

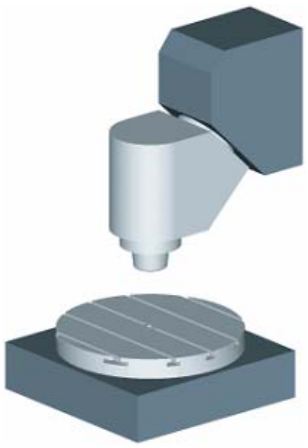
Kinematics	Swivel table		TABLE_45
Rotary axes			
Rotary axis 1	B	Mode	Auto
Angular range	0.000		180.000
Rotary axis 2	C	Mode	Auto
Angular range	0.000		360.000

*) Calculation of rotary axis vector V1: β = -45 degrees
V1Y= sin(-45)= -0.7071
V1z= cos(-45)= 0.7071
V1Y and V1z can be normalized to -1 and 1.

7.5.9 Example of the commissioning of a swivel head / rotary table

Swivel head / rotary table "MIXED_45"

The vectors relate to the initial setting of the kinematics:



Rotary axis vector V1:	Rotary axis B rotates around Y **and** around Z.
Rotary axis vector V2:	Rotary axis C rotates around Z.
Offset vector I2:	Distance from the reference point of the tool adapter to the pivot point / intersection of **rotary axis 1**
Offset vector I1:	Closure of vector chain I1=-I2
Offset vector I3:	Distance from the reference point of the machine to the pivot point / intersection of **rotary axis 2**
Offset vector I4:	Closure of vector chain I4=-I3

Side view of the machine:

Spindle (tool adapter) is positioned on a block dimension above the top edge of the table (rotary axis C) or the center of the table. A measuring rod in the spindle is used to determine the turning center of rotary axis C.

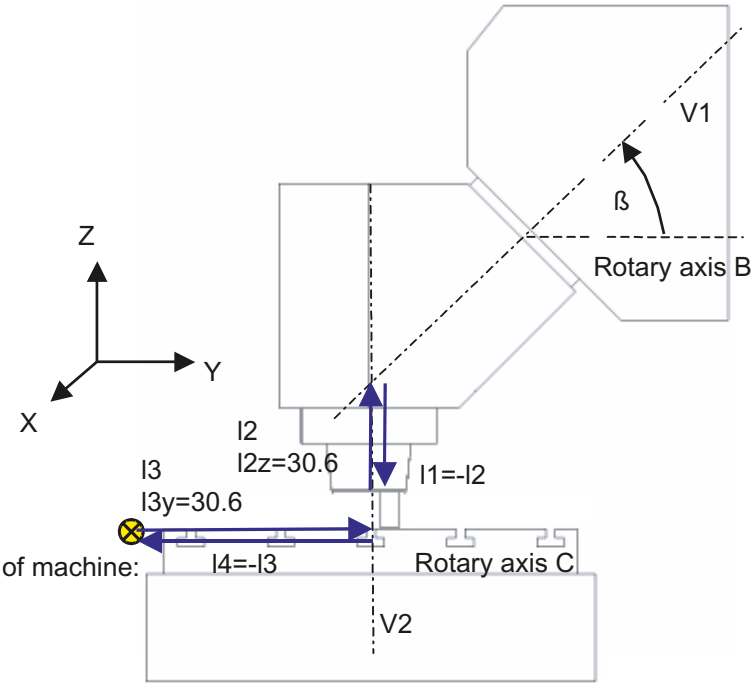


Figure 7-10 Swivel head / rotary table "MIXED_45"

Commissioning softkey "Swivel", kinematics (example 4):

Kinematics	**Mixed kinematics**		**MIXED_45**
	X	Y	Z
Offset vector I1	0.000	0.000	-30.600
Rotary axis vector V1	0.000	1.000[1)]	1.000 *)
Offset vector I2	0.000	0.000	30.600
Offset vector I3	300.000	150.000	0.000
Rotary axis vector V2	0.000	0.000	-1.000
Offset vector I4	-300.000	-150.000	0.000
Display version			
Swivel mode	Axis by axis		
Direction	Rotary axis 1		
Correct tool	yes		

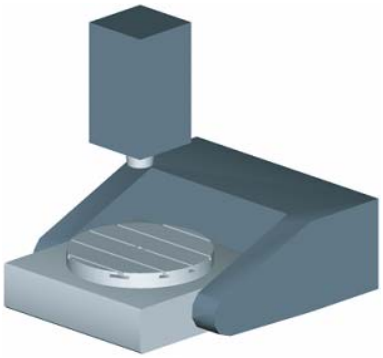
Kinematics	**Mixed kinematics**		**MIXED_45**
Rotary axes			
Rotary axis 1	B	Mode	Auto
Angular range	0.000		180.000
Rotary axis 2	C	Mode	Auto
Angular range	0.000		360.000

*) Calculation of rotary axis vector V1: β = 45 degrees
V1Y= sin(45)= -0.7071
V1z= cos(45)= 0.7071
V1Y and V1z can be normalized to 1.

7.5.10 Example of the commissioning of a swivel table

Swivel table "TABLE_5"

The vectors relate to the initial setting of the kinematics:



Rotary axis vector V1:	Rotary axis A rotates around X.
Rotary axis vector V2:	Rotary axis C rotates around Z.
Offset vector I2:	Distance from the reference point of the machine to the pivot point / intersection of **rotary axis 1**
Offset vector I3:	Distance from the pivot point of rotary axis 1 to the pivot point / intersection of **rotary axis 2**
Offset vector I4:	Closure of vector chain I4=-(I2+I3)

Side view of the machine from the X direction:

Spindle (tool adapter) is positioned on a block dimension above the top edge of the table (rotary axis C) or the center of the table. A measuring rod in the spindle is used to determine the turning center of rotary axis C.

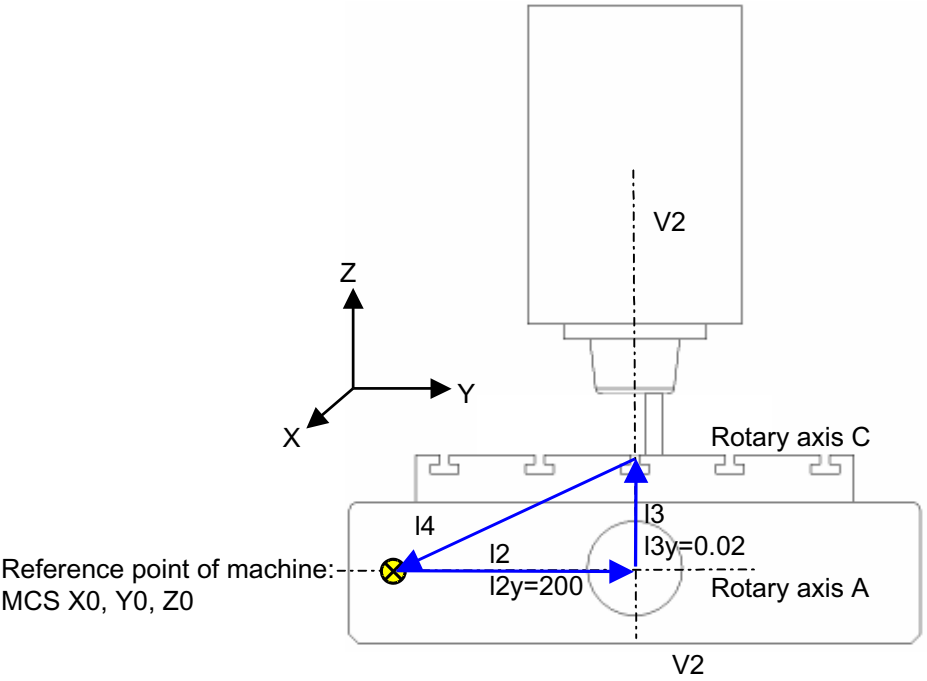


Figure 7-11 Swivel table "TABLE_5" (side view)

Front view of the machine from the Y direction:

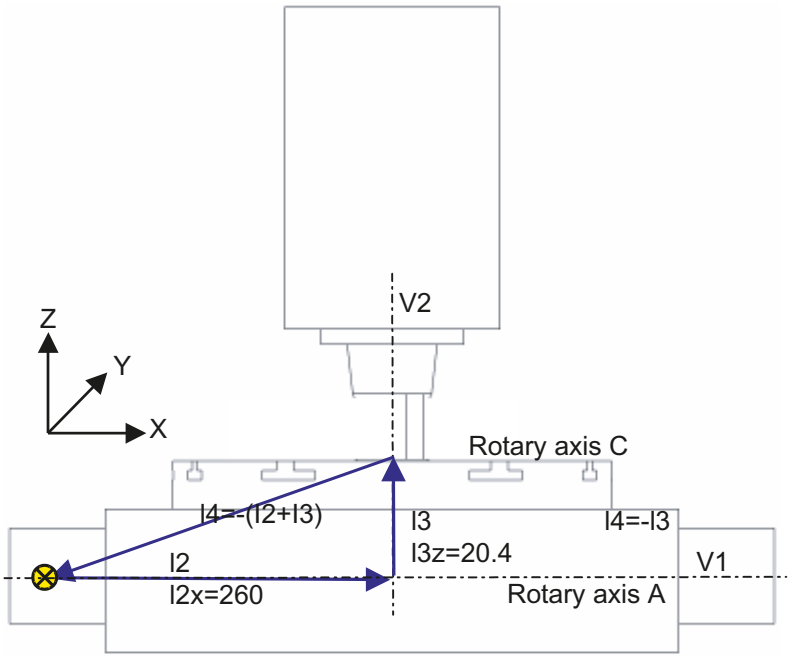


Figure 7-12 Swivel table "TABLE_5" (front view)

Commissioning softkey "Swivel", kinematics (example 5):

Kinematics	**Swivel table**		**TABLE_5**
	X	Y	Z
Offset vector I2	260.000	200.000	0.000
Rotary axis vector V1	-1.000	0.000	0.000
Offset vector I3	0.000	0.020	20.400
Rotary axis vector V2	0.000	0.000	-1.000
Offset vector I4	-260.000	-200.020	-20.400
Display version			
Swivel mode	Axis by axis		
Direction	Rotary axis 1		
Correct tool	No		
Rotary axes			
Rotary axis 1	A	Mode	Auto
Angular range	-90.000		90.000
Rotary axis 2	C	Mode	Auto
Angular range	0.000		360.000

7.5.11 Manufacturer cycle CUST_800.SPF

Overview

During swiveling, all axis positions are approached using the CUST_800.SPF cycle.
The call is exclusively made from the swivel cycle CYCLE800 or from the cycles E_TCARR (ShopMill) or F_TCARR (ShopTurn).

In cycle CUST_800.SPF, the function markers (_M2: to _M59) are prepared and documented.

Parameters of the CUST_800.SPF manufacturer cycle

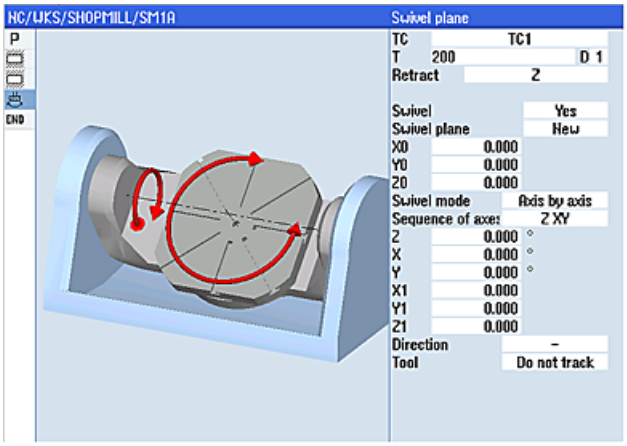
CUST_800 (INT _MODE, INT _TC1, REAL _A1, REAL _A2, INT _TC2, REAL _T_POS)
SAVE DISPLOF

_MODE	A jump is made to markers _M2 to _M59.
_TC1	Number of the swivel head/table
_A1	Angle of rotary axis 1
_A2	Angle of rotary axis 2
_TC2	1. Feed evaluation in percent (%) for swiveling in JOG mode
	2. Number of the new swivel head/table replacement under ShopMill
_T_POS	Incremental position during retraction in the incremental tool direction (see marker _M44, _M45)

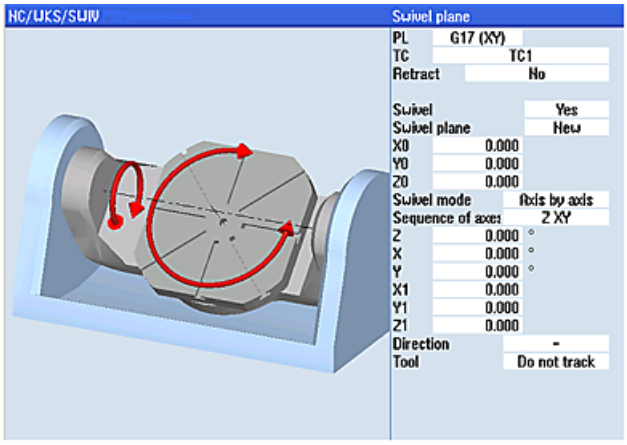
Retract prior to swiveling

If the CUST_800.SPF cycle is not modified, the Z axis (marker _M41) or the Z axis followed by the X, Y axes (marker _M42) are first traversed in the MCS to the positions when retracting prior to swiveling. The freely available position values are specified in the system variables $TC_CARR38[n] to $TC_CARR40[n]. When retracting, the active tool cutting edge is deselected (D0) and is reselected after retraction.

If retraction in the tool direction had been declared, the tool axis is retracted to the software end position (maximum in tool direction) or by an incremental distance away from the tool in the tool direction. The tool lengths are taken into account accordingly.



Input dialog with ShopMill (ShopTurn)



CYCLE800 input dialog

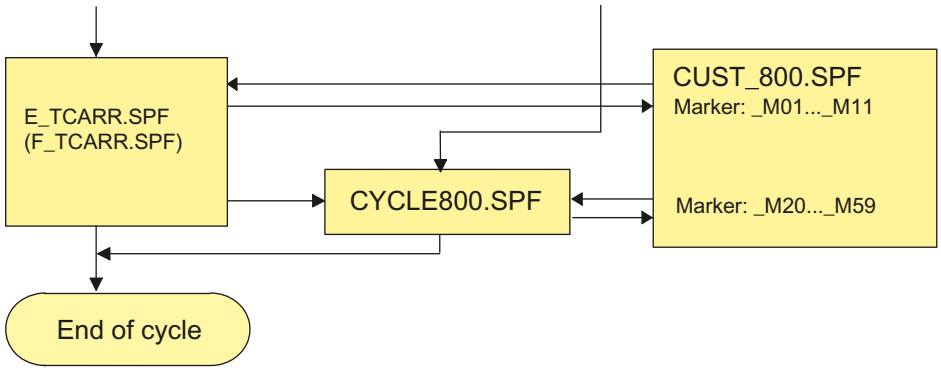


Figure 7-13 CYCLE800 programming

Structure CYCLE800

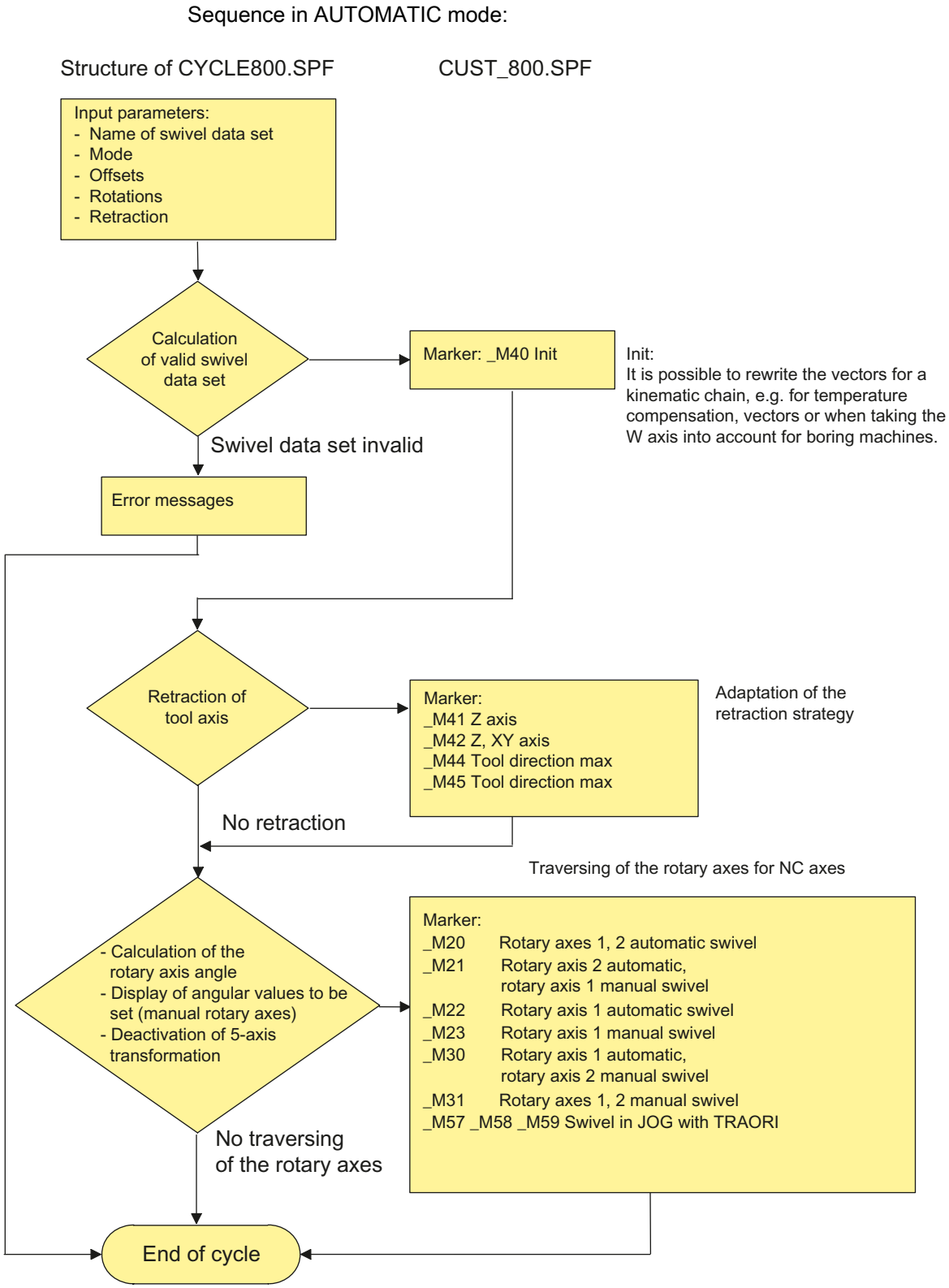


Figure 7-14 Structure CYCLE800.SPF / CUST_800.SPF

Structure E_TCARR.SPF (F_TCARR.SPF)

The following structure refers tot the swivel data set change and the related tool change tool change during milling or turning.

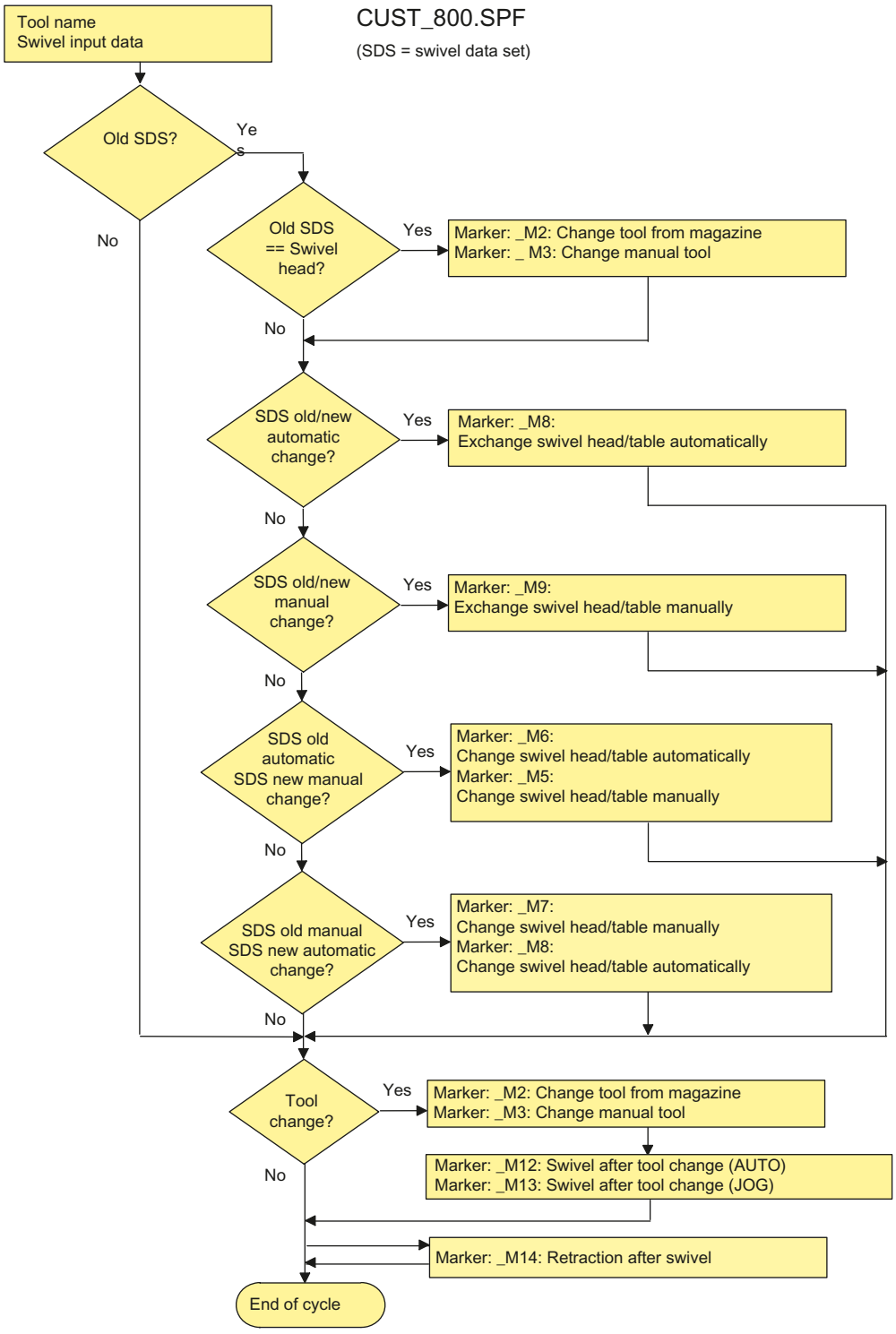


Figure 7-15 Structure E_TCARR.SPF

Markers _M2 to _M13

If the swivel data set or tool is changed, the linear axes are retracted using the last retraction mode (modal).

If this behavior is not desired in milling/turning, the corresponding calls must be commented out with a semicolon (;). The E_SWIV_H or F_SWIV_H cycle is called in milling/turning (see markers _M2 to _M9) in the CUST_800.SPF manufacturer cycle.

Parameters: E_SWIV_H (Par1, Par2, Par3)

- Par1: Number of swivel data set (_TC1)
- Par2: Angle of 1st rotary axis
- Par3: Angle of 2nd rotary axis

Modification examples

If the rotary axes (swivel head) are not to be positioned during swivel data change / tool change, the call of the E_SWIV_H cycle can be commented out at the relevant markers. If the rotary axes are to move to a certain position, an angle value can be transferred to parameters Par 2, Par 3.

Markers _M14 to _M15

Depending on the values of the retraction plane and the programmed swivel plane, it is possible that the linear axes now travel the swiveled retraction plane while running up from the current position to the software limit switches after a block search. To avoid this problem, marker _M14 in the CUST_800.SPF is called after swiveling. The E_SP_RP(30) cycle preset there runs up to the milling retraction plane, whereby travel may be along the software limit switches. An appropriate retraction after block search can be set at marker _M15.

Markers _M20 to _M31

Markers _M20 to _M31 are distinguished by machine kinematics with two rotary axes or one rotary axis. A distinction is also made between automatic rotary axes (known to the NCU) and manual (semi-automatic) rotary axes. There is only ever one valid marker for swiveling with the active swivel data set.

Marker _M35

Run through _M35 for block search and a swivel data set with manual rotary axes.

Marker _M46

Retraction before swiveling after a block search can be set at marker _M46. Variable _E_VER is 1 if it is a milling technology program.

Markers _M57 to _M59

Markers _M57 to _M59 are used for swiveling in JOG mode and active 5-axis transformation (TRAORI).

"Correct tool"

“Correct tool” requires that a 5-axis transformation is set up, which is equivalent to the corresponding swivel data set. The programming section for "Correct tool" is integrated in the markers _M20, _M21, _M22 and _M30. The first 5-axis transformation is called with TRAORI(1).

Tool change + swivel

In general, the swivel (CYCLE800) and tool change functions for a machine are independent of each other. Thus, the swiveled work plane can be retained in a technological sequence with multiple tools, e.g. centering, drilling, tapping.

If the rotary axes of the active swivel data set are involved in the mechanical sequence of the tool change or have to be retracted, this must be taken into account in the tool change program. After the tool change, the rotary axis positions are approached as prior to the tool change. If linear axes (geometry axes) are also involved in the tool change, the rotations in the NC (swivel frame) must not be deleted. Rather, the linear axes can be positioned as machine axes using the G153 or SUPA commands.

Swivel without active tool offset

If swiveling the rotary axes without active tool cutting edge (D0) is not possible, then you can adapt this in cycle CUST_800.SPF:

```
_M40:
IF ((NOT $P_TOOL) AND _TC1)
 LOOP
 MSG ("no tool cutting edge active")
 M0
 STOPRE
 ENDLOOP
ENDIF
GOTOF_MEND
```

7.6 High Speed Settings (Advanced Surface)

7.6.1 Configuring the High Speed Settings function (CYCLE832)

Function

The High Speed Settings function is used to preset data for the machining of sculptured surfaces so that optimum machining is possible. The "Advanced Surface" function is implemented in the "High Speed Settings" cycle (CYCLE832).

The call of CYCLE832 contains the following parameters:

- Tolerance
- Machining type
- Version=1 (permanently preset)

Tolerance value

The tolerance value for the linear axes (geometry axes) is transferred to the NC with the NC command CTOL. CTOL = root(3) * tolerance value.

If rotary axes are involved in the machining (multi-axis transformation), the tolerance value is transferred to the NC with a factor on the NC command OTOL. This factor can be set in the following channel-specific setting data for each machining type:

SD55440 $SCS_MILL_TOL_FACTOR_NORM	
Factor, tolerance of the rotary axes for CYCLE 832, G group 59.	
= 10	

SD55441 $SCS_MILL_TOL_FACTOR_ROUGH	
Factor, tolerance of the rotary axes for roughing of the G group 59.	
= 10	

SD55442 $SCS_MILL_TOL_FACTOR_SEMIFIN	
Factor, tolerance of the rotary axes for roughing-finishing of the G group 59.	
= 10	

SD55443 $SCS_MILL_TOL_FACTOR_FINISH	
Factor, tolerance of the rotary axes for finishing of the G group 59.	
= 10	

Machining type and technology G group 59

The machining types of the technology G group 59 are permanently assigned in CYCLE832 or in CUST_832.SPF:

Machining type	Technology G group 59	Array index
Deselection	DYNNORM	0
Roughing	DYNROUGH	2
Pre-finishing	DYNSEMIFIN	3
Finishing	DYNFINISH	4

Dynamic parameters are adapted to the respective machining operation with the technology G groups during commissioning.

Note

During the optimization of the machine axes, the values (array index) of the following machine data must be set correctly.

Machining type, deselection

When CYCLE832 is deselected, the G groups are programmed for the settings during the program run time; these settings are declared in MD2150 $MC_GCODE_RESET_VALUES[ ].

If, when deselecting CYCLE832, a tolerance has not been programmed, the setting from the following channel-specific setting data is used.

SD55445 $SCS_MILL_TOL_VALUE_NORM
Tolerance value for deselection

7.6.2 How to adapt the High Speed Settings function (CYCLE832)

Adapting manufacturer cycle CUST_832.SPF

In contrast to the settings (G commands) by CYCLE832.SPF, these settings can be modified in the manufacturer cycle CUST_832.SPF.

Procedure:

1. Copy the CUST_832.SPF cycle from the directory:
   `/NC data / Cycles / Standard cycles`
2. Paste the CUST_832.SPF cycle into the following directory:
   `/NC data / Cycles / Manufacturer cycles`
3. Open the cycle. The following settings are programmed:
   ```
   SOFT
   COMPCAD
   G645
   FIFOCTRL
   UPATH
   ;FFWON
   ;ORISON
   ;OST
   ```
   DYNNORM, DYNFINISH, DYNSEMIFIN, DYNROUGH depending on the machining type.

   Corresponding markers are prepared in CUST_832.SPF:
   ```
   _M_NORM:
   _M_FINISH:
   _M_SEMIFINISH:
   _M_ROUGH:
   ```
   The programming of FGREF () is useful when machining with active multi-axis transformation (e.g. TRAORI). In this case, in CUST_832.SPF, variable _FGREF is pre-assigned a value of 10 mm. This value can also be modified. In CYCLE832.SPF, the value of variable _FGREF is written to the rotary axes involved in the machining, which are declared as orientation axis of a 5-axis transformation, using the FGREF(rotary axis) command. When G70/G700 is active, the value from _FGREF is converted into inches before writing to the command FGREF.

Note

More than three rotary axes (orientation axes) in CYCLE832:

In CYCLE832, a maximum of three rotary axes of the orientation transformation (TRAORI) are taken into account for FGREF.

Example:

If more than three rotary axes are declared in the channel for the orientation transformation, You can wrote the value to FGREF with the following syntax in the CUST_832.SPF:

```
FGREF[AA]=$AA_FGREF[C]
;C = rotary axis 1 (axis is taken into account by CYCLE832)
;AA = rotary axis 4
```

7.7 Measuring cycles and measurement functions

7.7.1 General settings for measuring

Requirement

Two types of electronic probes are used for measuring:

- Probe for workpiece measurement
- Probe for tool measurement

The electronic probe is only called probe in the following description.

You set the electrical polarity of the connected probe using the following general machine data:

MD13200[0] $MN_MEAS_PROBE_LOW_ACTIVE MD13200[1] $MN_MEAS_PROBE_LOW_ACTIVE	
Polarity change of the probe	
= 0	Probe in the non-deflected state 0 V (default setting) Probe in the deflected state 24 V
= 1	Probe in the non-deflected state 24 V Probe in the deflected state 0 V

MD13210 $MN_MEAS_TYPE =1
Measuring type for distributed drives
The distributed measuring is permanently preset (data class: SYSTEM).

See also

Connecting the probes (Page 136)

Testing the probe function

You can test the switching function of the probe by manually deflecting it and checking the following PLC interface signals: DB2700

To test the switching behavior and the measured value transfer, use an NC test program with, for example, the following NC commands:

MEAS	Measuring with delete distance-to-go
$AC_MEA[n]	Checking of the switching operation n = measuring input number
$AA_MW[axis name]	Measured value of the axes in workpiece coordinates
$AA_MM[axis name]	Measured value of the axes in machine coordinates

Example: Test program

```
Program code                                  Comment
%_N_PRUEF_MESSTASTER_MPF                  ;
$PATH=/_N_MPF_DIR                         ; Test program, probe connection
N00 DEF INT MTSIGNAL                      ; Bit memory to check the switching status
N05 G17 G54 T="3D_Taster" D1              ; Select tool geometry for probe
N10 M06                                   ; Activate tool
N15 G0 G90 X0 F150                        ; Starting position and measuring speed
N20 MEAS=1 G1 X100                        ; Measurement at measuring input 1 in the
                                            X axis
N30 MTSIGNAL=$AC_MEA[1]                   ; Switching operation at the 1st measuring
                                            input completed, YES / NO
N35 IF MTSIGNAL == 0 GOTOF _FEHL1         ; Signal evaluation
N40 R1=$AA_MM[X]                          ; Save measured value in machine
                                            coordinates at R1
N45 R2=$AA_MW[X]                          ; Save measured value in workpiece
                                            coordinates at R2
N50 M0                                    ; Check measured value in R1/R2
N55 M30
N60 _FEHL1: MSG ("Probe does not
switch!")
N65 M0
N70 M30
```

Measuring input of the PPU for the workpiece or tool probe

The assignment of the NC measuring input is defined in the following general machine data.

MD51606 $MNS_MEA_INPUT_PIECE_PROBE[0]	
Workpiece probe measuring input	
= 0	Workpiece probe at the 1st NC measuring input (default setting)
= 1	Workpiece probe at the 2nd NC measuring input

MD51607 $MNS_MEA_INPUT_TOOL_PROBE[0]	
Tool probe measuring input	
= 0	Tool probe at the 1st NC measuring input
= 1	Tool probe at the 2nd NC measuring input (default setting)

MD51614 $MNS_MEA_PROBE_LENGTH_RELATE	
Length reference when calibrating the probe length	
= 0	In the infeed axis, calibration is performed to the center of the probe ball
= 1	In the infeed axis, calibration is performed in relation to the circumference of the probe ball (default setting)

Note

After changing MD51614, the probe must be recalibrated.

MD51616 $MNS_MEA_CAL_MONITORING	
Monitoring the calibration status	
= 0	Without monitoring
= 1	With monitoring (default setting)

7.7.2 Manufacturer cycle CUST_MEACYC.SPF

Manufacturer and user cycle CUST_MEACYC.SPF

The CUST_MEACYC.SPF is part of the measuring cycle functionality. It is called in every measuring cycle before and after executing the measurement task. The CUST_MEACYC.SPF acts in the same way when measuring in the JOG mode and measuring in the AUTOMATIC mode.

You can use the CUST_MEACYC.SPF to program and execute sequences that are necessary before and/or after a measurement (e.g. activating/deactivating a probe).

7.7.3 Measuring in the JOG mode

Requirement

You have already made the settings from the previous section. General settings for measuring (Page 231)

Measure workpiece

Milling technology:

- The probe has been inserted in the tool spindle.
- The probe has been selected in the tool list as type 710 (3D probe milling).
- Probe is activated as tool in the current NC channel.

Setting the general setting data:

SD54798 $SNS_J_MEA_FUNCTION_MASK_PIECE= 4H	
Settings for the input mask, measuring in JOG, workpiece measurement.	
Bit 2 = 1	Activation of the function "Measuring with electronic workpiece probe".

Measure tool

To measure tools, an appropriate probe must be located in the machine space so that this can be reliably and safely reached with a tool in the spindle.

The following tool types are supported with measure tool:

- Milling technology: Tool types 1xx and 2xx
- Turning technology: Tool type 5xx, 1xx, 2xx

For the specified tool types, the tool lengths and the tool radii can be measured.

Setting the general setting data:

SD54799 $SNS_J_MEA_FUNCTION_MASK_TOOL= 4H	
Settings for the input mask, measuring in JOG, tool measurement.	
Bit 2 = 1	Activation of the function "Measuring with electronic tool probe".

Setting machine data

General machine data:

MD11602 $MN_ASUP_START_MASK	
Ignore stop reasons for ASUB.	
Bit 0 = 1	ASUB start possible in the JOG mode.

MD11604 $MN_ASUP_START_PRIO_LEVEL	
Priorities from which ASUP_START_MASK is effective.	
= 1 - 64	Priorities for ASUP_START_MASK.

Channel-specific machine data:

MD20050 $MC_AXCONF_GEOAX_ASIGN_TAB[i]	
Assignment, geometry axis to channel axis.	
Only valid for milling. Requirement: For measuring in the JOG mode all of the geometry axes are available; preferably XYZ.	
[0] = 1	1st real geometry axis X axis.
[1] = 2	2nd real geometry axis, Y axis.
[2] = 3	3rd real geometry axis, Z axis.

MD20110 $MC_RESET_MODE_MASK	
Defines the basic control settings after reset / TP end,	
= 4045H	Minimum value
Bit 0 = 1	Basic control setting after power on and reset.
Bit 2 = 1	
Bit 6 = 1	
Bit 14 = 1	

MD20112 $MC_START_MODE_MASK	
Definition of the basic control setting after part program start.	
= 400H	Minimum value
Bit 6 = 0	Definition of the basic control setting after NC start.

MD20310 $MC_TOOL_MANAGEMENT_MASK	
Activating tool manager functions	
= 4002H	Minimum value
Bit 1 = 1	Tool manager and monitoring functions active
Bit 14 = 1	Automatic tool change for reset and start.

Note

After setting the machine and setting data in the JOG operating mode, the preconditions described in this chapter have been fulfilled so that the following measurements can be performed using a workpiece probe:

- Measuring a workpiece: at a milling machine
- Measuring a tool: at a milling machine or at a lathe

A description is provided in the following sections as to which settings you can make in order to adapt measuring to the specific requirements of the machine.

7.7.4 JOG: Measure workpiece during milling

Measure workpiece

Measuring in the "Machine" operating area can be appropriately adapted to the specific requirements using the following channel-specific general machine data and channel-specific setting data.

General machine data:

MD51751 $MNS_J_MEA_M_DIST_MANUELL	
Measuring path in mm, before and after the measuring point.	
= 10	Default setting

MD51755 $MNS_J_MEA_MEASURING_FEED	
Measuring feedrate in mm/min for workpiece measurement and calibration.	
= 300	Default setting

MD51757 $MNS_J_MEA_COLL_MONIT_FEED	
Position feedrate in mm/min, in the working plane for active collision monitoring.	
= 1000	Default setting

MD51758 $MNS_J_MEA_COLL_MONIT_POS_FEED	
Position feedrate in mm/min, in the infeed axis for active collision monitoring.	
= 1000	Default setting

MD51770 $MNS_J_MEA_CAL_RING_DIAM[i]	
Pre-assignment of the calibration diameter in mm specifically for the calibration data sets.	
= -1	Default setting

MD51772 $MNS_J_MEA_CAL_HEIGHT_FEEDAX[i]	
Default setting of the calibration height in mm in the infeed axis, specifically for the calibration data sets.	
= -99999	Default setting

General setting data:

SD54798 $SNS_J_MEA_FUNCTION_MASK_PIECE	
Settings of the input masks for measuring in JOG	
Bit 0	n. a.
Bit 1	n. a.
Bit 2 = 1	Activates measurements with an electronic probe.
Bit 3 = 1	Selects the probe calibration data field, enable.
Bit 4	n. a.
Bit 5	Select WO as basis for making measurements
Bit 6 = 1	Selects WO in the base reference (SETFRAME), enable.
Bit 7 = 1	Selects WO in the channel-specific basic frames, enable.
Bit 8 = 1	Selects WO in the global basic frame, enable.
Bit 9 = 1	Selects WO in adjustable frames, enable (default setting).

Channel-specific setting data:

SD55770 $SCS_J_MEA_SET_COUPL_SP_COORD	
Behavior of the probe carrier spindle.	
= 0	Coupling of the probe carrier spindle with coordinate rotation around the infeed axis, default setting.
= 1	The position of the probe carrier spindle at the time that the cycle started is used as the initial position for the measurement.

Note

After changing this setting data, the probe must be recalibrated.

SD55761 $SCS_J_MEA_SET_NUM_OF_ATTEMPTS	
Number of measurements at a measuring point	
= 0	Five measurements at every measuring point. The arithmetic average is generated.
= 1	One measurement at every measuring point, default setting.

SD55762 $SCS_J_MEA_SET_RETRAC_MODE	
Retraction velocity from the measuring point	
= 0	Retraction velocity corresponding to the intermediate positioning, default setting.
= 1	Retraction with rapid traverse

SD55763 $SCS_J_MEA_SET_FEED_MODE	
Selection of the measuring feedrate	
= 0	Measuring with the measuring feedrate, default setting.
= 1	1. Measuring with the feedrate corresponding to the channel-specific setting data SD55633 $SCS_MEA_FEED_FAST_MEASURE. 2. Measuring with measuring feedrate

If you are using a monoprobe, set the following general machine and channel-specific setting data. A monoprobe only switches in one direction of motion.

MD51612 $MNS_MEA_MONO_COR_POS_ACTIVE	
Alignment of the switching direction for monoprobe, taking into account the offset angle.	
= 0	Without offset angle
= 1	With offset angle, default setting

SD55772 $SCS_J_MEA_SET_PROBE_MONO	
Selection of the probe type	
= 0	Probe is a multiprobe (3D probe) default setting.
= 1	Probe is a monoprobe.

7.7.5 JOG: Measure tool during milling

Tool probe

In the following machine data, index [i] stands for the number of the current data field (probe number -1) of the probe.

General machine data:

MD51774 $MNS_J_MEA_T_PROBE_TYPE[i]	
Probe version	
= 0	Cube, default setting
= 101	Disk in XY, working plane G17
= 201	Disk in ZX, working plane G18
= 301	Disk in YZ, working plane G19

The following general machine data is used to define in which axes and directions it is possible to calibrate a tool probe.

MD51776 $MNS_J_MEA_T_PROBE_ALLOW_AX_DIR[i]	
Axes and directions for "calibration"	
= 133	Default setting

Decimal place			
ONES	1st axis		
		= 0	axis not possible
		= 1	only minus direction
		= 2	only plus direction
		= 3	both directions
TENS	2nd axis		
		= 0	axis not possible
		= 1	only minus direction
		= 2	only plus direction
		= 3	both directions
HUNDREDS	3rd axis		
		= 0	axis not possible
		= 1	only minus direction
		= 2	only plus direction
		= 3	both directions

Example

If the general machine data MD51776[i] $MNS_J_MEA_T_PROBE_ALLOW_AX_DIR has the value 123, the tool probe is calibrated as follows in the G17 plane:

- X in both directions
- Y only in plus direction
- Z only in minus direction

MD51778 $MNS_J_MEA_T_PROBE_DIAM_LENGTH[i]	
Effective diameter of the tool probe for length measurement.	
= 0	Default setting

MD51780 $MNS_J_MEA_T_PROBE_DIAM_RAD[i]	
Effective diameter of the tool probe for radius measurement.	
= 0	Default setting

MD51782 $MNS_J_MEA_T_PROBE_T_EDGE_DIST[i]	
Distance between the upper edge of the tool probe and the lower edge of the tool (= calibration depth, measuring depth when measuring the milling cutter radius).	
= 2	Default setting

Measuring path / measuring feedrate

General machine data:

MD51752 $MNS_J_MEA_M_DIST_TOOL_LENGTH	
Measuring path to measure the tool length.	
= 2	Default setting

MD51753 $MNS_J_MEA_M_DIST_TOOL_RADIUS	
Measuring path to measure the tool radius.	
= 1	Default setting

MD51786 $MNS_J_MEA_T_PROBE_MEASURE_DIST	
Measuring path to calibrate the probe or for measurements with stationary spindle.	
= 10	Default setting

MD51787 $MNS_J_MEA_T_PROBE_MEASURE_FEED	
Calibrate measuring feedrate for probe and tool measurement with stationary spindle.	
= 100	Default setting

Monitoring when measuring with a rotating spindle

General setting data:

SD54670 $SNS_MEA_CM_MAX_PERI_SPEED[0]	
Maximum permissible peripheral speed of the tool to be measured.	
= 100	Default setting

SD54671 $SNS_MEA_CM_MAX_REVOLUTIONS[0]	
Maximum permissible tool speed of the tool to be measured. The speed is automatically reduced when exceeded.	
= 1000	Default setting

SD54672 $SNS_MEA_CM_MAX_FEEDRATE[0]	
Maximum permissible feedrate to probe the tool to be measured at the probe.	
= 20	Default setting

SD54673 SNS_MEA_CM_MIN_FEEDRATE[0]	
Minimum feedrate for the first probing of the tool to be measured at the probe. The avoids excessively small feedrates for large tool radii.	
= 1	Default setting

SD54674 $SNS_MEA_CM_SPIND_ROT_DIR[0]	
Spindle direction of rotation to measure tools.	
= 4	Spindle rotation the same as M4 (default setting)

NOTICE
If the spindle is already rotating when the measuring cycle is called, this direction of rotation remains independent of the setting of this data.

SD54675 $SNS_MEA_CM_FEEDFACTOR_1[0]	
Feedrate factor 1	
= 10	Default setting
= 0	Only single probing with the feedrate calculated by the cycle. However, as a minimum, the value from SD54673[0] $SNS_MEA_CM_MIN_FEEDRATE.
>= 1	First probing with feedrate. But at least with the value of SD54673[0] $SNS_MEA_CM_MIN_FEEDRATE) SD54675[0] $SNS_MEA_CM_FEEDFACTOR_1

SD54676 $SNS_MEA_CM_FEEDFACTOR_2[0]	
Feedrate factor 2	
= 0	Second probing with the feedrate calculated by the measuring cycle. This is only effective for SD54673[0] $SNS_MEA_CM_FEEDFACTOR_1 > 0 (default setting)
>= 1	Second probing with the calculated feedrate from SD54673[0] $SNS_MEA_CM_MIN_FEEDRATE feedrate factor 2. Third probing with the calculated feedrate.

NOTICE
Feedrate factor 2 should be less than feedrate factor 1.

SD54677 $SNS_MEA_CM_MEASURING_ACCURACY[0]	
Specified measuring accuracy: The value of this parameter always refers to the last probing of the tool at the probe!	
= 0.005	Default setting

7.7.6 JOG: Measure tool during turning

Measuring path / measuring feedrate

General machine data:

MD51786 $MNS_J_MEA_T_PROBE_MEASURE_DIST	
Measuring path to calibrate the probe or for measurements with stationary spindle.	
= 10	Default setting

MD51787 $MNS_J_MEA_T_PROBE_MEASURE_FEED	
Calibrate measuring feedrate for tool probe and tool measurement with stationary spindle.	
= 100	Default setting

Channel-specific setting data:

SD42950 $SC_TOOL_LENGTH_TYPE	
Assignment of the tool length offset independent of tool type.	
= 0	Measuring turning tools, type 5xx
= 2	Measuring turning tools, type 5xx, drilling and milling tools, type 1xx, 2xx (default setting)

7.7.7 Measuring in the AUTOMATIC mode

Requirement



Software option

In order to use the "Measuring in AUTOMATIC" function, you require the software option: "Measuring cycles"

You have already made the settings from this section: General settings for measuring (Page 231)

Workpiece measurement for milling:

- The probe has been inserted in the tool spindle.
- The probe has been selected in the tool list as type 710 (3D probe milling).
- The probe is activated in the current NC channel.

Workpiece measurement for turning:

- The tool of type 580 (3D probe, turning) is selected.
- The tool is activated in the current NC channel.

Measure tool

To measure tools, an appropriate probe must be located in the machine space so that this can be reliably and safely reached with a tool in the spindle.

The following tool types are supported with measure tool:

- Milling technology: Tool types 1xx and 2xx
- Turning technology: Tool type 5xx, 1xx, 2xx

For the specified tool types, the tool lengths and the tool radii can be measured.

Using the following channel-specific setting data, you can adapt the workpiece and tool measuring in the "Program" operating area. However, a basic change is not required.

SD55613 $SCS_MEA_RESULT_DISPLAY	
Selects the screen display of the measuring result	
= 0	No screen display of the measuring result (default setting).
= 1	Screen display of the measuring result is displayed for 8 seconds.
= 3	The measuring cycle stops an internal machine data, the measuring result is statically displayed on the screen! Continue with NC start, the measuring result screen is deselected.
= 4	The measuring result is only displayed on the screen for cycle alarms 61303, 61304, 61305, 61306. Continue with NC start, the measuring result display on the screen is deselected.

SD55623 $SCS_MEA_EMPIRIC_VALUE[i]	
Empirical values	
= 0	Default setting

SD55618 $SCS_MEA_SIM_ENABLE	
Simulation of measuring cycles	
= 0	The measuring cycles are skipped when simulation is called (default setting).
= 1	The measuring cycles are run through when simulation is called: • No corrections are performed and there is no logging. • The measurement result is not displayed on the screen.

SD55619 $SCS_MEA_SIM_MEASURE_DIFF	
Value for simulated measuring difference.	
= 0	For the simulation, a setpoint / actual value difference can be entered (default setting).

SD55600 $SCS_MEA_COLLISION_MONITORING	
Internal intermediate positioning of the measuring cycle is monitored for deflection of the probe.	
= 0	Without collision monitoring
= 1	With collision monitoring (default setting).

Configuring input masks for measuring cycles in the program editor

You can extend or restrict the entries and measuring variants with the following user variable _MZ_MASK[i]. The GUD is displayed in the operating area "Parameter" → "User variable" → "Global GUD" or "Channel GUD".

GUD parameter		Function
_MZ_MASK[2]		Input field for the number of measurements and measuring speed:
	= 0	Without input field (default setting)
	= 1	With input field
_MZ_MASK[5]		Selection box for probe type:
	= 0	Multiprobe (default setting)
	= 1	Monoprobe

Note

SGUD parameters

The SGUD parameters are not defined via the user interface.

Program the parameters with value assignment in the appropriate NC program in the "MDA" or "AUTOMATIC" mode.

Example: _MZ_MASK[2]=1

The value of parameter _MZ_MASK is also retained after NC program end, reset and power OFF/ON!

If you have created the prerequisites described in this section and you have set and checked the machine and setting data, you can perform measurements on the machine with a workpiece or tool probe in the AUTOMATIC mode!

See also

If you wish to set-up workpiece measurement on a lathe, then additional settings are required:

- AUTO: Measure workpiece during turning (Page 249)

If you wish to set up tool measurement using a tool probe, then you must also make additional settings:

- AUTO: Measure tool during milling (Page 250)
- AUTO: Measure tool during turning (CYCLE982) (Page 258)

A description is provided in the following sections as to which settings you can make in order to adapt measuring to the specific requirements of the machine.

7.7.8 AUTO: General settings for the workpiece measurement

Requirement

You have already made the settings from the following chapter: General settings for measuring (Page 231)

Settings

Using the following setting data, you can appropriately adapt workpiece measurement to the specific requirements in the "Program" operating area.

General setting data:

SD54655 $SNS_MEA_REPEAT_ACTIVE	
Repeating measurements for alarm - dimension difference or confidence range.	
= 0	Measurement is not repeated (default setting)
= 1	Measurements are repeated a maximum of four times

SD54656 $SNS_MEA_REPEAT_WITH_M0	
M0 for repeating measurement and alarm - dimension difference or confidence range.	
= 0	No M0 for alarm (default setting)
= 1	With M0 for alarm

SD54657 $SNS_MEA_TOL_ALARM_SET_M0	
M0 for alarm - oversize or undersize	
= 0	No M0 for alarm (default setting)
= 1	With M0 for alarm

Channel-specific setting data:

SD55606 $SCS_MEA_NUM_OF_MEASURE	
Number of repeated measurements at a measuring position, when the probe does not switch.	
= 0	Maximum of 5 measurement attempts (default setting)
= 1	Only 1 measurement attempt

SD55608 $SCS_MEA_RETRACTION_FEED	
Retraction velocity from the measuring point.	
= 0	Retraction velocity corresponding to the intermediate positioning (default setting)
= 1	Retraction with percentage rapid traverse corresponding to the settings from SD55630 $SCS_MEA_FEED_RAPID_IN_PERCENT. The setting of the channel-specific setting data SD55600 $SCS_MEA_COLLISION_MONITORING must be = 1.

SD55610 $SCS_MEA_FEED_TYP	
Selection of the measuring feedrate.	
= 0	Measuring with the measuring feedrate, default setting.
= 1	1st measurement with the feedrate corresponding to the channel-specific setting data SD55633 $SCS_MEA_FEED_FAST_MEASURE. 2nd measurement with measuring feedrate.

SD55630 $SCS_MEA_FEED_RAPID_IN_PERCENT	
Percentage reduction	
= 50	Default setting of the percentage reduction of the rapid traverse velocity for internal cycle intermediate positioning without collision monitoring. The setting of the channel-specific setting data SD55600 SCS_MEA_COLLISION_MONITORING must be = 0.

SD55631 $SCS_MEA_FEED_PLANE_VALUE	
Feedrate of the intermediate positioning in the working plane with active collision monitoring.	
= 1000	Default setting

SD55632 $SCS_MEA_FEED_FEEDAX_VALUE	
Feedrate of the intermediate positioning in the infeed axis with active collision monitoring.	
= 1000	Default setting

SD55633 $SCS_MEA_FEED_MEASURE	
Fast measuring feedrate.	
= 900	Default setting

7.7.9 AUTO: Measure workpiece during milling

Settings

Measuring in the "Program" operating area can be adapted corresponding to the specific requirements using the channel-specific setting data.

SD54660 $SNS_MEA_PROBE_BALL_RAD_IN_TOA	
Acceptance of the calibrated probe ball radius into the probe tool data.	
= 0	Without acceptance (default setting)
= 1	With acceptance

SD55602 $SCS_MEA_COUPL_SPIND_COORD	
Coupling of the spindle alignment with coordinate rotation in the active plane.	
= 0	No coupling of the spindle position (probe in the spindle) with coordinate rotation around the infeed axis (default setting).
= 1	Coupling of the spindle position (probe in the spindle) with coordinate rotation around the infeed axis.

SD55625 $SCS_MEA_AVERAGE_VALUE[i]	
Number of mean values.	
= 0	Default setting

The following setting data is written by the measuring cycles with the "calibrate" measuring function. User parameterization is not necessary here.

SD54600 $SNS_MEA_WP_BALL_DIAM[i]	Effective diameter of the probe ball of the workpiece probe.
SD54601 $SNS_MEA_WP_TRIG_MINUS_DIR_AX1[i]	Trigger point minus direction, 1st measuring axis in the plane.
SD54602 $SNS_MEA_WP_TRIG_PLUS_DIR_AX1[i]	Trigger point plus direction, 1st measuring axis in the plane.
SD54603 $SNS_MEA_WP_TRIG_MINUS_DIR_AX2[i]	Trigger point minus direction, 2nd measuring axis in the plane.
SD54604 $SNS_MEA_WP_TRIG_PLUS_DIR_AX2[i]	Trigger point plus direction, 2nd measuring axis in the plane.
SD54605 $SNS_MEA_WP_TRIG_MINUS_DIR_AX3[i]	Trigger point plus direction, 3rd measuring axis in the plane.
SD54606 $SNS_MEA_WP_TRIG_PLUS_DIR_AX3[i]	Trigger point minus direction, 3rd measuring axis opposite to the tool direction. In the default case = 0.
SD54607 $SNS_MEA_WP_POS_DEV_AX1[i]	Positional deviation. 1st measuring axis in the plane.
SD54608 $SNS_MEA_WP_POS_DEV_AX2[i]	Positional deviation. 2nd measuring axis in the plane.

However, after the probe has been calibrated, you can check these values and if required, evaluate the probe quality, e.g. for position deviations, no values > 0.1 mm should be reached. Otherwise, the probe must be mechanically readjusted.

Note

Observe the manufacturer's instructions for the probe.

7.7.10 AUTO: Measure workpiece during turning

Requirement

MD51610 $MNS_MEA_TOOLCARR_ENABLE	
Support of a probe or tool that is positioned with a toolholder that can be oriented (swivel-mounted).	
= 0	No support (default setting)
= 1	With support

MD52605 $MNS_MEA_TURN_CYC_SPECIAL_MODE	
Function for turning. Measure in the Y axis (3rd axis) and correct in the X axis (face axis).	
= 0	Default setting

Calibration data of the tool probe in relation to the MCS

Before calibration is started, the position of the tool probe in the machine coordinate system (MCS) must be entered into the following general setting data.

SD54615 $SNS_MEA_CAL_EDGE_BASE_AX1[i]	
Calibration slot base referred to the 1st measuring axis.	
= 0	

SD54617 $SNS_MEA_CAL_EDGE_PLUS_DIR_AX1[i]	
Calibration slot edge in the positive direction of the first measuring axis.	
= 0	

SD54618 $SNS_MEA_CAL_EDGE_MINUS_DIR_AX1[i]	
Calibration slot edge in the negative direction of the first measuring axis.	
= 0	

SD54619 $SNS_MEA_CAL_EDGE_BASE_AX2[i]	
Calibration slot base referred to the 2nd measuring axis.	
= 0	

SD54620 $SNS_MEA_CAL_EDGE_UPPERE_AX2[i]	
Upper calibration slot edge referred to the 2nd measuring axis.	
= 0	

SD54621 $SNS_MEA_CAL_EDGE_PLUS_DIR_AX2[i]	
Calibration slot edge in the positive direction of the 2nd measuring axis.	
= 0	

SD54422 $SNS_MEA_CAL_EDGE_MINUS_DIR_AX2[i]	
Calibration slot edge in the negative direction of the 2nd measuring axis.	
= 0	

Note

For a standard lathe with axes X and Z (G18), axis Z is the 1st measuring axis and axis X is the 2nd measuring axis.

7.7.11 AUTO: Measure tool during milling

Calibration data of the tool probe in relation to the MCS

Before calibration is started, the position of the tool probe in the machine coordinate system (MCS) must be entered into the following general setting data. In this case, the reference point is the outer diameter or the tool length of the active tool in the spindle. If there is no tool in the spindle, the reference points are the spindle center point and the tool reference point at the spindle.

Note

If you have already calibrated the tool probe in the JOG mode, then the calibration data has already been correctly entered.

The setting of the following machine and setting data must match:

- MD51776 $MNS_J_MEA_T_PROBE_ALLOW_AX_DIR[ii]
- SD54632 $SNS_MEA_TP_AX_DIR_AUTO_CAL[i]

Index [i] stands for the number of the actual data field (_PRNUM-1).

SD54625 $SNS_MEA_TP_TRIG_MINUS_DIR_AX1[i]	
Trigger point of the 1st measuring axis in the negative direction.	
= 0	

SD54626 $SNS_MEA_TP_TRIG_PLUS_DIR_AX1[i]	
Trigger point of the 1st measuring axis in the positive direction.	
= 0	

SD54627 $SNS_MEA_TP_TRIG_MINUS_DIR_AX2[i]	
Trigger point of the 2nd measuring axis in the negative direction.	
= 0	

SD54628 $SNS_MEA_TP_TRIG_PLUS_DIR_AX2[i]	
Trigger point of the 2nd measuring axis in the positive direction.	
= 0	

SD54629[i] $SNS_MEA_TP_TRIG_MINUS_DIR_AX3[i]	
Trigger point of the 3rd measuring axis in the negative direction.	
= 0	

SD54630 $SNS_MEA_TP_TRIG_PLUS_DIR_AX3[i]	
Trigger point of the 3rd measuring axis in the positive direction.	
= 0	

SD54631 $SNS_MEA_TP_EDGE_DISK_SIZE[i]	
Tool probe, edge length / disk diameter.	
= 0	

SD54632 $SNS_MEA_TP_AX_DIR_AUTO_CAL[i]	
Axes and directions for automatic calibration.	
= 133	

The general setting data SD54632 $SNS_MEA_TP_AX_DIR_AUTO_CAL, is used to define in which direction the tool probe on the axis can be calibrated:

Decimal place		
ONES	1st axis	
	= 0	axis not possible
	= 1	only minus direction
	= 2	only plus direction
	= 3	both directions
TENS	2nd axis	
	= 0	axis not possible
	= 1	only minus direction
	= 2	only plus direction
	= 3	both directions

Decimal place		
HUNDREDS	3rd axis	
	= 0	axis not possible
	= 1	only minus direction
	= 2	only plus direction
	= 3	both directions

Example

If the general machine data SD54632 $SNS_MEA_TP_AX_DIR_AUTO_CAL = 123 is set, then the tool probe is calibrated in the G17 plane:

- 1st axis: Z only in minus direction
- 2nd axis: Y only in plus direction
- 3rd axis: X in both directions

SD54633 $SNS_MEA_TP_TYPE[i]	
Probe version	
= 0	Cube (default setting)
= 101	Disk in XY, working plane G17
= 201	Disk in ZX, working plane G18
= 301	Disk in YZ, working plane G19

SD54634 $SNS_MEA_TP_CAL_MEASURE_DEPTH[i]	
Distance between the upper edge of the tool probe and lower edge of the tool (calibration depth, measuring depth for milling radius).	
= 2	Default setting

Calibration data of the tool probe in relation to the WCS

Before calibration is started, the position of the tool probe in the workpiece coordinate system (WCS) must be roughly entered into the following general setting data. In this case, the reference points are the outer diameter and the tool length of the active tool in the spindle. If there is no tool in the spindle, the reference points are the spindle center point and the tool reference point at the spindle.

NOTICE
When measuring tools, ensure that the data of the adjustable work offset and the basic reference always correspond to the data when calibrating (measuring in WCS). Always make measurements and calibrate with the same adjustable work offset.

SD54640 $SNS_MEA_TPW_TRIG_MINUS_DIR_AX1[i]	
Trigger point of the 1st measuring axis in the negative direction.	
= 0	

SD54641 $SNS_MEA_TPW_TRIG_PLUS_DIR_AX1[i]	
Trigger point of the 1st measuring axis in the positive direction.	
= 0	

SD54642 $SNS_MEA_TPW_TRIG_MINUS_DIR_AX2[i]	
Trigger point of the 2nd measuring axis in the negative direction.	
= 0	

SD54643 $SNS_MEA_TPW_TRIG_PLUS_DIR_AX2[i]	
Trigger point of the 2nd measuring axis in the positive direction.	
= 0	

SD54644 $SNS_MEA_TPW_TRIG_MINUS_DIR_AX3[i]	
Trigger point of the 3rd measuring axis in the negative direction.	
= 0	

SD54645 $SNS_MEA_TPW_TRIG_PLUS_DIR_AX3[i]	
Trigger point of the 3rd measuring axis in the positive direction.	
= 0	

SD54646 $SNS_MEA_TPW_EDGE_DISK_SIZE[i]	
Tool probe, edge length / disk diameter.	
= 0	

SD54647 $SNS_MEA_TPW_AX_DIR_AUTO_CAL[i]	
Automatic calibration, tool probe, enable axes/directions.	
= 133	(default setting)

The following general setting data is used to define the directions in which a tool probe can be calibrated for the particular axes.

Decimal place		
ONES	1st axis	
	= 0	axis not possible
	= 1	only minus direction
	= 2	only plus direction
	= 3	both directions
TENS	2nd axis	
	= 0	axis not possible
	= 1	only minus direction
	= 2	only plus direction
	= 3	both directions
HUNDREDS	3rd axis	
	= 0	axis not possible
	= 1	only minus direction
	= 2	only plus direction
	= 3	both directions

Example

If the general machine data SD54647 $SNS_MEA_TPW_AX_DIR_AUTO_CAL = 123 is set, the tool probe is calibrated in the G17 plane as follows:

- 1st axis: Z only in minus direction
- 2nd axis: Y only in plus direction
- 3rd axis: X in both directions

SD54648 $SNS_MEA_TPW_TYPE[i]	
Probe version	
= 0	Cube (default setting)
= 101	Disk in XY, working plane G17
= 201	Disk in ZX, working plane G18
= 301	Disk in YZ, working plane G19

SD54649 $SNS_MEA_TPW_CAL_MEASURE_DEPTH[i]	
Distance between the upper edge of the tool probe and lower edge of the tool (calibration depth, measuring depth for milling radius).	
= 2	Default setting

Monitoring when measuring with a rotating spindle

Setting data settings:

SD54670 $SNS_MEA_CM_MAX_PERI_SPEED[0]	
Maximum permissible peripheral speed of the tool to be measured.	
= 100	Default setting

SD54671 $SNS_MEA_CM_MAX_REVOLUTIONS[0]	
Maximum permissible tool speed of the tool to be measured. The speed is automatically reduced when exceeded.	
= 1000	Default setting

SD54672 $SNS_MEA_CM_MAX_FEEDRATE[0]	
Maximum permissible feedrate to probe the tool to be measured at the probe.	
= 20	Default setting

SD54673 $SNS_MEA_CM_MIN_FEEDRATE[0]	
Minimum feedrate for the first probing of the tool to be measured at the probe. This avoids excessively small feedrates for large tool radii.	
= 1	Default setting

SD54674 $SNS_MEA_CM_SPIND_ROT_DIR[0]	
Spindle direction of rotation to measure tools.	
4 = M4	Default setting

NOTICE
If the spindle is already rotating when the measuring cycle is called, this direction of rotation remains independent of the setting of this data.

SD54675 $SNS_MEA_CM_FEEDFACTOR_1[0]	
Feedrate factor 1	
= 10	Default setting
= 0	Only single probing with the feedrate calculated by the cycle. However, as a minimum, the value from SD54673[0] $SNS_MEA_CM_MIN_FEEDRATE.
≥ 1	First probing with feedrate. But at least with the value of SD54673[0] $SNS_MEA_CM_MIN_FEEDRATE) SD54675[0] $SNS_MEA_CM_FEEDFACTOR_1

SD54676 $SNS_MEA_CM_FEEDFACTOR_2[0]	
Feedrate factor 2	
= 0	Second probing with the feedrate calculated by the cycle, only effective for SD54673 $SNS_MEA_CM_FEEDFACTOR_1[0] > 0 (default setting)
≥ 1	Second probing with the calculated feedrate from SD54673 $SNS_MEA_CM_MIN_FEEDRATE[0] feedrate factor 2. Third probing with the calculated feedrate.

NOTICE
Feedrate factor 2 should be less than feedrate factor 1.

SD54677 $SNS_MEA_CM_MEASURING_ACCURACY[0]	
Specified measuring accuracy: The value of this parameter always refers to the last probing of the tool at the probe.	
= 0.005	Default setting

Measured value correction using correction tables

SD54691 $SNS_MEA_T_PROBE_OFFSET	
Activates the measuring result correction	
= 0	No data (default setting)
= 1	Correction in the cycle. This is only effective if SD54690 $SNS_MEA_T_PROBE_MANUFACTURER>0.
= 2	Correction using user-defined correction table

SD54689 $SNS_MEA_T_PROBE_MANUFACTURER	
Activate pre-configured compensation tables for several tool probe models (customer-specific).	
= 0	No data (default setting)
= 1	TT130 (Heidenhain)
= 2	TS27R (Renishaw)

Correction values for users

For the general setting data SD54691 $SNS_MEA_T_PROBE_OFFSET= 2, the following settings apply:

SD54695 to SD54700	Compensation values for radius measurement
SD54705 to SD54710	Compensation values for length measurement

SD54695 $SNS_MEA_RESULT_OFFSET_TAB_RAD1[i] **SD54705 $SNS_MEA_RESULT_OFFSET_TAB_LEN1[i]**	Radius measurement Length measurement
= 0	0
= 1	1st radius
= 2	2nd radius
= 3	3rd radius
= 4	4th radius

SD54696 $SNS_MEA_RESULT_OFFSET_TAB_RAD2[i] **SD54706 $SNS_MEA_RESULT_OFFSET_TAB_LEN2[i]**	Radius measurement Length measurement
= 0	1st peripheral speed.
= 1	Correction value for 1st radius/length measurement.
= 2	Correction value for 2nd radius/length measurement.
= 3	Correction value for 3rd radius/length measurement.
= 4	Correction value for 4th radius/length measurement.

SD54697 $SNS_MEA_RESULT_OFFSET_TAB_RAD3[i] **SD54707 $SNS_MEA_RESULT_OFFSET_TAB_LEN3[i]**	Radius measurement Length measurement
= 0	2nd peripheral speed.
= 1	Correction value for 1st radius/length measurement.
= 2	Correction value for 2nd radius/length measurement.
= 3	Correction value for 3rd radius/length measurement.
= 4	Correction value for 4th radius/length measurement.

SD54698 $SNS_MEA_RESULT_OFFSET_TAB_RAD4[i] **SD54708 $SNS_MEA_RESULT_OFFSET_TAB_LEN4[i]**	Radius measurement Length measurement
= 0	3rd peripheral speed.
= 1	Correction value for 1st radius/length measurement.
= 2	Correction value for 2nd radius/length measurement.
= 3	Correction value for 3rd radius/length measurement.
= 4	Correction value for 4th radius/length measurement.

SD54699 $SNS_MEA_RESULT_OFFSET_TAB_RAD5[i] **SD54709 $SNS_MEA_RESULT_OFFSET_TAB_LEN5[i]**	Radius measurement Length measurement
= 0	4th peripheral speed.
= 1	Correction value for 1st radius/length measurement.
= 2	Correction value for 2nd radius/length measurement.
= 3	Correction value for 3rd radius/length measurement.
= 4	Correction value for 4th radius/length measurement.

SD54700 $SNS_MEA_RESULT_OFFSET_TAB_RAD6[i] SD54710 $SNS_MEA_RESULT_OFFSET_TAB_LEN6[i]	Radius measurement Length measurement
= 0	5th peripheral speed.
= 1	Correction value for 1st radius/length measurement.
= 2	Correction value for 2nd radius/length measurement.
= 3	Correction value for 3rd radius/length measurement.
= 4	Correction value for 4th radius/length measurement.

7.7.12 AUTO: Measure tool during turning (CYCLE982)

Calibration data of the tool probe

Measurement in relation to the machine coordinate system (MCS):

Before calibration is started, the position of the tool probe in the machine coordinate system (MCS) must be entered into the following general setting data.

SD54625 $SNS_MEA_TP_TRIG_MINUS_DIR_AX1[i]	
Trigger point in minus direction of the 1st measuring axis (for G18 Z)	
= 0	

SD54626 $SNS_MEA_TP_TRIG_PLUS_DIR_AX1[i]	
Trigger point in plus direction of the 1st measuring axis (for G18 Z)	
= 0	

SD54627 $SNS_MEA_TP_TRIG_MINUS_DIR_AX2[i]	
Trigger point in minus direction of the 2nd measuring axis (for G18 X).	
= 0	

SD54628 $SNS_MEA_TP_TRIG_PLUS_DIR_AX2[i]	
Trigger point in plus direction of the 2nd measuring axis (for G18 X).	
= 0	

Index [i] stands for the number of the actual data field (_PRNUM-1)

Measurement in relation to the workpiece coordinate system (WCS):

Before calibration is started, the position of the tool probe in the workpiece coordinate system (WCS) must be roughly entered into the following general setting data. In this case, the reference point is the outer diameter or the tool length of the active tool in the spindle.

SD54640 $SNS_MEA_TPW_TRIG_MINUS_DIR_AX1[i]	
Trigger point minus direction of the 1st measuring axis (for G18 Z).	
= 0	

SD54641 $SNS_MEA_TPW_TRIG_PLUS_DIR_AX1[i]	
Trigger point plus direction of the 1st measuring axis (for G18 Z).	
= 0	

SD54642 $SNS_MEA_TPW_TRIG_MINUS_DIR_AX2[i]	
Trigger point minus direction of the 2nd measuring axis (for G18 X).	
= 0	

SD54643 $SNS_MEA_TPW_TRIG_PLUS_DIR_AX2[i]	
Trigger point plus direction of the 2nd measuring axis (for G18 X).	
= 0	

Index [i] stands for the number of the actual data field (_PRNUM-1)

Measure tool with "Toolholder that can be oriented" or "Swivel tool"

MD51610 $MNS_MEA_TOOLCARR_ENABLE	
Support for toolholders that can be orientated.	
= 0	No support for toolholders that can be orientated (default setting).
= 1	With support of a measuring probe or tool, positioned using a tool holder that can be oriented (kinematics type "T") referred to the special holder positions 0°, 90°, 180° and 270°.

If the general machine data MD51610 $MNS_MEA_TOOLCARR_ENABLE = 1, then the following setting applies:

MD51618 $MNS_MEA_CM_ROT_AX_POS_TOL	
Tolerance parameter for rotary axis settings.	
= 0.5	Default setting

The real angular position of rotary axes can deviate from that programmed (exact stop fine window). This deviation depends on the position control properties of the axis. The maximum deviation that can be expected at a specific axis should be entered into the parameter. If the tolerance is exceeded, the following alarm is output:

61442 Toolholder not parallel to the geometry axes.

Commissioning the PLC (Part 2) 8

8.1 Service Planner

Overview

Using the service planner dialog at the user interface or programming tool, time intervals and alarm sequences for the tasks to be processed (mainly machine maintenance tasks) can be edited, started, deactivated or reactivated.

The numeric data of the tasks is organized in data blocks and provided in the user interface for the HMI, PLC user program and programming tool. The designations of the particular task are managed and edited by the operating software and displayed together with the numerical data.

The PLC firmware accesses the user interface data blocks, processes the data and then provides the results in the form of remaining times as well as warnings and alarms in data blocks. The Service Planner is processed every minute in the PLC firmware. When the control is switched off, the actual data of the maintenance tasks is frozen. When the control is switched on, processing is continued with these retentively saved values.

The PLC user program evaluates the actual data and generates warning and alarm messages in numeric form either with or without Power OFF status. The alarm handler converts these messages with the appropriate PLC alarm text file oem_alarm_plc_<lng>.ts to a message for the operator which is displayed on the user interface (<lng> currently set language) and which can be logged, if required.

Configuration in the system

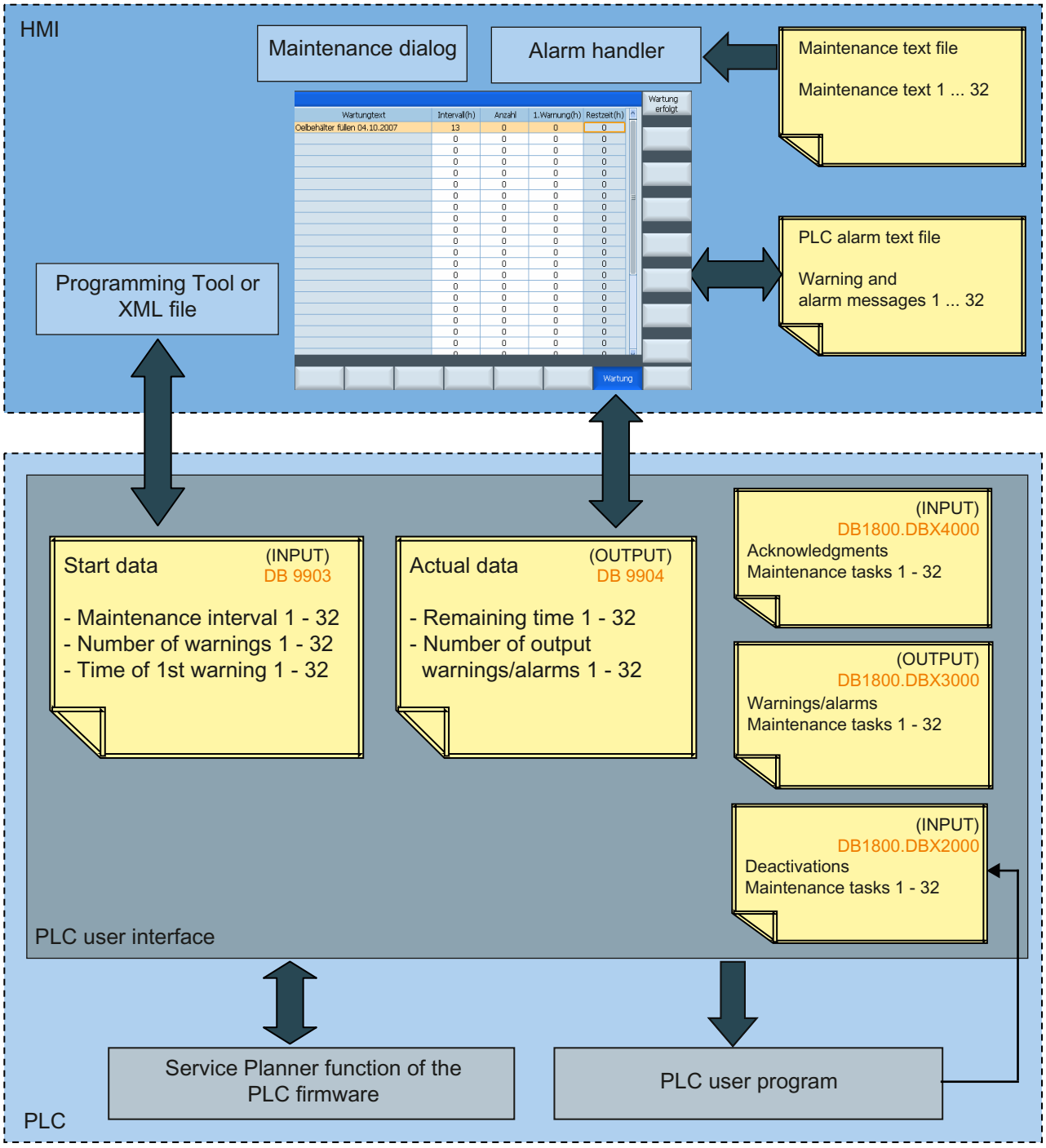


Figure 8-1 Service Planner: Configuration

8.1.1 Interfaces in the PLC user program

PLC user program

The PLC user program operates the user interface. This includes, in particular, the bit interfaces of the DB1800 and the evaluation of the remaining time in DB9904. You must ensure that appropriate messages are displayed for the warnings and alarms.

During the configuration of these messages, so-called alarm responses can be selected, e.g. the machine is "disabled" after a Power OFF message.

The alarm messages are configured in accordance with the specifications of the SINUMERIK 828D PLC messages. The texts are entered with the alarm text editor. Then the texts are available for the operating software:

- Evaluation of the alarm and actual data with the aim of generating PLC warning and alarm messages. Additional signals can be included in the evaluation logic.
- Optional linking of the deactivation bits with bit memory or I/O signals.

Note

A sample program is supplied in the PLC function library. This can be adapted by machine manufacturers to meet their requirements.

Interfaces to the PLC

The following data areas are available in the user interface:

Data block	Meaning
DB9903	Initial data
DB9904	Actual data
DB1800.DBB2000	Deactivate tasks
DB1800.DBB3000	Alarms
DB1800.DBB4000	Acknowledgments
DB1800.DBB5000	Acknowledgment block

DB9903: Initial data

DB9903	Initial data table [r16]							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBW0	Interval 1 [h]							
DBW2	Time of first warning 1 [h]							
DBW4	Number of warnings to be output 1							
DBW6	Reserved 1							
DBW8	Interval 2 [h]							
DBW10	Time of first warning 2 [h]							
DBW12	Number of warnings to be output 2							
DBW14	Reserved 2							
...	...							
DBW248	Interval 32 [h]							
DBW250	Time of first warning 32 [h]							
DBW252	Number of warnings to be output 32							
DBW254	Reserved 32							

Designation	Meaning
Interval	Number of hours after which the maintenance must be performed. When this time expires, the warning or alarm bit belonging to the task is set for the last time.
Time of the first warning	Number of hours after which the first warning is output. This time must be greater than or equal to the interval.
Number of warnings to be output	Number of n warnings to be output before the alarm. (The alarm bit is therefore set maximum (n+1)-times, i.e n-times as warning and 1-time as alarm.)
Reserved	Reserved for expansions.

Example:

Interval = 100

Time of the 1st warning = 80

Number of warnings to be output = 2

After the task is started, the warning/alarm bit is output for the first time after 80 hours, a second time after a further 10 hours (i.e. after a total of 90 hours), and the warning/alarm bit is set for the last time after 100 hours.

DB9904: Actual data

DB9904	Actual data table [r16]							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBW0	Remaining time 1 [h]							
DBW2	Number of warnings output 1							
DBW4	Reserved_1 1							
DBW6	Reserved_2 1							
DBW8	Remaining time 2 [h]							
DBW10	Number of warnings output 2							
DBW12	Reserved_1 2							
DBW14	Reserved_2 2							
...	...							
DBW248	Remaining time 32 [h]							
DBW250	Number of warnings output 32							
DBW252	Reserved_1 32							
DBW254	Reserved_2 32							

Designation	Meaning
Remaining time	Number of hours remaining after the start of the task until it expires. Remaining time ≠ 0 and associated alarm bit set: Warning Remaining time = 0 and associated alarm bit set: Alarm
Number of warnings output	Number n of warnings that have already been output. If the interval has expired completely, the output value is (n+1): n = "number of warnings to be output" 1 = alarm at the end of the interval
Reserved_1, ~_2	Reserved for expansions.

Example:

Interval = 100, time of the 1st warning = 80, number of warnings to be output = 2

After the task is started, the remaining time is decremented every hour.

- After 80 hours, the remaining time is 20 hours and the number of warnings that have been output is increased from 0 to 1.
- After a further 10 hours (i.e. a total of 90 hours), the remaining time is 10 hours and the number of warnings that have been output is increased from 1 to 2.
- After 100 hours, the remaining time is 0 and the number of warnings that have been output is 3 (= 2 warnings plus 1 alarm).

DB1800: Acknowledgments

DB1800	Acknowledgments [r/w]							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBB4000	Acknowl-edgment 8	Acknowl-edgment 7	Acknowl-edgment 6	Acknowl-edgment 5	Acknowl-edgment 4	Acknowl-edgment 3	Acknowl-edgment 2	Acknowl-edgment 1
DBB4001	Acknowl-edgment 16	Acknowl-edgment 15	Acknowl-edgment 14	Acknowl-edgment 13	Acknowl-edgment 12	Acknowl-edgment 11	Acknowl-edgment 10	Acknowl-edgment 9
DBB4002	Acknowl-edgment 24	Acknowl-edgment 23	Acknowl-edgment 22	Acknowl-edgment 21	Acknowl-edgment 20	Acknowl-edgment 19	Acknowl-edgment 18	Acknowl-edgment 17
DBB4003	Acknowl-edgment 32	Acknowl-edgment 31	Acknowl-edgment 30	Acknowl-edgment 29	Acknowl-edgment 28	Acknowl-edgment 27	Acknowl-edgment 26	Acknowl-edgment 25

Designation	Meaning
Acknowledgment n	The acknowledgment bit assigned to task n: Under the precondition that the corresponding acknowledgment blocking bit is not set, the setting of the acknowledgment bit restarts the task and, in particular, the actual data of the task is set: • Remaining time = interval • Number of warnings output = 0 The bit is automatically reset at the end of the PLC cycle.

Example:

Interval = 100, time of the 1st warning = 80, number of warnings to be output = 2

After the setting of the associated acknowledgment bit, the remaining time is set to the interval time and the number of output warnings is zero - assuming the associated acknowledgment blocking bit is not set.

DB1800: Alarms

DB1800	Warnings/Alarms [r]							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBB3000	Alarm 8	Alarm 7	Alarm 6	Alarm 5	Alarm 4	Alarm 3	Alarm 2	Alarm 1
DBB3001	Alarm 16	Alarm 15	Alarm 14	Alarm 13	Alarm 12	Alarm 11	Alarm 10	Alarm 9
DBB3002	Alarm 24	Alarm 23	Alarm 22	Alarm 21	Alarm 20	Alarm 19	Alarm 18	Alarm 17
DBB3003	Alarm 32	Alarm 31	Alarm 30	Alarm 29	Alarm 28	Alarm 27	Alarm 26	Alarm 25

Designation	Meaning
Alarm n	The alarm bit assigned to task n. The bit is set each time for one PLC cycle: As warning (remaining time ≠ 0) and as alarm (remaining time = 0).

DB1800: Deactivate tasks

DB1800	Task deactivation [r/w]							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBB2000	Deactivation 8	Deactivation 7	Deactivation 6	Deactivation 5	Deactivation 4	Deactivation 3	Deactivation 2	Deactivation 1
DBB2001	Deactivation 16	Deactivation 15	Deactivation 14	Deactivation 13	Deactivation 12	Deactivation 11	Deactivation 10	Deactivation 9
DBB2002	Deactivation 24	Deactivation 23	Deactivation 22	Deactivation 21	Deactivation 20	Deactivation 19	Deactivation 18	Deactivation 17
DBB2003	Deactivation 32	Deactivation 31	Deactivation 30	Deactivation 29	Deactivation 28	Deactivation 27	Deactivation 26	Deactivation 25

Designation	Meaning
Deactivation n	The deactivation bit assigned to task n. If the bit is set via the HMI or from the PLC user program, the current state of task n is frozen and no longer processed. **TRUE:** Task deactivated **FALSE:** Task active This means it is possible, for example, to adapt the maintenance interval according to the actual runtime of the modules.

DB1800: Acknowledgment block

DB1800	Acknowledgment block [r/w]							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBB5000	Acknowledgment block 8	Acknowledgment block 7	Acknowledgment block 6	Acknowledgment block 5	Acknowledgment block 4	Acknowledgment block 3	Acknowledgment block 2	Acknowledgment block 1
DBB5001	Acknowledgment block 16	Acknowledgment block 15	Acknowledgment block 14	Acknowledgment block 13	Acknowledgment block 12	Acknowledgment block 11	Acknowledgment block 10	Acknowledgment block 9
DBB5002	Acknowledgment block 24	Acknowledgment block 23	Acknowledgment block 22	Acknowledgment block 21	Acknowledgment block 20	Acknowledgment block 19	Acknowledgment block 18	Acknowledgment block 17
DBB5003	Acknowledgment block 32	Acknowledgment block 31	Acknowledgment block 30	Acknowledgment block 29	Acknowledgment block 28	Acknowledgment block 27	Acknowledgment block 26	Acknowledgment block 25

Designation	Meaning
Acknowledgment block n	The acknowledgment blocking bit assigned to task n. If the bit is set via the HMI or from the PLC user program, the task is not acknowledged even when the acknowledgment bit is set. **TRUE:** Acknowledgment of the task blocked **FALSE:** Acknowledgment of the task permitted In this way, it is possible, for example, to incorporate a sensor in the PLC user program, which signals that the maintenance task has been performed, and, if required, block the acknowledgment.

8.1.2 Interfaces to the operating software

Overview

To process these service/maintenance tasks, you have the following options:

- Managing the data in the PLC user program using the programming tool.
- Maintenance tasks are created via an XML interface.

Using this segmentation, inconsistencies between the PLC block and the XML scripts are avoided if the PLC block is changed in the PLC user program via the programming tool or as a result of series commissioning.

Configuration

You can configure up to 32 maintenance tasks. The following columns are shown in the configuring mode:

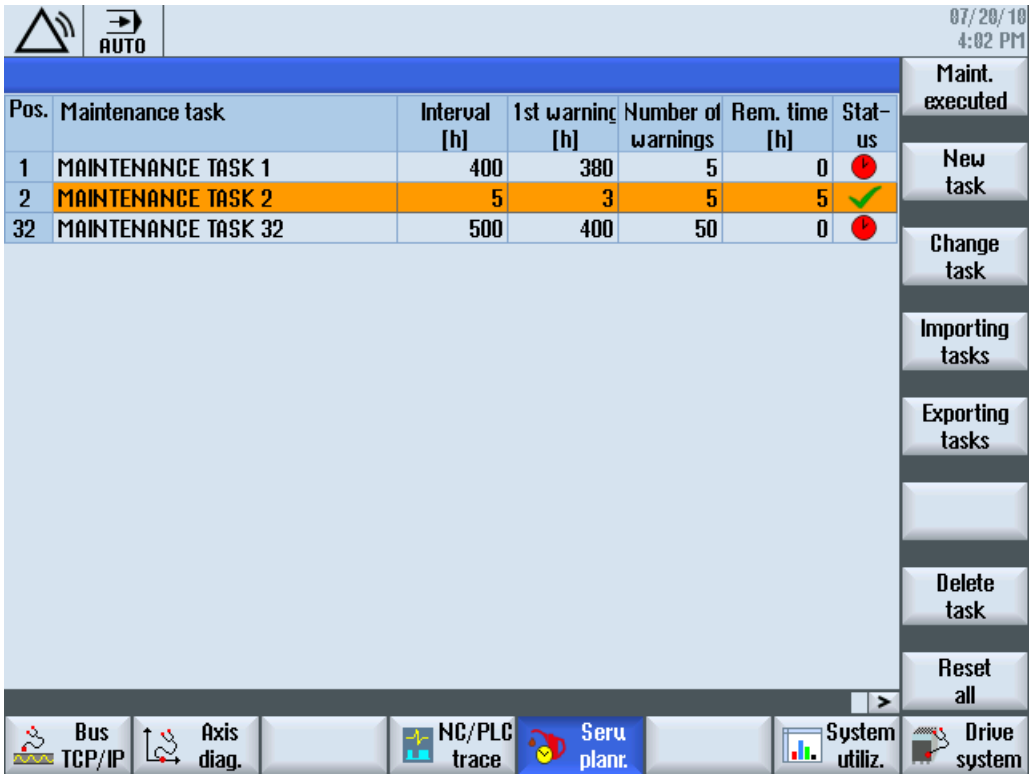
Designation of the column	Meaning
Maintenance task	Name of the maintenance task.
Interval [h]	Maximum time until the next maintenance in hours; if this value ≠ 0, this data set is accepted by the PLC as a valid maintenance task.
1st warning [h]	Time in hours after which the first warning is displayed; this value must be less than that of the interval.
Number of warnings	Number of warnings that are output by the PLC before the PLC sets the alarm bit for the last time after the interval has expired (remaining time == 0).
Remaining time [h]	Time until the interval expires in hours.
Status	• A green check mark indicates that the time up to the next maintenance is still running. • A red clock symbol indicates that there is a maintenance task to be performed.

The dialog is called with different contents depending on the access level.

Configuration mode

Access level 2: Service

Maintenance tasks can be created, changed and also deleted in this mode. The maintenance tasks can also be acknowledged. All columns are visible. Navigation between the columns is with <Tab> or <Key Left/Right>.



Note

Number assignment

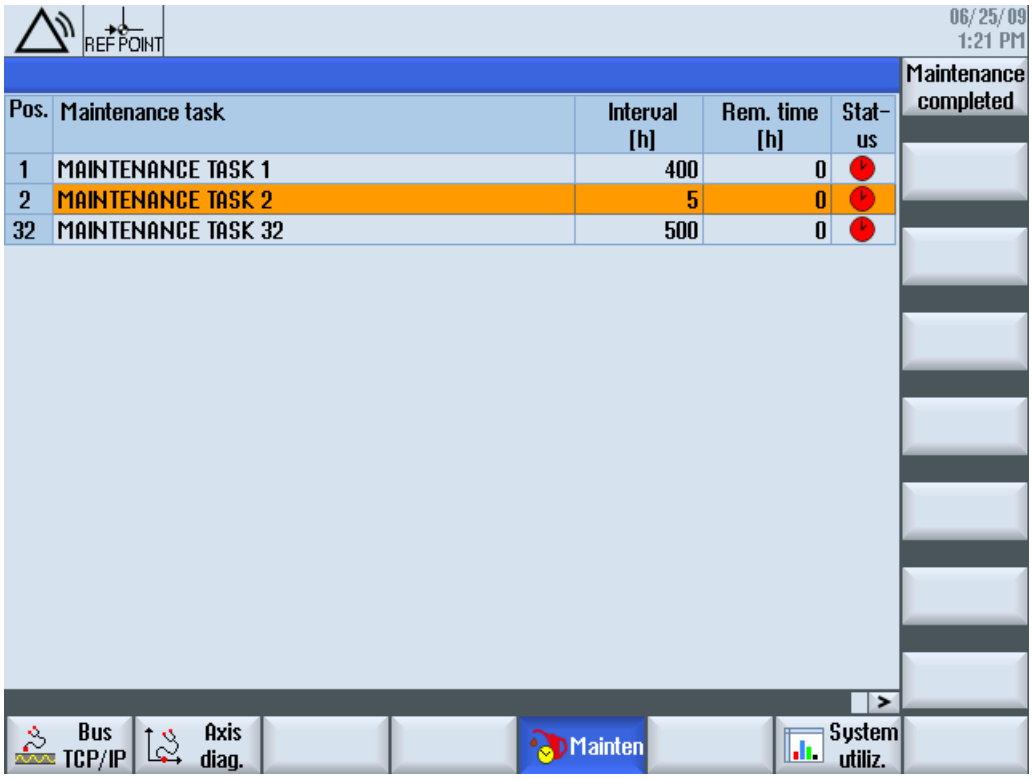
A number is automatically assigned when a task is created. If this is not desirable, then the tasks must be configured with the corresponding numbers.

This is recommended, for example, if task m is deleted by mistake and this has to be created again under the same number because of the evaluation in the PLC user program.

Standard mode

Access level 3: User

In the standard mode, the actual status of the maintenance tasks is displayed. The following columns are visible, but cannot be edited: "Maintenance task", "Interval", "Remaining time" and "Status".



8.1.3 This is how you import and export maintenance tasks

Overview

In the dialog, there are two soft keys that can be used to read-in and read-out the configuration files with the maintenance tasks:

- Importing maintenance tasks
- Exporting maintenance tasks

Configuring maintenance tasks

Procedure:

1. Generate a configuration file using the structure described below.

   The filename in lower-case letters can be freely selected.

2. As a minimum, generate the oem_maintenance_<lng>.ts, file, which contains the language-dependent designations for the maintenance tasks.
3. Copy the two files to a storage medium (CompactFlash card or USB FlashDrive).
4. Insert the storage medium into the appropriate slot at the front.
5. Press the "Import maintenance tasks" softkey.
6. Confirm with "OK".

   The files are copied. The language-dependent tasks are immediately visible.

Press the "Export maintenance tasks" softkey in order to make changes or supplements. This means that you copy these files to a storage medium, which you can then edit using an external editor.

Structure of the xml configuration file

The following identifiers are permissible:

Identifier	Meaning
<MAINTENANCE_TASK>	Main Tag
<TASK_ID>	Number of the maintenance task
<INTERVALL>	Maintenance interval time
<FIRST_WARNING>	Time interval until the first alarm is output.
<NUMBER_OF_WARNING>	Number of alarms

The file name can be freely selected, for example task.xml.

task.xml

```
<MAINTENANCE>
  <MAINTENANCE_TASK>
    <TASK_ID>0</TASK_ID>
    <INTERVALL>3</INTERVALL>
    <FIRST_WARNING>1</FIRST_WARNING>
    <NUMBER_OF_WARNING>1</NUMBER_OF_WARNING>
  </MAINTENANCE_TASK>
  <MAINTENANCE_TASK>
    <TASK_ID>1</TASK_ID>
    <INTERVALL>3</INTERVALL>
    <FIRST_WARNING>2</FIRST_WARNING>
    <NUMBER_OF_WARNING>1</NUMBER_OF_WARNING>
  </MAINTENANCE_TASK>
```

task.xml

```
  ...
  <MAINTENANCE_TASK>
    <TASK_ID>2</TASK_ID>
    <INTERVALL>3</INTERVALL>
    <FIRST_WARNING>2</FIRST_WARNING>
    <NUMBER_OF_WARNING>1</NUMBER_OF_WARNING>
  </MAINTENANCE_TASK
</MAINTENANCE>
```

Structure of the oem_maintenance_<lng>.ts file

This file has the extension ".ts" and contains all the language-dependent warning texts that were entered in the dialog. This file is available in the binary format (*.qm) after the system powers-up the next time.

Target directory: `/oem/sinumerik/hmi/lng`

oem_maintenance_<lng>.ts

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE TS>
<TS>
  <context>
    <name>maintenance</name>
    <message>
      <source>1</source>
      <translation>Maintenance task 1</translation>
      <chars>44</chars>
    </message>
    <message>
      <source>2</source>
      <translation>Maintenance task 2</translation>
      <chars>44</chars>
    </message>
. . .
    <message>
      <source>32</source>
      <translation>Maintenance task 32</translation>
      <chars>44</chars>
    </message>
  </context>
</TS>
```

Editing text entries

The maintenance texts are entered in the dialog with the values for the interval, time of the first warning and the number of warnings. In addition, the .ts file can also be edited in the alarm text editor if the necessary entry is available in the configuration file "oem_alarms_config.xml".

Integration in the existing language concept

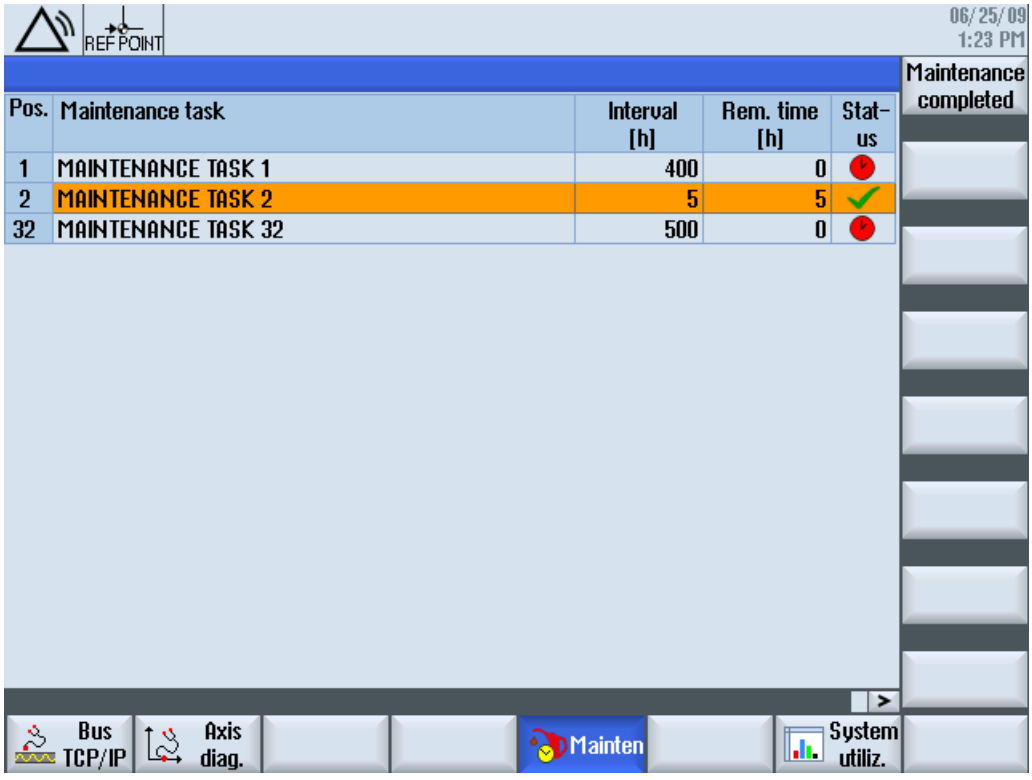
When started, the Service Planner reads the oem_maintenance_<lng>.ts file with the language set in the language selection menu. If this is not available, the English version is read which has to be available for the commissioning.

8.1.4 Acknowledging maintenance tasks

Acknowledging maintenance tasks

After completion of the maintenance activities, the maintenance tasks are acknowledged by the PLC user program or in the dialog of the operating software using the "Maintenance completed" softkey.

The acknowledgment bit for the maintenance task is set by making the acknowledgment in the dialog, the PLC then deletes "Number of output warnings/alarms" in the actual data and loads the interval value with the remaining time. This tells the user that the maintenance task has been successfully acknowledged.



Note

Acknowledgment of a maintenance task

The access level for the acknowledgment of a maintenance task is determined by MD51235 $MNS_ACCESS_RESET_SERV_PLANNER.

Default setting: Access level 2 "service"

Acknowledgment before the interval has expired

The maintenance interval can be acknowledged at any time. Premature acknowledgment means the premature start of a new maintenance interval.

Acknowledgment after the interval has expired

Acknowledgment of the maintenance interval restarts the task.

8.2 Easy Extend

8.2.1 Function of Easy Extend

Objective

Easy Extend provides you with a simple facility for commissioning, activating, deactivating or testing optional equipment. The available equipment and device states are displayed in a list by the control system. The system can manage a maximum of 64 devices.

Softkeys are used to activate or deactivate a device.

The Easy Extend function is available in the operating area "Parameter" → "Extension menu" → "Easy Extend".

Configuration

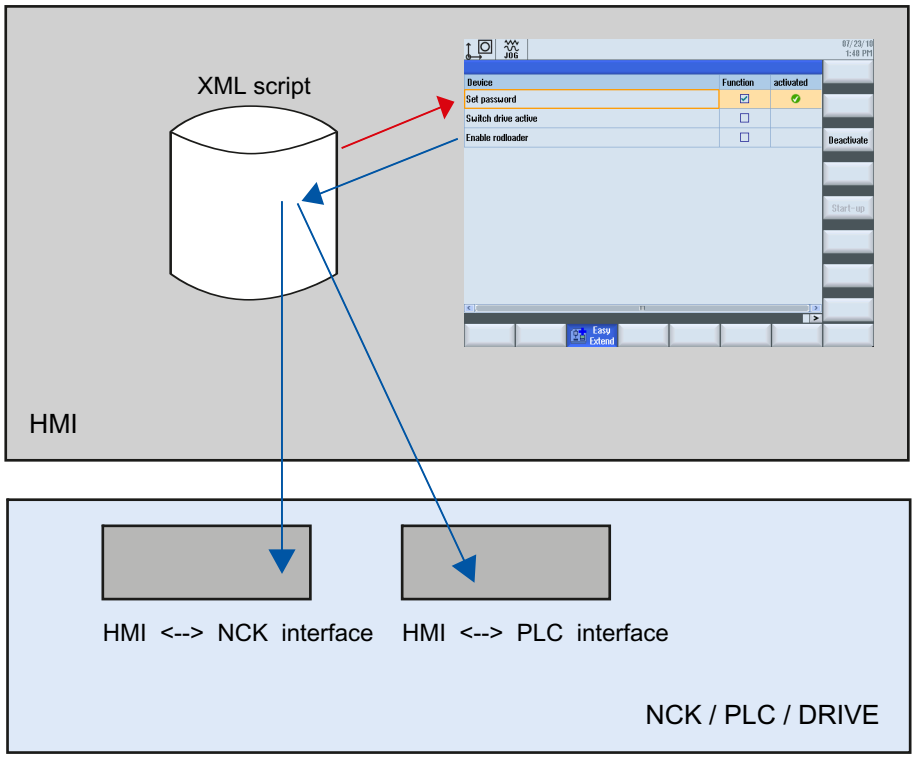


Figure 8-2 Mode of operation of Easy Extend

To use Easy Extend, the following functions should be configured by the machine manufacturer:

- **PLC ↔ HMI interface**

  The optional devices are managed via the interface between the user interface and the PLC.

- **Script processing**

  The machine manufacturer saves the sequences to be executed for commissioning, activating, deactivating or testing a device, in a statement script.

- **Parameter dialog (optional)**

  The parameter dialog shows device information that is saved in the script file.

Storage of the files

The Easy Extend files are stored on the system CompactFlash card in the "oem" (machine manufacturer) directory.

File	Name	Target directory
Text file	oem_aggregate_xxx.ts	`/oem/sinumerik/hmi/lng/`
Script file	agm.xml	`/oem/sinumerik/hmi/dvm`
Archive file	Any	`/oem/sinumerik/hmi/dvm/archives`
PLC user program	Any	PLC

8.2.2 Configuration in the PLC user program

Loading configurations

The configurations created are transferred to the manufacturer directory of the control, with the script and text file. Additionally, the corresponding PLC user program should be loaded.

Programming the equipment

Communication between the operator component and the PLC takes place in the PLC user program via **data block DB9905**, in which 128 words are reserved to manage a maximum of 64 devices.

Four bytes with the following meanings are used for each device:

Byte	Bit	Description	
0	0	== 1	Device has been started up (HMI acknowledgment)
	1	== 1	Device is to be activated (HMI request)
	2	== 1	Device is to be deactivated (HMI request)
	3-7	Reserved	

Byte	Bit	Description	
1	0-7	Reserved	
2	0	== 1	Device is active (PLC acknowledgment)
	1	== 1	Device has an error
	2-7	Reserved	
3	0-7	Unique identifier for the device	

Adding axes

If the machine is extended with machine axes, it is important to install the drive objects (DO) in a fixed sequence because the series startup archive contains the constellation of the machine manufacturer's reference machine and cannot be applied if the sequence is changed.

It is recommended that the following settings be selected for the "control components":

- NC data
- PLC data
- Drive data
  - ACX format (binary)

NOTICE
Series commissioning archive • In order to be able to use series startup archives in the Easy Extend script, these archives must be created **without** HMI data! **Changing the machine configuration** • Should there be any need to edit the drive machine data, this should be adapted in the control first. This procedure should be repeated for all devices and constellations.

See also

How to create and read in a series start-up archive (Page 414)

8.2.3 Option bits for machine manufacturers and dealers

Distribution of the option bits

The option bits are managed in blocks of 16 bits. According to the default setting, block 3 is reserved for dealers. This means that machine manufacturers have 48 options and dealers have 16 options.

The Easy Extend function uses the following machine data to manage options:

MD14510 $MN_USER_DATA_INT[0] to MD14510 $MN_USER_DATA_INT[3]

This assignment of the addresses can also be redefined in the script using the identifier **OPTION_MD** so that dealers can specify their own address area. If the dealer wishes to offer more than 16 options for this machine type, then the blocks not used should be determined. This must be coordinated with the machine manufacturer.

Block 3 must be programmed in the script using operations to communicate the dealer area. Further, a fixed device index should be assigned for every device.

Example, to redefine any area:

```
<option_md name="address identifier of the data" index="<index>"/>
  index - identifier of the area index:
    0: device 0 to 16 (default setting)
    1: device 17 to 32
    2: device 33 to 48
    3: device 49 to 64
```

Emulating the device at the PLC interface

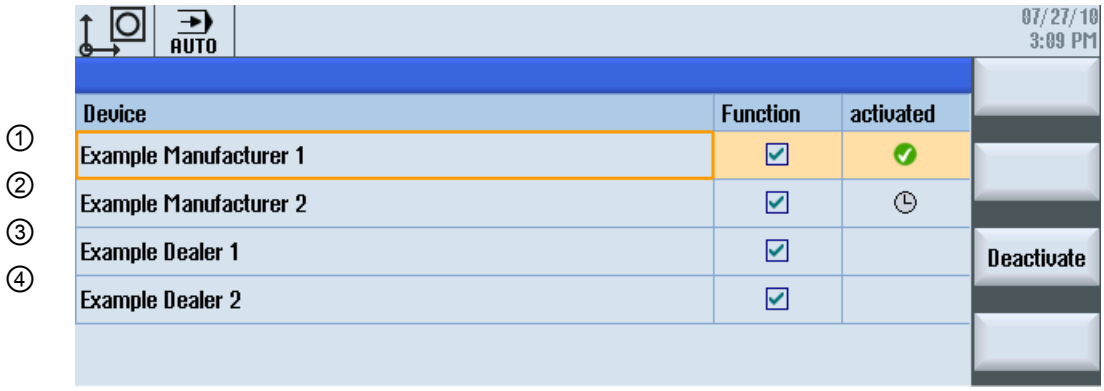
Each device must be assigned a unique index in order to permanently assign devices an option bit and to be able to access the PLC interface. The areas are permanently assigned using the attribute **option_bit**. If an assignment is not made, then the bits and the interface are defined by the index, which is assigned to the device in the following list:

Index	MD14510	Data block	Device designation
0	$MN_USER_DATA_INT[0] Bit 0	DB9905.DBB0	Device 1
1	$MN_USER_DATA_INT[0] Bit 1	DB9905.DBB4	Device 2
2	$MN_USER_DATA_INT[0] Bit 2	DB9905.DBB8	Device 3
3	$MN_USER_DATA_INT[0] Bit 3	DB9905.DBB12	Device 4
....	...	...	...
47	$MN_USER_DATA_INT[2] Bit 15	DB9905.DBB188	Device 48
48	$MN_USER_DATA_INT[3] Bit 0	DB9905.DBB192	Device 49
49	$MN_USER_DATA_INT[3] Bit 1	DB9905.DBB196	Device 50
....	...	...	...
62	$MN_USER_DATA_INT[3] Bit 14	DB9905.DBB248	Device 63
63	$MN_USER_DATA_INT[3] Bit 15	DB9905.DBB252	Device 64

Example of a fixed assignment:

```
<device option_bit="bit number">
...
</device>
```

Example



Blockwise assignment:

①	DB9905.DBB0	Device 1
②	DB9905.DBB4	Device 2
	...	
③	DB9905.DBB192	Device 49
④	DB9905.DBB196	Device 50

Assignment to free locations:

①	DB9905.DBB0	Device 1
②	DB9905.DBB4	Device 2
	...	
③	DB9905.DBB72	Device 19
④	DB9905.DBB76	Device 20
	...	

Feedback from the PLC

If the dealer cannot integrate his own PLC logic, then the PLC feedback signals must be set by the script.

See also

Attribute option_bit: Structure of the XML script (Page 289)

XML identifier: OPTION_MD (Page 301)

8.2.4 Display on the user interface

Dialogs on the user interface

The following dialogs are available for Easy Extend:

- The control offers a **configurable dialog**, in which the available devices are shown.
- If first commissioning has not taken place yet, the control opens the **commissioning dialog**.

If a commissioning procedure has been programmed for the device (XML statement: "START_UP") and the device has not been commissioned yet, the control starts the commissioning procedure.

This involves a complete data backup before the series startup archives saved in the script file are read in. Standard or data class archives are permitted as archive types: *.arc and *.ard

- In the event of an error, the commissioning engineer can decide whether to roll back the commissioning procedure or to rectify possible errors in machine configurations manually.
- Commissioning can be aborted early with the "Cancel" function. The control then copies the previously saved commissioning files back.

If the machine has to be switched off after successful completion of the commissioning, the XML statement "POWER_OFF" can be used to program that a corresponding message is output on the control.

8.2.5 Generating language-dependent text

Replacement characters for language-dependent text

The system offers the option of defining CONTROL properties (attribute values) during runtime. In order to use this function, the desired property must be set in a local variable and the variable name must be transferred to the tag as an attribute value preceded by $ characters.

Example:

```
<let name="caption_device_variable" type="string">my_device 3</let>


  <DEVICE option_bit="8">
    <!-- Direct: The entered expression is displayed. -->
    <NAME>my device 1</NAME>
  </DEVICE>


  <DEVICE option_bit="10">
    <!-- Indirect: The text saved for the text identifier caption_id_my_device2 is
displayed. -->
```

```
    <NAME>$$caption_id_my_device2</NAME>
    <form>
      <init>
        <!-- Indirect: The text saved for the text identifier
form_caption_device2 is displayed. -->
        <caption>$$form_caption_device2</caption>
      </init>
    </form>
  </DEVICE>
  <!-- Indirect: The content of the variable caption_device_variable is displayed. -
->

  <DEVICE option_bit="11">
    <NAME>$$$caption_device_variable</NAME>
  </DEVICE>
```

Structure of text file

The XML files with the language-dependent texts must be created in UTF8 format:

Example: oem_aggregate_eng.ts

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE TS>
<TS>
<context>
<name>EASY_EXTEND</name>
  <message>
    <source>form_caption_id_my_device2</source>
    <translation>my_device2</translation>
    <chars>64</chars>
  </message>
  <message>
    <source>form_caption_device2</source>
    <translation>my_input_form</translation>
    <chars>64</chars>
  </message>
</context>
</TS>
```

8.2.6 Examples

8.2.6.1 Example with control elements

Example of a combo box

If "combo box" is selected as the field type, the expressions to be displayed must also be defined. The <item> identifier should be used for this purpose. The combo box saves the index of the currently selected text in the variable belonging to the "CONTROL" (variable name). The index starts with 1.

```
<control name = "button1" xpos = "10" ypos = "10" fieldtype = "combobox">
<item>text1</item>
<item>text2</item>
<item>text3</item>
<item>text4</item>
</control>
```

Example of a value assignment

If an arbitrary integer value is to be assigned to an expression, the attribute value="value" should be added to the identifier. Rather than consecutive numbers, the control variable now contains the item's assigned value.

```
<control name = "button1" xpos = "10" ypos = "10" fieldtype = "combobox">
<item value = "10" >text1</item>
<item value = "20" >text2</item>
<item value = "12" >text3</item>
<item value = "1" >text4</item>

</control>
```

Note

"hotlink" attribute

The hotlink attribute results in a cyclic update of the corresponding control. This means that when a value is entered, the following update cycle overwrites the entered value. To avoid this behavior, the immediate saving of the entries must be activated with the DATA_ACCESS tag.

Another possibility is to take the the SOFTKEY_OK identifier into the form. This identifier is performed before the dialog is closed. In this block, the data comparison can be made between the control and reference variables with the UPDATE_CONTROLS statement.

8.2.6.2 Example with parameters to support the commissioning

Dialog with additional parameters

The input fields list selected drive parameters.

```
<DEVICE>
  <list_id>3</list_id>
  <name> "Test form" </name>
  <form>

  <init>
    <caption>Equipment Manager</caption>
    <control name = "edit1" xpos = "400" ypos = "34" refvar = "drive/dc/p105[DO5]"
/>
    <control name = "edit1" xpos = "400" ypos = "54" refvar =
"$MC_AXCONF_MACHAX_USED[4]" />
    <control name = "edit1" xpos = "400" ypos = "74" refvar = "drive/dc/p971[DO5]"
/>
    <control name = "edit1" xpos = "400" ypos = "94" refvar = "drive/dc/r2[DO5]" />
  </init>

  <paint>
  <text xpos = "40" ypos = "34">dc[DO5]/p105</text>
  <text xpos = "40" ypos = "54">$MC_AXCONF_MACHAX_USED[4]</text>
  <text xpos = "40" ypos = "74">dc[DO5]/p971</text>
  <text xpos = "40" ypos = "94">dc[DO5]/r2</text>
  </paint>
  </form>

</DEVICE>
```

Dialog with combo box

```
<form>

<init>
<caption>selected machine data</caption>
<DATA_ACCESS type="true" />
<!-- switch on the direct access to the NC variables -->
<control name = "edit1" xpos = "322" ypos = "34"
refvar="$MN_AXCONF_MACHAX_NAME_TAB[0]" />
<control name = "edit2" xpos = "322" ypos = "54"
refvar="$MN_AXCONF_MACHAX_NAME_TAB[1]" />
<control name = "edit3" xpos = "322" ypos = "74"
refvar="$MN_AXCONF_MACHAX_NAME_TAB[2]" />
<control name = "edit4" xpos = "322" ypos = "94"
refvar="$MN_AXCONF_MACHAX_NAME_TAB[3]" />

<control name = "edit5" xpos = "322" ypos = "114" refvar="$MA_IS_ROT_AX[AX1]"
hotlink="true" />
<control name = "edit6" xpos = "322" ypos = "134" refvar="$MA_IS_ROT_AX[AX2]"
hotlink="true" />
<control name = "edit7" xpos = "322" ypos = "154" refvar="$MA_IS_ROT_AX[AX3]"
hotlink="true" />

<!-- using the control type combo box to display the rotation axis value -->
<control name = "edit5" xpos = "322" ypos = "194" refvar="$MA_IS_ROT_AX[AX1]"
fieldtype = "combobox" hotlink="true" >
<item value= "0" >no</item>
<item value= "1" >yes</item>
</control>

<control name = "edit6" xpos = "322" ypos = "214" refvar="$MA_IS_ROT_AX[AX2]"
fieldtype = "combobox" hotlink="true" >
<item value= "0" >No</item>
<item value= "1" >yes</item>
</control>

<control name = "edit7" xpos = "322" ypos = "234" refvar="$MA_IS_ROT_AX[AX3]"
fieldtype = "combobox" hotlink="true" >
<item value= "0" >No</item>
<item value= "1" >yes</item>
</control>

</init>
```

```
<paint>
<text xpos = "23" ypos = "34">AXCONF_MACHAX_TAB[0]</text>
<text xpos = "23" ypos = "54">AXCONF_MACHAX_TAB[1]</text>
<text xpos = "23" ypos = "74">AXCONF_MACHAX_TAB[2]</text>
<text xpos = "23" ypos = "94">AXCONF_MACHAX_TAB[3]</text>
<text xpos = "23" ypos = "114">Is rot axis 1</text>
<text xpos = "23" ypos = "134">Is rot axis 2</text>
<text xpos = "23" ypos = "154">Is rot axis 3</text>

<text xpos = "23" ypos = "174">using combo box control</text>

<text xpos = "23" ypos = "194">Is rot axis 1</text>
<text xpos = "23" ypos = "214">Is rot axis 2</text>
<text xpos = "23" ypos = "234">Is rot axis 3</text>

</paint>

</form>
```

8.2.6.3 User example for a power unit

Activating the drive object

The drive object to be activated has already been commissioned and deactivated again by the machine manufacturer, to market the axis (axes) as an option.

To activate the axis carry out the following steps:

- Activate the drive object via p0105.
- Enable the 2nd axis in the channel machine data.
- Back up the drive machine data via p0971.
- Wait until the data has been written.
- Restart the NCK and the drives.

Programming:

```
<DEVICE>
  <list_id>1</list_id>
  <name> "Activate the drive" </name>

  <SET_ACTIVE>
```

```
    <data name = "drive/dc/p105[DO5]">1</data>
    <data name = "$MC_AXCONF_MACHAX_USED[4]">5</data>
    <data name = "drive/dc/p971[DO5]">1</data>
    <while>
      <condition> "drive/dc/p971[DO5]" !=0 </condition>
    <while>
    <control_reset resetnc ="true" resetdrive = "true"/>
  </SET_ACTIVE>

  <SET_INACTIVE>
    <data name = "drive/dc/p105[DO5]">0</data>
    <data name = "$MC_AXCONF_MACHAX_USED[4]">0</data>
    <data name = "drive/dc/p971[DO5]">1</data>
    </while>
      <condition> "drive/dc/p971[DO5]" !=0 </condition>
    </while>
    <control_reset resetnc ="true" resetdrive = "true"/>
  </SET_INACTIVE>

</DEVICE>
```

Activating the PLC-controlled device

The device is activated and deactivated by the user program using bit memory 100.0.

Programming:

```
<AGM>
<OPTION_MD NAME= "$MN_USER_DATA_INT[0]" />
  <DEVICE>
    <LIST_ID>1</LIST_ID>
    <NAME> "PLC Device" </NAME>
    <SET_ACTIVE>
    <SET_ACTIVE>
      <DATA NAME= "PLC/M100.0"> 1 </DATA>
    </SET_ACTIVE>
    <SET_INACTIVE>
      <DATA NAME= "PLC/M100.0"> 0 </DATA>
    </SET_INACTIVE>
  </DEVICE>
</AGM>
```

8.2.7 Description of the script language

Script language: Extended XML

XML (Extended Markup Language) is used as the script language, extended to include data processing and high-level language elements.

In contrast to standard XML, the language offers the following additional properties:

- Data storage for NC/PLC data, start-up data
- Reading and writing NC/PLC and drive data
- Execution of conditional jumps within an XML block
- Execution of program loops
- Execution of arithmetic operations
- Creation of local variables
- Loading/creation of a series startup archive
- Displaying messages

Additionally, SinuCom Installer script elements can be processed with the "up" execute statement.

Program parts of the script

The script is divided into the following areas:

- Identifier for Easy Extend
- Identifier for the device
- Identifier for commissioning the device
- Identifier for activating the device
- Identifier for deactivating the device
- Identifier for testing the device
- Identifier for machine data and high-level language elements
- Identifier for the parameter dialog

The individual identifiers are described in the following sections.

8.2.7.1 Special characters and operators

Displaying special characters

Characters with special meanings in XML syntax have to be rewritten if they are to be displayed correctly by a general XML interpreter.

The following characters are affected:

Character	Notation in XML	Meaning
<	>	Greater than
>	<	Less than
&	&	--
"	"	Quotation marks (straight)
'	'	Apostrophe

Permitted operators

The operation statement processes the following operations:

Operator	Notation in XML	Meaning
=	=	Assignment
==	==	Equal to
!	!	Not
!=	!=	Not equal to
>	>, >	Greater than
<	<, <	Less than
>=	>=, >=	Greater than or equal to
<=	<=, <=	Less than or equal to
|	|	Bit-by-bit OR operation
||	||	Logical OR operation
&	&	Logical or bit-by-bit AND operation
&&	&&	Logical AND operation
+	+	Addition
-	-	Subtraction
*	*	Multiplication
/	/	Division

8.2.7.2 Structure of the XML script

Overview

The following identifiers are available for the description of the device:

- Identifier for Easy Extend
- Identifier for the device
- Identifier for commissioning the device
- Identifier for activating the device
- Identifier for deactivating the device
- Identifier for testing the device

Description

Identifier <tag>	Meaning
AGM	Identifier for Easy Extend
DEVICE Attribute: option_bit	Identifier for the description of the device. The device is assigned a fixed bit number for the option management.
NAME	The identifier specifies the name of the device to be displayed in the dialog. If a text reference is used, the dialog displays the text which is saved for the identifier.
START_UP	The identifier contains a description of the sequences required for commissioning the device.
SET_ACTIVE	The identifier contains a description of the sequences required to activate the device.
SET_INACTIVE	The identifier contains a description of the sequences required to shut down the device.
TEST	The identifier contains the statements for testing the operating capability of a device.
UID	Unique numerical identifier to identify the device in the PLC ↔ HMI interface.
VERSION	Identifier for a version

Negative acknowledgment of the function execution

With the automatically provided variable "$actionresult", the system can inform the XML parser of a negative execution result. If the value is set to zero, the parser aborts the function processing.

Example

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE AGM>
<AGM>                                Identifier for Easy Extend
<DEVICE>
 <NAME> Device 1 </NAME>             Identifier for the device
 <START_UP>                          Identifier for commissioning the device
 …
 </START_UP>
 <SET_ACTIVE>                        Identifier for activating the device
 …
 </SET_ACTIVE>
 <SET_INACTIVE>                      Identifier for deactivating the device
 …
 </SET_INACTIVE>
 <TEST>                              Identifier for testing the device
 …
 </TEST>
</DEVICE>
…
</AGM>
```

8.2.7.3 CONTROL_RESET

Description

This identifier allows one or more control components to be restarted. Execution of the script is only continued when the control has resumed cyclic operation.

Programming

Identifier:	**CONTROL_RESET**	
Syntax:	**<CONTROL_RESET resetnc="TRUE" />**	
Attributes:	resetnc="true"	The NC component is restarted.
	resetdrive="true"	The drive components are restarted.

8.2.7.4 DATA

Description

Identifier for access to NCK, PLC and drive data.

Further details are described in Chapter: Addressing the parameters (Page 313)

Programming

Identifier:	DATA	
Syntax:	<data name ="<Address>" > value </data>	
Attributes:	name	Identifier for the variable name

8.2.7.5 DATA_ACCESS

Description

The identifier controls the behavior of the dialog when user inputs are being saved. The behavior should be defined within the INIT identifier. If this identifier is not used, inputs are always buffered.

Exception: The following attribute is set: hotlink = true

Programming

Identifier:	DATA_ACCESS	
Syntax:	--	
Attributes:	type=true	There is no buffering of the input values. The dialog copies the entered values directly to the reference variables.
	type=false	The values are only copied to the reference variables with the UPDATA_DATA type = "FALSE" identifier.

8.2.7.6 DATA_LIST

Description

This identifier enables the drive and machine data to be buffered or restored. Up to 20 temporary data lists can be created per device.

Note

The system deletes the data lists when the Easy Extend function is exited.

Programming

Identifier:	**DATA_LIST**	
Syntax:	<DATA_LIST action ="<read/write>" id="<list name>" > </DATA_LIST>	
Attributes:	action	Identifier for variable values:
	• action="read"	• The values of the listed variables are stored in a temporary memory.
	• action="append"	• The values of the listed variables are added to an existing list.
	• action="write"	• The backed up values of the variables are copied into the relevant machine data.
	id	Identifier for identifying the temporary memory

Example

```
< DATA_LIST action ="read" id="<name>">
nck/channel/parameter/r[2]
nck/channel/parameter/r[3]
nck/channel/parameter/r[4]
$MN_USER_DATA_INT[0]
…
</ DATA_LIST >
< DATA_LIST action ="write" id="<name>" />
```

8.2.7.7 DRIVE_VERSION

Description

Identifier for the drive version. The version number is copied to the `$driveversion` variable which is valid within the DEVICE identifier.

Further details are described in Chapter: Addressing the parameters (Page 313)

Programming

Identifier:	DRIVE_VERSION
Syntax:	--
Attributes:	--

8.2.7.8 FILE

Description

The identifier enables the reading in or creation of standard or data class archives.

- Reading in an archive:

  The file name of the archive must be specified for reading in an archive.

- Creating an archive:

  If the attribute create= "true" is specified, the function creates a standard archive (*.arc) under the specified name and stores the file in the …/dvm/archives directory.

  If the attribute class is also used, the system also creates a data class archive. The attributes class and group define the contents.

Programming

Identifier:	FILE	
Syntax:	<file name ="<archive name>" />	
	<file name ="<archive name>" create="true" class="<data classes>" group="<area>" />	
Attributes:	name	Identifier for the file name

class	Specifies the data classes that are to be contained in the archive. If several data classes are to be saved, the classes should be separated by a blank. The following data classes can be specified: • user • manufacturer • individual
create	A commissioning archive is created under the specified name in the …/dvm/archives/ directory. If the "class" attribute is not specified, it is a standard archive with NC/PLC, HMI and drive data.
group	Specifies the data groups that are to be contained in the archive. If several data groups are to be saved, the groups should be separated by a blank. The following data groups can be contained in the archive: • NC • PLC • HMI • DRIVES

Example

```
<!-- Create data class archive -->
<file name="user.arc" create="true"
class="user manufacturer individual"
group="nc plc hmi" />

<!—Read archive into the control à
<file name="user.arc" />
; or
<file name="user.ard" />
```

8.2.7.9 FUNCTION

Description

Function call: This identifier carries out the function specified under the attribute "name".

Programming

Identifier:	**FUNCTION**	
Syntax:	<FUNCTION name = "function name" />	
Attributes:	name	Function name
	return	Variable name for saving the result of the function
Call parameters:	Call parameters are transferred as values of the XML statement.	
	The listed variables must be separated by a comma. A maximum of 10 parameters can be transferred.	
	It is also possible to specify constants or text expressions as call parameters.	
	The identifier _T should be placed at the start as a means of identifying a text expression.	

Example

The calling function does not expect a return value

```
<FUNCTION name = "function name" return="variable name" />
; Parameter transfer
<FUNCTION name = "function name"> var1, var2, var3 </FUNCTION>
<FUNCTION name = "function name"> _T"Text", 1.0, 1 </FUNCTION>
```

8.2.7.10 FUNCTION_BODY

Description

Function body: This identifier forms the function body of a subfunction.

Programming

Identifier: **FUNCTION_BODY**

Syntax:

- Function body without parameter

```
<FUNCTION_BODY name = "function name" >
…
</ FUNCTION_BODY>
```

- Function body with parameter

```
<FUNCTION_BODY name="function_name" parameter="p1, p2, p3" >
…
<let name="tmp></let>
<op> tmp = p1 </op>
…
</ FUNCTION_BODY>
```

- Function body with return value

```
<FUNCTION_BODY name="function_name" parameter="p1, p2, p3" return="true" >
…
<let name="tmp></let>
<op> tmp = p1 </op>
…
<op> $return= tmp </op>
</ FUNCTION_BODY>
```

Attributes:		name	Name of the subfunction's function body
		return	If the attribute is set to true, the system creates the local variable $return. The function's return value which is forwarded to the calling function when exiting the function body should be copied to this variable.
	(optional)	parameter	The attribute lists the expected transfer parameters. The parameters must be separated by a comma. When the function body is called, the values of the parameters specified in the function call are copied to the transfer parameters listed.

Example

```
<function_body name="test" parameter="c1,c2,c3" return="true">
      <let name="tmp">0</let>
      <op> tmp = c1+c2+c3 </op>
      <op> $return= tmp </op>
```

```
</function_body>
…
       <let name="my_var"> 4 </let>
<function name="test" return=" my_var " > 2, 3, 4</function>
       <print text="result=%d"> my_var </print>
       …
       <op> tmp = c1+c2+c3 </op>
       <op> $return= tmp </op>
</function_body>
       <let name="my_var"> 4 </let>
<function name="test" return=" my_var " > 2, 3, 4</function>
       <print text="result=%d"> my_var </print>
       …
```

8.2.7.11 INCLUDE

Description

An XML description is included in this statement.

Programming

Identifier:	**INCLUDE**	
Syntax:	**<?include src="file name" ?>**	
Attributes:	src	Identifier for the file name

8.2.7.12 LET

Description

Identifier for a local variable under the specified name.

The variable remains valid until the end of the higher-level XML block.

Variables which are to be available globally should be created directly after the AGM tag.

Note

Preassignment of a STRING variable

Texts containing more than one line can be assigned to a string variable if the formatted text is transferred as a value. If a line is to end with a line feed <LF> , the characters "\n" should be added at the end of the line.

Programming

Identifier:	**LET**	
Syntax:	<let name ="<Name>" > preassignment </let>	
	<let name ="<Name>" type="<Variable type>"> preassignment </let>	
Attributes:	name	Identifier for the variable name
	type	Permissible identifiers for the variable type: • Integer (INT) • Double (DOUBLE) • Float (FLOAT) • String (STRING) **Default:** If no type is defined, the system creates an integer variable. <LET name = "VAR1" type = "INT" /> A variable can be initialized with a value. <LET name = "VAR1" type = "INT" > 10 </LET> If values comprising NCK or PLC variables are saved in a local variable, the assignment operation automatically adapts the format to that of the variables which have been loaded.
	permanent	If the attribute is set to TRUE, the variable value is saved permanently. This attribute only applies to global variables! **Syntax:** <let name ="<Name>" type="<Variable type>" permanent="TRUE" > preassignment </let>

Example

```
<LET name="text" type = "string"> F4000 G94\n
   G1 X20\n
   Z50\n
   M2\n
</LET>
```

8.2.7.13 MSGBOX

Description

The identifier opens a message window whose return value can be used for branching. If a text reference is used, the message window displays the text which is saved for the identifier.

Programming

Identifier:	**MSGBOX**	
Syntax:	<MSGBOX text="<Text>" caption="<Title>" retvalue="<Variable>" type="<Acknowledgment>" />	
	<MSGBOX text="<Text>" caption="<$$Text reference>" retvalue="<Variable>" type="<Acknowledgment>" />	
Attributes:	caption	Identifier for the title of the message window
	retvalue	Identifier for the name of the variable to which the return value is copied:
	retval=0	0: OK
	retval=1	1: Cancel
	type	Identifier for acknowledging the message
	type="btn_ok"	
	type="btn_cancel"	
	type="btn_okcancel"	

8.2.7.14 OP

Description

Identifier for an execute statement: All permissible operators may be executed. For accessing the NCK, PLC and drive data, the complete variable name is to be placed in quotation marks.

Programming

Identifier:	**OP**
Syntax:	<op> arithmetic operation </op>
Attributes:	--

Example

```
<OP> tmpVar = "PLC/MB170" </OP>
<OP> tmpVar = "PLC/MB170" + 5 </OP
```

Character string processing

The operation instruction is able to process character strings and assign the results to the string variable specified in the equation.

The identifier _T should be placed at the start as a means of identifying text terms. Formatting of variable values is also possible. The identifier _F should be placed at the start of the formatting regulation, followed by the format instruction.

The address is then specified for the variable.

Example

```
<LET name="buffer" type="string"></LET>
…
…
<op> buffer = _T"unformatted value R0= " + "nck/Channel/Parameter/R[0]" + _T" and "
+ _T"$$85051" + _T" formatted value R1 " + _F%9.3f"nck/Channel/Parameter/R[1]" </op>
```

8.2.7.15 OPTION_MD

Description

The identifier allows option machine data to be redefined. As delivered, the system uses MD14510 $MN_USER_DATA_INT[0] to $MN_USER_DATA_INT[3].

If the PLC user program manages the options, the appropriate data words must be provided in a data block or GUD.

The data is structured in bits. Starting with bit 0, there is a fixed assignment of the bits to the listed devices, i.e. bit 0 is assigned to device 1, bit 1 to device 2, etc. If more than 16 devices are managed, the address identifiers of the device groups 1-3 are assigned via the area index.

Note

Converting the value range

The value range of MD14510 $MN_USER_DATA_INT[i] is from -32768 to +32767. To activate the devices bit-by-bit via the machine data dialog, the bit combination must be converted to decimal representation.

Programming

Identifier:	**OPTION_MD**	
Syntax:	Area 0:	
	<option_md name = "Address identifier of the data" />	
	OR:	
	<option_md name = "Address identifier of the data" index= "0"/>	
	Area 1 to 3:	
	<option_md name = "Address identifier of the data" index= "Area index"/>	
Attributes:	name	Identifier for the address, e.g. $MN_USER_DATA_INT[0]
	index	Identifier for the area index:
		0 (default setting): Device 1 to 16
		1: Device 17 to 32
		2: Device 33 to 48
		3: Device 49 to 64

8.2.7.16 PASSWORD

Description

If this identifier is assigned to a device, a softkey appears when the option is not set requesting the input of a password for this device. The character string is processed by the PLC and the result forwarded to the HMI via the option data.

Programming

Identifier:	**PASSWORD**	
Syntax:	<password refvar = "variable name" />	
Attributes:	refvar	Name of the reference variable

Example:

```
<password refvar="plc/db9900.dbd0" />
```

8.2.7.17 PLC_INTERFACE

Description

This identifier permits the PLC ↔ HMI interface to be redefined. The system expects 128 addressable words.

Default: DB9905

Programming

Identifier:	**PLC_INTERFACE**	
Syntax:	<plc_interface name = "Address identifier of the data" />	
Attributes:	name	Identifier for the address, e.g. "plc/mb170"

Example: `plc/mb170`

8.2.7.18 POWER_OFF

Description

Identifier for a message prompting the operator to switch the machine off. The message text is permanently saved in the system.

Programming

Identifier:	POWER_OFF
Syntax:	<power_off />
Attributes:	--

8.2.7.19 PRINT

Description

The identifier outputs a text in the message line or copies the text to the specified variable. If the text contains formatting identifiers, the variable values are inserted at the appropriate place.

- The specified "%n" results in a line break in the displayed text.
- The character '%' results in the formatting of the variable specified as the value:

```
%[Flags] [Width] [.decimal places] type
```

Parameter	Application
Flags	Optional character to define the formatting for the task: • Right or left-justified (- left-justified) • Add leading zeros (0) • Fill with blanks
Width	The argument defines the minimum output width for a non-negative number. If the value to be output has fewer places than the argument defined, the missing spaces are filled with blanks. Decimal places: With floating-point numbers, the optional parameter defines the number of decimal places.
Type	The type character defines which data formats are transferred for the PRINT instruction. This character must be specified. The following data formats are supported: • d: Integer value • f: Floating-point number • s: String
Values	Number of variables whose values are to be inserted into the text. The variable types must match the corresponding type identifier for the formatting instruction.

Programming

Identifier:	**PRINT**	
Syntax:	<print name = "Variable name" text="text %formatting"> Variable, … </print>	
Attributes:	name	Name of the variable where the text is to be stored.
	text	Text

8.2.7.20 WAITING

Description

After a reset of the NC or the drive, there is a wait for the restart of the respective component.

Programming

Identifier:	**WAITING**	
Syntax:	<WAITING WAITINGFORNC ="TRUE" />	
Attributes:	waitingfornc="true"	There is a wait for the restart of the NC.
	waitingfordrive="true"	There is a wait for the restart of the drive.

8.2.7.21 ?up

Description

SinuCom Installer:

This section contains the script language for the SinuCom Installer. If the code from a SinuCom Installer file is to be included, the INCLUDE (Page 297) statement must be used.

Programming

Identifier:	**?up**
Syntax:	<?up <?include src="Filename" ?> ?>
Attributes:	--

8.2.7.22 XML identifiers for the dialog

Dialog for the parameterization

A dialog can be configured for each device so that additional parameters can be set or output during runtime. This is displayed by pressing the "Additional parameters" softkey.

The following dialog elements are available:

- Input dialog
- Dialog title
- Combined input/output field
- Text display
- Image display

Description

Identifier <tag>	Meaning
CAPTION	Identifier for the title of the dialog: Syntax: <caption> title </caption>
CLOSE	**Dialog message:** This identifier is executed before the dialog is closed.
FORM	Identifier for a user dialog. Attribute color: Color coding of the background color
INIT	**Dialog message:** Identifier for initializing the dialog. The identifier is executed immediately after the dialog is created. All the input elements and hotlinks for the dialog should be created here.
PAINT	**Dialog message:** Identifier for displaying all texts and images of a dialog. This identifier is executed when the dialog is shown.
TIMER	**Dialog message:** This identifier is called cyclically

Example

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE AGM>
<AGM>
<DEVICE>
 <NAME> Device 1 </NAME>
 <START_UP>
 …
 </START_UP>
 <SET_ACTIVE>
 …
 </SET_ACTIVE>
…
 <FORM>                                    Identifier for a user dialog
 <INIT>
 <CONTROL name = edit1 /CONTROL>           Identifier for an input field
 </INIT>
 <PAINT>                                   Identifier for text or image display
 <TEXT>hello world !</TEXT>
 </PAINT>
 </FORM>

</DEVICE>
…
</AGM>
```

8.2.7.23 BOX

Description

Identifier for drawing a filled rectangle at the specified position, colored as specified.

Programming

Identifier:	**BOX**	
Syntax:	<box xpos="X position" ypos = "Y position" width="X extension" height="Y extension" color="Color code" />	
Attributes:	xpos	Position in the X direction (zero point in the left upper corner)
	ypos	Position in the Y direction (zero point in the left upper corner)
	width	Extension in X direction (in pixels)
	height	Extension in Y direction (in pixels)
	color	Color

8.2.7.24 CONTROL

Description

Identifier for creating control elements.

Default: fieldtype="edit" The data can be edited.

Programming

Identifier:	**CONTROL**	
Syntax:	<control name = "edit1" xpos = "X position" ypos = "Y position" refvar="NC variable" hotlink="true" format="Format" display_format="FLOAT" />	
Attributes:	name	Name of the field: A local variable of the same name is created for each field.
	xpos	Position in the X direction (zero point in the left upper corner)
	ypos	Position in the Y direction (zero point in the left upper corner)

fieldtype • fieldtype="edit" • fieldtype="readonly" • fieldtype="combobox"	Field type: • The data can be edited. • The data can be read. • Identifiers are shown instead of the data.
format	The attribute defines the display format of the specified variables, e.g. left-justified or the number of decimal places.
display_format	The attribute defines the processing format of the specified variables. When accessing a PLC float variable, this attribute must be used, as the access is realized by reading a double word. The following data formats are permissible: • FLOAT • INT • DOUBLE • STRING
hotlink	Identifier for a field that is updated (=TRUE) immediately when the data is changed.
refvar	Identifier for a reference variable (NC, PLC or drive variable)

Note

Cyclic update

The "hotlink" attribute results in a cyclic update of the corresponding control.

This means that If a value is entered, the following update cycle overwrites the entered value.

To avoid this behavior, the immediate saving of the entries must be activated with the DATA_ACCESS identifier.

Example

If "combo box" is selected as the field type, the expressions to be displayed must also be defined. The <item> identifier should be used for this:

The combo box saves the index of the currently selected text in the variable belonging to the "CONTROL" (variable name). The index starts with 1.

Syntax: <item>expression</item>

```
<control name = "button1" xpos = "10" ypos = "10" fieldtype="combo box">
      <item>text1</item>
      <item>text2</item>
      <item>text3</item>
      <item>text4</item>
</control>

      ; If an arbitrary integer value is to be assigned to an expression, the
      attribute value="value" should be added to the item identifier.
      ; Rather than consecutive numbers, the control variable now contains the
      item's assigned value.

<control name = "button1" xpos = "10" ypos = "10" fieldtype="combo box">
      <item value = "10" >text1</item>
      <item value = "20" >text2</item>
      <item value = "12" >text3</item>
      <item value = "1" >text4</item>
</control>
```

8.2.7.25 IMG

Description

Identifier for displaying a pixel graphic in the directory: `../oem/sinumerik/hmi/dvm`

- The bitmaps must be stored in BMP or PNG format.
- Lower-case letters should be used for all file names.
- If the image display is to differ from the original in terms of size, the attributes width and height can be used to define the dimensions.

Programming

Identifier:	**IMG**	
Syntax:	<img name = "<Name>" xpos = "X position" ypos = "Y position" height = "Scaling in Y" width = "Scaling in X" />	
Attributes:	name	Extension in Y direction (in pixels)
	xpos	Position in the X direction (zero point in the left upper corner)
	ypos	Position in the Y direction (zero point in the left upper corner)

width	Scaling in X (optional)
height	Scaling in Y (optional)

8.2.7.26 PROPERTY

Description

The identifier specifies additional properties of an operator control.

Programming

Identifier:	**PROPERTY**	
Syntax:	<property attribute="<Value>" />	
Attributes:	max	Maximum input value
	min	Minimum input value
	default	Default

Example

```
<control name = "edit" xpos = "10" ypos = "10" width = "100" hotlink="true"
refvar="nck/Channel/GeometricAxis/actProgPos[1]" >
       <property min="0" />
       <property max="1000" />
</control>

<control name = "edit1" xpos = "10" ypos = "10" >
       <property min = "20" />
       <property max = "40" />
       <property default="25" />
</control>
```

8.2.7.27 REQUEST

Description

This identifier is only valid within the INIT statement of a form. This identifier is used to add a variable to the cyclic reading service (hotlink).

Programming

Identifier:	**REQUEST**	
Syntax:	<REQUEST name = "NC variable" />	
Attributes:	name	Address identifier

8.2.7.28 SOFTKEY_OK, SOFTKEY_CANCEL

Description

The identifier SOFTKEY_OK overwrites the standard behavior when closing a dialog by means of the "OK" softkey. The identifier SOFTKEY_CANCEL overwrites the standard behavior when closing a dialog by means of the "CANCEL" softkey.

The following functions can be performed within this identifier:

- Data manipulation
- Conditional processing
- Loop processing

Programming

Identifier:	**SOFTKEY_OK**
Syntax:	<SOFTKEY_OK>
	…
	</SOFTKEY_OK>
Identifier:	**SOFTKEY_CANCEL**
Syntax:	<SOFTKEY_CANCEL>
	…
	</SOFTKEY_CANCEL>

8.2.7.29 TEXT

Description

Identifier for displaying text

Programming

Identifier:	TEXT	
Syntax:	<text xpos="X position" ypos = "Y position" color="Color code"> Text </text>	
Attributes:	xpos	Position in the X direction (zero point in the left upper corner)
	ypos	Position in the Y direction (zero point in the left upper corner)
	color	Color

8.2.7.30 UPDATE_CONTROLS

Description

This identifier performs a comparison between the operator controls and the reference variables.

Programming

Identifier:	UPDATE_CONTROLS	
Syntax:	<update_controls type="<Direction>"/>	
Attributes:	type	The attribute defines the direction of the data comparison. • TRUE The data is read from the reference variables and copied to the operator controls. • FALSE The data is copied from the operator controls to the reference variables.

8.2.7.31 Addressing the parameters

Addressing parameters

Address identifiers for the desired data must be created to address NC variables, PLC blocks or drive data. An address consists of the subpaths component name and variable address. A slash should be used as a separating character.

Addressing PLC data blocks

Data format f:	B: Byte
	W: Word
	D: Double word
x address:	Valid PLC address identifier
Bit addressing:	b = Bit number

The following addresses are permissible:

DBx.DB(f)	Data block
`I(f)x`	Input
`Q(f)x`	Output
`M(f)x`	Bit memory
`V(f)x`	Variable

Data format identification is not applicable for bit addressing:

DBx.DBXx.b	Data block
`Ix.b`	Input
`Qx.b`	Output
`Mx.b`	Bit memory
`Vx.b`	Variable

Example:

```
<data name = "plc/mb170">1</data>
<data name = "plc/db9905.dbb0"> 0 </data>
<data name = "plc/i0.1"> 1 </data>
<op> plc/m19.2 = 1 </op>
```

Addressing NC variables

Addressing starts with the path section nck. This section is followed by the data address; its structure should be taken from the OEM Package Parameter Manual 2.

Example:

```
<let name = "tempStatus"></let>
<op> tempStatus = "nck/channel/state/chanstatus" </op>
```

Addressing the machine data and setting data

Machine data and setting data is identified by the character $, followed by the name of the data.

- Machine data:

  $Mx_<name[index, AX<axis_number>]>

- HMI machine data:

  $MxS_<name[index, AX<axis_number>]>

- Option data:

  $Ox_<name[index, AX<axis_number>]>

- Setting data:

  $Sx_<name[index, AX<axis_number>]>

  $SxS_<name[index, AX<axis_number>]>

Structure		Meaning
x:	N	General machine or setting data
	C	Channel-specific machine or setting data
	A	Axis-specific machine or setting data
index:		For a field, the parameter indicates the index of the data.
AX<axis_number>:		The required axis (<axis_number>) has to be specified for axis-specific data. Alternatively, the axis index can be read from a local variable using a "substitution character" $<variable name>: e.g. AX$localvariable

Example:

```
<data name ="$MN_AXCONF_MACHAX_NAME_TAB[0] ">X1</data>
```

- Direct addressing of the axis:

  ```
  <data name ="$MA_CTRLOUT_MODULE_NR[0, AX1] ">1</data>
  ```

- Indirect addressing of the axis:

  ```
  <let name ="axisIndex"> 1 </let>
  <data name ="$MA_CTRLOUT_MODULE_NR[0, AX$axisIndex] ">1</data>
  ```

Addressing the global user data

Addressing starts with the path section gud, followed by the specification of the area CHANNEL. This address section is followed by the specification of the GUD areas:

GUD areas	Assignment
sgud	Siemens GUD
mgud	Machine manufacturer GUD
ugud	User GUD

Then enter the GUD name. If an array is to be addressed, the name is followed by the array subscript in square brackets.

Example:

```
<data name ="gud/channel/mgud/syg_rm[0]">1</data>
<op>"gud/channel/mgud/syg_rm[0]" = 5*2 </op>
```

8.2.7.32 Addressing the drive objects

Syntax for Drive Objects (DO)

Addressing starts with the path section "drive". Then the drive device is specified: CU or DC. The parameter to be set is added to this section.

To address individual objects, the desired object should be entered in square brackets after the parameter.

```
Parameter number[do<DO-index>]
```

Example: `p0092[do1]`

Note

Numbering

The number of the drive object differs from the numbering used in the drive dialog, since the CU components, ALM and all connected hubs are integrated in the consecutive numbering.

The DO number can be determined as follows:

All connected drive objects are listed in field p0978 of the relevant CU. The numerical value of a field corresponds to the slot number of a drive object. It is a question of establishing the field index for the desired slot and adding one to this number. This value is the DO index needed for addressing purposes.

If there are drive objects connected to an NX, the index of the last CU drive object should be established first and the index of the NX drive object should be added to it.

Alternatively, the drive index can be read from a local variable, e.g. DO$localvariable, using a "substitution character" $<variable name>.

Example:

```
<data name ="drive/cu/p0092">1</data>
<data name ="drive/dc/p0092[do1] ">1</data>
```

Indirect addressing:

```
<let name = "driveIndex> 0 </let>
<op> driveIndex = $ctrlout_module_nr[0, AX1] </op>
<data name ="drive/dc[do$driveIndex]/p0092">1</data>
```

Addressing an NX

An NX module is treated as another CU, module addressing uses the CU specification, whereby the desired NX number incremented by one is specified in square brackets after the parameter.

```
Parameter number[CU<CU index>]
```

Example

```
<let name="r0002_content"></let>
<let name="p107_content"></let>
<!- Reading of value r0002 from the CU ->
<op> r0002_content = "drive/cu/r0002" </op>
<op> r0002_content = "drive/cu/r0002[CU1]" </op>
<!- Reading of value r0002 from the NX1 ->
<op> r0002_content = "drive/cu/r0002[CU2]" </op>
<!- Reading of value p107[0] from the CU ->
<op> p107_content = "drive/cu/p107[0]" </op>
   <print text="%d"> p107_content </print>
<!- Reading of value p107[0] from the NX1 ->
<op> p107_content = "drive/cu/p107[0,CU2]" </op>
   <print text="%d"> p107_content </print>
```

8.2.7.33 XML identifiers for statements

Program statements

The following identifiers are permissible for statements:

Identifier <tag>	Meaning
IF	Conditional statement (IF, THEN, ELSE) The THEN and ELSE tags are included in the IF tag. The condition which is stated in the CONDITION tag follows the IF tag. The further processing of the statements depends upon the result of the operation. If the result of the function is true, the THEN branch is executed and the ELSE branch is skipped. If the result of the function is false, the parser executes the ELSE branch. **Example:** <IF> <CONDITION> plc/mb170 != 7 </CONDITION> <THEN> <OP> plc/mb170 = 7 </OP> … </TEHN> <ELSE> … … </ELSE> </IF>
THEN	Statement for situations where the condition has been met (IF, THEN, ELSE).
ELSE	Statement for situations where the condition has not been met (IF, THEN, ELSE).
FOR	The FOR loop is executed as follows: 1. The expression **initialization** (INIT) is analyzed. 2. The expression **test** (CONDITION) is analyzed as a Boolean expression. If the value is false (FALSE) the FOR loop is ended. 3. The following statements are executed. 4. The expression **continuation** (INCREMENT) is analyzed. 5. Proceed with Step 2. `for (initialization, test, continuation) statements` **Syntax:** <FOR> <INIT>…</INIT> <CONDITION>…</CONDITION> <INCREMENT>…</INCREMENT> Statements … </FOR>

Identifier <tag>	Meaning
BREAK	Conditional cancellation of a loop
WHILE	The WHILE loop is used to execute a sequence of statements repeatedly while a condition is met. This condition is tested before the sequence of statements is executed. `while (test) statements` **Syntax:** <WHILE> <CONDITION>…</CONDITION> Statements … </WHILE>

DO_WHILE	The DO WHILE loop comprises a block of statements and a condition. The code within the statement block is executed first, then the condition is analyzed. If the condition is true, the function executes the code section again. This is continuously repeated until the condition is false.
	
	`do`
	` Statements`
	`while (test)`
	Syntax:
	<DO_WHILE>
	Statements
	…
	<CONDITION>…</CONDITION>
	</DO_WHILE>

SWITCH	The SWITCH statement describes a multiple choice. A term is evaluated once and compared with a number of constants. If the term matches the constants, the statements are processed within the CASE statement. The DEFAULT statement is processed when none of the constants listed match the expression. **Syntax:** <SWITCH> <condition> Expression </condition> <CASE value="<constant 1>" > Statements ... </CASE> <CASE value="<constant 2>" > Statements ... </CASE> <DEFAULT> Statements ... </DEFAULT> </SWITCH>

8.2.8 String functions

Overview of the functions

The script language offers various string functions. The function names are reserved and cannot be overloaded.

Name	Function
string.cmp	Comparing character strings (string.cmp (Page 319))
string.icmp	Comparing character strings without consideration of upper/lower case (string.icmp (Page 320))
string.left	Selecting number of characters from the left (string.left (Page 321))
string.right	Selecting number of characters from the right (string.right (Page 322))
string.middle	Selecting number of characters from the middle (string.middle (Page 322))
string.length	Determining the length of a character string (string.length (Page 323))
string.replace	Replacing character strings (string.replace (Page 324))
string.remove	Deleting character strings (string.remove (Page 324))
string.insert	Inserting a character string from index (string.insert (Page 326))
string.delete	Deleting a number of characters in a character string (string.delete (Page 325))
string.find	Finding a subset of a character string (forwards) (string.find (Page 326))
string.reversefind	Finding a subset of a character string (backwards) (string.reversefind (Page 327))
string.trimleft	Removing blanks from the left (string.trimleft (Page 328))
string.trimright	Removing blanks from the right (string.trimright (Page 328))

8.2.8.1 string.cmp

Description

Two strings are compared with each other.

The function gives a return value of zero if the strings are the same, a value less than zero if the first string is smaller than the second string or a value greater than zero if the second string is smaller then the first string.

Programming

Designation:	**string.cmp**	
Syntax:	<function name="string.cmp" retvar ="<int var>" > str1, str2 </function>	
Parameters:	str1	String
	str2	Comparison string
	rval	Result

Example

```
<let name="rval">0</let>
<let name="str1" type="string">A brown bear hunts a brown dog.</let>
<let name="str2" type="string">A brown bear hunts a brown dog.</let>
<function name="string.cmp" return="rval"> str1, str2 </function>

;Result: rval=0
```

8.2.8.2 string.icmp

Description

Two strings are compared (the comparison is not case-sensitive).

The function gives a return value of zero if the strings are the same, a value less than zero if the first string is smaller than the second string or a value greater than zero if the second string is smaller then the first string.

Programming

Designation:	**string.icmp**	
Syntax:	<function name="string.icmp" retvar ="<int var>" > str1, str2 </function>	
Parameters:	str1	String
	str2	Comparison string
	rval	Result

Example

```
<let name="rval">0</let>
<let name="str1" type="string">A brown bear hunts a brown dog.</let>
<let name="str2" type="string">A brown Bear hunts a brown Dog.</let>
<function name="string.icmp" return="rval"> str1, str2 </function>

;Result: rval=0
```

8.2.8.3 string.left

Description

The function extracts the first nCount characters from string 1 and copies them to the return variable.

Programming

Designation:	**string.left**	
Syntax:	<function name="string.left" return="< result string>"> str1, nCount </function>	
Parameters:	str1	String
	nCount	Number of characters

Example

```
<let name="str1" type="string">A brown bear hunts a brown dog.</let>
<let name="str2" type="string"></let>
<function name="string.left" return="str2"> str1, 12 </function>

;Result: str2="A brown bear"
```

8.2.8.4 string.right

Description

The function extracts the last nCount characters from string 1 and copies them to the return variable.

Programming

Designation:	**string.right**	
Syntax:	<function name="string.right" return="< result string>"> str1, nCount </function>	
Parameters:	str1	String
	nCount	Number of characters

Example

```
<let name="str1" type="string">A brown bear hunts a brown dog.</let>
<let name="str2" type="string"></let>
<function name="string.right" return="str2"> str1, 10 </function>

;Result: str2="brown dog"
```

8.2.8.5 string.middle

Description

The function extracts the specified number of characters from string 1, starting from the iFirst index, and copies them to the return variable.

Programming

Designation:	**string.middle**	
Syntax:	<function name="string.middle" return="< result string>"> str1, iFirst, nCount </function>	
Parameters:	str1	String
	iFirst	Start index
	nCount	Number of characters

Example

```
<let name="str1" type="string">A brown bear hunts a brown dog.</let>
<let name="str2" type="string"></let>
<function name="string.middle " return="str2"> str1, 2, 5 </function>

;Result: str2="brown"
```

8.2.8.6 string.length

Description

The function gives the number of characters in a string.

Programming

Designation:	**string.length**	
Syntax:	<function name="string.length" return="< int var>"> str1 </function>	
Parameters:	str1	String
	length	Result

Example

```
<let name="length">0</let>
<let name="str1" type="string">A brown bear hunts a brown dog.</let>
<function name="string.length" return="length"> str1 </function>

; Result: length=31
```

8.2.8.7 string.replace

Description

The function replaces all the substrings found with the new string.

Programming

Designation:	**string.replace**	
Syntax:	<function name="string.replace"> string, find string, new string </function>	
Parameters:	string	String
	find string	String to be replaced
	new string	New string

Example

```
<let name="str1" type="string">A brown bear hunts a brown dog. </let>
<function name="string.replace" > str1, _T"a brown dog" ,
_T"a big salmon"</function>

; Result: str1="A brown bear hunts a big salmon."
```

8.2.8.8 string.remove

Description

The function deletes all the substrings found.

Programming

Designation:	**string.remove**	
Syntax:	<function name="string.remove" > string, remove string </function>	
Parameters:	string	String
	remove string	Substring to be deleted

Example

```
<let name="index">0</let>
<let name="str1" type="string">A brown bear hunts a brown dog. </let>
<function name="string.remove" > str1, _T"a brown dog" </function>

; Result: str1="A brown bear hunts ."
```

8.2.8.9 string.delete

Description

The function deletes the defined number of characters starting from the start position specified.

Programming

Designation:	**string.delete**	
Syntax:	function name="string.delete"> string, start index , nCount </function>	
Parameters:	string	String
	start index	Start index
	nCount	Number of characters

Example

```
<let name="str1" type="string">A brown bear hunts. </let>
<function name="string.delete" > str1, 2, 5 </function>

;Result: str1="A bear hunts."
```

8.2.8.10 string.insert

Description

The function inserts a string at the index specified.

Programming

Designation:	**string.insert**	
Syntax:	<function name="string.insert"> string, index, insert string </function>	
Parameters:	string	String variable
	index	Number of characters to be inserted
	insert string	String to be inserted

Example

```
<let name="str1" type="string">A brown bear hunts. </let>
<let name="str2" type="string">a brown dog </let>
<function name="string.insert"> str1, 19, str2 </function>

;Result: str1="A brown bear hunts a brown dog."
```

8.2.8.11 string.find

Description

The function searches the transferred string for the first match with the substring. If the substring is found, the function provides the index to the first character (starting with zero), otherwise -1.

Programming

Designation:	**string.find**	
Syntax:	<function name="string.find" return="<int val>"> str1, find string </function>	
Parameters:	string	String variable
	find string	String to be found

Example

```
<let name="index">0</let>
<let name="str1" type="string">A brown bear hunts a brown dog. </let>
<function name="string.find" return="index"> str1, _T"brown" </function>

; Result: index=2
```

8.2.8.12 string.reversefind

Description

The function searches the transferred string for the last match with the substring. If the substring is found, the function provides the index to the first character (starting with zero), otherwise -1.

Programming

Designation:	**string.reversefind**	
Syntax:	<function name="string.reversefind" return="<int val>"> str1, find string </function>	
Parameters:	string	String variable
	find string	String to be found

Example

```
<let name="index">0</let>
<let name="str1" type="string">A brown bear hunts a brown dog. </let>
<function name="string.reversefind" return="index"> str1, _T"brown" </function>

; Result: index=21
```

8.2.8.13 string.trimleft

Description

The function trims the starting characters from a string.

Programming

Designation:	**string.trimleft**	
Syntax:	<function name="string.trimleft" > str1 </function>	
Parameters:	str1	String variable

Example

```
<let name="str1" type="string">        test trim left</let>
<function name="string.trimleft"> str1 </function>

;Result: str1="test trim left"
```

8.2.8.14 string.trimright

Description

The function trims the closing characters from a string.

Programming

Designation:	**string.trimright**	
Syntax:	<function name="string.trimright" > str1 </function>	
Parameters:	str1	String variable

Example

```
<let name="str1" type="string">  test trim right      </let>
<function name="string.trimright" > str1 </function>
```

```
;Result: str1=" test trim right"
```

8.2.9 Trigonometric functions

Overview of the functions

The script language offers various trigonometric functions. The function names are reserved and cannot be overloaded.

Trigonometric functions and inverse functions:

Name	Function
sin	Sine
cos	Cosine
tan	Tangent
arcsin	Arc sine
arccos	Arc cosine
arctan	Arc tangent

Sine, cosine, tangent description

The function calculates the sine, cosine, tangent of the value transferred.

Programming

Designation:	**sin**
Syntax:	<function name="sin" return="<double val>"> double </function >
Designation:	**cos**
Syntax:	<function name="cos" return="<double val>"> double </function >
Designation:	**tan**
Syntax:	<function name="tan" return="<double val>"> double </function >
Parameters:	double Angle (0° to 360°)

Example

```
<let name= "sin_val" type="double"></let>
<function name="sin" return="sin_val"> 20.0 </function>
```

Arc sine, arc cosine, arc tangent description

The function calculates the arc sine, arc cosine, arc tangent of the value transferred.

Programming of arcsin, arccos

Designation:	**arcsin**		
Syntax:	<function name="arcsin" return="<double val>"> double </function >		
Designation:	**arccos**		
Syntax:	<function name="arccos" return="<double val>"> double </function >		
Parameters:	double	x in the range from -1 to +1	
Value range:		arcsin	y in the range from -π/2 to +π/2
		arccos	y in the range from 0 to π

Programming of arctan

Designation:	**arctan**		
Syntax:	<function name="arctan" return="<double val>"> double </function >		
Parameters:	double	x arbitrary value	
Value range:		y in the range from -π/2 to +π/2	

Example

```
<let name= "arccos_val" type="double"></let>
<function name="arccos" return="arctan_val"> 0.47 </function>
```

Tool management 9

9.1 Fundamentals

Tool management (TM)

The tool management (TM) function ensures that the right tool is in the right place on the machine at all times.

Machines, magazines, loading positions and tool buffers (e.g. spindles, grippers) form a specific system in which the tools are stored and transported. The tool management continually informs the NCK of the current location of the tools and logs it using NC part programs, PLC or HMI initiated tool movements.

During tool management commissioning the specific machine system architecture is mapped in the control. For example, one or more magazines are set up that are able to pick up tools at their locations. The "workplace" of a tool is described in the control in the form of value pairs (magazine number and location number).

Note

Scope of delivery

Tool management is included in the scope of delivery for all controls (M/T version).

The function "Spare tools for tool management" (duplo tools) is an **option**.

See also

Additional references:

- SINUMERIK 828D Parameter Manual
- A comprehensive description of the tool and magazine parameters and the internal data structure is to be found in:

  → Description of Functions Tool Management of the SINUMERIK 840D sl

  As far as the range of functions is concerned, the NCK part of this documentation is also valid for the SINUMERIK 828D.

  The description of the PLC functions and the communication between NC and PLC in this documentation is **not** valid for the SINUMERIK 828D.
- The settings of the tool management user interface are described in:

  → Commissioning Manual Base Software and Operating Software (IM9)

9.1.1 Structure of the tool management

Function structure

The software components of the control have the following tasks in the tool management:

- HMI:
  - Tool data display, input/output
  - Magazine data display, input/output
  - Load/Unload relocation dialog
- NCK:

  The tool management administrates the magazine locations. These locations might be empty, loaded with tools or assigned to oversized tools in adjacent locations. Empty locations can be loaded with other tools. The tool management provides the machine manufacturer with optimized management of tools and magazine locations. Magazine management provides extended functions such as load, unload or position tools. It also includes searches for tools, magazine locations and search strategies for replacement tools.

  For the tool monitoring functions, while the active monitoring is running, tools are disabled and no longer used. To continue machining, an equivalent tool (duplo tool) that is not disabled is used, if available.
- PLC:
  - Execute tool change
  - Move tools in the magazine
  - Gripper control
  - Magazine control if applicable
  - Safety interlocks
  - Providing the structure of tool movements in transfer step tables
  - Acknowledgment of the tool movements with acknowledgment step tables

PLC user program

The PLC user program executes the tool management jobs and acknowledges all position changes of the tools (and magazines). The monitoring and prevention of collisions is the task of the PLC user program alone, for example:

- Multiple spindles are using the same magazine.
- The paths of simultaneous jobs cross.
- As long as a large tool is located in the shifter, the chain must not be moved.

PLC-Firmware

Functions of the PLC firmware:

- Assignment of tool management jobs to the PLC user program
- Communication of PLC user program acknowledgments to the tool management
- Transfer feedback signals for each acknowledgment (acknowledgment incorrect with error number of acknowledgment OK) to PLC user program.
- In addition: Register order status

9.1.2 Components of the tool management

Tool list, magazines, magazine list

Circular and chain magazines can be managed. Other types of magazines are mapped on these. Loading points or loading stations shall be used as the magazine type for loading and unloading.

A magazine buffer combines all other locations in which tools can be placed (spindle, gripper, ...).

Note

Default settings for tool magazines

The number of magazines, which the NCK can manage, is permanently set by the system:

- PPU24x.2: Number of magazines = 3
- PPU26x.2: Number of magazines = 3
- PPU28x.2: Number of magazines = 4

Since at least one buffer and one loading point must be available, the PPU24x.2/PPU26x.2 can manage one real magazine and the PPU28x.2 can manage two real magazines.

When the control powers up, depending on the technology, the following tool magazines are pre-selected and created:

- Milling: one chain magazine with 20 locations, double gripper and a tool carrier
- Turning: one revolver with 8 locations, a tool carrier

Magazines

Information is provided by the system for all locations in the magazine, describing the content and status of the locations.

The position of a tool is described by an identifier for the magazine and an identifier for the location. Magazines have an identifier and a number, magazine locations only a number. In a real magazine (chain, turret, etc.), the position of the tool is identified by the magazine number assigned during start-up and the location within the magazine.

Example:

The T number of the tool in magazine location 7 in magazine 1: $TC_MPP6[1,7]

Tool list

The tool list contains all the tools known to the NC. These are the tools in the magazine and unloaded tools whose data is to be retained. The tool management works with loaded tools from the tool list.

Magazine list

The magazine list is a location-oriented map of the tool magazine, gripper and spindle. The tool management only works with the tools from the magazine list. Additional tools without a magazine assignment can also be selected for tool changes. The tool must be inserted in the machine manually and removed again manually after machining (manual tool).

Loading magazine

The loading magazine is the 1st internal magazine and is assigned magazine number 9999. The loading magazine has loading points for loading and unloading tools.

For the allocation of locations, one is fixed, all other locations can be assigned freely. Location 1 in the loading magazine is used for the fixed assignment. Location 1 is reserved for loading/unloading to all spindles/tool holders.

All positioning and relocation jobs to any locations (not loading points) are still handled via location 1. The stated jobs, which refer to a particular loading point, are output at the interface of this loading point. The loading points are assigned to magazines during start-up ($TC_MDP1). A loading point is an open access point to the magazine, where a tool is manually loaded and unloaded from the magazine.

Buffer

Buffers are located in the 2nd internal magazine. The buffer includes the spindle, tool holder, gripper, loader and transfer location. The buffers are managed under magazine number 9998. Each buffer element is assigned a unique location. Any location numbers may be assigned. It is recommended that all spindles or tool holders be numbered in ascending order, starting at number 1. The assignment to real magazines or of spindles/tool holders to other buffers is made during start-up ($TC_MDP2, $TC_MLSR).

Chain magazine

The setting in MD22550: $MC_TOOL_CHANGE_MODE may only have the value 1 for these magazine types.

Chain magazines do not as a rule have any additional buffer available for transportation between magazine and spindle. These additional buffers can contain tools temporarily.

Description of the buffers and loading points:

Magazine	Location	Meaning
1	xx	Real magazine 1 (chain, plate, box), position xx
9998	1	Spindle
9998	2	Gripper
9998	3	Gripper
9998	4	Toolboy
9998	5	Shifter
9999	1	Loading point for spindle, manual tool
9999	2	Magazine loading point

Circular magazine

The setting of MD22550: $MC_TOOL_CHANGE_MODE normally has the value 0.

Circular magazines do not have any additional buffer with which tools can be transported from the magazine to the spindle. The tools on circular magazines are not physically transported into the spindle, but are moved into a defined position through rotation of the turret so that machining can take place with one particular tool. The tool is only transported to the spindle or tool holder in the software. Transporting the tool to the buffer 9998/1 (spindle) serves to inform the tool management that the turret holding the requested tool has been turned to the machining position.

The programming command T = identifier initiates the tool change. T = location can be programmed as an alternative. If T = location, no tool need actually be stored in the location.

Description of the buffers and loading points:

Magazine	Location	Meaning
1	xx	Real magazine 1 (circular), position xx
9998	1	Tool holder
9999	1	Loading point for tool holder, manual tool

If the value 1 is also set for the revolver in MD22550: $MC_TOOL_CHANGE_MODE, the statement made for the chain magazines also applies here.

Consider adjacent location

Consider adjacent location is used for oversized tools. When searching for empty locations (loading, tool change) the bits 4 ... 11 are then evaluated in the magazine location parameter $TC_MPP4 (half location occupied/reserved).

See also

You can find additional information in the "Machine data for the tool management (Page 353)" section.

9.1.3 Loading and unloading tools manually

Manual tools

Bit 1 of MD22562: $MC_TOOL_CHANGE_ERROR_MODE decides whether additional tools without magazine allocation can be selected during tool change. The automatically selected tool must be inserted in the machine manually and removed again manually after machining.

Responsibility of the operator

The operator must ensure that the data block for the tool on the spindle is in the NCK, or that he/she puts the appropriate tool onto the spindle for the data block stored in the NCK. Tools which are loaded manually during machining are referred to as "manual tools".

Note

The responsibility is on the user to comply with the safety regulations via the PLC program.

An alarm (17212, 17214 or 17216) is always output to indicate that a tool change involving a manual tool has been executed. The alarm is reset by the tool change acknowledgment of the PLC user program.

The following types of tools are manual tools:

- Oversized tools
- Tools that cannot be stored in the magazine.
- Tools that may not be handled by the gripper system.

9.2 PLC - NCK user interface

Overview

The tool management receives tasks for preparing and carrying out a tool change (T command, M06), a tool movement (MVTOOL) or magazine positioning (POSM) from the part program or from the HMI. From these tasks the TM defines the location change needed for the tool and assigns this to the PLC.

Program components and interfaces:

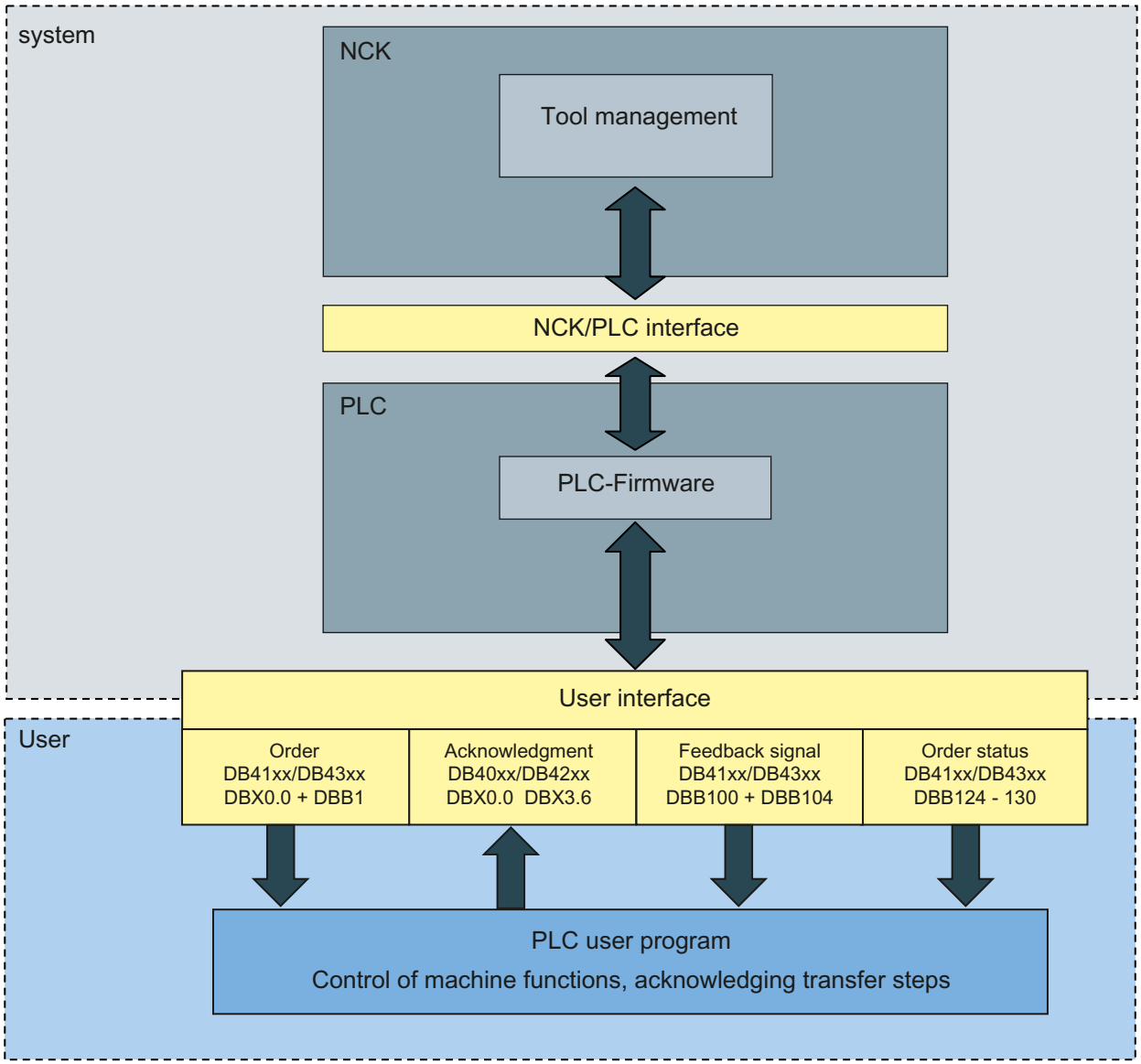


Figure 9-1 Interfaces of the tool management

The user interface provides separate data blocks for loading, unloading, relocating and magazine positioning on the one hand and tool change on the other hand.

9.2.1 Relocating, unloading, loading tool, positioning magazine

Tool and magazine movements

Per loading point there is one interface for:

- Jobs for loading, unloading, relocating a tool (MVTOOL) and for magazine positioning (POSM).

  Message to the PLC user program:

  – Job active

  – Job specification

  – Job description

  The jobs appear on the interface of the loading point from which the tool is to be exchanged.

- Acknowledgments of the PLC user program

  All acknowledgments for a job must take place in the interface of the same loading point. Acknowledgment errors should also be reset in this interface.

- Feedback from the tool management to the PLC user program

  Message to the PLC user program:

  – Status of acknowedgment

  – Error status

  – Map of the acknowledgment bits

- Job status

  Selected data from the last intermediate or end acknowledgment is saved. This data is needed by the PLC firmware for the next acknowledgment to the tool management and is readable for diagnostic purposes. This data can be used by the user program to restart after an abort (e.g. reset during a tool change).

Rule

The distribution of jobs to the interfaces takes place according to the following rule:

If the job contains a loading point (9999/x), that interface will be used.

Otherwise, the interface of the first loading point (9999/1) will be used.

All acknowledgments for a job must take place in the interface of the same loading point.

Interface signal	Meaning
xx: Load location	
DB40xx.DBX0.0 – DBX 3.6	PLC user program: Acknowledgments for loading/unloading/relocating or positioning the magazine
DB40xx.DBX9.0	PLC user program: Resetting the message "Acknowledgment error" /DB41xx.DBX100.1) and the diagnostic information in the feedback interface
DB41xx.DBX0.0	Tool management: Job for loading/unloading/relocating or positioning the magazine
DB41xx.DBB1	Tool management: Job specification
DB41xx.DBW 6 – DBW34	Job description
DB41xx.DBX100:0 DB41xx.DBX100.1	Positive feedback: Acknowledgment status, acknowledgment OK, 1 PLC cycle pending Negative feedback: Acknowledgment status, acknowledgment error, static pending
DB41xx.DBB104	Tool management: Feedback error status
DB41xx.DBX108.0 - DB41xx.DBX111.6	Map of acknowledgments for load, unload, relocate or position magazine. This map belongs to the positive or negative feedback and remains valid for the same time.
DB41xx.DBW124 – DBW130	Job status

Jobs

DB4100...41xx	Signals from tool management [r]							
xx: Load location								
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBB0								Job
DBB1				Job from NC program	Positioning	Relocating	Unloading	Loading
DBB2	Reserved							
DBB3	Reserved							
DBB4	Reserved							
DBB5	Reserved							
DBW6	Source magazine number (INT)							
DBW8	Source location number (INT)							
DBW10	Target magazine number (INT)							

DB4100...41xx	Signals from tool management [r]							
DBW12	Target location number (INT)							
DBW14: HMI → PLC								Load/unload without moving magazine

Signal description:

- Job:

  The interface contains a job. Job processing has not yet been completed with an end acknowledgment. This signal is reset after transmission of the end acknowledgment to the tool management.

- Loading:

  The "target location" parameter's magazine location should be loaded with a tool via the "source location" parameter's loading station.

- Unloading:

  The tool in the "source location" parameter's magazine location should be unloaded to the "target location" parameter's unloading station.

- Relocating:

  The tool in the "source location" parameter's magazine location should be relocated to the "target location" parameter's magazine location.

- Positioning:

  The "source location" parameter's magazine location should be positioned at the "target location" parameter's change/loading/unloading station. The tool remains in its magazine location.

- NC program positions magazine:

  The positioning job comes from the part program.

- Loading/unloading without moving magazine:

  HMI sets/deletes this signal when requested by the operator. If the bit is active, there must be no traversing motion of the magazine, only a mechanical unlocking/locking of the location. The load/unload command must be acknowledged after the action. For a position and relocate request, this signal is not valid for a traversing motion.

- Source location:

  Magazine and location number of a tool that traverses or should be positioned at a change or loading station.

- Target location:

  Magazine and location number to where a tool is moving or to where a magazine location should be positioned.

Acknowledgments

DB4000...40xx	Signals to tool management [r/w]							
xx: Load location								
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBB0	Acknowl-edgment step 7	Acknowl-edgment step 6	Acknowl-edgment step 5	Acknowl-edgment step 4	Acknowl-edgment step 3	Acknowl-edgment step 2	Acknowl-edgment step 1	Total acknowl-edgment
DBB1	Acknowl-edgment step 15	Acknowl-edgment step 14	Acknowl-edgment step 13	Acknowl-edgment step 12	Acknowl-edgment step 11	Acknowl-edgment step 10	Acknowl-edgment step 9	Acknowl-edgment step 8
DBB2	Acknowl-edgment step 23	Acknowl-edgment step 22	Acknowl-edgment step 21	Acknowl-edgment step 20	Acknowl-edgment step 19	Acknowl-edgment step 18	Acknowl-edgment step 17	Acknowl-edgment step 16
DBB3	Reserved	Acknowl-edgment step 30	Acknowl-edgment step 29	Acknowl-edgment step 28	Acknowl-edgment step 27	Acknowl-edgment step 26	Acknowl-edgment step 25	Acknowl-edgment step 24
DBB4	Reserved							
DBB5	Reserved							
DBB6	Reserved							
DBB7	Reserved							
DBB8	Reserved							
DBB9								Resetting the acknowledgment error

Signal description:

- Total acknowledgment:

  At a 0/1 edge the end acknowledgment, with status 99, is sent to the current job (job complete, all target positions have been reached). As long as the signal is present, no changes may be made to the data of this interface!

  This signal is reset by the PLC firmware after the acknowledgment has been transferred to the tool management.

- Acknowledgment step 1…30:

  At a 0/1 edge, the appropriate acknowledgment step from the acknowledgment step table is sent to the tool management. As long as the signal is present, no changes may be made to the data of this interface and the variable transfer-step table!

  This signal is reset by the PLC firmware after the acknowledgment has been transferred to the tool management.

- Resetting the acknowledgment error:

  Resetting the message "Acknowledgment error" /DB41xx.DBX100.1) and the diagnostic information in the feedback interface.

Feedback reports

DB4100...41xx	Signals from tool management [r]							
xx: Load location								
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBB100							Acknowl-edgment error	Acknowl-edgment OK
DBB101	Reserved							
DBB102	Reserved							
DBB103	Reserved							
DBW104	Error status (WORD)							
DBB106	Reserved							
DBB107	Reserved							
DBB108	Acknowl-edgment step 7	Acknowl-edgment step 6	Acknowl-edgment step 5	Acknowl-edgment step 4	Acknowl-edgment step 3	Acknowl-edgment step 2	Acknowl-edgment step 1	Acknowl-edgment step 0
DBB109	Acknowl-edgment step 15	Acknowl-edgment step 14	Acknowl-edgment step 13	Acknowl-edgment step 12	Acknowl-edgment step 11	Acknowl-edgment step 10	Acknowl-edgment step 9	Acknowl-edgment step 8
DBB110	Acknowl-edgment step 22	Acknowl-edgment step 22	Acknowl-edgment step 21	Acknowl-edgment step 20	Acknowl-edgment step 19	Acknowl-edgment step 18	Acknowl-edgment step 17	Acknowl-edgment step 16
DBB111	Reserved	Acknowl-edgment step 30	Acknowl-edgment step 29	Acknowl-edgment step 28	Acknowl-edgment step 27	Acknowl-edgment step 26	Acknowl-edgment step 25	Acknowl-edgment step 24

Signal description:

- **Acknowledgment OK (DB41xx.DBX100.0):**

  The acknowledgment of the PLC user program (area DB40xx.DBB0 to DBB3) was transferred without error to the tool management. This signal is reset after one PLC cycle.

- **Acknowledgment error (DB41xx.DBX100.1):**

  Negative feedback of acknowledgment status. There is an error in the acknowledgment of the PLC user program (area DB40xx.DBB0 to DBB3). The cause of the error is displayed in "Error status."

  The "acknowledgment error" bit is only set if the acknowledgment from the PLC firmware is accepted as error free and the tool management detects and signals an error in the acknowledged tool transfer (for example, when the target location for tool transfer is occupied).

  Errors that are detected directly by the tool management in the NC before an acknowledgment is sent by the PLC user program, do not result in the setting of bit 100.1.

  If an error is present that prevents transfer of the acknowledgment to the tool management (error status 1 to 7), the error is only output on the interface of the tool management and not by the NC (no NC alarm!).

  If necessary, signal such errors with a user PLC alarm in the PLC user program.

This signal remains pending statically until the error has been acknowledged (set the bit "Reset acknowledgment error" DB40xx.DBX9.0) by the user. The interface in DB40xx.DBB0 to DBB3 is disabled if bit "acknowledgment error" is pending. Incoming acknowledgment bits are not evaluated by the PLC firmware and are cleared when bit "Reset acknowledgment error" is set.

Error status:

If there is an error, the error status (DB41xx.DBB104) contains a diagnostic number unequal to zero.

Status	Meaning
0	No error
1	Multiple acknowledgment signals at the same time
2	Acknowledgment without job
3	Invalid transfer step number
4	There is no job for a position specification
5	The status does not permit a location change (acknowledgment status 0 is used)
7	An impermissible acknowledgment status has been used
Other values:	The number corresponds to the error message of the tool management in the NCK caused by this transfer

The error status is reset by acknowledgment of the error by the user.

Map of acknowledgments (DB41xx.DBB108 to DBB111)

The acknowledgments last set by the PLC user program (DB40xx.DBB0 to DBB3) are set and reset by the PLC firmware together with the bits "Acknowledgment OK" or "Acknowledgment error". If there is an error, the user uses these statically pending bits to see which acknowledgment step triggered the error. If the PLC user program incorrectly sets multiple acknowledgment bits, these are also entered one-to-one in the map.

Job status

DB4100...41xx	Signals from tool management [r]							
xx: Load location								
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBW124	Current magazine number of the tool (INT)							
DBW126	Current location number of the tool (INT)							
DBW128	Target magazine number of the tool (INT)							
DBW130	Target location number of the tool (INT)							

9.2.2 Tool change

Interface description

Per tool holder/spindle there is one interface for:

- Jobs for preparing and executing the tool change.

  Message to the PLC user program: Job active; job specification and job description.

  The jobs appear in the toolholder interface (of the spindle) where a tool is to be exchanged.

- Acknowledgments of the PLC user program

  All acknowledgments for a job must take place in the interface of the same tool holder (of the same spindle). Acknowledgment errors should also be reset in this interface.

- Feedback from the tool management to the PLC user program

  Message to the PLC user program: Acknowledgment status, error status, map of the acknowledgment bits.

- Job status

  Selected data from the last intermediate or end acknowledgment is saved. This data is needed by the PLC firmware for the next acknowledgment to the tool management and is readable for diagnostic purposes. This data can be used by the user program to restart after an abort (e.g. reset during a tool change).

End acknowledgment for tool change

A common end acknowledgment (tool change via T command using turret) or separate end acknowledgment (Txx and M206 in separate blocks, default milling) is possible for "Prepare tool change" and "Implement tool change." With the appropriate MD setting, the end acknowledgment for the job to prepare for a tool change permits the NCK preprocessing to continue.

The machine data with which the response of block preprocessing, main run, and the various acknowledgment responses are defined are described in Chapter Machine data for the tool management (Page 353).

The main NCK run can be continued with the end acknowledgment to the job "Carry out tool change". Therefore, these end acknowledgments should take place as early as possible. This can mean that the end acknowledgment can take place before the old tool is in the magazine (e.g. the new tool is in the spindle, the old tool in the toolboy). The rest of the steps to bring the old tool into the magazine must then be communicated asynchronously. The same interface should be used as for synchronous acknowledgments.

Interface signal	Meaning
xx: Spindle index / tool holder	
DB42xx.DBX0.0 – DBX 3.6	PLC user program: Preparing and carrying out acknowledgments for tool change

Interface signal	Meaning
DB42xx.DBX9.0	PLC user program: Resetting the message "Acknowledgment error" (DB43xx.DBX100.1) and the diagnostic information in the feedback interface
DB43xx.DBX0.0	Tool management: Job for "Prepare tool change" and "Execute tool change"
DB43xx.DBB1	Tool management job specification
DB43xx.DBW 6 – DBW34	Job description
DB43xx.DBX100.0 DB43xx.DBX100.1	Positive feedback: Acknowledgment status, acknowledgment OK, one PLC cycle pending Negative feedback: Acknowledgment status, acknowledgment error, static pending
DB43xx.DBX100:0 DB43xx.DBX100:1	Positive feedback: Acknowledgment status, one PLC cycle pending Negative feedback: Acknowledgment status, static pending.
DB43xx.DBX100:0	Tool management feedback: Acknowledgment status
DB43xx.DBB104	Tool management feedback: Error status
DB43xx.DBX108.0 - DB43xx.DBX111.6	Map of acknowledgments for tool change: This map belongs to the positive or negative feedback and remains valid for the same time.
DB43xx.DBW124 – DBW138	Job status

Jobs

DB4300...43xx	Signals from tool management [r]							
xx: Tool holder								
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBB0	--	--	--	--	--	--	--	Job
DBB1	Tool remains in the spindle	Unload manual tool	Load manual tool	No old tool	T0	Prepare change	Change tool (initiated by: M06)	Fixed-location coded
DBB2	Reserved							
DBB3	Reserved							
DBB4	Reserved							
DBB5	Reserved							
DBW6	Source magazine number for new tool (INT)							
DBW8	Source location number for new tool (INT)							
DBW10	Reserved							
DBW12	Reserved							
DBW14	Reserved							
DBW16	Reserved							
DBW18	Target magazine number for old tool (INT)							
DBW20	Target location number for old tool (INT)							

DB4300...43xx	Signals from tool management [r]							
DBW22	Location type (INT)							
DBW24	Size left (INT)							
DBW26	Size right (INT)							
DBW28	Reserved							
DBW30	Reserved							
DBB32	Tool status for new tool							
	--	--	--	Master tool	To be loaded	To be unloaded	Blocked	Identifier for tool
DBB33	Tool status for new tool							
	Tool has been in use	Tool fixed-location-coded	Tool being changed	Prewarn-ing limit reached	Measuring tools	Tool disabled	Tool enabled	Active tool
DBW34	New tool: Internal T number of NCK (INT)							
DBW36	Reserved							
DBW38	Reserved							
DBW40	Reserved							
DBW42	Reserved							
DBW44	Free parameter 1 (DWORD)							
DBW48	Free parameter 2 (DWORD)							
DBW52	Free parameter 3 (DWORD)							

Signal description:

- Job:

  The interface contains a job. Job processing has not yet been completed with an end acknowledgment. This signal is reset after transmission of the end acknowledgment to the tool management.

- Fixed-location coded: The new tool is fixed-location-coded.

- Execute tool change:

  The new tool is to be loaded into the tool holder / the spindle. The old tool is to be brought back to a magazine location. This job always requires an end acknowledgment.

- Prepare a tool change:

  Initialize new tools. If necessary, position magazine location for old tool at the changing point. This job requires an individual end acknowledgment. If there is a parallel job "Execute change," end acknowledgment for the preparation is not necessary.

- T0: T0 has been programmed (empty tool holder / spindle).

- No old tool:

  Tool change into the previously empty tool holder / spindle.

- Load manual tool:

  A manual tool is to be loaded. The HMI displays the tool which is to be loaded.

- Unload manual tool:

  The tool is to be changed via manual operation.

- Tool remains in spindle:

  The bit is set at a change from tool holder → spindle to tool holder → spindle. Triggers can be, for example, reset start mode or block search.

- Source location for the new tool:

  Magazine and location number from where the new tool comes (mostly a location in a real magazine).

- Target location for the old tool:

  Magazine and location number to where the old tool is to be transported (mostly a location in a real magazine).

- Origin of the new tool:
  - Internal T number: Internal T number of the new tool
  - Tool status: Tool status of the new tool
  - Location type: Location type of the new tool
  - Size: Size (right, left, up, down) of the new tool
  - User-definable parameters: Three user-definable parameters which are transferred by the part program to the PLC user program.

Acknowledgments

DB4200 ... 42xx	**Signals to tool management [r/w]**							
xx: Tool holder								
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBB0	Acknowl-edgment step 7	Acknowl-edgment step 6	Acknowl-edgment step 5	Acknowl-edgment step 4	Acknowl-edgment step 3	Acknowl-edgment step 2	Acknowl-edgment step 1	Total acknowl-edgment
DBB1	Acknowl-edgment step 15	Acknowl-edgment step 14	Acknowl-edgment step 13	Acknowl-edgment step 12	Acknowl-edgment step 11	Acknowl-edgment step 10	Acknowl-edgment step 9	Acknowl-edgment step 8
DBB2	Acknowl-edgment step 23	Acknowl-edgment step 22	Acknowl-edgment step 21	Acknowl-edgment step 20	Acknowl-edgment step 19	Acknowl-edgment step 18	Acknowl-edgment step 17	Acknowl-edgment step 16
DBB3	Reserved	Acknowl-edgment step 30	Acknowl-edgment step 29	Acknowl-edgment step 28	Acknowl-edgment step 27	Acknowl-edgment step 26	Acknowl-edgment step 25	Acknowl-edgment step 24
DBB4	Reserved							
DBB5	Reserved							
DBB6	Reserved							
DBB7	Reserved							
DBB8	Reserved							
DBB9								Resetting the acknowledg-ment error

Signal description:

- Total acknowledgment:

  At a 0/1 edge the end acknowledgment, with status 99, is sent to the current job (job complete, all target positions have been reached). As long as the signal is present, no changes may be made to the data of this interface!

  This signal is reset by the PLC firmware after the acknowledgment has been transferred to the tool management.

- Acknowledgment step 1…30:

  At a 0/1 edge, the appropriate acknowledgment step from the acknowledgment step table is sent to the tool management. As long as the signal is present, no changes may be made to the data of this interface and the variable transfer-step table!

  This signal is reset by the PLC firmware after the acknowledgment has been transferred to the tool management.

- Resetting the acknowledgment error:

  Resetting the message Acknowledgment error (DB43xx.DBX100.1) and the diagnostic information in the feedback interface.

Feedback reports

DB4300...43xx	Signals from tool management [r]							
xx: Load location								
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBB100							Acknowl-edgment error	Acknowl-edgment OK
DBB101	Reserved							
DBB102	Reserved							
DBB103	Reserved							
DBW104	Error status (WORD)							
DBB106	Reserved							
DBB107	Reserved							
DBB108	Acknowl-edgment step 7	Acknowl-edgment step 6	Acknowl-edgment step 5	Acknowl-edgment step 4	Acknowl-edgment step 3	Acknowl-edgment step 2	Acknowl-edgment step 1	Acknowl-edgment step 0
DBB109	Acknowl-edgment step 15	Acknowl-edgment step 14	Acknowl-edgment step 13	Acknowl-edgment step 12	Acknowl-edgment step 11	Acknowl-edgment step 10	Acknowl-edgment step 9	Acknowl-edgment step 8
DBB110	Acknowl-edgment step 23	Acknowl-edgment step 22	Acknowl-edgment step 21	Acknowl-edgment step 20	Acknowl-edgment step 19	Acknowl-edgment step 18	Acknowl-edgment step 17	Acknowl-edgment step 16
DBB111	Reserved	Acknowl-edgment step 30	Acknowl-edgment step 29	Acknowl-edgment step 28	Acknowl-edgment step 27	Acknowl-edgment step 26	Acknowl-edgment step 25	Acknowl-edgment step 24

Signal description:

- **Acknowledgment OK (DB43xx.DBX100.0): Positive feedback of acknowledgment status.**

  The acknowledgment of the PLC user program (area DB42xx.DBB0 to DBB3) was transferred without error to the tool management. This signal is reset after one PLC cycle.

- **Acknowledgment error (DB43xx.DBX100.1): Negative feedback of acknowledgment status.**

  There is an error in the acknowledgment of the PLC user program (area DB42xx.DBB0 to DBB3). The cause of the error is displayed in "Error status."

  The "acknowledgment error" bit is only set if the acknowledgment from the PLC firmware is accepted as error free and the tool management detects and signals an error in the acknowledged tool transfer (for example, when the target location for tool transfer is occupied).

  Errors that are detected directly by the tool management in the NC before an acknowledgment is sent by the PLC user program, do not result in the setting of bit 100.1.

  If an error is present that prevents transfer of the acknowledgment to the tool management (error status 1 to 7), the error is only output on the interface of the tool management and not by the NC (no NC alarm!).

  If necessary, signal such errors with a user PLC alarm in the PLC user program.

  This signal remains pending statically until the error has been acknowledged (set the bit "Reset acknowledgment error" DB4200.DBX9.0) by the user. The interface in DB42xx.DBB0 to DBB3 is disabled if bit "acknowledgment error" is pending. Incoming acknowledgment bits are not evaluated by the PLC firmware and are also cleared when bit "Reset acknowledgment error" is set.

Error status:

If there is an error, the error status (DB43xx.DBB104) contains a diagnostic number unequal to zero.

Status	Meaning
0	No error
1	Multiple acknowledgment signals at the same time
2	Acknowledgment without job
3	Invalid transfer step number
4	There is no job for a position specification
5	The status does not permit a location change (acknowledgment status 0 is used)
7	An impermissible acknowledgment status has been used
Other values:	The number corresponds to the error message of the tool management in the NCK caused by this transfer.

The error status is reset by acknowledgment of the error by the user.

Map of acknowledgments (DB43xx.DBB108 to DBB111)

The acknowledgments last set by the PLC user program (DB42xx.DBB0 to DBB3) are set and reset by the PLC firmware together with the bits "Acknowledgment OK" or "Acknowledgment error." If there is an error, the user uses these statically pending bits to see which acknowledgment step triggered the error. If the PLC user program incorrectly set multiple acknowledgment bits, these are also entered one-to-one in the map.

Job status

DB4300 ... 43xx	Signals from tool management [r]							
xx: Tool holder								
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBW124	Current magazine number for new tool (INT)							
DBW126	Current location number for new tool (INT)							
DBW128	Target magazine number for new tool (INT)							
DBW130	Target location number for new tool (INT)							
DBW132	Current magazine number for old tool (INT)							
DBW134	Current location number for old tool (INT)							
DBW136	Target magazine number for old tool (INT)							
DBW138	Target location number for old tool (INT)							

9.2.3 Transfer-step and acknowledgment-step tables

Configurable step tables

There are configurable tables available in the data blocks TM_CTS (DB9900), TM_VTS (DB9901) and TM_ACK (DB9902), which are used to describe the tool movement sequences.

Interface signal	Name	Meaning
DB9900	TM_CTS	Constant transfer-step table (configurable)
DB9901	TM_VTS	Variable transfer-step table (can be configured and written to from PLC user program)
DB9902	TM_ACK	Acknowledgment-step table (configurable)

The data blocks DB40xx, 41xx, 42xx and 43xx are system blocks and are created automatically by the control.

The data blocks DB9900, DB9901 and DB9902 are made available by the Programming Tool under Libraries / Special data blocks. The blocks are not yet filled with the necessary data. The user must copy them into the PLC project and edit them.

Transfer-step tables

The individual tool movements are defined as transfer steps – tool from magazine location x/y to magazine location m/n. Acknowledgment steps can be defined with these transfer steps. DB9900 contains permanently configured transfer steps (constant transfer-step table). DB 9901 can be changed by the PLC user program; for example, for acknowledging intermediate steps like magazine locations for tool change preparation (variable transfer-step table).

DB9900	**Constant transfer-step table [r]**							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBW0	Transfer step 1 Source magazine number (INT)							
DBW2	Transfer step 1 Source location number (INT)							
DBW4	Transfer step 1 Target magazine number (INT)							
DBW6	Transfer step 1 Target location number (INT)							
DBW8	Transfer step 2 Source magazine number (INT)							
DBW10	Transfer step 2 Source location number (INT)							
DBW12	Transfer step 2 Target magazine number (INT)							
DBW14	Transfer step 2 Target location number (INT)							
	…							
DBW504	Transfer step 64 Source magazine number (INT)							
DBW506	Transfer step 64 Source location number (INT)							
DBW508	Transfer step 64 Target magazine number (INT)							
DBW510	Transfer step 64 Target location number (INT)							

DB9901	**Variable transfer-step table [rw]**							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBW0	Transfer step 101 Source magazine number (INT)							
DBW2	Transfer step 101 Source location number (INT)							
DBW4	Transfer step 101 Target magazine number (INT)							
DBW6	Transfer step 101 Target location number (INT)							
DBB 8	Transfer step 102 Source magazine number (INT)							
DBW10	Transfer step 102 Source location number (INT)							
DBW12	Transfer step 102 Target magazine number (INT)							
DBW14	Transfer step 102 Target location number (INT)							
	…							
DBW504	Transfer step 164 Source magazine number (INT)							
DBW506	Transfer step 164 Source location number (INT)							
DBW508	Transfer step 164 Target magazine number (INT)							
DBW510	Transfer step 164 Target location number (INT)							

Acknowledgment step table

Each entry indexes two transfer steps (for the new and old tool) and gives the corresponding status reached. The acknowledgment-step table in DB9902 is used jointly for acknowledgments on the interface of the loading point and on the interface of the tool holder.

DB9902	Acknowledgment-step table [r]							
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBB0	Acknowledgment step 1 Transfer step for new tool (BYTE)							
DBB1	Acknowledgment step 1 Transfer step for old tool (BYTE)							
DBB2	Acknowledgment step 1 Acknowledgment status (BYTE)							
DBB3	Acknowledgment step 1 reserved							
DBB4	Acknowledgment step 2 Transfer step for new tool (BYTE)							
DBB5	Acknowledgment step 2 Transfer step for old tool (BYTE)							
DBB6	Acknowledgment step 2 Acknowledgment status (BYTE)							
DBB7	Acknowledgment step 2 reserved							
	…							
DBB116	Acknowledgment step 30 Transfer step for new tool (BYTE)							
DBB117	Acknowledgment step 30 Transfer step for old tool (BYTE)							
DBB118	Acknowledgment step 30 Acknowledgment status (BYTE)							
DBB119	Acknowledgment step 30 reserved							

See also

PLC Program Blocks (Page 360)

9.3 Machine data for the tool management

Machine data (default setting)

The following machine data has already been preset for tool management or will be set with "default data" during booting: These settings can be changed as necessary.

MD number	Designation	Value	
10715[0]	M_NO_FCT_CYLE	6	(M version)
10716[0]	M_NO_FCT_CYLCLE_NAME	L6	(M version)
10717	T_NO_FCT_CYLCLE_NAME	TCHANGE	(T version)
17500	MAXNUM_REPLACEMENT_TOOLS	0	
20124	TOOL_MANAGEMENT_TOOLHOLDER	1	
20270	CUTTING_EDGE_DEFAULT	1	
20310	TOOL_MANAGEMENT_MASK	181400F 180400F	(T version) (M version)
22550	TOOL_CHANGE_MODE	0 1	(T version) (M version)
22560	TOOL_CHANGE_MCODE	206	
22562	TOOL_CHANGE_ERROR_MODE	0	(for manual tool)

MD20270: $MC_CUTTING_EDGE_DEFAULT

If no cutting edge is programmed after a tool change, the cutting edge number set in $MC_CUTTING_EDGE_DEFAULT is used.

MD20270: $MC_CUTTING_EDGE_DEFAULT	
	Basic setting of tool cutting edge without programming (DWORD)
> 0	Number of the cutting edge that is selected with M206 Cutting edge selection is also active if followed by D programming.
= 1	Default setting
= 0	No cutting edge is initially active after a tool change. Any tool offset active before tool change is deselected (corresponds to D0!). Cutting edge selection only active with D programming.
= -1	Tool edge number of old tool also applies to new tool.
= -2	Tool edge offset of old tool remains active until D is programmed.

The setting in MD20270 affects the block preparation of the NC. To prevent preprocessing stop from occurring when tool change command is called until it is acknowledged again, perform NC functions without tool offset in the tool change subprogram, for example, traversing axes or output of auxiliary functions.

Example:

Requirement: MD20270: $MC_CUTTING_EDGE_DEFAULT= 0 or = -2

After the tool change command M206 the axes can continue travel without having to wait for the tool change acknowledgment and execute traversing blocks without tool compensation. Travel only stops in a block with compensation selected (D no.) until end of tool change is signaled by the PLC.

Sequence in the part program:

```
N10 T="Drill18"           ; Tool change preparation
N20 M6                    ; Tool change subprogram called
Tool change subprogram L6:
N10 M206                  ; Tool change
N20 D0                    : Compensation deselected
N40 Y150 M79              ; Traverse machine axes
N50 G01 D1 X10            ; Activate tool compensation.
                          ; Check whether tool has been changed.
                          ; Preprocessing stop is maintained until tool change
                          preparation is completed. The main run waits at N50 (D1)
                          until tool change has been executed and acknowledged.
```

MD20310: $MC_TOOL_MANAGEMENT_MASK

Settings:

MD20310: $MC_TOOL_MANAGEMENT_MASK		
		Activating the tool management functions
T and M version:		
Bit 0	= 1	Tool management active: The functions of tool management are enabled for the current channel.
Bit 1	= 1	Monitoring functions of tool management active: The functions for monitoring tools (tool life and workpiece count) are enabled.
Bit 2	= 1	OEM functions active
Bit 3	= 1	Adjacent location consideration active
Bit 14	= 1	Tool and offset selection according to the settings in: MD20110 $MC_RESET_MODE_MASK MD20112 $MC_START_MODE_MASK
Bit 23	= 1	With offset selection no synchronization with main run.

MD20310: $MC_TOOL_MANAGEMENT_MASK		
Bit 24	= 1	Using an asynchronous transfer with acknowledgment status 201, you can move a tool to a location reserved for another tool with "reserved for tool from buffer". This location reservation is then removed before the movement is executed ("Reserved for new tool to be loaded" (bit value="H8") remains effective).
Additionally only for T version:		
Bit 16	= 1	T location number is active.

If a locked tool is at the programmed location, the location of a replacement tool (if available) is output as job from the tool management when the option "Spare tools for tool management" option is set.

Channel MD52270: $MCS_TM_FUNCTION_MASK

Settings:

SD52270: $MCS_TM_FUNCTION_MASK	
	Tool management function mask
Bit 0:	Creating tool at the magazine location is not permitted. Tools can only be created outside of the magazine.
Bit 1:	Load/unload lock, if the machine is not in the reset state. Tools can only be loaded/unloaded when the relevant channel is in the reset state.
Bit 2:	Load/unload lock for EMERGENCY STOP. Tools can only be loaded/unloaded when EMERGENCY STOP is not active.
Bit 3:	Tool in/out of spindle load/unload locked. Tools cannot be loaded to or unloaded from the spindle.
Bit 4:	Loading is performed directly into the spindle. Tools are only loaded directly into the spindle.
Bit 5:	Reserved
Bit 6:	Reserved
Bit 7:	Create tool using the T number. The T number of the tool must be entered when creating a tool.
Bit 8:	Hide tool relocation. The "Relocate tool" function is hidden in the user interface.
Bit 9:	Hide magazine positioning. The "Position magazine" function is hidden in the user interface.
Bit 10:	Reactivate tool with magazine positioning. Before Reactivate, the tool is positioned at the loading point.
Bit 11:	Reactivate tool in all monitoring types. When reactivating a tool, all the monitoring types for this tool enabled in the NC are reactivated. I.e. also the monitoring types that are not set for the relevant tool, but are only in the background.
Bit 12:	Hide reactivating tool. The "Reactivate tool" function is hidden in the user interface.

Settings at the user interface

The settings for dialogs of the tool management at the user interface are generally made using SD54215: $SNS_TM_FUNCTION_MASK_SET:

SD54215: $SNS_TM_FUNCTION_MASK_SET	
	Tool management function mask
Bit 0:	Diameter display for rotating tools. For rotating tools, the diameter is displayed and not the radius.
Bit 1:	M4 is the default direction of rotation for all turning tools. When creating turning tools, the direction of rotation is pre-assigned with M4.
Bit 2:	Create tool without name suggestion.
Bit 3:	Input lock, tool name and tool type for loaded tools. For loaded tools, the tool name and the tool type can no longer be changed.
Bit 4:	Input lock for loaded tools if the channel is not in the reset state.
Bit 5:	Calculate by adding tool wear entries. Wear data is entered in addition to the already existing wear value.
Bit 6:	Numerical input of the tool identification. Only numbers are permitted for the input of the tool identification.
Bit 7:	Hide tool monitoring parameters. The tool monitoring parameters are hidden in the user interface.
Bit 8:	Diameter display for face axis → Geometry. The geometry value of the face axis is displayed as diameter value.
Bit 9:	Diameter display for face axis → Wear. The wear value of the face axis is displayed as diameter value.
Bit 10:	Enable tool load/relocate to buffer storage locations. The magazine number can be entered into the "Load" dialog. It is therefore possible to access the buffer storage via magazine number 9998.

Tool call for tool measuring in JOG

For special applications (e.g. measuring cycles in JOG), it is necessary to load a specific tool (e.g. miller_15, duplo number 2) - independent of its status - onto the spindle or the toolholder; this can be the case, for example, if a tool was being used but was locked by the tool monitoring function. For the new release, it should now be measured.

The NC command TCA is available to call a tool independent of the status and by selecting duplo and tool holder number; this is used when measuring tools in JOG.

```
TCA ("Tool name", duplo No., toolholder No.)
```

TCA behaves like the T command with respect to alarm and command output to the PLC.

Boundary conditions:

The following boundary conditions must be observed for an active T function replacement (default setting for turning):

- TCA cannot be substituted (T replacement cycle).
- The cycle defined in MD10717: $MN_T_NO_FCT_CYCLE_NAME and which is executed for a T call, is not started by the TCA command.

  To ensure that a tool change cycle also starts when TCA is programmed, the language command TCA must be redefined in _TCA (default setting NC).
- The machine functions of the TCHANGE.SPF must be transferred into the TCA.SPF cycle. The TCA.SPF cycle is in the Toolbox CD.

Example:

The following example describes the sequence:

```
PROC TCA(STRING[64] _TOOL_NAME,INT _DUPLO,INT _TH_NO)
                                          ;VERSION: 01.00.07. Oct 22, 2009
                                          ;CHANGE: 01.00.07. Oct 22, 2009
TCA(_TOOL_NAME,_DUPLO,_TH_NO)             ;Tool call to NC
; here, insert the machine functions for the change
M17
```

Note

The offset selection corresponding to MD20270: $MC_CUTTING_EDGE_DEFAULT acts in the same way as for the T command.

TCA and D must not be programmed in the same block.

Default setting for turning technology

Two machine data items determine the function for turning technology:

- **MD22550: $MC_TOOL_CHANGE_MODE = 0**

  Setting for a revolver magazine: The new tool is changed immediately with the T function. No additional M command is used. No distinction is made between Prepare tool change and Execute tool change.

  The function "Manual tools" is not enabled for this case.

- **MD20310: $MC_TOOL_MANAGEMENT_MASK = 81400F (bit 16=1)**

  Bit 16 is used to set the tool programming type:

  T = "x" with x as tool identifier

  Tx, with x as location number of the magazine containing the tool used for machining.

  When the function is active, T1 selects the tool in location number 1 instead of the tool with identifier "1". Then the identifier of the tool in this location is ascertained (e.g. "FINISHING TOOL") The procedure is as if T="FINISHING_TOOL" had been programmed.

  If T = location number, no tool need actually be stored in the location.

- **MD10717: $MN_T_NO_FCT_CYCLE_NAME = TCHANGE**

  Name of tool change cycle for T function replacement.

  For a description, refer to the Chapter Example: Tool change cycle for turning machine (Page 388)

- **ISO dialects**

  Default setting for programming the tool offset:

MD10888	$MN_EXTERN_DIGITS_TOOL_NO = 0
MD10889	$MN_EXTERN_DIGITS_OFFSET_NO = 2
MD10890	$MN_EXTERN_TOOLPROG_MODE = 4

Programming T101 and T0101 leads to the same result: T1 H01

Configuring the magazine

The magazine configuration can be created either with the start-up tool or with a configuration program. The configuration program is selected and started as a normal part program.

An NC POWER ON is needed after changing the magazine configuration. The changed configuration is only displayed after restarting the NC.

See also

Examples:

- Configuration of a chain magazine with a dual gripper (Page 393)
- Configuration of a revolver magazine (Page 382)

  The program is also to be found on the Toolbox CD.

References

Further information:

- SINUMERIK 828D List Manual: Machine data and interface signals
- SINUMERIK 802D sl/828D/840D sl: Function Manual: ISO dialects
  - The description "Substitution by a replacement cycle for a tool selection block" is valid for the ISO mode (G291).
  - The description "Tool change and tool offsets" is valid when programming the tool offset.

9.4 PLC Program Blocks

9.4.1 Acknowledgment process

Information to tool management

The tool management expects acknowledgment of its orders in order to track and carry the actual tool positions. At least one acknowledgment is required for each order. This is sufficient for many applications.

Acknowledgment takes place either via the table defined in DB9902, or in one step with the total acknowledgment (DB40xx/42xx DBX0.0) after a completely finished tool management order using a 0/1 edge (set) of the corresponding bits in the user interface.

As long as the acknowledgment signal is present, no changes may be made to the data of this interface! This signal is reset by the PLC firmware after the acknowledgment has been transmitted to the tool management. In certain circumstances resetting may take place after several PLC cycles.

There are further advantages if the tool management is also informed of intermediate positions:

- Information about intermediate positions:

  If the intermediate tool positions are known in the tool management, the allocation of the buffer magazine may be queried. This makes powering up easier after switching off and on again or after canceling a command (e.g. via reset). If the tool currently being changed is needed again immediately, it can be loaded back into the spindle from a buffer location without loading it into the magazine first.

- Information about magazine positions:

  If the tool management knows at which magazine location there is a transfer point (changing point for the spindle, loading point), it can determine the shortest paths in the magazine for finding an empty location or for selecting a new tool. During orders the tool management can usually recognize the magazine position from intermediate acknowledgments (e.g. tool transfer between the real magazine and the buffer) or from the end acknowledgment (e.g. "magazine positioning" order complete). If the magazine is positioned by the PLC user program itself (e.g. by means of HMI or machine keys) without a tool management order, this must be communicated to the tool management via synchronous signals.

9.4.2 Types of acknowledgment

Tool and magazine movements

The tool management distinguishes between synchronous acknowledgment and an asynchronous job-independent message.

Synchronous acknowledgment

- Acknowledgment of the intermediate steps of a job (tool management registers the current position changes of tools, part program must wait).

  The tool management is informed of intermediate steps of a job by an intermediate acknowledgment. In intermediate acknowledgments, only the target position of the intermediate step is relevant. The source position is known from the job or the last intermediate acknowledgment. During a tool change, two tools (new and old) can also be acknowledged simultaneously. Intermediate acknowledgments are only possible **before** the end acknowledgment.

- End acknowledgment of a job (part program can be continued)

  An end acknowledgment is necessary for every job. The end acknowledgment allows the part program to continue and frees up the job interface for new jobs. It should take place as soon as possible (e.g. as soon as the new tool is in the spindle and a collision is no longer expected). The tool management can be informed asynchronously of further steps after the end acknowledgment of a job (e.g. the return path of the old tool into the magazine).

Asynchronous job-independent message

Position change of a tool or magazine ("asynchronous message", e.g. when the PLC changes a tool position via machine control panel operation without tool change job).

An asynchronous message can be used to inform the tool management of a tool or magazine movement, independently of a job. An asynchronous message must always contain a source position (from) **and** target position (to).

Tool movements within a magazine (relocate tool) can only be carried out with magazine locations which are actually occupied. Empty transfers are not permissible. Two asynchronous transfers can be implemented in a single message. In this case the interface must be used for tool change DB42xx.

Effect of acknowledgments

Effect of acknowledgments on the job and the part program:

- Intermediate and end acknowledgment take place synchronously with the job.
  - The part program must wait.
  - No new job can come yet.
- Message of an asynchronous transfer:
  - The part program continues to run.
  - The message is completely independent of any job.

9.4.3 Acknowledgment states

Acknowledgment states and their meaning:

The status indicated by the respective type of acknowledgment is shown in the following table:

Acknowledgment		Meaning
Synchronous end acknowledgment	1	Job finished at the specified position: The tools are in the positions specified. The part program can be continued.
	3	Job canceled: The job is canceled, previously acknowledged tool position changes will be kept. The cancel command itself does not trigger any position acknowledgments or changes in the tool management.
	6	End acknowledgment for "Move tool" from the real magazine to a buffer (gripper, spindle) with reservation of the old location in the magazine for this tool. Same meaning as status 1.
	7	Repeat "prepare tool change" job: The tool change was informed of a new tool position in advance. The "prepare tool change" job is to be recalculated with this position. This is permissible only for preparation commands which have not yet been acknowledged.
	99	Total acknowledgment: Job complete, all positions reached. All the tools concerned are in the positions specified in the job. The part program can be continued. All target positions from the job have been reached.
Synchronous intermediate acknowledgment	105	Intermediate position for tool: The tools are moved from the source position specified in the job, or from the last acknowledged intermediate position, to the specified target position.
Communicating an asynchronous transfer	201	Communicate tool movement: The tool is moved from the source position to the specified target position. When moving from a location in a real magazine to an intermediate buffer location, the source location is reserved for the tool. Also observe MD20310, bit 24 (Page 353).
	204	Communicate magazine position: The magazine location is in the change/load/unload point of the specified target location.

Overview of evaluated table parameters

Acknowledgment status		1	3	6	7	99	105	201	204
New tool	Transfer step	x	-	x	-	-	x	x	x
	• from magazine	-	-	-	-	-	-	xx	xx
	• from location	-	-	-	-	-	-	xx	xx
	• to magazine	xx	-	xx	-	-	xx	xx	zz
	• to location	xx	-	xx	-	-	xx	xx	zz
Old tool	Transfer step	x	-	-	-	-	x	x	-
	• from magazine	-	-	-	-	-	-	xx	-
	• from location	-	-	-	-	-	-	xx	-
	• to magazine	xx	-	-	-	-	xx	xx	-
	• to location	xx	-	-	-	-	xx	xx	-

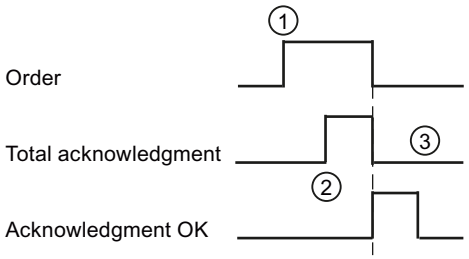
Legend:

- Date not relevant
x Number (1…n) of the transfer step from the transfer-step table
xx Magazine number, location number of the tool
zz Magazine number, location number of the load / unload or changing point

The status overview issues the following information:

- Depending on their meaning, the states 1, 6, 105, 201 and 204 should be combined in the acknowledgment table with the transfer steps to make useful acknowledgment steps.
- If status 1 is coded with both transfer step numbers = 0, this acknowledgment step serves as an end acknowledgment of the current status reached via intermediate acknowledgments.
- If a tool has been moved for a real magazine to a buffer (relocate, MVTOOL), acknowledgment with status 6 or total acknowledgment reserves the source location for this tool ($TC_MPP4 bit 1 and bit 2). The behavior is the same as for the removal of a tool from the magazine during tool change. With status 1, there is no reservation of the source location during relocation or MVTOOL.
- Statuses 3 and 7 need only to be coded once in the acknowledgment-step table, as no transfers steps are evaluated.
- Status 99 does not need to be coded, it is specified by the "total acknowledgment" bit.

Typical sequence of any job with total acknowledgment



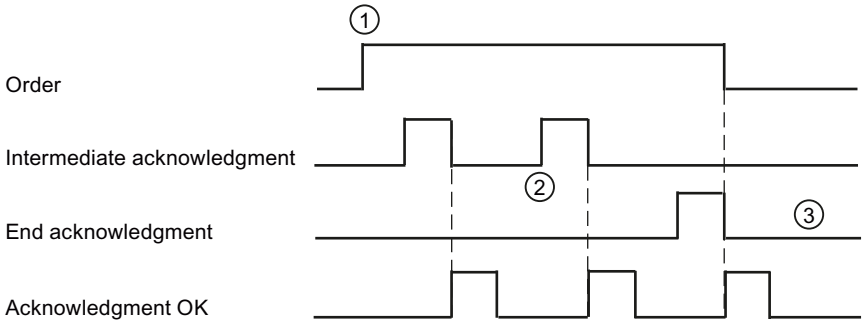
Signal description:

① The PLC user program recognizes from the 0/1 edge of the signal DB43xx.DBX0.0 (job) that a new job has been assigned by the tool management.

② The PLC user program sets the acknowledgment signal in DB42xx.DBX0.0 (total acknowledgment). With activation of the 0/1 edge, the PLC firmware starts transferring the acknowledgment to the tool management.

③ After successful transmission of the acknowledgment to the tool management, the PLC firmware sets a PLC cycle to 1 for the signal "acknowledgment OK" and at the same time the job signal and the acknowledgment bit are reset to 0.

Typical sequence of any job with total acknowledgment and end acknowledgment



Signal description:

① The PLC user program recognizes from the 0/1 edge of the signal DB43xx.DBX0.0 (job) that a new job has been assigned by the tool management.

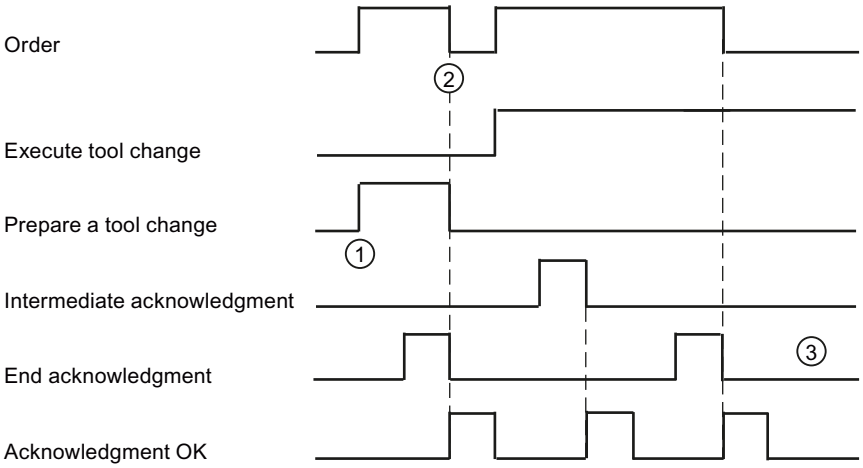
② The PLC user program acknowledges the transfer steps, configured in DB9900, DB9901 and DB9902, with acknowledgment status 105. The tool management updates the tool positions using the transfer steps of the acknowledgments.

③ The execution of the job is acknowledged by the PLC user program with acknowledgment status 1. After successful transmission of the acknowledgment to the tool management, the PLC firmware sets a PLC cycle to 1 for the signal "acknowledgment OK" and at the same time the job signal and the acknowledgment bit are reset to 0.

Sequence of the tool management with block splitting (machine data setting milling)

```
Txx M6 ;
```

Program L6 is called with M6 (basic setting)



Signal description:

① The PLC user program receives a new job. The job "prepare tool change" and the job "execute tool change" are issued one after the other. Txx and M206 have been programmed in separate NC blocks. Only the job "prepare tool change" is present in the interface in DB43xx. The job "execute tool change" is output only after the end acknowledgment for the job for tool preparation.

② Bit DB43xx.DBX0.0 (job) is reset with acknowledgment of the "prepare tool change" job. If the change command (M206) has already run through the NC main run, the new job is immediately output at the interface.

③ The "execute tool change" job is acknowledged as a normal job. The end acknowledgment OK is returned and the bit for the job is simultaneously reset.

The description of the job ("execute tool change" and "prepare tool change") is not reset. Byte 1 of DB43xx is not overwritten until the next job.

Note

MD20270, MD20310:

The response of the interface in DB43xx.DBB1 and of NC block processing is affected by the setting in MD20270: $MC_CUTTING_EDGE_DEFAULT and MD20310: $MC_TOOL_MANAGEMENT_MASK bit 5, 6, 7, and 8.

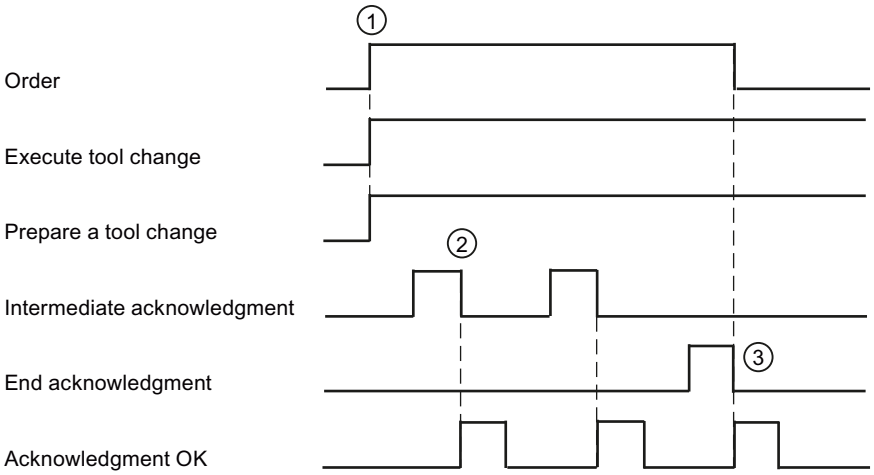
This sequence described here corresponds to the presetting of the machine data.

Sequence of the tool management without block splitting (machine data setting milling)

```
Txx Myy ;
```

Myy is the setting from MD22560: $MC_TOOL_CHANGE_M_CODE

This type of programming is not recommended!



Signal description:

① The PLC user program receives a new job. The job "prepare tool change" and the job "execute tool change" are issued simultaneously. Txx and M206 have been programmed in one NC block.

② Several intermediate steps are acknowledged. The state of the job remains unchanged. The positions of the tools are updated using the transfer steps of the acknowledgments from the tool management.

③ The end acknowledgment OK is returned and the bit for the job is simultaneously reset.

The description of the job ("execute tool change" and "prepare tool change") is not reset. Byte 1 of DB43xx is not overwritten until the next job.

Note

MD20310: $MC_TOOL_MANAGEMENT_MASK

The response of the interface in DB43xx.DBB1 is affected by the setting in MD20310 bit 10.

This sequence described here corresponds to the presetting of the machine data.

9.4.4 Configuring step tables

Configuring step tables

You can find the transfer-step tables (TM_CTS, DB9900 and TM_VTS, DB9901) and the acknowledgment-step table (TM_ACK, DB9902) in the Programming Tool under "Libraries" → "Special data blocks". The blocks are copied into a project with a double-click.

The structure of the data blocks is permanently fixed.

The blocks are not yet filled with the necessary data. The user must edit them in the Programming Tool, via the menu "View" → "Data block". The constant tables (TM_CTS, DB9900 and TM_ACK, DB9902) are configured by writing the initial data block values in the Programming Tool.

The initial data block values are loaded into the control, along with the PLC user program. Changed initial values are not activated until the PLC is restarted!

Configuring the transfer steps

Any changes to the tool and magazine positions must be communicated to the tool management by the PLC user program. A table of all mechanical single movements which should be acknowledged / communicated helps with this. For each tool transfer, the table contains the corresponding tool start and target position, or for positioning a magazine location at a transfer point (change, load, unload point), the magazine position and the name of the transfer point.

- The transfer steps 1 … 64 are permanently configured in TM_CTS (DB9900) and can be changed only by reloading.
- The transfer steps 101 … 164 in TM_VTS (DB9901) can be completely or partially overwritten by the PLC user program (e.g. by entering the current magazine location).

Coding for position from a job

In the constant transfer-step table, the locations of real magazines are not identified by their actual values (e. g. 1/14 for magazine 1 location 14) but with symbolic values (0/1) or (0/2). Otherwise the transfer-step table would be huge for large magazines.

These symbolic values have the following meaning:

Magazine / Location	Meaning
(0 / 1)	The source position of the only or new tool from the job should be used.
(0 / 2)	The target position of the old tool from the job should be used.
(0 / 3)	The target position of the only tool or new tool from the job should be used.

This symbolic notation form can only be used for synchronous intermediate and end acknowledgments, since asynchronous messages do not have a job as a reference.

Example: Constant transfer-step table

Transfer step	Address DB9900	Name	Start value	Comment
1	0.0	SrcMag_1	0	Source magazine number of the transfer step
	2.0	SrcPos_1	1	Source position number of the transfer step
	4.0	DstMag_1	0	Target magazine number of the transfer step
	6.0	DstPos_1	1	Target position number of the transfer step
2	8.0	SrcMag_2	0	Source magazine number of the transfer step
	10.0	SrcPos_2	1	Source position number of the transfer step
	12.0	DstMag_2	9998	Target magazine number of the transfer step
	14.0	DstPos_2	2	Target position number of the transfer step

Example as a complete step

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
1	0	1	0	1	Prepare Tool: The magazine is positioned at the changing point of the new tool.
2	0	1	9998	2	Tool change: Tool from magazine to gripper 1
3	9998	1	9998	3	Tool change: Tool from spindle to gripper 2
4	9998	2	9998	1	Tool change: Tool from gripper 1 to spindle
5	9998	3	0	2	Tool change: Tool from gripper 2 to magazine

Example: Variable transfer-step table

Transfer step	Address DB9901	Name	Start value	Comment
101	0.0	SrcMag_101	1	Source magazine number of the transfer step
	2.0	SrcPos_101	0	Source position number of the transfer step
	4.0	DstMag_101	9998	Target magazine number of the transfer step
	6.0	DstPos_101	1	Target position number of the transfer step
102	8.0	SrcMag_102	1	Source magazine number of the transfer step
	10.0	SrcPos_102	0	Source position number of the transfer step
	12.0	DstMag_102	9998	Target magazine number of the transfer step
	14.0	DstPos_102	2	Target position number of the transfer step

Example as a complete step

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
101	1	0	9998	1	"Prepare tool change": The magazine is positioned at the changing point. The source position must be entered by the PLC user program.
102	1	0	9998	2	"Prepare tool change": Tool from magazine to buffer. The source position must be entered by the PLC user program.

9.4.5 Configuring acknowledgment steps

Configuring the acknowledgment steps

The PLC 31 provides acknowledgment steps to acknowledge tool and magazine movements. These are activated via the corresponding bits on the user interface. The data of these acknowledgment steps (with the exception of special case acknowledgment step 9: total acknowledgment) are stored in acknowledgment step table TM_ACK (DB9902). The transfer steps (number of the transfer step from the transfer step table) for the old and new tool and an acknowledgment status are combined in one acknowledgment status.

It is important to include an acknowledgment step with the status 3 in this table, so that potential errors can be reset. The numbers for the transfer in this acknowledgment step are 0.

Special meaning of transfer step 0

Only the transfer steps assigned to the acknowledgment step are executed. If only one or no transfer step is assigned, no tool transfer will be carried out for the tool with transfer step = 0. The tool is not available or remains at its original location.

Example: Acknowledgment step table

Acknowledgment step	Address DB9902	Name	Start value	Comment
1	0.0	TsNewT_1	0	Transfer step number of the new tool
	1.0	TsOldT_1	0	Transfer step number of the old tool
	2.0	State_1	3	Status to NCK
2	4.0	TsNewT_2	1	Transfer step number of the new tool
	5.0	TsOldT_2	0	Transfer step number of the old tool
	6.0	State_2	1	Status to NCK

Example as a complete step

Acknowledgment step	Transfer step		Acknowledgment status	Comment
	New tool	Old tool		
1	0	0	3	"Cancel order" command.
2	2	0	1	Prepare Tool: The tool is taken from the magazine (changing point) and placed into gripper 1.

Acknowledgment is given by setting the corresponding bits in the user interface:

- DB40xx for loading/unloading / relocating or positioning the magazine
- DB42xx for "prepare tool change" and "execute tool change"

After processing, the acknowledgment bit is reset one PLC cycle by the PLC firmware.

In the same data block where the acknowledgments have taken place, for a PLC cycle the feedback message is output in bit 100.0 (acknowledgment OK), or as a static signal in bit 100.1 (acknowledgment error); the error status is output in byte 104 for an acknowledgment error, and in bytes DBB108 to DBB111 the acknowledgment bits last set are output. The user can use these bits to see which acknowledgment step triggered the error. If the PLC user program incorrectly set multiple acknowledgment bits, these are also entered one-to-one in the map. Resetting of the error status occurs when the user acknowledges the error in DB40xx.DBX9.0 or DB42xx.DBX9.0.

9.4.6 Adjust the PLC user program

Adjust the PLC user program

The control of mechanical processes, monitoring and prevention of potential collisions and the acknowledgment of tool position changes are the task of the PLC user program.

See also

In the PLC project on the Toolbox CD you can find examples of transfer steps and their acknowledgments, for a turning machine with a circular magazine and a milling machine with a chain magazine and dual gripper:

- Application example for milling machine (Page 393)
- Application example for turning machine (Page 382)

These blocks are function examples for the acknowledgment of different tool management jobs.

Acknowledging jobs

Many tool movements or tool management jobs can be directly acknowledged using total acknowledgment at bit 0.0 of the interface, without previous intermediate acknowledgment.

For example:

- Turns of a circular magazine
- Loading/unloading (only for systems without additional buffers such as handling systems, loaders, etc.)
- Changing manual tools
- Positioning a magazine

Rules for acknowledgment

If intermediate steps are useful, several rules must be observed during acknowledgment:

The PLC user program must ensure that all acknowledgments are transferred correctly to the tool management.

- Only one acknowledgment signal at a time may be sent to the tool management.
- Synchronous acknowledgments are permissible only for the pending job
- Only valid transfer-step numbers may be used (1 - 64, 100 - 164). For asynchronous messages, at least one transfer step must be input for status 201, for status 204 a transfer step for the new tool must be input.
- Coded positions in the transfer steps may only be used for synchronous acknowledgments and only with the values 0/1, 0/2 or 0/3.
- No illegal acknowledgment states may be used.
- Magazine positioning with a job may only receive a synchronous acknowledgment (end acknowledgment). Intermediate positions must be reported to the tool management with asynchronous messages.
- Acknowledgment signals must be reset by the PLC basic program. After an acknowledgment bit is set, the user interface may not be changed until the feedback message in DB41xx/DB43xx DBB100!
- Asynchronous messages with two transfer steps must be acknowledged on the tool change interface (DB42xx).

9.4.7 Information on magazine location

Overview

It is possible to read up to eight NC variables in one job using the existing NC services interface (DB1200).

Variable $TC_MPP2 with index 7: Location type

Parameter assignment:

Read variable from the NCK	Address	Signal	Valid values
Job	DB1200.DBX0.0	Start	0/1
	DB1200.DBX0.1	Write variable	0
	DB1200.DBB1	Number of variables	1 … 8
Parameter	DB120x.DBW1000	Variables index	7
	DB120x.DBW1002	Location number	1 … 31999
	DB120x.DBW1004	Magazine number	1 … 9999
Result	DB1200.DBX2000.0	Job completed	0/1
	DB1200.DBX2000.1	Error in job	0/1
	DB120x.DBX3000.0	Variable is valid	0/1

Read variable from the NCK	Address	Signal	Valid values
	DB120x.DBB3001	Access result	0/3/5/10
	DB120x.DBW3004	Data from NCK variable	n

Legend:

n > 0:	Location type for virtual location
n = 0:	"match all" (buffer)
n = 9999:	undefined (no virtual location)

Variable I$TC_MPP4 with index 8: Location state

Parameter assignment:

Read variable from the NCK	Address	Signal	Valid values
Job	DB1200.DBX0.0	Start	0/1
	DB1200.DBX0.1	Write variable	0
	DB1200.DBB1	Number of variables	1 … 8
Parameter	DB120x.DBW1000	Variables index	8
	DB120x.DBW1002	Location number	1 … 31999
	DB120x.DBW1004	Magazine number	1 … 9999
Result	DB1200.DBX2000.0	Job completed	0/1
	DB1200.DBX2000.1	Error in job	0/1
	DB120x.DBX3000.0	Variable is valid	0/1
	DB120x.DBB3001	Access result	0/3/5/10
	DB120x.DBW3004	Data from NCK variable	n

Legend:

n = 1	locked
n = 2	free (<> occupied)
n = 4	reserved for tool in buffer
n = 8	reserved for tool to be loaded
n = 16	occupied in left half location
n = 32	occupied in right half location
n = 64	occupied in upper half location
n = 128	occupied in lower half location

Variable $TC_MPP6 with index 9: T number of the tool at this location

Parameter assignment:

Read variable from the NCK	Address	Signal	Valid values
Job	DB1200.DBX0.0	Start	0/1
	DB1200.DBX0.1	Write variable	0
	DB1200.DBB1	Number of variables	1 … 8
Parameter	DB120x.DBW1000	Variables index	9
	DB120x.DBW1002	Location number	1 … 31999
	DB120x.DBW1004	Magazine number	1 … 9999
Result	DB1200.DBX2000.0	Job completed	0/1
	DB1200.DBX2000.1	Error in job	0/1
	DB120x.DBX3000.0	Variable is valid	0/1
	DB120x.DBB3001	Access result	0/3/5/10
	DB120x.DBW3004	Data from NCK variable	n
	n = T number of the tool on the parameterized location		

Error (for all variables)

In case of error, DB120x.DBX3000.0 = 0 and an entry is made in the access result:

Values in DB120x.DBB3001	
0	No error
3	Illegal access to object
5	Invalid address
10	Object does not exist

9.4.8 PI service: TMMVTL

Function

With PI service TMMVTL, it is possible to initiate a job to relocate a tool from the PLC. After an error-free "PI Start," the tool management performs an empty location search in the target magazine for the tool on the defined source location. Subsequently the PLC receives a job for relocating the tool (user interface DB41xx.DBB0).

Parameter assignment

Starting program instance services in the NCK area:

PI service	Address	Signal	Valid values
Job	DB1200.DBX4000.0	Start	0/1
	DB1200.DBB4001	PI index	5
Parameter	DB1200.DBW4004	Tool number (internal T number)	1 ... 31999
	DB1200.DBW4006	Source location number	1 ... 31999
	DB1200.DBW4008	Source magazine number	1 ... 31999
	DB1200.DBW4010	Target location number	-1
	DB1200.DBW4012	Target magazine number	1 ... 32000
Result	DB1200.DBX5000.0	Job completed	0/1
	DB1200.DBX5000.1	Error in job	0/1

The tool can be specified either via a T number or by means of the location and magazine numbers. The value -1 is entered at unused specification points.

With the parameter, target location number =-1, a search is made in the complete magazine for an empty location for the tool according to the search strategy that has been selected. If a target location is specified, then a check is made as to whether the location with the specified target location number is free and suitable for the particular tool.

For a target magazine number = -1, a search is made in a buffer for the tool corresponding to the assignment obtained from $TC_MDP2.

Application

Examples:

- When using buffers to return the tool (for example Toolboy and/or shifter), an explicit empty location search in the magazine may be needed during the asynchronous return transport. In this case the PLC does not have to note the original location, the PI service searches for a suitable location.
- A tool is to be moved from a background magazine to the front magazine.

9.5 Example: Loading/unloading

Programming

For loading, tools are placed directly in the magazine or the spindle; for unloading, they are removed directly from the magazine. Normally, a single acknowledgment from the operator or the PLC user program (tool holder is closed again) is sufficient as a message that the process is complete. There is no need to configure a transfer step. The total acknowledgment can be set in DB40xx.DBX0.0.

Acknowledgment to tool management:

Acknowledgment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
xxx	DB4000.DBX0.0	--	--	(99)

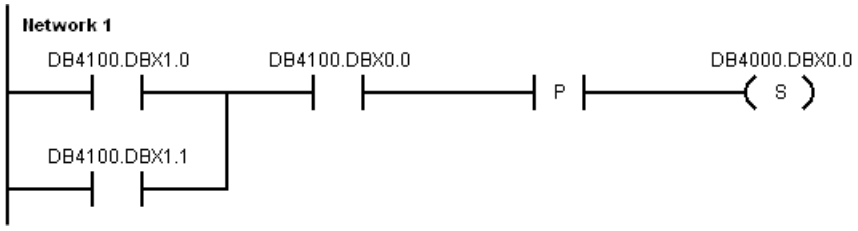


Figure 9-2 Programming in PLC user program

Loading procedures using handling systems or transporting the tool from the spindle into the magazine can be performed using further asynchronous messages.

Different loading sequences are described below:

Loading via the spindle with preselection of a magazine location

You can create a new tool directly on the chosen free magazine location or, using the "load" dialog, move a tool from the tool list, which is not located in the magazine, to a magazine location.

1. A job from the tool management is always output at the interface for the loading point. You must acknowledge this job.
2. Then move this tool, which is not yet positioned in the magazine, into the spindle with Txx M6 or via an asynchronous transfer.
3. Then manually place the tool into the spindle and deposit it in the magazine with T0 M6.

The procedure is **always** possible irrespective of whether manual tools are allowed. If manual tools are permitted, this sequence of operations must always be observed when loading via the spindle.

The tools can be oversized or fixed-location coded.

Loading via the spindle with preselection of a magazine location

You can create a new tool directly on the spindle or, using the "load" dialog, move a tool from the tool list, which is not located in the magazine, to the spindle.

1. A job from the tool management is always output at the interface for the loading point. You must acknowledge this job.
2. Then manually place the tool into the spindle and deposit it in the magazine with T0 M6. A free location is selected by the tool management in which the tool can be deposited.

This sequence is only possible if the manual tools function is not set
MD22562: $MC_TOOL_CHANGE_ERROR_MODE Bit 1=0 (default setting).

The tools can be oversized or fixed-location coded.

Loading directly into the magazine

Position the relevant magazine location next to the loading position. You can create a new tool directly on the chosen free magazine location or, using the "load" dialog, move a tool from the tool list, which is not located in the magazine, to the selected magazine location.

1. A job from the tool management is always output at the interface for the loading point. You must acknowledge this job.
2. Now place the tool in the magazine.

This sequence is not subject to any restrictions or constraints.

9.6 Example: Change manual tools

Programming

In MD22562: $MC_TOOL_CHANGE_ERROR_MODE Bit 1=1, additional tools without magazine allocation have been selected by the NC part program. The selected tool must be inserted in the machine manually and removed again manually after machining ("manual tools").

The operator must ensure that the data block for the tool on the spindle is in the NCK, or that he/she puts the appropriate tool onto the spindle for the data block stored in the NCK.

Note

The responsibility is on the user to comply with the safety regulations via the PLC user program.

The PLC user program is informed with DB43xx.DBX1.5 and DBX1.6 whether a manual tool is involved in a tool change job. With alarm 17212: "Channel %1, Manual tool %2, Duplo No. %3, Load to toolholder %4" or Alarm 17214. "Remove manual tool from spindle/tool holder," the operator is requested to execute the tool change.

The alarms are reset after a tool change by the acknowledgment from the PLC.

Starting position 1

The manual tool in the spindle should be exchanged for another manual tool

Job from tool management to the PLC user program (tool change):

DB4300.DBX0.0, DBX1.2, DBX1.5 and DBX1.6 ("Prepare tool change")

DB43xx.DBW6	Source magazine number	9999
DB43xx.DBW8	Source location number	1
DB43xx.DBW10	Target magazine number	9999
DB43xx.DBW12	Target location number	1

Acknowledgment "Prepare tool change":

Acknowledgment step	Acknowledgment bit	Transfer step for the new tool	Transfer step for the old tool	Status
xxx	DB4200.DBXx.x	0	0	1

Displayed on the interface:

DB4300.DBX0.0 /1.1, DBX1.5 and DBX1.6 ("Execute")

The job remains otherwise unchanged, the tools are still at the starting positions.

An intermediate step can be inserted for acknowledging the removal of the old tool:

Synchronous acknowledgment: The tool is no longer in the spindle:

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
6 DB9900.DBW40	9998	1	9999	1	Tool removed from the spindle

Acknowledgment step	Acknowledgment bit	Transfer step for the new tool	Transfer step for the old tool	Status
xxx	DB4200.DBXx.x	0	6	105

After intermediate acknowledgment of the empty spindle and insertion of the new tool in the spindle, the tool change is terminated with a total acknowledgment:

Acknowledgment step	Acknowledgment bit	Transfer step for the new tool	Transfer step for the old tool	Status
	DB4200.DBX0.0			(99)

Starting position 2

A manual tool is in the spindle and is to be replaced by a tool from the magazine

Job from tool management to the PLC user program (tool change):

DB4300.DBX0.0, DBX1.2 and DBX1.6 ("Prepare tool change")

DB43xx.DBW6	Source magazine number	1
DB43xx.DBW8	Source location number	6
DB43xx.DBW10	Target magazine number	9999
DB43xx.DBW12	Target location number	1

Acknowledgment "Prepare tool change":

Acknowledgment step	Acknowledgment bit	Transfer step for the new tool	Transfer step for the old tool	Status
xxx	DB4200.DBXx.x	0	0	1

Displayed on the interface:

DB4300.DBX0.0 /1.1 and 1.6 ("Execute tool change")

The job remains otherwise unchanged, the tools are still at the starting positions.

Synchronous acknowledgment: The old tool is no longer in the spindle:

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
6 DB9900.DBW40	9998	1	9999	1	Asynchronous message, unload the tool from the spindle

Acknowledgment step	Acknowledgment bit	Transfer step for the new tool	Transfer step for the old tool	Status
xxx	DB4200.DBXx.x	0	6	105

The spindle is now empty, the old tool is outside the magazine.

Next step: Synchronous acknowledgment, new tool to gripper 1

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
3 DB9900.DBW16	0	1	9998	2	New tool to gripper 1

Acknowledgment step	Acknowledgment bit	Transfer step for the new tool	Transfer step for the old tool	Status
xxx	DB4200.DBXx.x	3	0	105

The job is unchanged.

Next step: Synchronous acknowledgment, new tool from gripper 1 to spindle:

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
4 DB9900.DBW24	9998	2	9998	1	New tool from gripper 1 to spindle

Acknowledgment step	Acknowledgment bit	Transfer step for the new tool	Transfer step for the old tool	Status
xxx	DB4200.DBXx.x	4	0	105

This completes the tool movement.

End acknowledgment:

Acknowledgment step	Acknowledgment bit	Transfer step for the new tool	Transfer step for the old tool	Status
xxx	DB4200.DBXx.x	0	0	1

The step New tool from gripper 1 to spindle can be omitted and replaced by a total acknowledgment. This also informs the tool management that all tools are in their target positions.

Acknowledg-ment step	Acknowledgment bit	Transfer step for the new tool	Transfer step for the old tool	Status
	DB4200.DBX0.0			(99)

9.7 Application example for turning machine

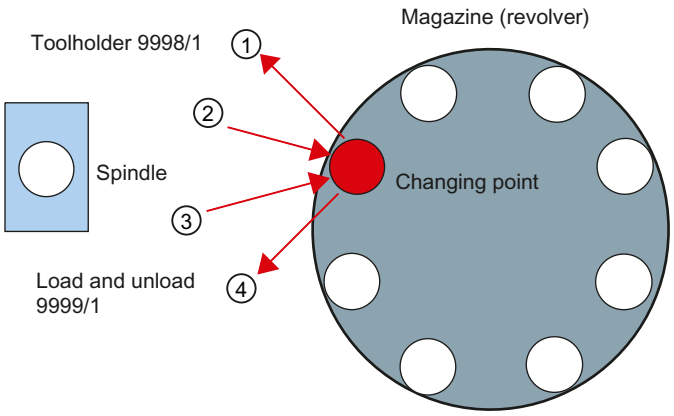
9.7.1 Example: Turning machine with revolver magazine (MAGKONF_MPF)

Example file

You can find the program for configuring the magazine in the Toolbox.

The program can be read into the control and should be adjusted for the specific machine concerned.

Configuration



1 ... 4 Transfer steps

Figure 9-3 Turning machine with revolver magazine

Description of the program

To start with, all the old magazine definitions and tools are deleted. As the program sequence continues, all magazines and buffers are newly created and assigned by writing the magazine parameters.

Search strategies for tools and magazine locations can be defined in N70.

For revolver magazines it is advisable to define all locations as fixed-location coded.
For magazine type 3 N320 $TC_MAP3[NUM_MAG] = 81 (Bit6 = 1) is set.

The chain magazine locations are set up from N430 to N500. Magazine location type = 0 means that the magazine location can be loaded with tools of different location types.

The buffers are set up from N520 onwards.

From N920, the buffers are assigned to the spindle / tool holder and the magazine.

After the magazine configuration program has finished running, restart the NC using (NCK reset).

See also

You will find a precise description of the parameters used in the Tool Management Function Manual of SINUMERIK 840D sl.

Example MAG_CONF_MPF

1. Plant configuration:
   - 1 revolver magazine with 8 locations (can be set up in N40)
   - 1 loading point
   - 3 buffer locations (can be set up in N50, assignments from N540)
2. Part program:

```
;MAG_CONF_MPF
N10 def int NUM_MAG,MAG_TYPE, LOCATIONS,
PLACE, NUM_BUFFER, NUM_LOAD, PLACE_SEARCH
;
N20 NUM_MAG = 1                                ;Number of the magazine
N30 MAG_TYPE = 3                               ;Magazine type (1: chain,
                                               3: revolver, 5: box magazine)
N40 LOCATIONS = 8                              ;Number of magazine locations
N50 NUM_BUFFER = 1                             ;Number of buffers (spindle,
                                               gripper)
N60 NUM_LOAD = 1                               ; Number of loading points
N70 PLACE_SEARCH = 257                         ; Type of search strategy
;= 257 Bit13=0 no exchange of the old tool on the location of the new tool
;Setting for pickup magazine
;=12289 Bit13=1 exchange old tool on the location of the new tool
;Setting for chain magazine
N80;
N90;
;Check parameters
N100 STOPRE
N110 if ((NUM_MAG==0)or(LOCATIONS==0))
N120 Err1:STOPRE
N130 MSG("Wrong Parameter --> Cancel")
N140 G04 F4
N150 STOPRE
N160 M0
N170 GOTOB Err1
N180 endif
N190; Magazine configuration
N200;
N210;
```

```
N220; Delete old data when magazine 1 is set up
N230 if NUM_MAG ==1
N240 $TC_MAP1[0]=0                              ; Delete magazine
N250 $TC_DP1[0,0]=0                             ; Delete Tools
N260 STOPRE
N270 endif
; Configuration
;
N280 $TC_MAMP2= PLACE_SEARCH                    ; Type of search strategy
;
; Magazine
; Set up magazine
N290 $TC_MAP1[NUM_MAG]= MAG_TYPE
N300 $TC_MAP2[NUM_MAG]="MAGAZINE"<<NUM_MAG
N310 if MAG_TYPE == 3
N320 $TC_MAP3[NUM_MAG]=81                       ; Magazine status, all locations
                                                fixed-location coded for revolver
                                                magazine
N330 else
N340 $TC_MAP3[NUM_MAG]=17                       ; Magazine status
N350 endif
N360 $TC_MAP4[NUM_MAG]=-1
N370 $TC_MAP5[NUM_MAG]=-1
N380 $TC_MAP6[NUM_MAG]=1                        ; Number of lines in magazine
N390 $TC_MAP8[NUM_MAG]=0
N400 $TC_MAP9[NUM_MAG]=0
N410 $TC_MAP7[NUM_MAG]=LOCATIONS                ; Number of magazine locations
N420 $TC_MAP10[NUM_MAG]=PLACE_SEARCH
;
; Magazine locations
N430 for PLACE=1 to LOCATIONS
N440 STOPRE
N450 $TC_MPP1[NUM_MAG,PLACE]=1                  ; Location type
N460 $TC_MPP2[NUM_MAG,PLACE]=0                  ; Location type, 0 compatible with
                                                every tool location type
N470 $TC_MPP3[NUM_MAG,PLACE]=1                  ; Consider adjacent location On (Off
                                                would be 0)
N480 $TC_MPP4[NUM_MAG,PLACE]=2                  ; Location status
N490 $TC_MPP5[NUM_MAG,PLACE]=PLACE              ; Location type index
N500 endfor
N510 STOPRE
;
N520; Definition of buffer magazine (always number 9998)
;
N530 $TC_MAP1[9998]=7                           ; Magazine type 7: Buffer
```

```
N540 $TC_MAP2[9998]="BUFFER"<<NUM_MAG
N550 $TC_MAP3[9998]=17                          ; Magazine status
N560 $TC_MAP6[9998]=1                           ; Number of lines
N570 $TC_MAP7[9998]=NUM_BUFFER                  ; Number of locations
;
; Locations in the buffer
;Spindle
N580 $TC_MPP1[9998,1]=2                         ; Location type (here spindle)
N590 $TC_MPP2[9998,1]=0                         ; Location type (here always 0)
N600 $TC_MPP3[9998,1]=0                         ; Consider adjacent location Off
N610 $TC_MPP4[9998,1]=2                         ; Location status
N620 $TC_MPP5[9998,1]=1                         ; Location type index
;
N630; Gripper
N640 FOR PLACE=2 to NUM_BUFFER
N650 STOPRE
N660 $TC_MPP1[9998,PLACE]=3                     ;(here gripper)
N670 $TC_MPP2[9998,PLACE]=0                     ;(here always 0)
N680 $TC_MPP3[9998,PLACE]=0                     ; Consider adjacent location Off
N690 $TC_MPP4[9998,PLACE]=2                     ; Location status
N700 $TC_MPP5[9998,PLACE]=PLACE                 ; Location type index
N710 endfor
N720 STOPRE
;
;
N730; Definition of loading magazine (always number 9999)
;
N740 $TC_MAP1[9999]=9                           ; Magazine type 9: Loading magazine
N750 $TC_MAP2[9999]="LOADING
MAGAZINE"<<NUM_MAG
N760 $TC_MAP3[9999]=17                          ; Magazine status
N770 $TC_MAP4[9999]=-1
N780 $TC_MAP5[9999]=-1
N790 $TC_MAP6[9999]=1                           ; Number of lines
N800 $TC_MAP7[9999]=NUM_LOAD                    ; Number of locations
N810 STOPRE;
;
N820; Loading magazine locations
;
N830 for PLACE=1 to NUM_LOAD
N840 STOPRE
N850 $TC_MPP1[9999,PLACE]=7                     ; Location type Loading point
N860 $TC_MPP2[9999,PLACE]=0                     ; Location type (here always 0)
N870 $TC_MPP3[9999,PLACE]=0                     ; Consider adjacent location Off
```

```
N880 $TC_MPP4[9999,PLACE]=2                    ; Location status
N890 $TC_MPP5[9999,PLACE]=PLACE                ; Location type index
N900 endfor
N910 STOPRE
;
;
N920; Offsets (clearances) ; Clearances to magazine
;
; Buffer
N930 for PLACE=1 to NUM_BUFFER
N940 $TC_MDP2[1,PLACE]=0
N950 endfor
N960 STOPRE
;
;Loading points
N970 for PLACE=1 to NUM_LOAD
N980 stopre
N990 $TC_MDP1[1,PLACE]=0
N1000 endfor

N1010 M30                                      ; End
```

Display in operating software

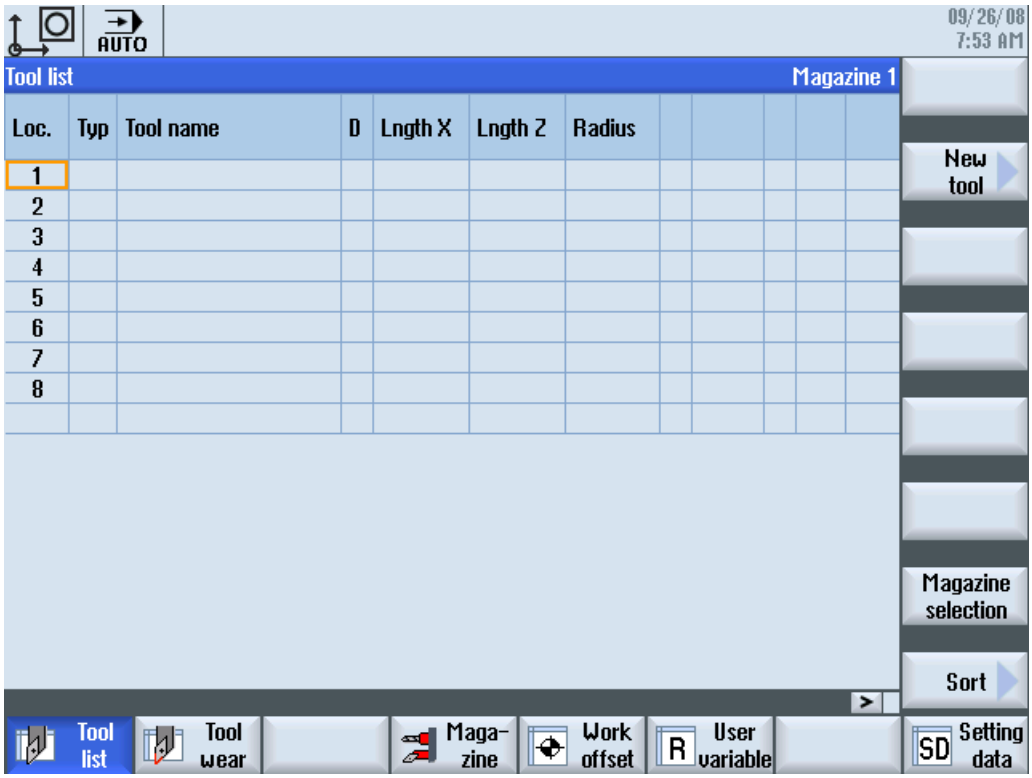


Figure 9-4 Turning machine tool list

9.7.2 Example: Acknowledgment steps (turning machine)

Acknowledgment steps

Generally, the mechanical sequences on a turning machine are simpler than those for a milling machine. In the configuration described in the previous chapter without additional buffer, tool changes can be acknowledged after the mechanical movements without transfer steps. The PLC user program must detect incoming jobs in the job interfaces and execute the mechanical movements.

Total acknowledgment to tool management:

Acknowledgment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
--	DB4000.DBX0.0	--	--	(99)
--	DB4200.DBX0.0	--	--	(99)

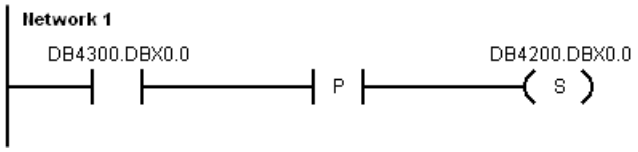


Figure 9-5 Programming in PLC user program

An asynchronous message can be used for magazine movements without a job from the tool management.

Acknowledgment to tool management:

Acknowledgment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
xxx	DB4000.DBXx.x	101	0	204

Transfer step	Source		Objective		Comment
	Magazine	Location	Magazine	Location	
101 DB9901.DBW0	1	**n**	9998	1	The variable location in Magazine 1 is positioned at the changing point to the spindle.

n is the actual location number (n ≠ 0) to be entered by the PLC user program in the variable transfer table.

9.7.3 Example: Tool change cycle for turning machine

Transfer variables

With MD10717: $MN_T_NO_FCT_CYCLE_NAME (name of cycle to be executed instead of the T function, e.g. "TCHANGE"), you can set that a cycle is run when the T command is called. The cycle is run through at every T call, irrespective of whether a new tool or the already active tool is called. The position of the revolver can be set on the programmed tool in this cycle (POSM). This is necessary when the revolver has been positioned manually after a tool selection and the tool has not been deselected. In this case, the NC does not issue a new job for tool change at the interface.

Sample program

Prerequisite is that the tool management has been informed of each magazine movement. The example has been created for machine data with the turning technology default setting. The "Spare tools for tool management" option is not active.

Transfer variables of the T replacement cycle:

Tag	Description
$SC_T	T number of the tool (numerical)
$SC_T_Prog	Bool variable that shows whether a T word is available in $C_T.
$C_TS	Identifier for tool (string)
$C_TS_Prog	Bool variable that shows whether an identifier is available in $C_TS.
$C_TE	Address extension of the T word
$C_D	Programmed D number
$C_D_Prog	Bool variable that shows whether an offset number is available in $C_D.
$C_DL	Programmed additive/setup offset
$C_DL_Prog	Bool variable that shows whether an offset number is available in $C_DL.

In the following example, a job for positioning the magazine with POSM is output at the interface. During the magazine positioning, the block processing of the NC must be controlled by the PLC user program. In most cases, it is useful to set the load or feed disable during this time. As soon as the position setpoint specified in the job is reached (asynchronous message of the magazine position), the job is terminated with total acknowledgment.

```
PROC L6 SAVE SBLOF DISPLOF
  IF $C_T_PROG==1                    ; T is numeric
    IF $C_T==0                       ; T=0
      T=0
    ENDIF
    IF $C_T>0
      IF $C_T<=$TC_MAP7[1]           ; Does the magazine location exist?
      POSM($C_T)                     ; Position magazine
      ENDIF
      T=$C_T                         ; T programming of location number
    ENDIF
  ENDIF

  IF $C_TS_PROG==1                   ; T is an identifier
    _TNO_NEW=GETT($C_TS,1)           ; Querying of T number
    IF _TNO_NEW>0                    ; Does the T number exist?
      _TL_NEW=$A_MYMLN[_TNO_NEW]     ; Querying of location number
    ENDIF
    IF _TL_NEW>0                     ; Is the tool in the magazine?
      POSM(_TL_NEW)                  ; Position magazine
    ENDIF
    T=$C_TS                          ; T programming without address extension
  ENDIF
M17
```

See also

The function is available irrespective of tool management and is described in full in: Function Manual, Basic Functions, "Mode Group, Channel, Program Operation, Reset Behavior (K1)."

9.7.4 Example: Turning machine with counterspindle

Magazine configuration

In the magazine configuration the magazine is assigned twice as many locations as actually exist, e.g. in the case of revolver with 12 locations, 24 locations are set up. Location 1-12 for the main spindle, location 13-24 for the counterspindle.

The user program positions the magazine in such a way that, for example, the same position is approached for location 1 and location 13. So each real magazine location corresponds to a virtual magazine location for the main spindle and a virtual magazine location for the counterspindle.

9.7.5 Example: Test for empty buffer

Procedure

Read the T number of a tool in gripper 1 and 2:

1. In the PLC user program, enter the parameters in DB1200.
2. In DB1200.DBX0.0, set the start for reading the location states.

Once the order has been successfully executed, the results are entered starting from DB1200.DBB3000:

Read variable from the NCK	Address	Signal	Values
Parameters	DB1200.DBW1000	Variables index	9
	DB1200.DBW1002	Location number	2
	DB1200.DBW1004	Magazine number	9998
	DB1201.DBW1000	Variables index	9
	DB1201.DBW1002	Location number	3
	DB1201.DBW1004	Magazine number	9998
Order	DB1200.DBX0.1	Write variable	0
	DB1200.DBX0.2	PI service	0
	DB1200.DBB1	Number of variables	2
	DB1200.DBX0.0	Start	→ 1
Result	DB1200.DBB2000.0	Order completed	1
	DB1200.DBX2000.1	Error in order	0
	DB1200.DBX3000.0	Variable is valid	1
	DB1200.DBB3001	Access result	0

Read variable from the NCK	Address	Signal	Values
	DB1200.DBW3004	Data from NCK variable	n
	DB1201.DBX3000.0	Variable is valid	1
	DB1201.DBB3001	Access result	n
	DB1201.DBW3004	Data from NCK variable	0

9.7.6 Example: Transporting a tool from a buffer into the magazine

Procedure

A tool is to be moved from a buffer (for example, of a gripper) into the magazine. The empty location search for the tool from gripper 1 (magazine 9998, location 2) is executed with PI Service TMMVTL and an order to relocate the tool is generated.

In the PLC user program, enter the parameters in DB1200, and in DB1200.DBX4000.0, set the start to read the PI service.

Start PI Services in the NCK area	Address	Signal	Values
Parameter	DB1200.DBW4004	Tool number	0
	DB1200.DBW4006	Source location number	2
	DB1200.DBW4008	Source magazine number	9998
	DB1201.DBW4010	Target location number	-1
	DB1201.DBW4012	Target magazine number	1
Job	DB1200.DBB4001	PI index	5
	DB1200.DBX4000.0	Start	→ 1
Result	DB1200.DBB5000.0	Job completed	1
	DB1200.DBX5000.1	Error in job	0

DB410x.DBX0.0 delivers an order to relocate the tool from the buffer. The target location in magazine 1 is in DB4100.DBW12. With it, the PLC user program can execute the necessary sequence.

9.7.7 Example: Repeat "Prepare tool change" order

Sequence: Repeat the command

For a milling machine with toolboy and shifter, the asynchronous transfer message and order can cross:

- The toolboy was acknowledged to the tool management as the target position for the old tool with the end acknowledgment.
- The part program is continued with the new tool and after a very short time needs the previous (old) tool once more.
- The tool management generates the next preparation order for the tool change with the toolboy as the source position for what will then be the new tool.
- At the same time the PLC user program transfered the tool from the toolboy into the shifter.
- The PLC user progamm communicates the tool movement from the toolboy to the shifter asynchronously and starts onward transport to the magazine.
- In the next cycle the AWP recognizes the new order to relocate the tool from the toolboy to the spindle.

But the tool is no longer in the toolboy! The PLC user program must detect such a condition (monitoring: does toolboy or shifter contain a tool?). It can now cancel the return of the tool into the magazine. A repeat order (Status 7) can then be requested from the tool management:

- Meanwhile, the tool management has received the message that the tool is in the shifter.
- It generates a new preparation order with the source position shifter for the new tool.

NOTICE
The acknowledgment to repeat the order "Prepare tool change" may only be issued **before** the end acknowledgment of the "Prepare tool change" order!

Acknowledgment to tool management:

Acknowledgment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
xxx	DB4000.DBXx.x	0	0	7

9.8 Application example for milling machine

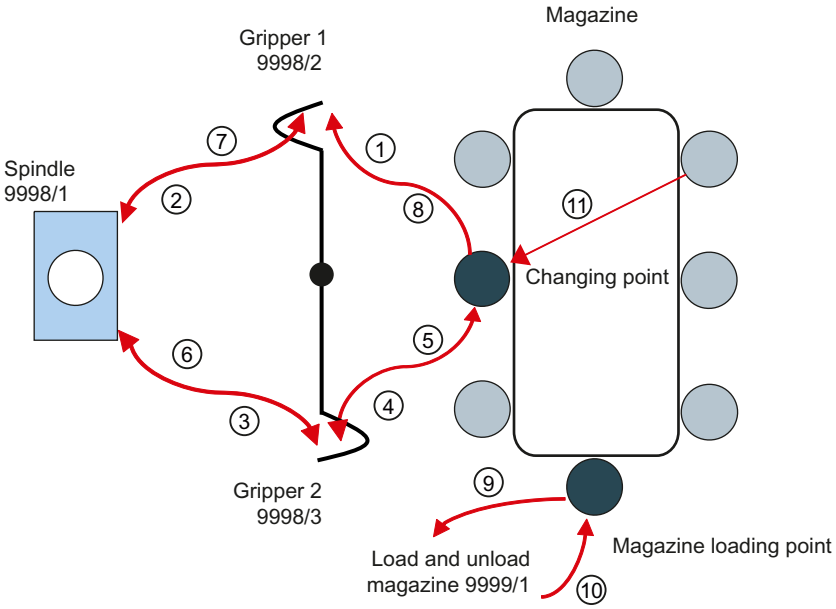
9.8.1 Example: Milling machine with chain magazine and dual gripper (MAGKONF_EXAMPLE_MPF)

Example file

You can find the program for configuring the magazine in the Toolbox.

The program can be read into the control and should be adjusted for the specific machine concerned.

Configuration



1 ... 11 Transfer steps

Figure 9-6 Milling machine with chain magazine

Description of the buffers and loading points:

Magazine	Location	Meaning
1	xx	Real magazine (chain, plate, box), position xx
9998	1	Spindle
9998	2	Gripper 1
9998	3	Gripper 2
9999	1	Magazine loading point

Description of the program

To start with, all the old magazine definitions and tools are deleted. As the program sequence continues, all magazines and buffers are newly created and assigned by writing the magazine parameters.

Search strategies for tools and magazine locations can be selected in N70. Here it is determined whether the tool can be placed directly in the new tool location, when changing out of the spindle. This makes the tool change possible in a single mechanical sequence, so shorter changeover times can be achieved. This sequence is not possible for pickup magazines.

The chain magazine locations are set up from N430 to N500. Magazine location type = 0 means that the magazine location can be loaded with tools of different location types.

The buffers are set up from N520 onwards. If additional buffers are available (toolboy, shifter...), the number should be changed in N50.

Additionally available loading points should be dealt with in the same way in N60.

From N920, the buffers are assigned to the spindle / tool holder and the magazine.

After the magazine configuration program has finished running, restart the NC (NCK reset).

See also

Further information:

- The configuration of a disk-type magazine with fixed-location coding is performed as for the configuration of a revolver magazine: Example: Turning machine with revolver magazine (MAGKONF_MPF) (Page 382)
- You will find a precise description of the parameters used in the Tool Management Function Manual of SINUMERIK 840D sl.

Example MAG_CONF_MPF

1. Plant configuration:
   - 1 chain magazine with 8 locations (can be set up in N40)
   - 1 loading point
   - 3 buffer locations (can be set up in N50, assignments from N540)
2. Part program:

```
;MAG_CONF_MPF
N10 def int NUM_MAG,MAG_TYPE, LOCATIONS,
PLACE, NUM_BUFFER, NUM_LOAD, PLACE_SEARCH
;
N20 NUM_MAG = 1                          ;Number of the magazine
N30 MAG_TYPE = 1                         ;Magazine type (1: chain,
                                         3: revolver, 5: box magazine)
N40 LOCATIONS = 8                        ;Number of magazine locations
N50 NUM_BUFFER = 3                       ;Number of buffers (spindle,
                                         gripper)
```

```
N60 NUM_LOAD = 1                                ; Number of loading points
N70 PLACE_SEARCH = 12289                        ; Type of search strategy
;= 257 Bit13=0 no exchange of the old tool on the location of the new tool
;Setting for pickup magazine
;=12289 Bit13=1 exchange old tool on the location of the new tool
;Setting for chain magazine
N80;
N90;
;Check parameters
N100 STOPRE
N110 if ((NUM_MAG==0)or(LOCATIONS==0))
N120 Err1:STOPRE
N130 MSG("Wrong Parameter --> Cancel")
N140 G04 F4
N150 STOPRE
N160 M0
N170 GOTOB Err1
N180 endif
N190; Magazine configuration
N200;
N210;
N220; Delete old data when magazine 1 is set up
N230 if NUM_MAG ==1
N240 $TC_MAP1[0]=0                              ; Delete magazine
N250 $TC_DP1[0,0]=0                             ; Delete Tools
N260 STOPRE
N270 endif
; Configuration
;
N280 $TC_MAMP2= PLACE_SEARCH                    ; Type of search strategy
;
; Magazine
; Set up magazine
N290 $TC_MAP1[NUM_MAG]= MAG_TYPE
N300 $TC_MAP2[NUM_MAG]="MAGAZINE"<<NUM_MAG
N310 if MAG_TYPE == 3
N320 $TC_MAP3[NUM_MAG]=81                       ; Magazine status, all locations
                                                fixed-location coded for revolver
                                                magazine
N330 else
N340 $TC_MAP3[NUM_MAG]=17                       ; Magazine status
N350 endif
N360 $TC_MAP4[NUM_MAG]=-1
N370 $TC_MAP5[NUM_MAG]=-1
```

```
N380 $TC_MAP6[NUM_MAG]=1                          ; Number of lines in magazine
N390 $TC_MAP8[NUM_MAG]=0
N400 $TC_MAP9[NUM_MAG]=0
N410 $TC_MAP7[NUM_MAG]=LOCATIONS                  ; Number of magazine locations
N420 $TC_MAP10[NUM_MAG]=PLACE_SEARCH
;
; Magazine locations
N430 for PLACE=1 to LOCATIONS
N440 STOPRE
N450 $TC_MPP1[NUM_MAG,PLACE]=1                    ; Location type
N460 $TC_MPP2[NUM_MAG,PLACE]=0                    ; Location type, 0 compatible with
                                                  every tool location type
N470 $TC_MPP3[NUM_MAG,PLACE]=1                    ; Consider adjacent location On (Off
                                                  would be 0)
N480 $TC_MPP4[NUM_MAG,PLACE]=2                    ; Location status
N490 $TC_MPP5[NUM_MAG,PLACE]=PLACE                ; Location type index
N500 endfor
N510 STOPRE
;
N520; Definition of buffer magazine (always number 9998)
;
N530 $TC_MAP1[9998]=7                             ; Magazine type 7: Buffer
N540 $TC_MAP2[9998]="BUFFER"<<NUM_MAG
N550 $TC_MAP3[9998]=17                            ; Magazine status
N560 $TC_MAP6[9998]=1                             ; Number of lines
N570 $TC_MAP7[9998]=NUM_BUFFER                    ; Number of locations
;
; Locations in the buffer
;Spindle
N580 $TC_MPP1[9998,1]=2                           ; Location type (here spindle)
N590 $TC_MPP2[9998,1]=0                           ; Location type (here always 0)
N600 $TC_MPP3[9998,1]=0                           ; Consider adjacent location Off
N610 $TC_MPP4[9998,1]=2                           ; Location status
N620 $TC_MPP5[9998,1]=1                           ; Location type index
;
N630; Gripper
N640 FOR PLACE=2 to NUM_BUFFER
N650 STOPRE
N660 $TC_MPP1[9998,PLACE]=3                       ;(here gripper)
N670 $TC_MPP2[9998,PLACE]=0                       ;(here always 0)
N680 $TC_MPP3[9998,PLACE]=0                       ; Consider adjacent location Off
N690 $TC_MPP4[9998,PLACE]=2                       ; Location status
N700 $TC_MPP5[9998,PLACE]=PLACE                   ; Location type index
N710 endfor
```

```
N720 STOPRE
;
;
N730; Definition of loading magazine (always number 9999)
;
N740 $TC_MAP1[9999]=9                          ; Magazine type 9: Loading magazine
N750 $TC_MAP2[9999]="LOADING
MAGAZINE"<<NUM_MAG
N760 $TC_MAP3[9999]=17                         ; Magazine status
N770 $TC_MAP4[9999]=-1
N780 $TC_MAP5[9999]=-1
N790 $TC_MAP6[9999]=1                          ; Number of lines
N800 $TC_MAP7[9999]=NUM_LOAD                   ; Number of locations
N810 STOPRE;
;
N820; Loading magazine locations
;
N830 for PLACE=1 to NUM_LOAD
N840 STOPRE
N850 $TC_MPP1[9999,PLACE]=7                    ; Location type Loading point
N860 $TC_MPP2[9999,PLACE]=0                    ; Location type (here always 0)
N870 $TC_MPP3[9999,PLACE]=0                    ; Consider adjacent location Off
N880 $TC_MPP4[9999,PLACE]=2                    ; Location status
N890 $TC_MPP5[9999,PLACE]=PLACE                ; Location type index
N900 endfor
N910 STOPRE
;
;
N920; Offsets (clearances) ; Clearances to magazine
;
; Buffer
N930 for PLACE=1 to NUM_BUFFER
N940 $TC_MDP2[1,PLACE]=0
N950 endfor
N960 STOPRE
;
;Loading points
N970 for PLACE=1 to NUM_LOAD
N980 stopre
N990 $TC_MDP1[1,PLACE]=0
N1000 endfor

N1010 M30                                      ; End
```

Display in operating software

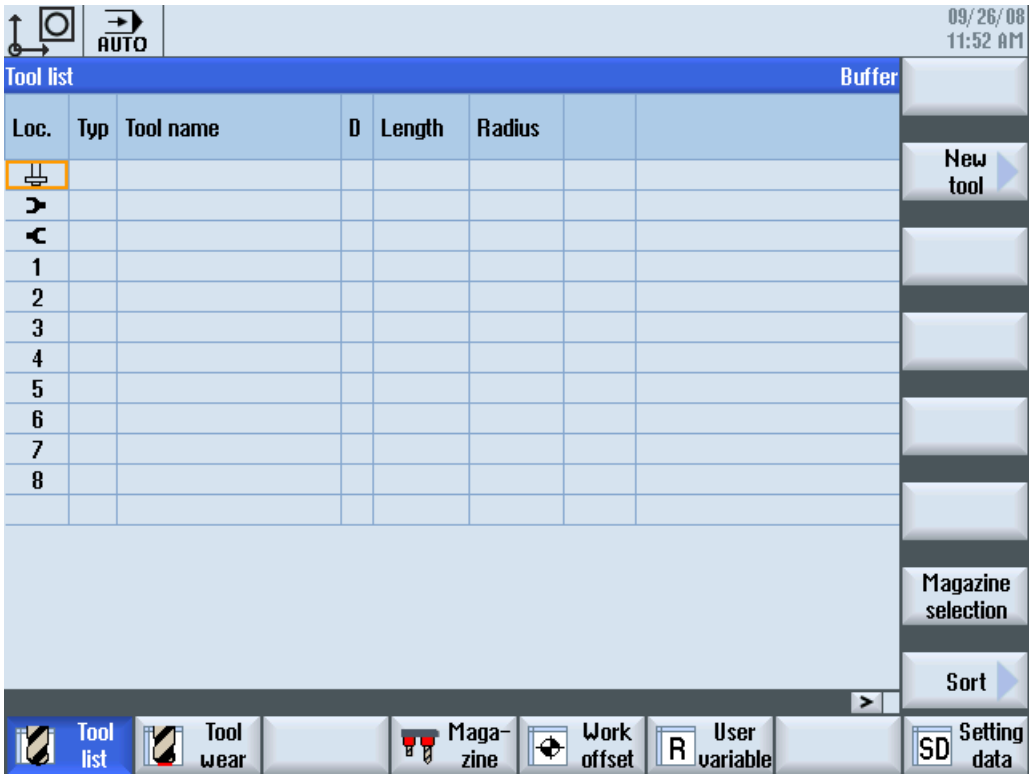


Figure 9-7 Tool list Milling machine

9.8.2 Flow chart: Tool change

Tool change program sequence (PLC)

The sequence described here describes the change between magazine and spindle. The changing of manual tools as well as loading and unloading are not taken into account.

Both procedures are described in Chapter:

The machine data default setting is selected such that the job for "Prepare tool change" is triggered with the T command on the interface:

```
N10 T = "Tool name" M6
```

The block preprocessing is not interrupted. M6 is used to start the tool change subprogram (L6) at the same time. As soon as the job for "Prepare tool change" has been acknowledged and in the tool change subprogram the M code for tool change output has been reached, the "Execute tool change" job is output to the interface (block splitting).

A tool change command (M206) must always precede a prepare tool change command. A change command without a previous "Prepare tool change" job will not initiate a job from the tool management.

You will find the expression 1:1 change in the program sequence. This means that the tool change will be carried out in a single cycle. The tool from the spindle (old tool) will be set down at the magazine location of the new tool. No additional magazine positioning is required. In this case, the target location of the old tool in the tool management job is the same as the source location of the new tool (DB43xx.DBW6 and DBW8 are equal to DBW18 and DBW20).

A 1:1 change is not possible for:

- Tools with different location types
- Different tool sizes
- Fixed-location tools

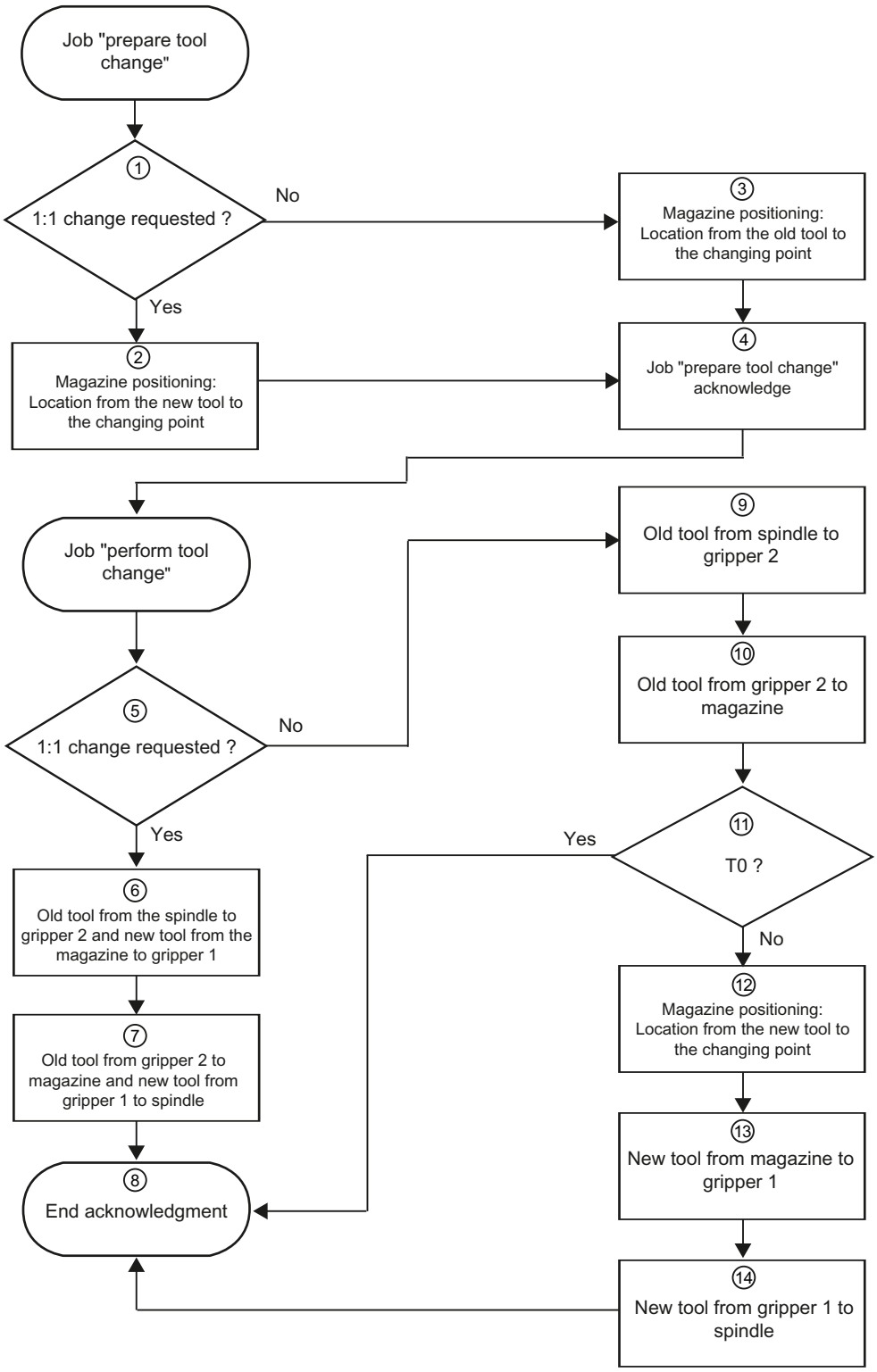
By programming T0 in the NC program, a tool change is initiated without a new tool. Only the spindle tool is transported to the magazine (empty the spindle).

① ... ⑭ refer to the steps in the flow chart below.

You can see the different acknowledgment options from the steps. Job-related acknowledgments and asynchronous messages are used.

Please take the transfer steps used from the table in Chapter Example: Acknowledgment steps (milling machine) (Page 408)

"Tool change" flow chart



1: 1 change: The old tool is deposited in the location of the new tool.

Figure 9-8 Flow Diagram

Description of the Procedure

- NC program:

  T command or T command with simultaneous tool change call (M6)

- Interface signals:

  Job from TM: DB43xx.DBX0.0 (job bit) and

  DB43xx.DBB1 (command bits): Prepare change (DB43xx.DBX1.2)

- Magazine and location number of the tools to be moved:

  DB43xx.DBW6 to DBW20: Source location of the new tool, target location of the old tool

Step 1: 1:1 Change requested

Request: Source location of the new tool == Target location of the old tool, Normal case:

There is a tool in the tool holder and a new tool is being requested. Both tools have the same location type and the same tool size in the magazine list, the tools are not fixed-location coded.

The old tool is set down on the magazine location of the new tool, in a direct exchange (1:1 change). If there is no old tool in the tool holder (DB4300.DBX1.4), the same sequence takes place. In this case, the magazine positions the new tool to the changing point.

→ continue with Step 2

Request: Source location of new tool >< Target location of old tool, Special cases:

The TM is informed by DB43xx.DBX1.3 whether the active tool from the tool holder should be removed (T0). In this case, the magazine positions the storage location of the old tool (that is currently in the spindle) to the changing point.

The magazine location for the old tool is also positioned to the changing point if the old tool cannot be stored on the new tool location (1:1 change not possible). The cause could be different location types or tool sizes, or that a fixed-location coded tool is involved. In this case, the tool change is carried out in two steps. First the old tool in the magazine is set down and then the new tool is transported to the spindle.

→ continue with Step 3

Step 2: Position magazine, location from new tool to changing point.

← Previous step: Step 1

For example, the magazine movement can be executed via an NC rotary axis controlled by a PLC. The movement should be notified to the tool management. The magazine position is thus updated at the user interface, in the views of the tool and magazine lists.

The target position is read from DB4300.DBW8 (location number for new tool – source) and written to DB9901.DBW2 (transfer step 101). If there is magazine coincidence at the target position, the step is acknowledged asynchronously:

Acknowledg-ment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
4	DB4200.DBX0.4	101	0	204

Acknowledgment to TM:

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
101 DB9901.DBW0	1	n	9998	1	The variable location in magazine 1 is positioned at the changing point to the spindle.

n: is here the actual location number (n ≠ 0) to be entered by the PLC user program in the variable transfer table.

→ continue with Step 4

Step 3: Positioning the magazine; location from the old tool to the changing point.

← Previous step: Step 1

Same as Step 2, however, the magazine target location is obtained from DB4300.DBW20 (location number for old tool - target).

Acknowledgment to TM:

Acknowledgment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
4	DB4200.DBX0.4	101	0	204

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
101 DB9901.DBW0	1	n	9998	1	The variable location in magazine 1 is positioned at the changing point to the spindle.

n: is here the actual location number (n ≠ 0) to be entered by the PLC user program in the variable transfer table.

→ continue with Step 4

Step 4: Acknowledge prepare tool change job

← Previous step: Step 2 or Step 3

With this, the preparation for the tool change is complete for many systems.

Acknowledgment to TM:

Acknowledg-ment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
1	DB4200.DBX0.1	0	0	1

→ continue with Step 5

- NC program:

  M206 initiates the job for execute tool change
- Interface signals:

  Job from TM: DB43xx.DBX0.0 (job bit)

  DB43xx.DBB1 (command bits): Execute change (DB43xx.DBX1.1)
- Magazine and location number of the tools to be moved:

  DB43xx.DBW6 to DBW20: Source location of the new tool, target location of the old tool

Step 5: 1:1 change requested

← Previous step: No previous step, entry point in the sequence for "Execute tool change" job

Similarly to Step 1, it is checked whether a direct change or a change in two steps is to be executed:

- 1:1 change possible: **→ continue with Step 6**
- 1:1 change not possible: **→ continue with Step 9**

Step 6: Old tool from the spindle to gripper 2 and new tool from the magazine to gripper 1

← Previous step: Step 5

The PLC program controls the machine functions with which the gripper movements, tool clamping, etc. are performed. As soon as the mechanical movements are completed and acknowledged in the PLC user program, the tool movements are acknowledged to the tool management.

Acknowledgment to TM:

Acknowledg-ment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
5	DB4200.DBX0.5	1	2	105

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
1 DB9900.DBW0	0	1	9998	2	New tool from magazine to gripper 1 Step 6 or Step 13
2 DB9900.DBW8	9998	1	9998	3	Old tool from spindle to gripper 2 Step 6 or Step 9

→ continue with Step 7.

Step 7: Old tool from gripper 2 to magazine and new tool from gripper 1 to spindle

← Previous step: Step 6

The PLC program controls the machine functions for the gripper movements, tool clamping, etc. As soon as the mechanical movements are completed and acknowledged in the PLC user program, the tool movements are acknowledged to the tool management.

Acknowledgment to tool management:

Acknowledgment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
7	DB4200.DBX0.7	3	4	105

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
3 DB9900.DBW16	9998	2	9998	1	New tool from gripper 1 to spindle Step 7 or Step 14
4 DB9900.DBW24	9998	3	0	2	Old tool from gripper 2 to magazine Step 7 or Step 10

→ continue with Step 8

Step 8: End acknowledgment

← Previous step: Step 7 or Step 14

End acknowledgment takes place with tool changeover in the initial setting, or in a state where the machine can continue machining. Here it is possible that there are still mechanical movements to be executed before the tool change can be completed.

Acknowledgment to tool management:

Acknowledgment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
1	DB4200.DBX0.1	0	0	1

→ continue with Step 9

Step 9: Old tool from spindle to gripper 2

← Previous step: Step 5

The PLC program controls the machine functions with which the gripper movements, tool clamping, etc. are performed. As soon as the mechanical movements are completed and acknowledged in the PLC user program, the tool movements are acknowledged to the tool management.

Acknowledgment to tool management:

Acknowledgment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
8	DB4200.DBX1.0	0	2	105

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
2 DB9900.DBW8	9998	1	9998	3	Old tool from spindle to gripper 2 Step 6 or Step 9

→ continue with Step 10

Step 10: Old tool from gripper 2 to magazine

← Previous step: Step 9

The PLC program controls the machine functions with which the gripper movements, tool clamping, etc. are performed. As soon as the mechanical movements are completed and acknowledged in the PLC user program, the tool movements are acknowledged to the tool management.

Acknowledgment to tool management:

Acknowledgment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
9	DB4200.DBX1.1	0	4	105

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
4 DB9900.DBW24	9998	3	0	2	Old tool from gripper 2 to magazine Step 7 or Step 10

→ continue with Step 11

Step 11: T0 ?

← Previous step: Step 10

Request: Is T0 set in the tool change job?

DB43xx.DBX1.3

If only the tool holder should be emptied, the tool change can be completed:

→ continue with Step 8

Do you want to place a new tool in the tool holder?

→ continue with Step 12

Step 12: Magazine position location from new tool to changing point

← Previous step: Step 11

Sequence as in Step 2

Acknowledgment to tool management:

Acknowledgment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
4	DB4200.DBX0.4	101	0	204

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
101 DB9901.DBW0	1	n	9998	1	The variable location in magazine 1 is positioned at the changing point to the spindle.

n: is here the actual location number (n ≠ 0) to be entered by the PLC user program in the variable transfer table.

→ continue with Step 13

Step 13: New tool from magazine to gripper 1

← Previous step: Step 12

The PLC program controls the machine functions used to execute gripper movements, tool clamping etc.

Acknowledgment to tool management:

Acknowledgment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
10	DB4200.DBX1.2	1	0	105

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
1 DB9900.DBW0	0	1	9998	2	New tool from magazine to gripper 1 Step 6 or Step 13

→ continue with Step 14

Step 14: New tool from gripper 1 to spindle

← Previous step: Step 13

The PLC user program controls the machine functions with which the gripper movements, tool clamping, etc. are performed. As soon as the mechanical movements are completed and acknowledged in the PLC user program, the tool movements are acknowledged to the tool management.

The tool change can be ended:

Acknowledgment to tool management:

Acknowledgment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
11	DB4200.DBX1.3	3	0	105

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
3 DB9900.DBW16	9998	2	9998	1	New tool from gripper 1 to spindle Step 7 or Step 14

→ continue with Step 8

9.8.3 Example: Acknowledgment steps (milling machine)

Constant transfer step table

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
1 DB9900.DBW0	0	1	9998	2	New tool from magazine to gripper 1 Step ⑥ or ⑬
2 DB9900.DBW8	9998		9998	3	Step ⑥ or ⑨
3 DB9900.DBW16	9998		9998	1	Step ⑦ or ⑩
4 DB9900.DBW24	9998	3	0	2	Step ⑦ or ⑭
5 DB9900.DBW32	0	2	9998	1	Storage location of the old tool
6 DB9900.DBW40	0	1	9998	1	Magazine location with the new tool to the changing point Step ② or ⑫
7 DB9900.DBW48	--	--	--	--	

Variable transfer step table

Transfer step	from		to		Comment
	Magazine	Location	Magazine	Location	
101 DB9901.DBW0	1	n	9998	1	The variable location in Magazine 1 is positioned at the changing point to the spindle.
102 DB9901.DBW8	--	--	--	--	

n : here is the actual location number (n ≠ 0) to be entered by the PLC user program in the variable transfer table.

Acknowledgment step table

Acknowledgment step	Transfer step		Acknowledg-ment status	Comment
	Old tool	New tool		
1 DB9902.DBW0	0	0	1	End acknowledgment, Step ④ and ⑧
2 DB9902.DBW4	0	0	3	Cancel order
3 DB9902.DBW8	0	0	105	Intermediate acknowledgment for subsequent order, Step ④
4 DB9902.DBW12	101	0	204	Variable magazine location to changing point
5 DB9902.DBW16	1	2	105	Intermediate acknowledgment Step ⑥
6 DB9902.DBW20	0	5	105	Intermediate acknowledgment Step ③
7 DB9902.DBW24	3	4	105	Intermediate acknowledgment Step ⑦
8 DB9902.DBW28	0	2	105	Intermediate acknowledgment Step ⑨
9 DB9902.DBW32	0	4	105	Intermediate acknowledgment Step ⑩
10 DB9902.DBW36	1	0	105	Intermediate acknowledgment Step ⑬
11 DB9902.DBW40	3	0	105	Intermediate acknowledgment Step ⑭
12 DB9902.DBW44	--	--	--	

Note: The step numbers ① ... ⑭ refer to the flow chart in Chapter Flow chart: Tool change (Page 399)

9.8.4 Example: Tool change cycle for milling machine

Sample program

```
PROC L6 SAVE DISPLOF
;-------------------------------------------------------------
; Example of a tool change cycle for machine manufacturers
;-------------------------------------------------------------
DEF INT _WZ_IN_SP,_WZ_VOR
DEF REAL _SPP= ...                        ; spindle position
;
  IF(NOT $P_SEARCH)                       ; if no block search
    _WZ_IN_SP=$TC_MPP6[9998,1]            ; tool in spindle
    GETSELT(_WZ_VOR)                      ; previously selected tool
;
    IF(_WZ_IN_SP<>_WZ_VOR)                ; if another tool
      SPOS=_SPP                           ; position spindle
      G0                                  ; approach tool change position:
      G75 Z=0
      WAITS(1)
    ENDIF
  ELSE
  ENDIF
;
; Load tool: Tool management and PLC
M206
M17
;-------------------------------------------------------------
; END
;-------------------------------------------------------------
```

Easy Archive 10

Overview

In line with the data segregation in data classes, an archive can be created separately for each data area and for each data class.

The data class "System" is an exception. This data is set permanently and becomes effective at the first installation or a default initialization. For this reason, data backup of system data is unnecessary, as this data class contains no data created during commissioning or during the machine run-time.

A system archive delivered by Siemens can, for example, contain a new NCK version or an integrated HMI version or also a cycle hotfix.

NOTICE
Protection of the system data All system data and data contained in the "System" data class in the HMI, NCK, PLC and drive areas may not be affected. System data cannot be changed through operation or through the writing of a part program or subroutine or cycle or through reading in an archive. When you load user data with a USB FlashDrive, the data volume must not be larger than 4 MB!

Identification of archives

Every archive contains the following identifiers:

- Data class: M, I, U
- Type of control: 828D TE or 828D ME
- Time stamp: Date and time when the archive was created
- Version designation: Software version with which this archive was created
- Serial number of the system CompactFlash card

These markings allow filters to be set when reading in, determining which archive may be read in to which control variant, depending on the data class and the software version.

Compatibility of the data

SINUMERIK 802D sl PLC archives can be transferred to a SINUMERIK 828D control variant using the Programming Tool.

10.1 Series start-up and archiving

When should you back up the start-up data?

The following times are recommended for performing a data backup:

- After start-up
- After changing machine-specific settings
- After the replacement of a hardware component
- Before a software upgrade

Backing up and restoring data

To back up and restore data, select in the "Start-up" operating area:

- "Save data" softkey for an internal data backup of the entire memory
- "Series startup" softkey
  - Create series start-up
  - Read in the series start-up

Data areas

The following data areas are backed up during series start-up:

Components	Data
HMI data	• Applications • Display machine data • Settings • Online help • PLC projects • Logs • Texts • Dictionaries • Version data • Templates • Cycle storage
PLC data	• User program • MAIN (main program) • DB (data blocks)

Components	Data
NC data	• Compile cycles • Standard and user cycles • Definitions and macros • Machine data • Setting data • Option data • Global (GUD) and local (LUD) user data • Tool and magazine data • Protection zone data • R parameters • Work offsets • Compensation data • Part programs • Subprograms • Workpieces

Note

The series start-up archive is saved as an archive, taking into account the data classes (file type ARD). The drive data is saved as binary data which cannot be edited.

Memory areas for archives

Archives can be stored in the following memory areas:

- On the user CompactFlash card: any directory
- On the CompactFlash card at:

  `/user/sinumerik/data/archive`

  or:

  `/oem/sinumerik/data/archive`
- On a USB FlashDrive

NOTICE

USB FlashDrives

USB flash drives are not suitable as persistent memory media.

10.2 How to create and read in a series start-up archive

Overview

Control components can be saved individually or jointly. Creating a separate series start-up archive for each component is recommended, so that the files can be read in again independently of each other.

Requirement

The access level "Service" is required.

Create "Series start-up"

Procedure:

1. Select the "Start-up" operating area.
2. Press the menu forward key and then the "Series start-up" softkey.

   The "Series Start-up" window opens.
3. Select the option "Create series start-up" with the <SELECT> key and confirm with "OK".

   The "Create Series Start-up" window opens:

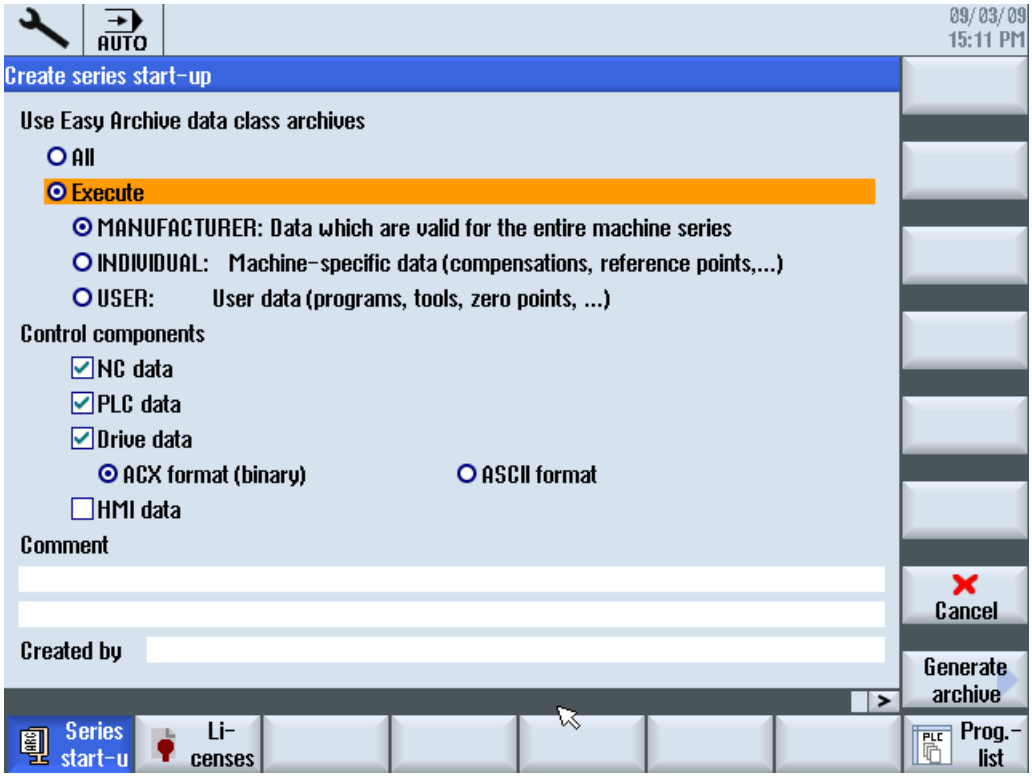


Figure 10-1 Create archive

4. Choose whether the data classes should be "ignored" or "considered" using the <SELECT> key:
   - Select "ignored" to archive all data belonging to the control component.
   - Select "considered" to write only the data in the data classes selected under "Selection" in the archive.
5. Mark the control component(s) for the archive.
6. Make use of the option to enter a comment and the creator of the archive.
7. Press the "Generate archive" softkey.

   The "Generate Archive: Select Storage Location" window opens.
8. Select a directory or press the "New directory" softkey to create a new subdirectory.

   The "New Directory" window opens.
9. Enter a name and confirm with "OK."

   The directory is created subordinate to the selected folder.

   The "Generate Archive: Name" window opens.
10. Enter a name and confirm with "OK."

An archive file is created in the chosen directory.

Read in "Series start-up"

Requirement: The access level "User" is necessary for reading in an archive.

Procedure:

1. Select the "Start-up" operating area.
2. Press the menu forward key and then the "Series start-up" softkey.

   The "Series Start-up" window opens.
3. Select the option "Read in series start-up" with the <SELECT> key and confirm with "OK".

   The "Select Start-up Archive" window opens.
4. Select the archive and confirm with "OK".
5. To read in the archive, confirm the prompt with "OK".

   The "Read In Archive" window opens and a progress message box appears for the read-in process.
6. Press the "Cancel" softkey to cancel the read-in process.

10.3 This is how you back up just the machine data that have changed

Requirement

The access level "Service" is required.

Difference backup only for machine data that have changed

Using the general MD11210 $MN_UPLOAD_MD_CHANGES_ONLY, you can set whether all machine data should be backed up in the archive, or only the machine data that differ from the default setting:

MD11210 $MN_UPLOAD_MD_CHANGES_ONLY	
Backing up just machine data that have changed	
= FFH (default setting)	
Bit 0	
. . .	
Bit 7	

NOTICE
Effect of MD11210
MD11210 $MN_UPLOAD_MD_CHANGES_ONLY is only effective for type *.arc archives.

Generating an archive

Procedure:

1. Insert the storage medium (CompactFlash Card or USB FlashDrive) in the appropriate slot at the front of the control.
2. To generate a type *.arc archive press the keys <Ctrl> + <Alt> + C.

Result:

An archive is generated on the storage medium:

The archive name has the following structure: CompleteArchive_Date_Time.arc

Example:

CompleteArchive2010-08-11_08-36-15.arc stands for an archive generated on 11.08.2010 at 8:36:15.

10.4 Example: Data archiving "Easy Archive" (use case)

Easy Archive

With "Easy Archive", the SINUMERIK 828D has a fundamentally new procedure for data archiving. This procedure is tailored exactly to the needs of series machine manufacture. "Easy Archive" is based on a strict separation between SINUMERIK system software, customized OEM data (machine data, manufacturer cycles) and operator data (part programs, tool offsets). There is a further separation for customized data; for data that is the same for all machines of that type, and for data that has been adjusted for an individual machine.

This will be clarified with an example:

User example

A machine manufacturer builds a vertical machining center in series. The customized data is created on a prototype machine. This prototype machine's customized data set will later be ported to all series machines (cloned). After porting the data, however, individual settings are made on each individual machine.

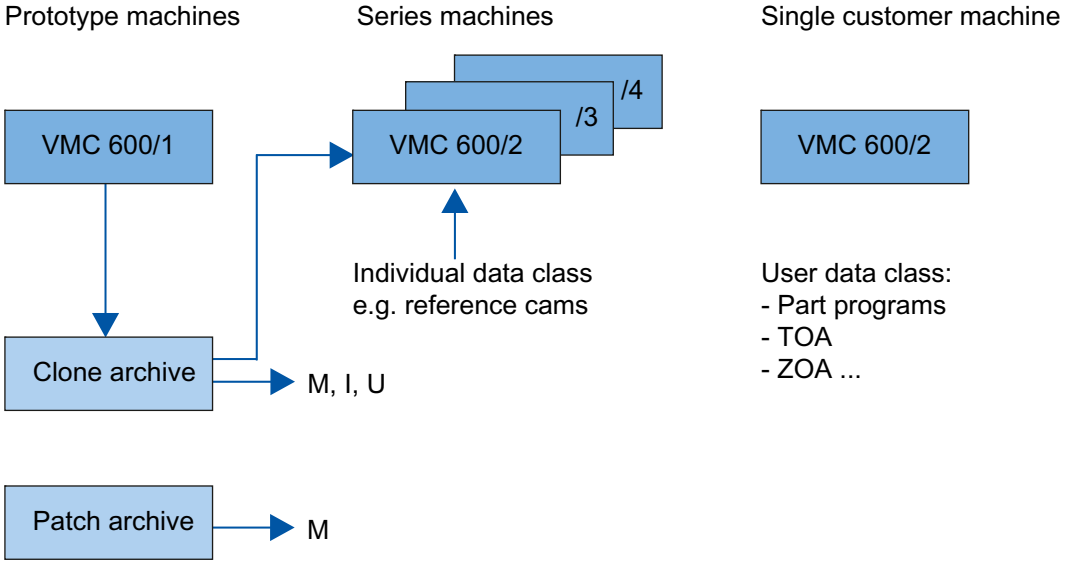


Figure 10-2 User example

For example, reference cams and ball bearing spindles are measured and entered as individual customized data. If an error occurs at the end user, the error is reproduced and solved on the prototype machine. If a complete archive from the prototype machine is now transferred to the affected machine, the individual customized data of this machine is overwritten by the individual customized data of the prototype machine.

For the SINUMERIK 828D, the machine manufacturer's customized data, to which no individual change has been made, may be archived separately. If this archive is transferred to the affected machine, it is ensured that the individual customized data and the end user data are retained. So the machine manufacturer's update process is simplified considerably.

Advantages

The advantage of the "Easy Archive" is that archive creation occurs directly at the SINUMERIK 828D user interface. So no separate PC is needed for the archive.

By separating the system data from the customized and user data, the SINUMERIK 828D system updates can be completely carried out by the OEMs, without changes to the customized data. A system update can be carried out by the end user themselves in a short time.

10.5 Parameterizing the serial interface

Data exchange

Data exchange via the serial interface is possible from the following operating areas:

- "Program manager" operating area
- "Start-up" operating area → "System data" softkey

Press the following softkeys to set the interface parameters:



Note

If the interface is already assigned, e.g. because a modem is connected, data exchange via the serial interface is not possible and a message is output.

Description of parameters

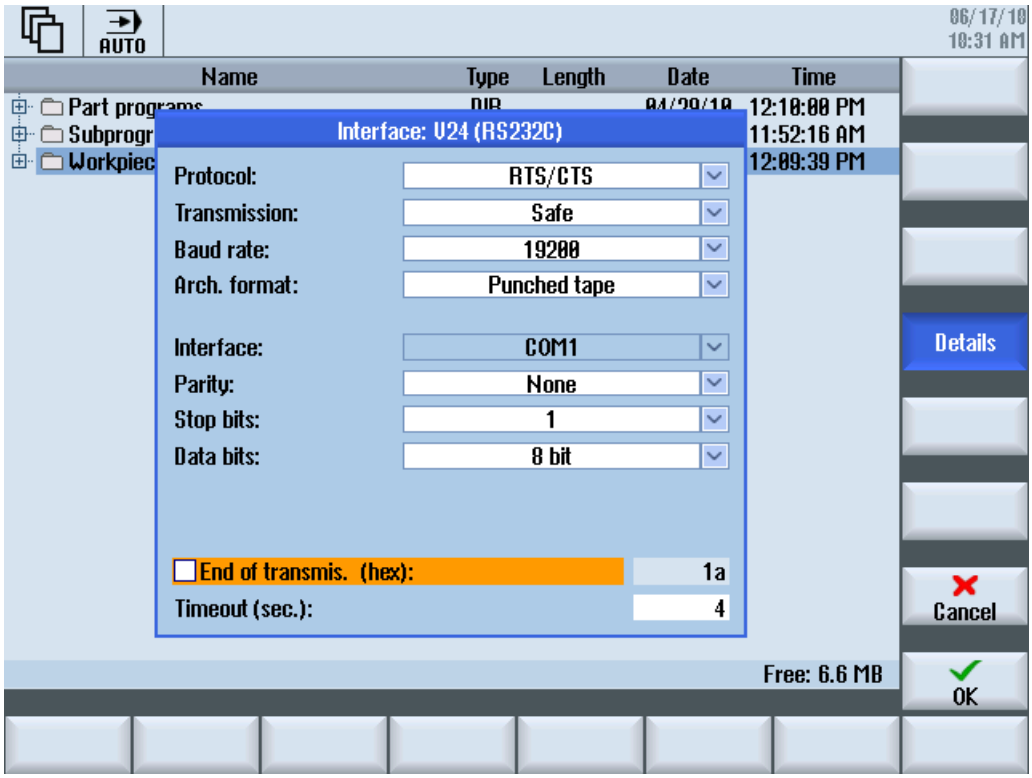


Figure 10-3 Setting parameters

Parameter	Permissible values	
Protocol:	**RTS/CTS (default setting)** Xon/Xoff	
Transmission:	Normal **backed up (default setting)**	
Baud rate:	**19200 (default setting)** 300 ... 19200 max.	
Archive format:	Punched tape Binary format (PC format) → only with RTS/CTS protocol	
Interface:	COM1	
Parity:	**None (default setting)** Even Odd	
Stop bits:	**1 (default setting)** 2	
Data bits:	5 bits 6 bits 7 bits **8 bits (default setting)**	
Xon (only for setting Xon/Xoff)		11
Xoff (only for setting Xon/Xoff)		13
End of transmission (hex)		1a
Time monitoring (sec)		4

Deactivating V24

The following machine data should be set to deactivate the interface for data exchange:

MD51233 $MNS_ENABLE_GSM_MODEM = 1

The softkeys are then no longer displayed.

Lists

A

A.1 Table: Language code in file names

Supported languages

Standard languages:

Language	Abbreviation in file name
Chinese (simplified)	chs
Chinese (traditional)	cht
German	deu
English	eng
Spanish	esp
French	fra
Italian	Ita
Korean	kor
Portuguese (Brazil)	ptb

Other languages:

Language	Abbreviation in file name
Czech	csy
Danish	dan
Finnish	fin
Hungarian	hun
Japanese	jpn
Dutch	nld
Polish	plk
Rumanian	rom
Russian	rus
Slovakian	sky
Swedish	sve
Turkish	trk

A.2 List of the alarm number ranges

Number ranges of alarms

Number range	Description	Source ID	Source URL
000 000 – 009 999	General alarms	0	/NCK
010 000 – 019 999	Channel alarms		
020 000 – 029 999	Axis/spindle alarms		
030 000 – 039 999	General functional alarms		
040 000 – 059 999	- reserved -		
060 000 – 064 999	Siemens cycle alarms		
065 000 – 069 999	User cycle alarms		
070 000 – 079 999	Compile cycle alarms (manufacturer and OEM)		
080 000 – 084 999	Message texts for Siemens cycles		
085 000 – 089 999	Message texts for user cycles		
090 000 – 099 999	- reserved -		
100 000 – 129 000	System	10000	/HMI
130 000 – 139 000	OEM		
140 000 – 199 999	- reserved -		
200 000 – 299 999	Drive: SINAMICS	0	/NCK
300 000 – 399 999	- reserved -		
400 000 – 499 999	General alarms	51	/PLC/PMC
500 000 – 599 999	Channel alarms		
600 000 – 699 000	Axis/spindle alarms		
700 000 – 799 999	User alarms		
800 000 – 899 999	- reserved -		
810 000 – 810 009	System error messages	50	/PLC/DiagBuffer
900 000 – 999 999	- reserved -	0	/NCK

A.3 List of the color codes

RGB colors

The colors configured are defined by the RGB value:

- The RGB value starts with the character "#" (hexadecimal).
- The RGB value is entered in the format "#RRGGBB".

Each R, G or B represents a single-digit hexadecimal number.

Color code

Color	Code (hexadecimal) ⇓	Code (decimal)		
Black	#000000	0	0	0
Red	#FF0000	255	0	0
Green	#00FF00	0	255	0
Blue	#0000FF	0	0	255
White	#FFFFFF	255	255	255

A.4 List of abbreviations

Abbreviation	Meaning	Explanation
ALM	Active Line Module	
ASCII	American Standard Code for Information Interchange	American coding standard for the exchange of information
AUTO	"Automatic" operating mode	
BAG	Mode group	
BERO	Proximity limit switch with feedback oscillator	
BICO	Binector Connector	Interconnection technology for the drive
CEC	Cross Error Compensation	
CNC	Computerized Numerical Control	Computerized numerical control
DB	Data Block in the PLC	
DBB	Data Block Byte in the PLC	
dbSI	drive based Safety Integrated	Safety functions integrated in the drive
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	
DO	Drive object	Drive Object
DRAM	Dynamic Random Access Memory	Dynamic memory block
DRF	Differential Resolver Function	Differential resolver function (handwheel)
DRY	DRY run	DRY run feedrate
ESR	Extended Stop and Retract	
FIFO	First In - First Out	Method of storing and retrieving data in a memory
GUD	Global User Data	Global user data
Hardware	Hardware	
HD	Hard Disk	Hard disk
HSC	High-Speed Cutting	
IGBT	Insulated Gate Bipolar Transistor	
IME	Input Method Editor	Entering Asian characters
INC	Increment	Increment
INI	Initializing Data	Initializing data
IPO	Interpolator	
IRT	Isochronous Real Time	Isochronous communication
ISO	International Standardization Organisation	International Standards Organization
JOG	"Jogging" operating mode	Jogging via the direction keys
LEC	Leadscrew Error Compensation	Leadscrew error compensation
LED	Light Emitting Diode	Light-emitting diode
LUD	Local User Data	Local user data
MAIN	Main program	Main program (OB1, PLC)
MB	Megabyte	

Abbreviation	Meaning	Explanation
MCP	Machine Control Panel	Machine control panel
MCS	Machine Coordinate System	
MD	Machine data	
MDA	"Manual Data Automatic" operating mode	Manual input
MLFB	Machine-Readable Product Code	
MPF	Main Program File	Main program (NC part program)
MPI	Multi Point Interface	Multi-Point Interface
NCK	Numerical Control Kernel	Numerical control kernel
NCU	Numerical Control Unit	NCK hardware unit
OEM	Original Equipment Manufacturer	
OPI	Operator Panel Interface	
PCU	Programmable Control Unit	
PG	Programming device	
PI	Program Instance	
PLC	Programmable Logic Control	Programmable logic controller
POU	Program organization unit	in the PLC user program
PPU	Panel Processing Unit	Panel-based control
PZD	Process data for drives	
QEC	Quadrant Error Compensation	Quadrant error compensation
REF POINT	"Reference point approach" in JOG mode	
REPOS	"Repositioning" in JOG mode	
RPA	R-Parameter Active	Memory area on the NCK for R parameter numbers
RTC	Real Time Clock	Real-time clock
SBL	Single Block	Single block
SBR	Subroutine	Subroutine (PLC)
SD	Setting Data	
SDB	System Data Block	
SEA	Setting Data Active	Identifier (file type) for setting data
SK	Softkey	
SLM	Smart Line Module	
SPF	Subprogram file	Subprogram (NC)
SRAM	Static Random Access Memory	Static memory block
SW	Software	
TEA	Testing Data Aktive	Identifier for machine data
TM	Tool Management	
TMMG	Tool Magazine Management	
TO	Tool Offset	Tool offset
TOA	Tool Offset Active	Identifier (file type) for tool offsets
VPM	Voltage Protection Module	
VSM	Voltage Sensing Module	
WCS	Workpiece Coordinate System	

Abbreviation	Meaning	Explanation
WO	Work Offset	
ZOA	Zero Offset Active	Identifier (file type) for work offset data

Appendix B

B.1 Definitions for license management

Important terms

The terms below are important for understanding the license management of SINUMERIK software products:

Term	Description
Software product	The term software product is generally used to describe a product that is installed on a piece of hardware to process data. Within the license management of SINUMERIK software products, a corresponding license is required to use each software product.
Hardware	In the context of license management of SINUMERIK software products, hardware refers to the component of a SINUMERIK control system to which licenses are assigned on the basis of its unique identifier. License information is also saved to non-volatile memory on this component. Examples: • SINUMERIK 828D/840D sl: CompactFlash card • SINUMERIK 840Di sl: MCI board
License	A license gives the user a legal right to use the software product. Evidence of this right is provided by the following: • CoL (Certificate of License) • License key
CoL (Certificate of License)	The CoL is the proof of the license. The product may only be used by the holder of the license or authorized persons. The CoL includes the following data relevant for the license management: • Product name • License number • Delivery note number • Hardware serial number **Note:** The hardware serial number is located only on a CoL of the system software or if the license was ordered bundled, in other words the system software came together with options.
License number	The license number is the feature of a license that is used for its unique identification.

Term	Description
CompactFlash card	As the carrier of all non-volatile data in a SINUMERIK solution line control system, the CompactFlash card represents the identity of this control system. The CompactFlash Card includes the following data relevant for the license management: • Hardware serial number • License information including the license key
Hardware serial number	The hardware serial number is a permanent part of the CompactFlash card. It is used to identify a control system uniquely. The hardware serial number can be determined by: • CoL (see: Certificate of License "Note") • Dialog in the operating software • Printing on the CompactFlash Card system
License key	The License Key is the "technical representative" of the sum of all the licenses that are assigned to one particular piece of hardware, which is uniquely marked by its hardware serial number.
Option	An option is a SINUMERIK software product that is not included in the basic version and that requires the purchase of a license for its use.
Product	A product is marked by the data below within the license management of SINUMERIK software products: • Product designation • Order number • License number

B.2 Documentation overview

Documentation overview, SINUMERIK 828D

General documentation



Sales brochure

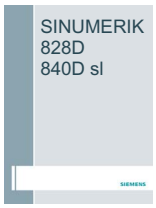


EMC directives

User documentation



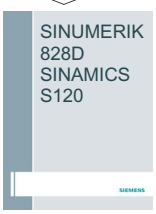
Operating Manual
- Turning
- Milling



Programming Manual
- Basic information
- Production planning
- Easy Screen

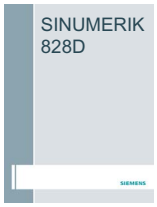


Programming Manual
- ISO turning
- ISO milling



Diagnostics Manual

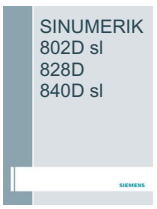
Manufacturer/service documentation



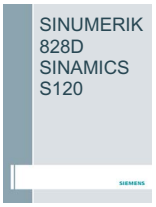
Manual
Commissioning Manual
Service Manual



Function Manual
- Basic functions
- Extended functions



Function Manual
ISO dialects



List Manual
- Machine data and interface signals
- Detailed Parameter description

Electronic documentation



DOConCD
DOConWEB



Industry Mall

Glossary

Active Line Module (ALM)

A controlled, self-commutating infeed/feedback unit (with IGBTs in infeed/feedback direction) which supplies a DC-link voltage for the → Motor Modules.

Basic infeed

Overall functionality of an infeed with → Basic Line Module, including the additional components required (filters, switching devices, etc.).

Basic Line Module

Unregulated line infeed unit (diode bridge or thyristor bridge, without feedback) for rectifying the line voltage of the → DC link.

CompactFlash card

Memory card with non-volatile memory: The CompactFlash card can be plugged into the → Control Unit from outside.

Control Unit

Central open and closed-loop control module. The following Control Units are available:

- SIMOTION Control Units, e.g. D425 and D435
- SINAMICS Control Units, e.g. CU320
- SINUMERIK solution line Control Units, e.g. NCU 7x0, PPU

Double Motor Module (DMM)

Two motors can be connected to and operated with a Double Motor Module. See → Motor Module.

Drive

The drive includes the (electric or hydraulic) motor, the actuator (converter, valve), the control unit, the measuring system and the supply components (line infeed module, pressure accumulator). For electric drives, a distinction is made between a converter system and an inverter system. With a converter system (e.g. → MICROMASTER 4), the line infeed, the actuator, and the control component form a single device from the point of view of the user. With an inverter system (e.g. → SINAMICS S), the supply is ensured by means of → Line Modules, thereby realizing a DC link to which the → Inverters (→ Motor Modules) are connected. The (→ Control Unit) is implemented as a separate device and connected to the other components by means of → DRIVE-CLiQ.

Drive component

Hardware component connected to a → Control Unit via → DRIVE-CLiQ or in some other way. Drive components include: → Motor Modules, → Line Modules, → Motors, → Sensor Modules and → Terminal Modules. The overall arrangement of a Control Unit including the connected drive components is known as the → Drive unit.

Drive line-up

A drive line-up comprises a → Control Unit as well as the → Motor Modules and → Line Modules connected via → DRIVE-CLiQ.

Drive object (DO)

A drive object is a self-contained software function with its own → parameters and, if necessary, its own → Faults and → Alarms. Drive objects may exist by default (e.g. onboard I/O), can be created individually (e.g. → Terminal Board 30, TB30) or also as multiples (e.g. → Servo control). As a rule, each drive object has its own window for parameterization and diagnostic purposes.

Drive parameters

Parameters of a drive axis that include, for example, the parameters of the corresponding controllers, as well as the motor and encoder data. The parameters of the higher-level technology functions (positioning, ramp-function generator), however, are called → Application parameters.

Drive system

A drive system includes all components of a family of products (e.g. SINAMICS) belonging to a drive. A drive system includes components such as → Line Modules, → Motor Modules, → Encoders, → Motors, → Terminal Modules and → Sensor Modules, as well as complementary components such as reactors, filters, lines, etc.

Drive unit

The drive unit includes all components, which are connected via → DRIVE-CLiQ and required for carrying out a drive task: → Motor Module → Control Unit → Line Module and the required → Firmware and → Motors, but not complementary components (such as filters and reactors). Several → Drives can be implemented in one drive unit. See → Drive system

DRIVE-CLiQ

Abbreviation for "Drive Component Link with IQ".

Communication system for connecting the various components of a SINAMICS drive system, such as the → Control Unit, → Line Modules, → Motor Modules, → Motors and speed/position encoders.

DRIVE-CLiQ is based on Industrial Ethernet using twisted-pair lines. The DRIVE-CLiQ line provides the send and receive signals, as well as the +24 V power supply.

Encoder

Records and makes positions available for electronic processing. Depending on their mechanical design, encoders can be incorporated in the → Motor (→ Motor encoder) or mounted on the external mechanics (→ External encoder). Depending on the type of movement, a distinction is made between rotary encoders (also called shaft encoders) and translatory encoders (e.g. → Linear encoder). In terms of measured value provision, we distinguish between → Absolute encoders (code sensors) and → Incremental encoders. See → Incremental encoder TTL/HTL → Incremental encoder sin/cos 1 Vpp → Resolver.

External encoder

Position encoder that is not built in or mounted on the → Motor, but is attached outside to the working machine or via a mechanical intermediate element. The external encoder (externally mounted encoder) is used for direct position detection.

Individual (I) data class

This data class covers data concerning a particular machine and is also created during commissioning by the OEM, or later by the dealer. This data class will be referred to as "I" in the rest of this document.

Infeed

Input component of a converter system for generating a DC-link voltage for feeding one or several → Motor Modules, including all components required, such as → Line Modules, fuses, reactors, line filters and firmware, as well as proportional computing power (if required) in a → Control Unit.

Line Module

A Line Module is a power component which creates the DC-link voltage for one or several → Motor Modules from a three-phase mains voltage. The following three Line Module types are used for SINAMICS: → Basic Line Module, → Smart Line Module and → Active Line Module.

The overall function of an infeed, including the required additional components such as → Line reactor, proportional computing power in a → Control Unit, switching devices, etc. is called → Basic infeed, → Smart infeed, and → Active infeed.

Manufacturer (M) data class

This data class covers all data defined by the machine manufacturer (OEM) when a machine in a model series is commissioned for the first time. For the rest of this document it will be referred to as "M".

Motor

For the electric motors that can be driven by → SINAMICS, a basic distinction is made between rotary and linear motors with regard to their direction of motion, and between synchronous and induction motors with regard to their electromagnetic operating principle. For SINAMICS, the motors are connected to a → Motor Module. See → Synchronous motor → Induction motor → Motor encoder → External encoder.

Motor encoder

An → Encoder integrated in the motor or attached to the motor, e.g. → Resolver, → Incremental encoder TTL/HTL or → Incremental encoder sin/cos 1 Vpp. The encoder is used for detecting the motor speed. In the case of synchronous motors, also for detection of the rotor position angle (the commutation angle for the motor currents). For drives without an additional → Direct position measuring system, it is also used as a → Position encoder for position control. In addition to the motor encoders, there are → External encoders for → Direct position sensing.

Motor Module

A motor module is a power unit (DC-AC inverter) that provides the power supply for the motor(s) connected to it. Power is supplied via the → DC link of the → Drive unit. A Motor Module must be connected to a → Control Unit via → DRIVE-CLiQ. The open-loop and closed-loop control functions of the Motor Module are stored in the Control Unit. There are Single Motor Modules and → Double Motor Modules.

Parameter

Variable quantity within the drive system that the user can read and, in some cases, write. For → SINAMICS, a parameter meets all specifications defined for drive parameters in the → PROFIdrive profile. See → Display parameter → Adjustable parameter

Sensor Module

Hardware module for evaluating speed/position encoder signals and providing detected actual values as numerical values at a → DRIVE-CLiQ socket. There are three mechanical variants of Sensor Modules:

- SMCxx = Sensor Module Cabinet-Mounted = Sensor Module for the snap-on mounting in the control cabinet.
- SME = Sensor Module Externally Mounted = Sensor Module with high degree of protection for mounting outside the control cabinet.

Servo control

This type of closed-loop control enables operation with a high dynamic response and precision for → Motors with → Motor encoders. In addition to speed control, position control can be included.

Servo drive

An electric servo drive consists of a motor, a → Motor Module and a → Servo control and, in most cases, a speed and position encoder. Electric servo drives generally work very precisely and with a high dynamic response. They are designed for cycle times of up to 100 ms. In many cases, they have a very short-time overload capacity, thus allowing particularly fast acceleration. Servo drives are available as rotary and as linear drives.

Smart Line Module (SLM)

Unregulated line infeed/feedback unit with a diode bridge for feeding; stall-protected, line-commutated feedback via IGBTs. The Smart Line Module provides the DC-link voltage for the → Motor Modules.

System (S) data class

This data class covers data in the Siemens and System directories on the CompactFlash Card. For the rest of this document it will be referred to as "S".

User (U) data class

This data class covers all user data as well as data that is created during the machine runtime, for example the maintenance interval timer. This data class will be referred to as "U" in the rest of this document.

User views

User views are user-specific groups of machine data. They are used to call all relevant machine data in a certain operating state from various areas for processing.

The user views are stored on the CompactFlash card with the following path:

`user/sinumerik/hmi/template/user_views`

The following user views are already available as template:

- Electrical_Startup
- Mechanical_Startup
- Optimizing_Axis

Index

$

$MC_CUTTING_EDGE_DEFAULT (MD20270), 353
$MC_TOOL_CHANGE_ERROR_MODE (MD22562), 336, 353, 377, 378
$MC_TOOL_CHANGE_MCODE (MD22560), 353
$MC_TOOL_CHANGE_MODE (MD22550), 335, 353
$MC_TOOL_MANAGEMENT_MASK (MD20310), 354
$MC_TOOL_MANAGEMENT_TOOLHOLDER (MD20124), 353
$MCS_DISP_COORDINATE_SYSTEM (MD52000), 148
$MCS_TM_FUNCTION_MASK (MD52270), 355
$MN_M_NO_FCT_CYCLE_NAME (MD10716), 353
$MN_MAXNUM_REPLACEMENT_TOOLS (MD17500), 353
$MN_PLC_DEACT_IMAGE_LADDR_IN (MD12986), 80
$MN_PLC_DEACT_IMAGE_LADDR_OUT (MD12987), 80
$MN_T_NO_FCT_CYCLE_NAME (MD10717), 358
$MN_UPLOAD_MD_CHANGES_ONLY (MD11210), 416
$MN_USER_DATA_INT (MD14510), 277, 301
$MN_USER_DATA_PLC_ALARM (MD14516), 55
$MNS_ACCESS_RESET_SERV_PLANNER (MD51235), 274
$SNS_TM_FUNCTION_MASK_SET (SD54215), 356

1

1:1 Change, (--> Tool change)

A

Access levels, 39
Acknowledgment
  Maintenance task, 266
  Process, 360
  Rules, 372
  Status, 362
  synchronous, 361
  TM, 341
Acknowledgment block, 267
Acknowledgment step, 370
Acknowledgment-step table, 352
Actual data table, 265
Addressing
  DO, 315
  GUD, 315
  MD, 314
  NC variables, 314
  NX, 316
  Parameter, 313
  Setting data, 314
Adjacent location, 335
Advanced Surface, 143, 144, 228
Alarms
  Backing up permanently, 58
  Configuring variables, 56
  Language code, 421
  Log, 57
  Number ranges, 422
  Structure, 54
Archive
  Memory area, 413
  Series start-up, 414
Axis configuration
  Lathe with milling tools, 173
  Milling machine, 163

B

Buffer, 334

C

Certificate of License (CoL), 427
Chain magazine, 334
Circular magazine, 335
Circumferential groove, 162
Company network, 29
Configuring the measuring system, 111
Contour grooving CYCLE930, 169
Contour milling (CYCLE63), 161
Contour turning CYCLE950, CYCLE952, 169
Coordinate system MD52000, 148
CUST_800.SPF, 151
CUST_832.SPF, 151, 230
CUST_MEACYC.SPF, 151, 233
CUST_TECHCYC.SPF, 151, 184, 195
Cycle support, 148
CYCLE79, 162
CYCLE800, 155

CYCLE800 example
Cardanic swivel head, 213
Cardanic table, 215
Changeable swivel head, 211
Swivel head / rotary table, 217
Swivel table, 219
CYCLE832, 143
CYCLE84, 158
CYCLE840, 158
CYCLE930, 169
CYCLE950, 169
CYCLE951, 169
CYCLE952, 169
CYCLE982, 258
CYCLE99, 169
Cylinder surface transformation
under milling, 162
under turning, 175
with groove side offset, 177
without groove side correction, 176

D

Data area, 412
Service Planner, 263
Data classes, 32
DB1800, 266
DB9903, 264
DB9904, 265
Definition files, 33
Dictionary
Editing, 48
importing, 48
Direct connection, 26
Drilling, 158
Drive
Circuitry, 134
Configuration, 100
Parameter, 122
DRIVE-CLiQ, 100
Wiring rules, 126

E

Easy Archive, 417
Easy Extend, 275
Options bits, 277
EE_IFC (DB9905), 276
End acknowledgment, 361, 362
Ethernet interface, 29
Example
Axis configuration of a lathe, 173
Change manual tool (1), 378
Change manual tool (2), 379
Measure tool in AUTO mode during milling, 252
Milling machine, 393
Milling machine axis configuration, 163
Milling part program, 394
Residual material machining, 172
Turning machine, 382
Turning part program, 383

F

Feedback signal, 342

G

Groove side offset, 163
GUD, 33

H

Hardware serial number, 428
High Speed Cutting (HSC), 143
High Speed Settings (CYCLE832), 228
Hirth tooth system, 200, 214

I

I/O module DIP switch, 82
Infeed, 108
Initial data table, 264
Input Method Editor (IME), 47
Intermediate acknowledgment, 361, 362
IP address, 82
ISO Dialect, 142

J

Job status, 350

K

Kinematics
Check list, 205
Swivel data set, 206
with Hirth tooth system, 200

L

Language code, 421
License, 51, 427
License key, 51, 428
License management, 427
License number, 427
Loading magazine, 334

M

Machine data, 139
  Effectiveness, 141
  Unit, 140
Machining type G group 59, 229
Macros, 33
Magazine, 333, 367
  Configuration, 390
Magazine list, 334
Manual tools, 336
MD10715[0]
  M_NO_FCT_CYCLE, 353
MD10716[0]
  M_NO_FCT_CYCLE_NAME, 353
MD10717
  $MN_T_NO_FCT_CYCLE_NAME, 358
MD11210
  $MN_UPLOAD_MD_CHANGES_ONLY, 416
MD12986
  PLC_DEACT_IMAGE_LADDR_IN, 80
MD12987
  PLC_DEACT_IMAGE_LADDR_OUT, 80
MD14510
  $MN_USER_DATA_INT[i], 277, 301
MD14516
  USER_DATA_PLC_ALARM, 55
MD17500
  MAXNUM_REPLACEMENT_TOOLS, 353
MD20124
  TOOL_MANAGEMENT_TOOLHOLDER, 353
MD20270
  CUTTING_EDGE_DEFAULT, 353
MD20310
  TOOL_MANAGEMENT_MASK, 354
MD22550
  TOOL_CHANGE_MODE, 335, 353
MD22560
  TOOL_CHANGE_MCODE, 353
MD22562
  TOOL_CHANGE_ERROR_MODE, 336, 377, 378
MD51235
  ACCESS_RESET_SERV_PLANNER, 274
MD52000
  $MCS_DISP_COORDINATE_SYSTEM, 148
MD52270
  TM_FUNCTION_MASK, 355
Measure tool
  In AUTO mode during turning, 258
  In JOG mode during turning, 242
  when milling, 250
Measure workpiece
  AUTO settings, 245
  In AUTO mode during milling, 247
  In JOG mode during milling, 236
  under turning, 249
Measured value correction using correction tables, 256
Measurement path, 242
Measuring feedrate, 242
Message, asynchronous, 361
Milling
  Cylinder surface transformation, 162
  Measure tool in JOG, 239
  Measure workpiece, 247
  Measure workpiece in JOG, 236
Multi-edge CYCLE79, 162

N

NC command TCA, 357
NCK variables, 372
Network connection, 26

O

Option, 51
  3D simulation 1 (finished part), 153
  Extended technology functions, 147
  Inclined axis, 181, 197
  Measuring cycles, 243
  Residual material identification and machining, 172
  ShopMill/ShopTurn, 147, 160, 166, 184, 203
  Simultaneous recording (real-time simulation), 153
  TRANSMIT and peripheral surface transformation, 162, 175, 178, 195, 196
  Travel to fixed stop (with force control), 191

P

password
  Change, 41
  Set, 41
PI service, 375
PLC user alarms, 54

PLC user program, 332
Aligning, 371
PLC-Firmware, 333
Probe, 136
Assigning NC measuring input, 233
Example of a test program, 232
Measure tool, 231
Measure workpiece, 231
Testing the function, 232
PROG_EVENT, 152
programGUIDE, 148
Programming Tool, 15

R

RCS Commander, 15
Residual material machining (turning), 172
RGB colors, 423
Rotary axis vectors, 206
RTC capacitor, 19

S

SD54215
TM_FUNCTION_MASK_SET, 356
Serial interface, 419
Series start-up, 412
Service Planner
Configuration mode, 269
Standard mode, 270
Setting data, 139
Setting date/time, 42
Setting-up measuring in JOG, 234
ShopTurn
Cylinder surface transformation, 195
Face end machining, 196
Setting up a counterspindle, 191
Tapping, 160
Simulation, 154, 155
Deactivate, 154
Simultaneous recording, activating the runtime, 154
Source ID, 422
Source URL, 422
STARTER drive/commissioning software, 15
Stock removal, corner CYCLE951, 169
String functions, 319
Swivel
activating, 198
Commission kinematic chain, 206
Configuring the input dialog, 198
under ShopMill, 204
System languages, 43
System variables, 208

T

Tapping, 158
TCP/IP, 29
Technology
Drilling, 158
Milling, 161
Swivel, 198
Terminal assignment
X122, 130
X132, 131
X242, 132
X252, 132
Thread-cutting CYCLE99, 169
TMMVTL (PI service), 375
Tolerance value, 228
Tool change, 344, 399
End acknowledgment, 344
Tool list, 334
Tool management, 331
Tool measuring in JOG, 357
Toolbox, 15
Toolholder that can be oriented, 259
Total acknowledgment, 362
TRAANG, 183
Inclined axis, 197
TRACON, 183
TRACYL, 183
Cylinder surface transformation, 195
Transfer step, 367
Transfer-step table
Constant, 351
variable, 351
TRANSMIT, 183
TRANSMIT with Y axis, 180
Trigonometric functions, 329
Turning
Cylinder surface transformation (TRACYL), 175
Inclined axis (TRAANG), 181
Measure workpiece, 249

U

USB FlashDrive, 413
User example, 417
User interface, 337
User views, 141

V

V24
Deactivate, 420
Interface, 419
Parameter, 419
Variable
Location state, 373
Location type, 372
T number, 374

W

Wiring rules DRIVE-CLiQ, 126

X

XML
Identifiers, 289
Operators, 288
Special characters, 288
Statements, 317
XML identifier
?up, 305
AGM, 289
BOX, 307
CAPTION, 305
CLOSE, 305
CONTROL, 307
CONTROL_RESET, 290
DATA, 291
DATA_ACCESS, 291
DATA_LIST, 292
DEVICE, 289
DRIVE_VERSION, 293
FILE, 293
FORM, 305
FUNCTION, 295
FUNCTION_BODY, 296
IMG, 309
INCLUDE, 297
INIT, 305
LET, 298
MSGBOX, 299
NAME, 289
OP, 300
OPTION_MD, 301
PAINT, 305
PASSWORD, 302
PLC_INTERFACE, 302
POWER_OFF, 303
PRINT, 304
PROPERTY, 310
REQUEST, 311
SET_ACTIVE, 289
SET_INACTIVE, 289
SOFTKEY_CANCEL, 311
SOFTKEY_OK, 311
START_UP, 289
TEST, 289
TEXT, 312
UID, 289
UPDATE_CONTROLS, 312
VERSION, 289
WAITING, 304