SIEMENS

SINUMERIK

SINUMERIK 828D
Tool management

Function Manual

Introduction 1

Fundamental safety instructions 2

Fundamentals 3

Machine data for the tool management 4

NC-PLC interface of the tool management 5

PLC Program Blocks 6

Example: Loading/ unloading 7

Example: Change manual tools 8

Application example for turning machine 9

Application example for milling machine 10

NC programming 11

SINUMERIK Operate user interface 12

Code carrier connection - Tool Ident Connection (option) 13

Appendix A

Valid for

Control system
SINUMERIK 828

CNC software version 5.25 SP1

07/2025
A5E48765003B AK

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**
indicates that minor personal injury can result if proper precautions are not taken.

NOTICE
indicates that property damage can result if proper precautions are not taken.

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, in particular its warning notices and safety instructions. Qualified persons are those who, because of their training and experience, are familiar with the installation, assembly, commissioning, operation, decommissioning and disassembly of the product and can recognize risks and avoid possible hazards.

Proper use of Siemens products

Note the following:

⚠ **WARNING**
Siemens products may only be used for the application described in the catalog and the associated usage information. 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 complied with. The information in the relevant documentation must be observed.

Trademarks

All names identified by ® are registered trademarks of Siemens Aktiengesellschaft. 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 Aktiengesellschaft
Digital Industries
Postfach 48 48
90026 NÜRNBERG
GERMANY

A5E48765003B AK
Ⓟ 07/2025 Subject to change

Copyright © Siemens 2019 - 2025.
All rights reserved

Table of contents

1 Introduction 9

1.1 About SINUMERIK 9

1.2 About this documentation 9

1.3 Documentation on the internet 11
1.3.1 Documentation overview SINUMERIK ONE 11
1.3.2 Documentation overview SINUMERIK 840D sl 12
1.3.3 Documentation overview SINUMERIK 828D 13
1.3.4 MC documentation overview 14
1.3.5 Documentation overview SINUMERIK operator components 14

1.4 Feedback on the technical documentation 15

1.5 mySupport documentation 15

1.6 Service and Support 16

1.7 Using OpenSSL 17

1.8 Compliance with the General Data Protection Regulation 17

2 Fundamental safety instructions 19

2.1 General safety instructions 19

2.2 Warranty and liability for application examples 19

2.3 Cybersecurity information 19

3 Fundamentals 21

3.1 Tool Management - Overview 21

3.2 Structure of the tool management 22

3.3 Components of the tool management 23

3.4 Multitool 27

3.5 Manual tools 27

4 Machine data for the tool management 31

5 NC-PLC interface of the tool management 35

5.1 Relocating, unloading, loading tool, positioning magazine 37

5.2 Tool change 44

5.3 Transfer-step and acknowledgment-step tables 53

6 PLC Program Blocks 57

6.1 Acknowledgment process 57

6.2 Types of acknowledgment 57

6.3 Acknowledgment states 59

6.4 Configuring step tables ... 63
6.5 Configuring acknowledgment steps ... 66
6.6 Adapting the PLC program ... 67
6.7 Information on magazine location ... 68
6.8 PI service: TMMVTL ... 71
7 Example: Loading/unloading ... 73
8 Example: Change manual tools ... 75
9 Application example for turning machine ... 79
9.1 Example: Turning machine with revolver magazine (MAGKONF_MPF) ... 79
9.2 Example: Acknowledgment steps (turning machine) ... 84
9.3 Example: Tool change cycle for turning machine ... 85
9.4 Example: Block search and MCALL ... 87
9.5 Example: Turning machine with counterspindle ... 88
9.6 Example: Test for empty buffer ... 88
9.7 Example: Transporting a tool from a buffer into the magazine ... 89
9.8 Example: Repeat "Prepare tool change" order ... 90
10 Application example for milling machine ... 91
10.1 Example: Milling machine with chain magazine and dual gripper (MAGKONF_EXAMPLE_MPF) ... 91
10.2 Flow chart: Tool change ... 96
10.3 Example: Acknowledgment steps (milling machine) ... 105
10.4 Example: Tool change cycle for milling machine ... 106
10.5 Monitoring the maximum speed of a tool ... 107
11 NC programming ... 115
11.1 Overview of OPI and system variables ... 115
11.2 Use of the magazine data ... 117
11.3 Use of the tool and cutting edge data ... 119
11.4 Cutting edge data ... 121
11.4.1 Cutting edge data ... 121
11.4.2 Cutting edge parameter $TC_DP ... 122
11.4.3 User cutting edge data $TC_DPC ... 124
11.4.4 Cutting edge-related tool monitoring $TC_MOP ... 125
11.4.5 User cutting edge monitoring $TC_MOPC ... 126
11.4.6 Location of use-dependent offsets fine (additive offsets) $TC_SCP ... 127
11.4.7 Location-dependent offsets coarse (setting-up offsets) $TC_ECP ... 127
11.5 Tool data ... 128
11.5.1 Overview ... 128
11.5.2 Tool-related data $TC_TP ... 128

11.5.3 Tool-related grinding data $TC_TPG ... 135
11.5.4 Tool-related user data $TC_TPC ... 136

11.6 Magazine data ... 137
11.6.1 Overview of magazine data ... 137
11.6.2 Magazine description data $TC_MAP ... 137
11.6.3 Magazine user data $TC_MAPC ... 140
11.6.4 Magazine location data $TC_MPP ... 140
11.6.5 Magazine location user data $TC_MPPC ... 144
11.6.6 Magazine location type hierarchy $TC_MPTH ... 145
11.6.7 Distance to change position $TC_MDP ... 146
11.6.8 Magazine blocks $TC_MAMP ... 149
11.6.9 Assignment of buffers to spindles $TC_MLSR ... 157

11.7 Multitool data ... 159
11.7.1 $TC_MTP - multitool data ... 159
11.7.2 $TC_MTPP - Multitool location data ... 168
11.7.3 Multitool-related user data ... 170
11.7.4 Multitool-location-related user data ... 170

11.8 Adapter data $TC_ADPT ... 171

11.9 User-definable variables ... 172

11.10 NC language commands ... 173
11.10.1 CHKDNO - Uniqueness check for D number ... 173
11.10.2 CHKDM - Uniqueness check within a magazine ... 174
11.10.3 GETACTTD - Determining the tool number of a unique D No. ... 175
11.10.4 GETDNO - Read out D numbers ... 176
11.10.5 SETDNO - Set or change D number ... 177
11.10.6 DZERO - Invalidate D numbers ... 177
11.10.7 DELDL - Delete additive offsets ... 178
11.10.8 NEWT - Create new tool ... 178
11.10.9 NEWMT Create new multitool ... 179
11.10.10 DELT Delete tool ... 181
11.10.11 DELMT - Delete multitool ... 181
11.10.12 $TC_MTPN - Delete one or all multitools ... 182
11.10.13 Reading the GETT - T no. ... 183
11.10.14 SETPIECE - decrementing the workpiece counter ... 184
11.10.15 GETSELT - Read the selected T number ... 186
11.10.16 GETEXET - Reading of the loaded T number ... 189
11.10.17 $P_MTHSDC - leading tool holder with regard to the D offset selection ... 192
11.10.18 $P_TH_OF_D - leading tool holder with regard to the current D offset ... 194
11.10.19 GETACTT - Read the active internal T number ... 195
11.10.20 SETMS - Set leading spindle ... 196
11.10.21 SETMTH - Set leading tool holder number ... 197
11.10.22 POSM - Positioning the magazine ... 200
11.10.23 POSMT - position multitool on toolholder to location number ... 203
11.10.24 MVTOOL - NC command to move tool ... 205
11.10.25 SETTA - Activate tool from wear group ... 208
11.10.26 SETTIA - Deactivate tool from wear group ... 210
11.10.27 RESETMON - NC command for setpoint activation ... 212
11.10.28 DELTC - Delete toolholder data record ... 216
11.10.29 TCA - Tool selection/tool change irrespective of tool status ... 217
11.10.30 TCI - Load tool from buffer into magazine ... 220

11.10.31 GETFREELOC - Search for empty location ... 222
11.10.32 DELMLRES - Delete location status "reserved for tool in the buffer" ... 228
11.10.33 DELMLOWNER - Delete owner magazine location of the tool ... 229
11.10.34 $P_USEKT - Tool change only with tools of subgroup ... 231
11.10.35 TOOLGNT/TOOLGT - Tool groups ... 234
11.10.36 $P_TMNOIS - is number, T number, magazine number, or MT number ... 235
11.10.37 $P_TOOLEXIST - Determine existence of a tool ... 236
11.10.38 $A_TOOLMN - read magazine no. from tool ... 237
11.10.39 $P_MTOOLN/$P_MTOOLMT- Determine the number of multitools/MT numbers ... 238
11.10.40 $P_MTOOLNT/$P_MTOOLT - Number of tools in the multitool ... 239
11.10.41 $A_TOOLMLN - Read magazine location no. from tool ... 240
11.10.42 $P_TOOLND - Read number of cutting edges for tool ... 241
11.10.43 $A_MONIFACT - factor for tool life monitoring ... 241
11.10.44 $AC_MONMIN - Factor for tool search ... 243
11.10.45 $P_TOOLNG - Number of tool groups ... 246
11.10.46 $A_MYMN/$A_MYMLN - Owner magazine/location of the tool ... 247
11.10.47 $A_MYMTN/$A_MYMTLN - $A_TOOLMTN/$A_TOOLMTLN - tools in the multitool ... 248
11.10.48 $P_TOOLNT/$P_TOOLT - T numbers ... 249
11.10.49 $P_TOOLD - D numbers ... 250
11.10.50 $P_TOOLNDL - Number of defined DL offsets ... 250
11.10.51 $A_USEDND - Workpiece counting ... 251
11.10.52 $A_USEDT - Workpiece counting ... 252
11.10.53 $A_USEDD - Workpiece count ... 253
11.10.54 $P_MAGN/$P_MAG - magazine ... 254
11.10.55 $P_MAGNDIS/$P_MAGDISS/$P_MAGDISL - magazine distance tables ... 255
11.10.56 $P_MAGNS/$P_MAGS - tool holder ... 256
11.10.57 $P_MAGNREL/$P_MAGREL - Assigned buffer ... 257
11.10.58 Example of magazine configuration system variables ... 258
11.10.59 $P_MAGNH/$P_MAGNHLT/$P_MAGHLT - Location type hierarchies ... 260
11.10.60 $P_MAGNA/$P_MAGA - tool adapter ... 263
11.10.61 Additional language commands ... 265
11.10.62 Variables for subroutine replacement technique ... 275
11.10.63 Variables for tool change in synchronized action ... 276

11.11 Definitions when programming data ... 278
11.11.1 Tool change ... 278
11.11.2 Cutting edge selection ... 279
11.11.3 Tool transfer from program test ... 280

11.12 Programming T=location number ... 280

11.13 Call multiple turrets with "T=location number" ... 282

11.14 Programming examples ... 283

11.15 Overview of the remaining OPI blocks of tool management ... 283
11.15.1 Magazine data, directory ... 283
11.15.2 Tool data, directory ... 284
11.15.3 Parameterization, return parameters _N_TMGETT, _N_TSEARC ... 284
11.15.4 Working offsets ... 285
11.15.5 PI services and language commands for tool management ... 286

12 SINUMERIK Operate user interface ... 291

12.1 Configuring the user interface ... 291
12.1.1 General settings ... 294

12.1.2 Configuring tool lists ... 300
12.1.2.1 Tool parameter identifiers ... 304
12.1.2.2 Cutting edge parameter identifiers ... 306
12.1.2.3 Monitoring parameter identifiers ... 311
12.1.2.4 Grinding parameter identifiers ... 311
12.1.2.5 Magazine location parameter identifiers ... 312
12.1.2.6 Magazine location adapter parameter identifiers ... 313
12.1.2.7 Multitool parameter identifiers ... 316
12.1.2.8 Multitool location parameter identifiers ... 317
12.1.2.9 Identifiers of the angle head parameters ... 318
12.1.2.10 Identifiers of the angle head location parameters ... 319
12.1.3 Configuring list parameters ... 320
12.1.4 List of tool types ... 324
12.1.5 Configure tool types ... 327
12.1.6 Configuring the "Additional Data" window ... 331
12.1.7 Configure the "New tool - favorites" window ... 333
12.1.8 Configuring the "New Tool" window ... 334
12.1.9 Configuring the "New Multitool" window ... 335
12.1.10 Configuring the "New angle head" window ... 335
12.1.11 Configuring the "New Tool from Code Carrier" window ... 336
12.1.12 Configuring the "New Tool from File" window ... 336
12.1.13 Configuring the "New tool from TTD file" window ... 336
12.1.14 Configuring default values for new tools ... 337
12.1.15 Configuring the "Details" window ... 344
12.1.16 Assigning a name for magazines and magazine locations ... 346
12.1.17 Configuring the "Assign Tooltips for Multitool Locations" window ... 348
12.1.18 Assigning a name for magazine location types ... 348
12.1.19 Assignment of magazines and magazine locations to channels ... 350
12.1.20 Assignment between magazines and magazine locations and coordinate systems ... 351
12.1.21 Disabling the softkey depending on filters ... 353
12.1.22 Configuring the "Load Station Selection" window ... 354
12.1.23 Coolant and tool-specific functions ... 354
12.1.24 Reason for the tool change when reactivating ... 358
12.1.25 Configure the code carrier connection ... 361
12.1.26 Configuring Save to file ... 364
12.1.27 Configure creation from a TTD file ... 366

12.2 Creating OEM texts ... 367
12.2.1 Identifiers of the standard texts ... 368
12.2.2 Examples of OEM texts ... 370

12.3 Examples ... 373
12.3.1 Example: Configuring an OEM tool list ... 373
12.3.1.1 Adapting the configuration file ... 373
12.3.1.2 Adaptation of the customer text file ... 375
12.3.2 Example: Configuring magazine location types with names ... 377
12.3.2.1 Adapting the configuration file ... 377
12.3.2.2 Adaptation of the customer text file ... 379

12.4 Working with two tool holders ... 381

13 Code carrier connection - Tool Ident Connection (option) ... 385

13.1 Installation ... 385
13.1.1 System requirements ... 385

13.1.2 Installation of Tool Ident Connection ... 386
13.1.2.1 Activation of Tool Ident Connection ... 386
13.2 Configuration ... 388
13.2.1 Configuration schematic ... 388
13.2.2 Configuration of the tool lists ... 388
13.2.3 TDIIdentCfg.xml ... 389
13.2.4 Read/write device at the PLC ... 392
13.2.5 Data conversion ... 392
13.2.5.1 Converting data ... 392
13.2.5.2 Messages during conversion ... 394
13.2.6 Conversion files wkonvert.txt and DefToolDat.txt ... 394
13.2.6.1 Structure of the description file wkonvert.txt ... 394
13.2.6.2 Examples for the description files wkonvert.txt ... 399
13.2.6.3 Tool data initialization file DefToolDat.txt ... 402
13.2.7 Conversion rule based on the MCIS-C script language ... 404
13.2.7.1 Structure of the conversion rule ... 404
13.2.7.2 Access to tool data ... 406
13.2.7.3 Auxiliary functions for numerical coding ... 408
13.2.7.4 Configuration of ToolSpec.xml ... 413
13.2.7.5 Error handling ... 414
13.2.7.6 Description of the MCIS-C script language ... 415
13.2.7.7 Auxiliary functions in conversion specifications ... 416
13.2.8 Examples ... 417
13.3 PLC interface in DB1900 ... 421
13.3.1 Function call of the PLC interface ... 421
13.3.2 Assignment of the PLC for transfer parameters ... 422
13.3.3 Assignment of the PLC for return parameters ... 422
13.3.4 PLC interface for the Parameter operating area ... 423
13.4 Command codes ... 424
13.4.1 Functional scope of the PLC interface (K codes) ... 424
13.4.2 Parameter description ... 425
13.4.3 Loading from the code carrier (K code = 0) ... 425
13.4.4 Unload to code carrier (K code = 1) ... 426
13.4.5 Loading a tool from the tool list (K code = 7) ... 428
13.4.6 Unloading a tool in the tool list (K code = 8) ... 429
13.4.7 Empty location search (K code = 9) ... 430
13.4.8 Delete tool in NC (K code = 10) ... 431
13.4.9 Updating the code carrier (K code = 14) ... 432
13.4.10 Reading data from the code carrier (K code = 15) ... 433
13.4.11 Error codes ... 434
13.5 Examples ... 440
13.5.1 Loading the code carrier via PLC interface ... 440
13.5.2 Unloading to the code carrier via PLC interface ... 444
A Appendix ... 447
A.1 List of abbreviations ... 447
Index ... 449

1 Introduction

1.1 About SINUMERIK

From simple, standardized CNC machines to premium modular machine designs – the SINUMERIK CNCs offer the right solution for all machine concepts. Whether for individual parts or mass production, simple or complex workpieces – SINUMERIK is the highly dynamic automation solution, integrated for all areas of production. From prototype construction and tool design to mold making, all the way to large-scale series production.

Visit our website for more information SINUMERIK (https://www.siemens.com/sinumerik).

1.2 About this documentation

This documentation is part of the documentation category of SINUMERIK Function Manuals.

SINUMERIK Function Manuals

The SINUMERIK Function Manuals describe the NC functions of a SINUMERIK control.

The target group includes configuring engineers, technologists, commissioning engineers and programmers.

Each Function Manual covers a specific topic area and contains all the function descriptions for this topic area.

The following table shows which Function Manuals are available for your SINUMERIK control and which topic areas a Function Manual covers:

Function Manual	Topic area
Basic functions	Basic functions of a CNC control
Axes and spindles	Axis and spindle functions, axis couplings
Transformations	Transformation functions
Monitoring and compensating	Axis monitoring and compensation functions, collision avoidance
Tools	Functions for selection, correction and monitoring of tools
Synchronized actions	Functionality of synchronized actions
PLC	Structure and functions of the PLC
Tool management	Function, commissioning and programming of tool management

Overview of contents

You will find an overview of the main chapters of the descriptions of functions that are contained in a Function Manual on the title page.

Validity

The title page also contains all information regarding the validity of a document, i.e. for which SINUMERIK control and for which software version this edition of the Function Manual is valid.

System data

In the descriptions of functions, the system data relevant for a function (machine data, setting data, system variables, interface signals and alarms) are only described with the level of detail that is strictly necessary to understand the function. You will find detailed information on this data in the relevant List Manuals and for alarms in the Diagnostics Manual.

PLC blocks

Interface signals are listed in the function descriptions. These can be used to control functionality via the PLC. You can find detailed descriptions of the PLC functionality, such as data interface and function interface, in the PLC Function Manual for SINUMERIK 828D.

Note

The functionality of the PLC blocks is implemented via subprograms. Examples of such subprograms are freely available to the user via the PLC toolbox.

Interface signals

In this document, the following form is used for addressing interface signals.

General graphical conventions	Examples
<signal address> (<signal name>)	DB38xx.DBX2000.0..2 (actual gear stage A ... C) DB38xx.DBX2000.3 (gear is switched over)

Note

Quantity structure

Explanations concerning the NC/PLC interface are based on the absolute maximum number of the following components:

- Mode groups (DB300x)
- Channels (DB32xx/DB33xx)
- Axes/spindles (DB38xx/DB39xx)

Standard scope

This documentation only describes the functionality of the standard version. This may differ from the scope of the functionality of the system that is actually supplied. Please refer to the ordering documentation only for the functionality of the supplied drive system.

It may be possible to execute other functions in the system which are not described in this documentation. This does not, however, represent an obligation to supply such functions with a new control or when servicing.

For reasons of clarity, this documentation cannot include all of the detailed information on all product types. Further, this documentation cannot take into consideration every conceivable type of installation, operation and service/maintenance.

The machine manufacturer must document any additions or modifications they make to the product themselves.

Websites of third-party companies

This document may contain hyperlinks to third-party websites. Siemens is not responsible for and shall not be liable for these websites and their content. Siemens has no control over the information which appears on these websites and is not responsible for the content and information provided there. The user bears the risk for their use.

1.3 Documentation on the internet

1.3.1 Documentation overview SINUMERIK ONE

Comprehensive documentation about the functions provided in SINUMERIK ONE Version 6.13 and higher is provided in the Documentation overview SINUMERIK ONE (https://support.industry.siemens.com/cs/ww/en/view/109768483).



You can display documents or download them in PDF and HTML5 format.

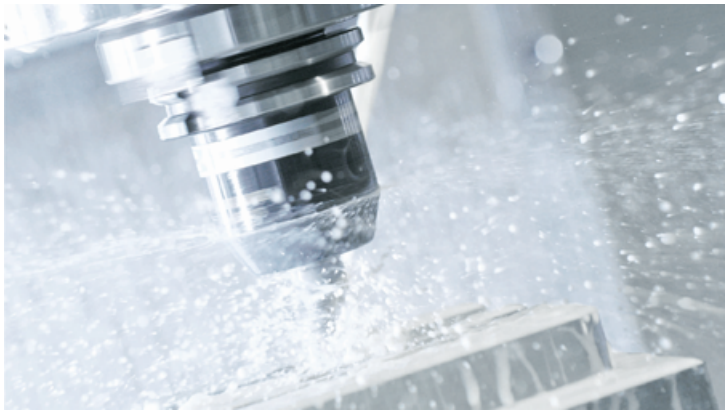
The documentation is divided into the following categories:

- User: Operating
- User: Programming
- Manufacturer/Service: Functions
- Manufacturer/Service: Hardware

- Manufacturer/Service: Configuring/Commissioning
- Manufacturer/Service: Safety Integrated
- Information and training
- Manufacturer/Service: SINAMICS

1.3.2 Documentation overview SINUMERIK 840D sl

You will find extensive documentation on the functions of SINUMERIK 840D sl from version 4.8 SP4 at 840D sl documentation overview (https://support.industry.siemens.com/cs/ww/en/view/109766213).



You can display documents or download them in PDF and HTML5 format.

The documentation is divided into the following categories:

- User: Operating
- User: Programming
- Manufacturer/Service: Functions
- Manufacturer/Service: Hardware
- Manufacturer/Service: Configuring/Commissioning
- Manufacturer/Service: Safety Integrated
- Manufacturer/Service: SINUMERIK Integrate/MindApp
- Information and training
- Manufacturer/Service: SINAMICS

1.3.3 Documentation overview SINUMERIK 828D

Comprehensive documentation about the functions provided in SINUMERIK 828D Version 4.8 SP4 and higher is provided in the 828D documentation overview (https://support.industry.siemens.com/cs/ww/en/view/109766724).



You can display documents or download them in PDF and HTML5 format.

The documentation is divided into the following categories:

- User: Operating
- User: Programming
- Manufacturer/Service: Configuring
- Manufacturer/Service: Commissioning
- Manufacturer/Service: Functions
- Manufacturer/Service: Safety Integrated
- SINUMERIK Integrate/MindApp
- Info & Training

1.3.4 MC documentation overview

Comprehensive documentation about the functions provided in SINUMERIK MC Version 1.12 and higher is provided in the SINUMERIK MC documentation overview (https://support.industry.siemens.com/cs/ww/en/view/109771043).



You can display documents or download them in PDF and HTML5 format.

The documentation is divided into the following categories:

- User: Operating
- User: Programming
- Manufacturer/Service: Commissioning
- Manufacturer/Service: Functions
- Manufacturer/Service: Hardware
- Manufacturer/Service: Safety Integrated
- Information and training

1.3.5 Documentation overview SINUMERIK operator components

Comprehensive documentation about the SINUMERIK operator components is provided in the Documentation overview SINUMERIK operator components (https://support.industry.siemens.com/cs/document/109783841/technische-dokumentation-zu-sinumerik-bedienkomponenten?dti=0&lc=en-WW).

You can display documents or download them in PDF and HTML5 format.

The documentation is divided into the following categories:

- Operator panels
- Handheld unit/Mini handheld devices
- Machine control panels
- Further operator components

An overview of the most important documents, entries and links on the topic of "SINUMERIK" is provided at SINUMERIK Overview - Topic Page (https://support.industry.siemens.com/cs/document/109766201/sinumerik-an-overview-of-the-most-important-documents-and-links?dti=0&lc=en-WW).

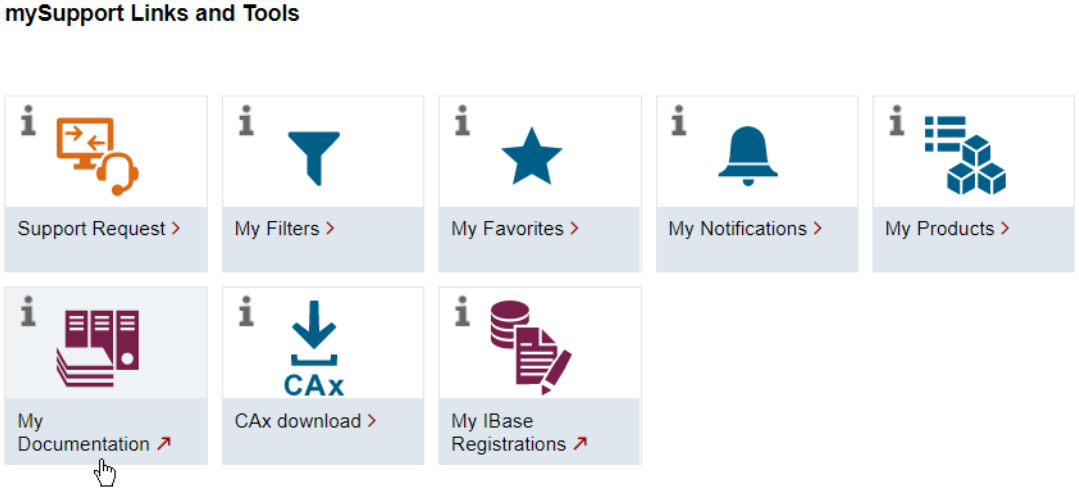
1.4 Feedback on the technical documentation

If you have any questions, suggestions, or corrections regarding the technical documentation published in the Siemens Industry Online Support, use the "Provide feedback" link which appears at the end of the entry.

1.5 mySupport documentation

With the "mySupport documentation" web-based system you can compile your own individual documentation based on Siemens content, and adapt it for your own machine documentation.

To start the application, click on the "My Documentation" tile on the "mySupport links and tools" (https://support.industry.siemens.com/cs/ww/en/my) portal page:



The configured manual can be exported in RTF, PDF or XML format.

Note

Siemens content that supports the mySupport documentation application can be identified by the presence of the "Configure" link.

1.6 Service and Support

Product support

You can find more information about products on the internet:

Product support (https://support.industry.siemens.com/cs/ww/en/)

The following is provided at this address:

- Up-to-date product information (product announcements)
- FAQs (frequently asked questions)
- Manuals
- Downloads
- Newsletters with the latest information about your products
- Global forum for information and best practice sharing between users and specialists
- Local contact persons via our Contacts at Siemens database (→ "Contact")
- Information about field services, repairs, spare parts, and much more (→ "Field Service")

Technical support

Country-specific telephone numbers for technical support are provided on the internet at address (https://support.industry.siemens.com/cs/ww/en/sc/4868) in the "Contact" area.

If you have any technical questions, please use the online form in the "Support Request" area.

Training

You can find information on SITRAIN at the following address (https://www.siemens.com/sitrain).
SITRAIN offers training courses for automation and drives products, systems and solutions from Siemens.

Siemens support on the go



With the award-winning "Industry Online Support" app, you can access more than 300,000 documents for Siemens Industry products – any time and from anywhere. The app can support you in areas including:

- Resolving problems when implementing a project
- Troubleshooting when faults develop
- Expanding a system or planning a new system

Furthermore, you have access to the Technical Forum and other articles from our experts:

- FAQs
- Application examples
- Manuals
- Certificates
- Product announcements and much more

The "Industry Online Support" app is available for Apple iOS and Android.

1.7 Using OpenSSL

This product can contain the following software:

- Software developed by the OpenSSL project for use in the OpenSSL toolkit
- Cryptographic software created by Eric Young.
- Software developed by Eric Young

You can find more information on the internet:

- OpenSSL (https://www.openssl.org)
- Cryptsoft (https://www.cryptsoft.com)

1.8 Compliance with the General Data Protection Regulation

Siemens observes standard data protection principles, in particular the data minimization rules (privacy by design).

For this product, this means:

The product does not process or store any personal data, only technical function data (e.g. time stamps). If the user links this data with other data (e.g. shift plans) or if he/she stores person-related data on the same data medium (e.g. hard disk), thus personalizing this data, he/she must ensure compliance with the applicable data protection stipulations.

Fundamental safety instructions 2

2.1 General safety instructions

⚠ WARNING

Danger to life if the safety instructions and residual risks are not observed

If the safety instructions and residual risks in the associated hardware documentation are not observed, accidents involving severe injuries or death can occur.

- Observe the safety instructions given in the hardware documentation.
- Consider the residual risks for the risk evaluation.

⚠ WARNING

Malfunctions of the machine as a result of incorrect or changed parameter settings

As a result of incorrect or changed parameterization, machines can malfunction, which in turn can lead to injuries or death.

- Protect the parameterization against unauthorized access.
- Handle possible malfunctions by taking suitable measures, e.g. EMERGENCY STOP or EMERGENCY OFF.

2.2 Warranty and liability for application examples

Application examples are not binding and do not claim to be complete regarding configuration, equipment, or any eventuality which may arise. Application examples do not represent customer-specific solutions, but merely serve to provide assistance with typical tasks.

As the user you yourself are responsible for ensuring that the products described are operated correctly. Application examples do not relieve you of your responsibility for safe handling when using, installing, operating and maintaining the equipment.

2.3 Cybersecurity information

Siemens provides products and solutions with industrial cybersecurity functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial cybersecurity concept. Siemens' products and solutions constitute one element of such a concept.

Customers are responsible for preventing unauthorized access to their plants, systems, machines and networks. Such systems, machines and components should only be connected to an enterprise network or the internet if and to the extent such a connection is necessary and only when appropriate security measures (e.g. firewalls and/or network segmentation) are in place.

For additional information on industrial cybersecurity measures that may be implemented, please visit
https://www.siemens.com/cybersecurity-industry.

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends that product updates are applied as soon as they are available and that the latest product versions are used. Use of product versions that are no longer supported, and failure to apply the latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Cybersecurity RSS Feed under
https://new.siemens.com/cert.

Further information is provided on the Internet:

Configuration Manual Industrial Cybersecurity (https://support.industry.siemens.com/cs/ww/en/view/109975311)

⚠ WARNING

Unsafe operating states resulting from software manipulation

Software manipulations, e.g. viruses, Trojans, or worms, can cause unsafe operating states in your system that may lead to death, serious injury, and property damage.

- Keep the software up to date.
- Incorporate the automation and drive components into a state-of-the-art, integrated industrial cybersecurity concept for the installation or machine.
- Make sure that you include all installed products in the integrated industrial cybersecurity concept.
- Protect files stored on exchangeable storage media from malicious software by with suitable protection measures, e.g. virus scanners.
- Carefully check all cybersecurity-related settings once commissioning has been completed.

Fundamentals

3

3.1 Tool Management - Overview

Tool management (TOOLMAN)

The tool management (TM) function ensures that the right tool is in the right location 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 NC of the current location of the tools and logs the tool movements initiated by the part program, PLC or SINUMERIK Operate.

During tool management commissioning the specific machine system architecture is mapped in the control. 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

The tool management is included in the scope of delivery for all controls.

- The "Replacement tools for tool management" function (6FC5800-0AM78-0YB0) is an option subject to license.
- Function "Monitoring for max. tool speed/acceleration" (6FC5800-0AS08-0YB0) is an option the requires a license.

Replacement tools

If the "Replacement tools for tool management" option is used, replacement tools can be created and used.

Tools with the same name form a tool group and are distinguished by the duplo number. The maximum number of replacement tools of a tool group can be set via MD17500 $MN_MAXNUM_REPLACEMENT_TOOLS. The name of the tool group is programmed in the part program. Generally, the tool with the status "active" is used if the tool group has several tools. If this is blocked, one of the other tools from the same tool group is used.

Further information

A comprehensive description of the tool and magazine parameters and the data structure can be found in:

- SINUMERIK 828D Machine Data (https://support.industry.siemens.com/cs/document/109803036) List Manual
- SINUMERIK 828D NC Variables and Interface Signals (https://support.industry.siemens.com/cs/document/109802877) List Manual
- The user interface of the tool management is described in: SINUMERIK Operate (https://support.industry.siemens.com/cs/document/109803077) Commissioning Manual IM9

See also

SIOS_840Dsl_FH_WZV (https://support.industry.siemens.com/cs/ww/en/view/109763234)

3.2 Structure of the tool management

Function structure

The software components of the control have the following tasks in the tool management:

- **NC:**
  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 optimum 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 (replacement tool) that is not blocked is used automatically, when available.
  - TO area: All data that is required for the tool management (tool and magazine data) is in the NC in the TO area.
  - TO unit (TOA): For each channel, a maximum of one separate TO unit can be set up as subset of the NC TO area. A TO unit represents an independent area of the tool management. There is no connection between the TO units. Several channels can be assigned to one TO unit. However, one channel cannot be assigned to several TO units. Two-channel systems are preset, so that each channel is assigned its own TO unit (MD28085 $MC_MM_LINK_TOA_UNIT).

- **PLC:**
  - Execute tool change
  - Move tools in the magazine
  - Gripper control
  - Magazine control if applicable
  - Safety interlocks
  - Provision of the structure of tool movements in transfer step tables
  - Acknowledgment of the tool movements with acknowledgment step tables
- **SINUMERIK Operate:**
  - Tool data display, input/output
  - Magazine data display, input/output
  - Load/unload relocation dialog

PLC program

The PLC 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 program alone, for example:

- Multiple spindles are using the same magazine.
- The paths of simultaneous pending 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 program
- Communication of PLC program acknowledgments to the tool management
- Transfer of feedback signals for each acknowledgment (acknowledgment incorrect with error number or acknowledgment OK) to the PLC program.
- In addition: Register job status

3.3 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, ...).

Default setting for tool magazines:

The number of magazines which the NCK can manage is specified by the system:

	SW24x	**SW26x**	**SW28x**
Number of tools	128	256	768
Number of magazines	3 (1 real magazine)	3 (1 real magazine)	4 (2 real magazines)
Number of magazine locations	84	84	260
Number of tool cutting edges	256	512	1536

As there must always be a buffer and a loading point, no more than two real magazines can be managed.

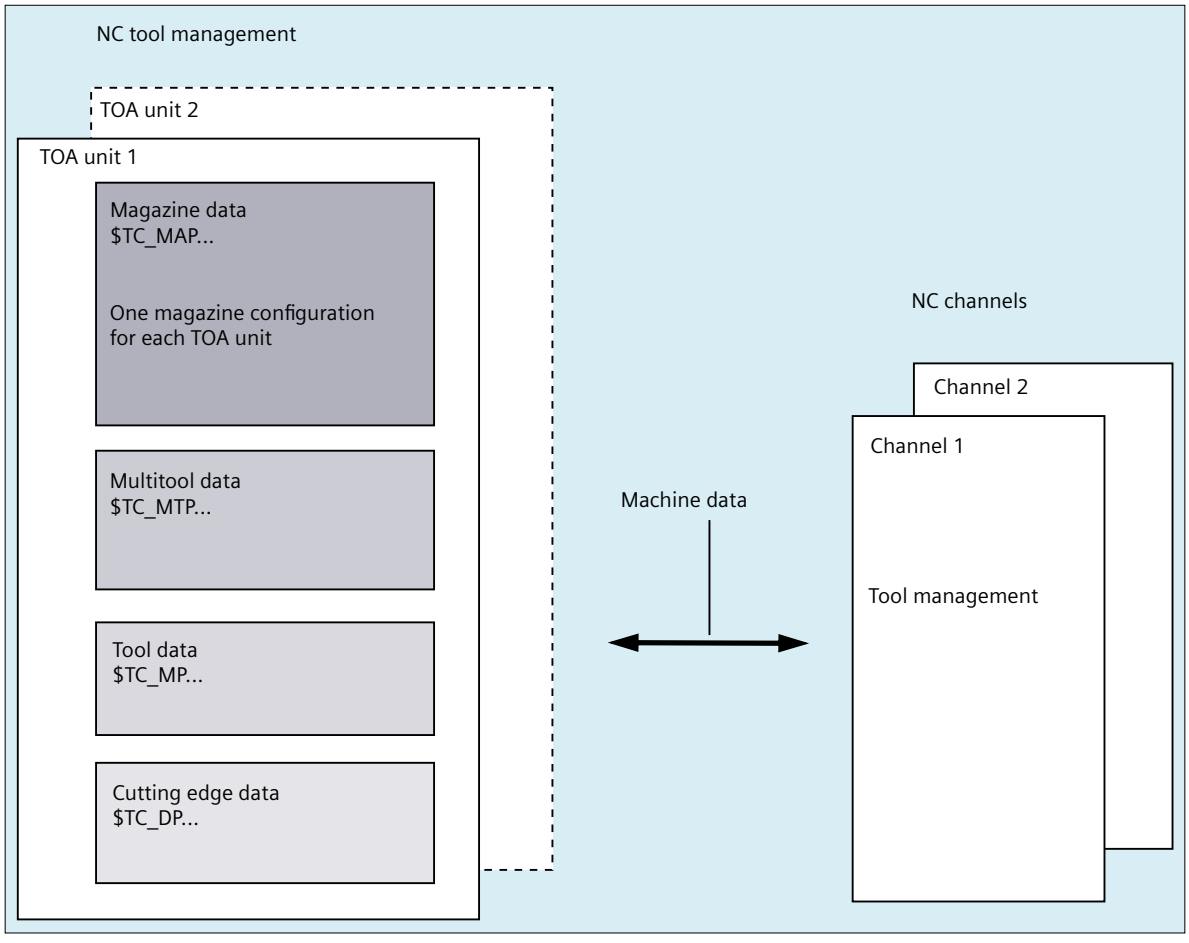


Figure 3-1 Assignment of TO unit to NC channel

Turning technology: Special case for SW28x

Number of magazines = 6, of which 1 real magazine per channel if channel 1 uses TO unit 1 and channel 2 uses TO unit 2 (default setting).

If both channels use TO unit 1 jointly, up to 4 real magazines can be set up.

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, circular, 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 must be 1 for this magazine type.

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	Tool boy
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 is usually 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 turret 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).

3.4 Multitool

Multitool in the turning technology

A multitool is a mini turret with a number of tools that is loaded to a magazine and unloaded from a magazine as one tool in the turning technology. The T selection in the part program also detects and checks the tools of a multitool when searching for a tool.

Brief description:

- Using a multitool you have the possibility of storing more than one tool at a magazine location.
- A multitool has a defined number of locations to accept tools.
- Regarding loading and unloading in a magazine, the multitool is treated as one tool.
- The multitool is considered to be an entity in its own right that is manually equipped, and which is loaded, unloaded or transported as a whole. The equipping of a multitool with tools or the removal of tools from the multitool do not create a tool management request to the PLC interface.
- The multitool knows the parameters "Number of locations" and "Type of clearance determination of the locations".
- The type of clearance for the multitool can be defined via the location number or via the angle of the location.
- Multitool names come from the same name space as the tool names and magazine names.
- Different multitools must have different names, i.e. a duplo number is not defined.

The functionality effects the following interfaces:

- NC programming language (data definition for the tool selection, tool change, tool movements)
- NC variables (data definition and visualization)
- NC/PLC interface

The description of the programming and the multitool data can be found in:
SINUMERIK 840D sl "Tool Management (https://support.industry.siemens.com/cs/de/en/view/109752352)" Function Manual

3.5 Manual tools

Manual tools

Manual tools are tools whose data is completely available in the NC, but are not loaded into the magazine. MD22562 $MC_TOOL_CHANGE_ERROR_MODE bit 1 is used to set as to whether manual tools are permissible in the system. The automatically selected tool must be inserted in the machine manually and removed again manually after machining.

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.

Responsibility of the operator

The machine operator must ensure that:

- The data record of the tool positioned on the spindle is actually in the NC.
- The tool that corresponds to the data record in the NC is placed in the spindle.

⚠ CAUTION
Machine safety Using the PLC program, the machine operator must ensure that safety regulations are complied with.

Unique identification of manual tools

A manual tool is assigned a unique identifier using $TC_TP8[T_No] bit 15 =1. If this state is set, a search is not made for an empty location in a magazine for the tool - and this (old) tool is directly transported to the virtual magazine location 9999/1 when the tool is changed. In the tool change subprogram, detecting the identifier can be performed with a bit-by-bit AND:

IF $TC_TP8[_TOOL_IN_SPINDLE] B_AND 'B1000000000000000' GOTOF HANDTOOL

Setting the "manual tool" state:

The "manual tool" state of a tool is set if the following conditions are fulfilled:

- The machine data is set: MD22562 $MC_TOOL_CHANGE_ERROR_MODE Bit 1 = 1
- As part of a tool change, an unloaded changing point was selected.

Resetting the "manual tool" state:

The "manual tool" state of a tool is reset in the following cases:

- The tool is unloaded, i.e. the tool no longer has a current magazine location. This can especially occur as a result of:
  - Unloading the old tool as part of the tool change.
  - Transporting a manual tool to a loading position (unloading as a result of an operator action at SINUMERIK Operate).
  - The command to select or load a manual tool was interrupted and the tool was still not moved to a magazine location.
- The tool was brought to a location in a real magazine or the loading magazine.

Note

Alarms for manual tools

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 program.

The alarms can be hidden: MD11415 $MN_SUPPRESS_ALARM_MASK_2 Bit 16 = 1

Loading a tool via SINUMERIK Operate

A tool which was, for example, loaded to the spindle via SINUMERIK Operate, is not a manual tool, and as a consequence is placed with T0/M06 in the magazine.

Machine data for the tool management

4

Default setting for machine data

The following machine data for tool management has already been preset or will be set with "NCK default data" during run-up:

Turning technology:

Number	Name	Default setting
MD10717	$MN_T_NO_FCT_CYCLE_NAME	TCHANGE
MD17500	$MN_MAXNUM_REPLACEMENT_TOOLS	1
MD20124	$MC_TOOL_MANAGEMENT_TOOLHOLDER	1
MD20310	$MC_TOOL_MANAGEMENT_MASK	181400F
MD22550	$MC_TOOL_CHANGE_MODE	0
MD22560	$MC_TOOL_CHANGE_MCODE	6
MD22562	$MC_TOOL_CHANGE_ERROR_MODE	0 (for manual tool)

Milling technology:

Number	Name	Default setting
MD10715[0]	$MN_M_NO_FCT_CYCLE	6
MD10716[0]	$MN_M_NO_FCT_CYCLE_NAME	L6
MD17500	$MN_MAXNUM_REPLACEMENT_TOOLS	1
MD20124	$MC_TOOL_MANAGEMENT_TOOLHOLDER	1
MD20310	$MC_TOOL_MANAGEMENT_MASK	180400F
MD22550	$MC_TOOL_CHANGE_MODE	1
MD22560	$MC_TOOL_CHANGE_MCODE	206
MD22562	$MC_TOOL_CHANGE_ERROR_MODE	0 (for manual tool)

MD20270: $MC_CUTTING_EDGE_DEFAULT

Basic setting of tool cutting edge without programming (DWORD): 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	
> 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.

MD20270: $MC_CUTTING_EDGE_DEFAULT	
= -1	The tool edge number of the old tool also applies to the new tool
= -2	The tool edge offset of the old tool remains active until D is programmed.

The setting in MD20270 affects the block preparation of the NC. When the tool change command is called, to prevent a preprocessing stop until the tool change has been acknowledged, perform NC functions without tool offset in the tool change subprogram, for example traversing axes or output of auxiliary functions.

Example: MD20270: $MC_CUTTING_EDGE_DEFAULT= 0 or = -2

After the tool change command M206, the axes can continue traveling 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.

Part program:

```
N10 T="Drill18"           ; Tool change preparation
N20 M6                    ; Tool change subprogram called
Tool change subprogram L6:
N10 D0                    : Compensation deselected
N20 M206                  ; Tool change
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 prep-
                          aration is completed. The main run waits at N50 (D1) until
                          tool change has been executed and acknowledged.
```

MD20310: $MC_TOOL_MANAGEMENT_MASK

Activating the tool management functions:

MD20310: $MC_TOOL_MANAGEMENT_MASK		
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.
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).

MD20310: $MC_TOOL_MANAGEMENT_MASK		
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 a job from the tool management when the "Replacement tools for tool management" option is set.

MD52270: $MCS_TM_FUNCTION_MASK

Tool management function mask

MD52270: $MCS_TM_FUNCTION_MASK	
Bit 0:	Creating a tool at the magazine location is not permitted. Tools can only be created outside 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 tool 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 reactivation, 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. That also applies to the monitoring types that are not set for the tool in question, but are only in the background.
Bit 12:	Hide reactivate tool. The "Reactivate tool" function is hidden in the user interface.

Default setting for turning technology

The following machine data 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.
  A description is provided in Chapter Example: Tool change cycle for turning machine (Page 85)

ISO dialects in turning technology:

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

Further information can be found in: SINUMERIK 828D/840D sl ISO Dialects (https://support.industry.siemens.com/cs/de/de/view/65371498/en) Function Manual

- The description "Substitution by a replacement cycle for a tool selection block" is valid for ISO mode (G291).
- The description "Tool change and tool offsets" is valid when programming the tool offset.

Configuring the magazine

A Power-On Reset is necessary after changing the magazine configuration. The changed configuration is only displayed after restarting the NC.

Examples:

Chapter Configuration of a chain magazine with a dual gripper (Page 91)

Chapter Configuration of a revolver magazine: (Page 79) The program is in the toolbox for SINUMERIK 828D in the "Examples" directory.

NC-PLC interface of the tool management

5

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 SINUMERIK Operate. From these tasks the TM defines the location change needed for the tool and assigns this to the PLC.

The NC/PLC interface provides separate data blocks for loading, unloading, relocating and magazine positioning on the one hand and tool change on the other hand.

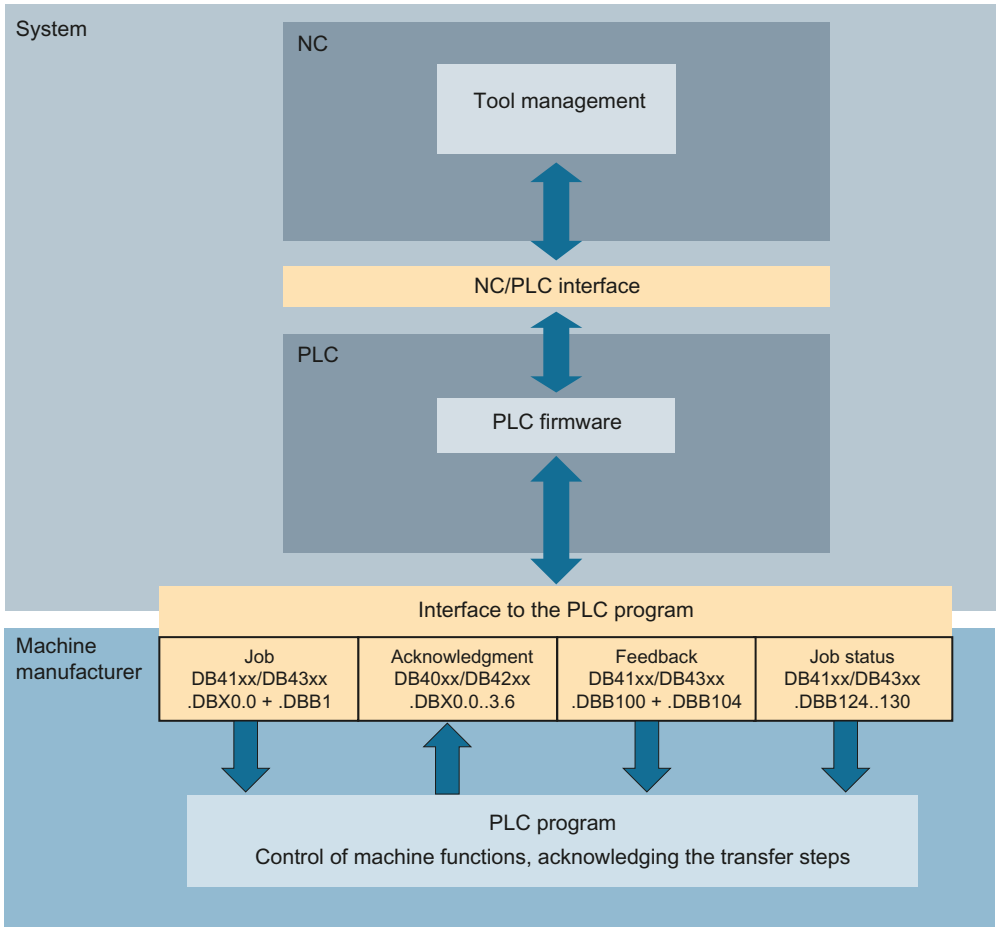


Figure 5-1 Interfaces of the tool management

Loading/unloading and relocating

Loading/unloading and relocating: Tool movement (MVTOOL) or magazine positioning (POSM) from the part program or SINUMERIK Operate

xx: Loading station index

y : Index of the TO unit

- 40xx y000 Acknowledgments from the PLC program for loading/unloading and relocating, magazine positioning
- 41xx y000 jobs to the tool management for loading/unloading and relocating, positioning the magazine
- 41xx y100 feedback from tool management
- 41xx y120 Job status

Tool change

Tool change (T command, M06)

xx: Spindle index

y : Index of the TO unit

- 42xx y000 Prepare and execute acknowledgments from the PLC program for the tool change
- 43xx y000 prepare and execute jobs to tool management for tool change
- 43xx y100 feedback from tool management
- 43xx y120 Job status

Transfer step and acknowledgment step tables:

- 9900 0000 Constant transfer step table (configurable)
- 9901 0000 Variable transfer step table (configurable and modifiable)
- 9902 0000 Acknowledgment step table (configurable)

Example:

There is a separate data area in the respective data blocks for each TO unit:

- Tool holder 1 in TO unit 1: DB4x00.0xxx
- Tool holder 2 in TO unit 1: DB4x01.0xxx
- Tool holder 1 in TO unit 2: DB4x00.1xxx
- Tool holder 2 in TO unit 2: DB4x01.1xxx

5.1 Relocating, unloading, loading tool, positioning magazine

Tool and magazine movements

Per loading point there is one interface for:

- **Jobs** for loading, unloading and relocating a tool (MVTOOL) and for magazine positioning (POSM).
  Message to the PLC 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 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 signal** from the tool management to the PLC program
  Message to the PLC program:
  - Status of acknowledgment
  - 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 PLC program to restart after an abort (e.g. reset during a tool change).

All acknowledgments for a job must take place in the interface of the same loading point.

Interface signal	Meaning
DB40xx.DBXy0.0..3.6	PLC program: Acknowledgments for tool loading/unloading, relocating, or for positioning the magazine
DB40xx.DBXy9.0	PLC program: Resetting the message "Acknowledgment error" (DB41xx.DBX100.1) and the diagnostic information in the feedback interface
DB41xx.DBXy0.0	Tool management: Job for tool loading/unloading, relocating, or for positioning the magazine
DB41xx.DBBy1	Tool management: Job specification
DB41xx.DBWy6..34	Job description
DB41xx.DBXy100.0..1	Positive feedback signal: Acknowledgment status, acknowledgment OK, 1 PLC cycle pending Negative feedback: Acknowledgment status, acknowledgment error, static pending

Interface signal	Meaning
DB41xx.DBBy104	Tool management: Feedback error status
DB41xx.DBXy108.0..111.6	Map of acknowledgments for tool loading, unloading, relocating or positioning the magazine. This map belongs to the positive or negative feedback and remains valid for the same time.
DB41xx.DBWy124..130	Job status

y [= 0 or 1]: TO unit 1 or TO unit 2

xx: Load location

Jobs: Interface description

DB41xx	Signals from tool management [r]							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBBy000	Multitool: Data in the multitool area							Job is still active.
DBBy001		Multitool: Positioning is an NC activity	Multitool: Positioning	Positioning is an NC activity	Positioning	Relocating	Unloading	Loading
DBBy002	Reserved							
DBBy003	Reserved							
DBBy004	Reserved							
DBBy005	Reserved							
DBWy006	Source magazine number (INT)							
DBWy008	Source location number (INT)							
DBWy010	Target magazine number (INT)							
DBWy012	Target location number (INT)							
DBWy014: HMI → PLC								Load/ unload without moving magazine
DBWy016								
...								
DBWy056	Multitool: Type (INT)							
DBWy058	Multitool: Number of locations (INT)							
DBWy060	Multitool: Tool distance (FLOAT)							
DBWy062								
DBWy064	Multitool: Number (INT)							
DBWy066	Multitool: Location number (INT)							
DBWy068	Multitool: Spindle number/tool holder number (INT)							

y [= 0 or 1]: TO unit 1 or TO unit 2

xx: Load location

Jobs: Signal description

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.
- Multitool: Data in the multitool area
  The interface contains information about the multitool specified in the job.
- 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.
- Positioning is an NC activity:
  The positioning job comes from the part program.
- Multitool: Positioning
  The tool in the multitool is moved to the machining position. The multitool remains in the tool holder / spindle.
- Multitool: Positioning is an NC activity:
  The positioning job comes from the part program.
- 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.
- Loading/unloading without moving magazine:
  The signal is set by the HMI when reading in setup data or for the "Unload all tools" function in addition to the loading and unloading jobs. The signal can be used in the PLC program to interlock undesirable magazine movements (e.g. set only total acknowledgment, no further action). The signal is reset by the system with a valid end acknowledgment for the job to the tool management.
- Multitool: Type:
  Type of distance coding: 1 = location number, 2 = length, 3 = angle
- Multitool: Number of locations
  Number of tool locations available in the multitool

- Multitool: Tool distance
  Value for the distance coding: Tool location number in the multitool, or angle or distance of the tool within the multitool
- Multitool: Number
  T number of the multitool
- Multitool: Location number
  Consecutively numbered location in the multitool
- Multitool: Spindle number/tool holder number
  Spindle number / tool holder number associated with the multitool

Acknowledgments: Interface description

DB40xx	**Signals to tool management [r/w]**							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBBy000	Acknowledgment step 7	Acknowledgment step 6	Acknowledgment step 5	Acknowledgment step 4	Acknowledgment step 3	Acknowledgment step 2	Acknowledgment step 1	Total acknowledgment
DBBy001	Acknowledgment step 15	Acknowledgment step 14	Acknowledgment step 13	Acknowledgment step 12	Acknowledgment step 11	Acknowledgment step 10	Acknowledgment step 9	Acknowledgment step 8
DBBy002	Acknowledgment step 23	Acknowledgment step 22	Acknowledgment step 21	Acknowledgment step 20	Acknowledgment step 19	Acknowledgment step 18	Acknowledgment step 17	Acknowledgment step 16
DBBy003	Reserved	Acknowledgment step 30	Acknowledgment step 29	Acknowledgment step 28	Acknowledgment step 27	Acknowledgment step 26	Acknowledgment step 25	Acknowledgment step 24
DBWy004	Multitool: Location number (INT)							
DBBy006	Reserved							
DBBy007	Reserved							
DBBy008	Reserved							
DBBy009							Job acknowledgment	Resetting the acknowledgment error
DBWy010 • • •	Reserved							
DBWy026	Reserved							
DBBy028	Acknowledgment step 38	Acknowledgment step 37	Acknowledgment step 36	Acknowledgment step 35	Acknowledgment step 34	Acknowledgment step 33	Acknowledgment step 32	Acknowledgment step 31
DBBy029	Acknowledgment step 46	Acknowledgment step 45	Acknowledgment step 44	Acknowledgment step 43	Acknowledgment step 42	Acknowledgment step 41	Acknowledgment step 40	Acknowledgment step 39

DB40xx	Signals to tool management [r/w]							
DBBy030	Acknowl-edgment step 54	Acknowl-edgment step 53	Acknowl-edgment step 52	Acknowl-edgment step 51	Acknowl-edgment step 50	Acknowl-edgment step 49	Acknowl-edgment step 48	Acknowl-edgment step 47
DBBy031	Reserved	Reserved	Acknowl-edgment step 60	Acknowl-edgment step 59	Acknowl-edgment step 58	Acknowl-edgment step 57	Acknowl-edgment step 56	Acknowl-edgment step 55

xx: Load location

y [= 0 or 1]: TO unit 1 or TO unit 2

Acknowledgments: Signal description

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...60:
  In the case of a 0/1 edge, the corresponding acknowledgement step from the acknowledgement 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.
- Multitool: Location number
  The PLC program returns the active location number to the NC tool management.
- Resetting the acknowledgment error:
  Resetting the message "Acknowledgment error" (DB41xx.DBX100.1) and the diagnostic information in the feedback interface.
- Job acknowledgment:
  This bit is only effective for complete acknowledgment DB4000.DBX0.0 = 1
  DB4000.DBX9.1 = 0 ⇒ job acknowledgment with status 99
  DB4000.DBX9.1 = 1 ⇒ job acknowledgment with status 3

Feedback signals: Interface description

DB41xx	Signals from tool management [r]							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBBy100							Acknowl-edgment error	Acknowl-edgment OK
DBBy101	Reserved							
DBWy102	Reserved							
DBWy104	Error status (WORD)							
DBBy106	Reserved							

DB41xx	**Signals from tool management [r]**							
DBBy107	Reserved							
DBBy108	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
DBBy109	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
DBBy110	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
DBBy111	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
DBWy112	Multitool: Location number (INT)							
• • •								
DBBy152	Acknowl-edgment step 38	Acknowl-edgment step 37	Acknowl-edgment step 36	Acknowl-edgment step 35	Acknowl-edgment step 34	Acknowl-edgment step 33	Acknowl-edgment step 32	Acknowl-edgment step 31
DBBy153	Acknowl-edgment step 46	Acknowl-edgment step 45	Acknowl-edgment step 44	Acknowl-edgment step 43	Acknowl-edgment step 42	Acknowl-edgment step 41	Acknowl-edgment step 40	Acknowl-edgment step 39
DBBy154	Acknowl-edgment step 54	Acknowl-edgment step 53	Acknowl-edgment step 52	Acknowl-edgment step 51	Acknowl-edgment step 50	Acknowl-edgment step 49	Acknowl-edgment step 48	Acknowl-edgment step 47
DBBy155	Reserved	Reserved	Acknowl-edgment step 60	Acknowl-edgment step 59	Acknowl-edgment step 58	Acknowl-edgment step 57	Acknowl-edgment step 56	Acknowl-edgment step 55

y [= 0 or 1]: TO unit 1 or TO unit 2

xx: Load location

Feedback signals: Signal description

Signal description:

- **Acknowledgment OK (DB41xx.DBXy100.0):**
  The acknowledgment of the PLC program (area DB40xx.DBB0..3) was transferred without error to the tool management. This signal is reset after one PLC cycle.
- **Acknowledgment error (DB41xx.DBXy100.1):**
  Negative feedback acknowledgment status. The acknowledgement of the PLC program (area DB40xx.DBB0..3) is faulty. 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 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 ... 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 manufacturer PLC alarm in the PLC program.
  This signal remains pending statically until the error has been acknowledged (set the bit "reset acknowledgment error" DB40xx.DBX9.0) in the PLC program. The interface in DB40xx.DBB0..3 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.
- **Multitool: Location number**
  Multitool: Location number that the PLC program entered as active location number when acknowledging in the acknowledgment interface.

Error status

When an error occurs, a diagnostics number not equal to zero is displayed in DB41xx.DBBy104 with the following meaning:

Diagnostics number	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 NC caused by this transfer.

The error status is reset to zero when acknowledging the error.

Map of the acknowledgements (DB41xx.DBBy108..111)

The acknowledgments last set by the PLC program (DB40xx.DBB0..3) are set and reset by the PLC firmware together with the bits "Acknowledgment OK" or "Acknowledgment error". If there is an error, these static bits are used to identify which acknowledgment step triggered the error. If several acknowledgment bits are set in the PLC program, then these are also entered into the image.

Job status

DB41xx	Signals from tool management [r]							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBWy120	Reserved							
DBWy122	Reserved							
DBWy124	Current magazine number of the tool (INT)							
DBWy126	Current location number of the tool (INT)							
DBWy128	Target magazine number of the tool (INT)							
DBWy130	Target location number of the tool (INT)							
DBWy132	Reserved							
DBWy134	Reserved							
DBWy136	Reserved							
DBWy138	Reserved							
DBWy140	Multitool type (INT)							
DBWy142	Multitool number of locations (INT)							
DBWy144	Multitool tool distance (FLOAT)							
DBWy146								
DBWy148	Multitool location number (INT)							

y [= 0 or 1]: TO unit 1 or TO unit 2

xx: Load location

5.2 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 program: Job active; job specification and job description.
  The jobs appear in the tool holder interface (of the spindle) where a tool is to be exchanged.
- Acknowledgments of the PLC 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 signal from the tool management to the PLC program
  Message to the PLC 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 PLC 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 NC 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 31).

The main NC 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 tool boy). 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
DB42xx.DBXy0.0..3.6	PLC program: Preparing and carrying out acknowledgments for tool change
DB42xx.DBXy9.0	PLC program: Resetting the message "Acknowledgment error" (DB43xx.DBX100.1) and the diagnostic information in the feedback interface
DB43xx.DBXy0.0	Tool management: Job for "Prepare tool change" and "Execute tool change"
DB43xx.DBBy1	Tool management job specification
DB43xx.DBWy6..34	Job description
DB43xx.DBXy100.0..1	Positive feedback acknowledgment status, acknowledgment OK, 1 PLC cycle pending Negative feedback acknowledgment status, acknowledgment error, static pending
DB43xx.DBXy100.0..1	Positive feedback acknowledgment status, 1 PLC cycle pending Negative feedback: Acknowledgment status, static pending.
DB43xx.DBXy100.0	Tool management feedback: Acknowledgment status
DB43xx.DBBy104	Tool management feedback error status

Interface signal	Meaning
DB43xx.DBXy108.0..111.6	Map of acknowledgments for tool change: This map belongs to the positive or negative feedback and remains valid for the same time.
DB43xx.DBWy124..138	Job status

y [= 0 or 1]: TO unit 1 or TO unit 2

xx: Index of the spindle/tool holder

Jobs: Interface description

DB43xx	Signals from tool management [r]							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBBy0	Multitool: Data in the multitool area	--	--	--	--	--	--	Job is still active.
DBBy1	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
DBBy2	Reserved							
DBBy3	Reserved							
DBBy4	Reserved							
DBBy5	Reserved							
DBWy6	Source magazine number for new tool (INT)							
DBWy8	Source location number for new tool (INT)							
DBWy10	Reserved							
DBWy12	Reserved							
DBWy14	Reserved							
DBWy16	Reserved							
DBWy18	Target magazine number for old tool (INT)							
DBWy20	Target location number for old tool (INT)							
DBWy22	Location type (INT)							
DBWy24	Size left (INT)							
DBWy26	Size right (INT)							
DBWy28	Reserved							
DBWy30	Reserved							
DBBy32	Tool status for a new tool:							
	Tool has been in use	Tool fixed-location coded	Tool being changed	Prewarning limit reached	Measuring tools	Tool disabled	Tool enabled	Active tool
DBBy33	Tool status for a new tool:							
	Manual tool	1:1 change	--	Master tool	To be loaded	To be unloaded	Disabled	Identification for tools in the buffer
DBWy34	New tool: Internal T number of the NC (INT)							
DBWy36	Reserved							

DB43xx	Signals from tool management [r]
DBWy38	Reserved
DBWy40	Reserved
DBWy42	Reserved
DBWy44	Free parameter 1 (DWORD)
DBWy46	
DBWy48	Free parameter 2 (DWORD)
DBWy50	
DBWy52	Free parameter 3 (DWORD)
DBWy54	
DBWy56	Multitool: Type (INT)
DBWy58	Multitool: Number of locations (INT)
DBWy60	Multitool: Tool distance (FLOAT)
DBWy62	
DBWy64	Multitool: Number (source) of new tool (INT)
DBWy66	Multitool: Location number (source) of new tool (INT)
DBWy68	Multitool: Number (target) of old tool (INT)
DBWy70	Multitool: Location number (target) of the old tool (INT)
DBWy72	Location type (INT)
DBWy74	Size left (INT)
DBWy76	Size right (INT)
DBWy78	Size top (INT)
DBWy80	Size bottom (INT)
DBWy82	Tool status for new tool (INT)
DBWy84	T number of tool (INT)
DBWy86	Spindle number/tool holder number (INT)
DBWy88	Multitool: original magazine number (INT)
DBWy90	Multitool: original location number (INT)

y [= 0 or 1]: TO unit 1 or TO unit 2

xx: Tool holder

Jobs: Signal description

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.
- Multitool: Data in the multitool area
  The interface contains information about the multitool specified in the job.
- 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. SINUMERIK Operate 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) of the new tool
- User-definable parameters: 3 user-definable parameters, which are transferred by the part program to the PLC program.
- Multitool: Type:
  Type of distance coding: 1 = location number, 2 = length, 3 = angle
- Multitool: Number of locations
  Number of tool locations available in the multitool
- Multitool: Tool distance
  Value for the distance coding: Tool location number in the multitool, or angle or distance of the tool within the multitool
- Multitool: Number (source)/Multitool: location number (source) of the new tool
  Magazine number/location number from where the new multitool comes (usually a location in a real magazine).
- Multitool: Number (target)/Multitool: location number (target) of the old tool
  Magazine number/location number to where the old multitool is to be transported (usually a location in a real magazine).
- Location type: Location type of the new tool
- Size left: space required for the multitool in the magazine at the left

- Size right: space required for the multitool in the magazine at the right
- Size top: space required for the multitool in the magazine at the top
- Size bottom: space required for the multitool in the magazine at the top
- Tool status
- T number of the tool
- Spindle number/tool holder number
- Multitool: original magazine number
- Multitool: original location number

Acknowledgments: Interface description

DB42xx	Signals to tool management [r/w]							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBBy0	Acknowledgment step 7	Acknowledgment step 6	Acknowledgment step 5	Acknowledgment step 4	Acknowledgment step 3	Acknowledgment step 2	Acknowledgment step 1	Total acknowledgment
DBBy1	Acknowledgment step 15	Acknowledgment step 14	Acknowledgment step 13	Acknowledgment step 12	Acknowledgment step 11	Acknowledgment step 10	Acknowledgment step 9	Acknowledgment step 8
DBBy2	Acknowledgment step 23	Acknowledgment step 22	Acknowledgment step 21	Acknowledgment step 20	Acknowledgment step 19	Acknowledgment step 18	Acknowledgment step 17	Acknowledgment step 16
DBBy3	Reserved	Acknowledgment step 30	Acknowledgment step 29	Acknowledgment step 28	Acknowledgment step 27	Acknowledgment step 26	Acknowledgment step 25	Acknowledgment step 24
DBWy4	Multitool: Location number (INT)							
DBBy6	Reserved							
DBBy7	Reserved							
DBBy8	Reserved							
DBBy9							Job acknowledgment	Resetting the acknowledgment error
DBWy10 • • •	Reserved							
DBWy26	Reserved							
DBBy28	Acknowledgment step 38	Acknowledgment step 37	Acknowledgment step 36	Acknowledgment step 35	Acknowledgment step 34	Acknowledgment step 33	Acknowledgment step 32	Acknowledgment step 31
DBBy29	Acknowledgment step 46	Acknowledgment step 45	Acknowledgment step 44	Acknowledgment step 43	Acknowledgment step 42	Acknowledgment step 41	Acknowledgment step 40	Acknowledgment step 39

DB42xx	Signals to tool management [r/w]							
DBBy30	Acknowl-edgment step 54	Acknowl-edgment step 53	Acknowl-edgment step 52	Acknowl-edgment step 51	Acknowl-edgment step 50	Acknowl-edgment step 49	Acknowl-edgment step 48	Acknowledg-ment step 47
DBBy31	Reserved	Reserved	Acknowl-edgment step 60	Acknowl-edgment step 59	Acknowl-edgment step 58	Acknowl-edgment step 57	Acknowl-edgment step 56	Acknowledg-ment step 55

xx: Load location

y [= 0 or 1]: TO unit 1 or TO unit 2

Acknowledgments: Signal description

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 ... 60:
  In the case of a 0/1 edge, the corresponding acknowledgement step from the acknowledgement 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.
- Multitool: Location number
  The PLC program returns the active location number to the NC tool management.
- Resetting the acknowledgment error:
  Resetting the message Acknowledgment error (DB43xx.DBX100.1) and the diagnostic information in the feedback interface.
- Job acknowledgment:
  This bit is only effective for complete acknowledgment DB4200.DBX0.0 = 1
  DB4200.DBX9.1 = 0 ⇒ job acknowledgment with status 99
  DB4200.DBX9.1 = 1 ⇒ job acknowledgment with status 3

Feedback signals: Interface description

DB43xx	Signals from tool management [r]							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBBy100							Acknowl-edgment error	Acknowl-edgment OK
DBBy101	Reserved							
DBWy102	Reserved							
DBWy104	Error status (WORD)							
DBWy106	Reserved							

DB43xx	Signals from tool management [r]							
DBBy108	Acknowledg-ment 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
DBBy109	Acknowledg-ment 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
DBBy110	Acknowledg-ment 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
DBBy111	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
DBBy112	Multitool: Location number (INT)							
• • •								
DBBy152	Acknowledg-ment step 38	Acknowl-edgment step 37	Acknowl-edgment step 36	Acknowl-edgment step 35	Acknowl-edgment step 34	Acknowl-edgment step 33	Acknowl-edgment step 32	Acknowl-edgment step 31
DBBy153	Acknowledg-ment step 46	Acknowl-edgment step 45	Acknowl-edgment step 44	Acknowl-edgment step 43	Acknowl-edgment step 42	Acknowl-edgment step 41	Acknowl-edgment step 40	Acknowl-edgment step 39
DBBy154	Acknowledg-ment step 54	Acknowl-edgment step 53	Acknowl-edgment step 52	Acknowl-edgment step 51	Acknowl-edgment step 50	Acknowl-edgment step 49	Acknowl-edgment step 48	Acknowl-edgment step 47
DBBy155	Reserved	Reserved	Acknowl-edgment step 60	Acknowl-edgment step 59	Acknowl-edgment step 58	Acknowl-edgment step 57	Acknowl-edgment step 56	Acknowl-edgment step 55

y [= 0 or 1]: TO unit 1 or TO unit 2

xx: Tool holder

Feedback signals: Signal description

Signal description:

- **Acknowledgment OK (DB43xx.DBX100.0): Positive feedback of acknowledgment status.**
  The acknowledgment of the PLC program (area DB42xx.DBB0..3) was transferred without error to the tool management. This signal is reset after one PLC cycle.
- **Acknowledgment error (DB43xx.DBX100.1): Negative feedback acknowledgment status.**
  The acknowledgement of the PLC program (area DB42xx.DBB0..3) is faulty. 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 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 ... 7), the error is only output on the interface of the tool management and not by the NC (no NC alarm!).
  When required, the machine OEM signals this error by the PLC program initiating a PLC alarm.
  This signal remain static until the error is acknowledged (by setting the bit "Reset acknowledgment error" DB4200.DBX9.0 by the manufacturer). The NC/PLC interface in DB42xx.DBB0..3 is disabled if bit "acknowledgment error" is active. Incoming acknowledgment bits are not evaluated by the PLC firmware and are also cleared when bit "Reset acknowledgment error" is set.
- **Multitool: Location number**
  Multitool: Location number that the PLC program entered as active location number when acknowledging in the acknowledgment interface.

Error status

When an error occurs, a diagnostics number not equal to zero is displayed in DB43xx.DBB104 with the following meaning:

Diagnostics number	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 NC caused by this transfer.

The error status is reset when acknowledging the error.

Map of the acknowledgements (DB43xx.DBB108..111)

The acknowledgments last set by the PLC program (DB42xx.DBB0..3) are set and reset by the PLC firmware together with the bits "Acknowledgment OK" or "Acknowledgment error". If there is an error, these static bits are used to identify which acknowledgment step triggered the error. If several acknowledgment bits are set in the PLC program, then these are also entered into the image.

Job status

DB43xx	Signals from tool management [r]							
Byte	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
DBWy120	Reserved							
DBWy122	Reserved							
DBWy124	Current magazine number for new tool (INT)							
DBWy126	Current location number for new tool (INT)							
DBWy128	Target magazine number for new tool (INT)							
DBWy130	Target location number for new tool (INT)							
DBWy132	Current magazine number for old tool (INT)							
DBWy134	Current location number for old tool (INT)							
DBWy136	Target magazine number for old tool (INT)							
DBWy138	Target location number for old tool (INT)							
DBWy140	Multitool: Type (INT)							
DBWy142	Multitool: Number of locations (INT)							
DBWy144	Multitool: Tool distance (FLOAT)							
DBWy146								
DBWy148	Multitool: Location number (INT)							

y [= 0 or 1]: TO unit 1 or TO unit 2

xx: Tool holder

5.3 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 the PLC 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, and must be copied to the PLC project and processed.

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). DB9901 can be changed by the PLC program; for example, for acknowledging intermediate steps such as 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							
	...							
DBB236	Acknowledgment step 60 transfer step for new tool (BYTE)							
DBB237	Acknowledgment step 60 transfer step for old tool (BYTE)							
DBB238	Acknowledgment step 60 acknowledgment status (BYTE)							
DBB239	Acknowledgment step 60 reserved							

See also

PLC Program Blocks (Page 57)

PLC Program Blocks

6

6.1 Acknowledgment process

Information to tool management

The tool management expects acknowledgment of its jobs in order to track and manage actual tool positions. At least one acknowledgment is required for each job. 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.DBX0.0/42xx.DBX0.0) after a completely finished tool management job 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 transferred to the tool management. Under certain circumstances the reset may take place after several PLC cycles.

There are additional 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 the power up strategy 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. For jobs, 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 final acknowledgment (e.g. "magazine positioning" job complete). If the magazine is positioned by the PLC program itself (e.g. using machine keys) without a tool management job, this must be communicated to the tool management using asynchronous signals.

6.2 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.

6.3 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.
	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 31).
	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	7	99	105	201	204
New tool	Transfer step	x	-	-	-	x	x	x
	• from magazine	-	-	-	-	-	xx	xx
	• from location	-	-	-	-	-	xx	xx
	• to magazine	xx	-	-	-	xx	xx	zz
	• to location	xx	-	-	-	xx	xx	zz

Acknowledgment status		1	3	7	99	105	201	204
Old tool	Transfer step	x	-	-	-	x	x	-
	• from magazine	-	-	-	-	-	xx	-
	• from location	-	-	-	-	-	xx	-
	• to magazine	xx	-	-	-	xx	xx	-
	• to location	xx	-	-	-	xx	xx	-

Legend:

- Data item not relevant

x : Number (1...n) of the transfer step from the transfer-step table

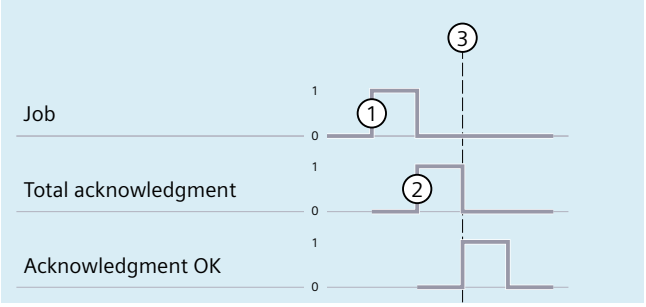
xx: Magazine number, location number of the tool

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.
- States 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



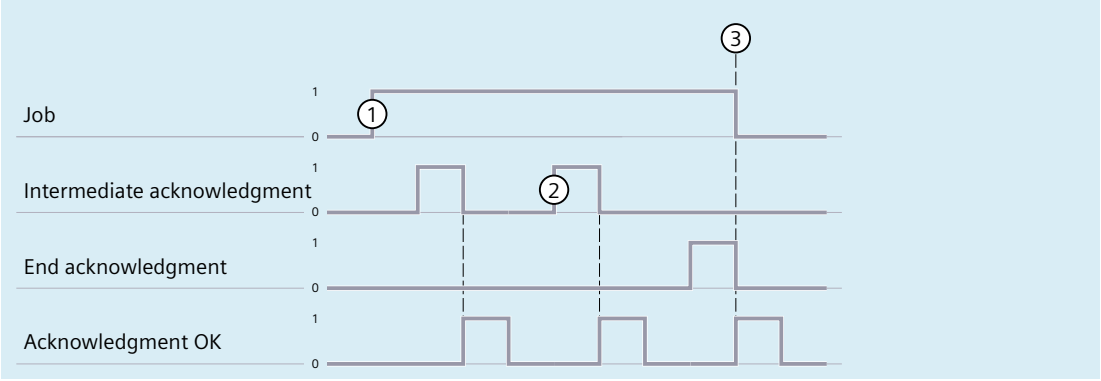
Signal description:

① The PLC 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 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 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.

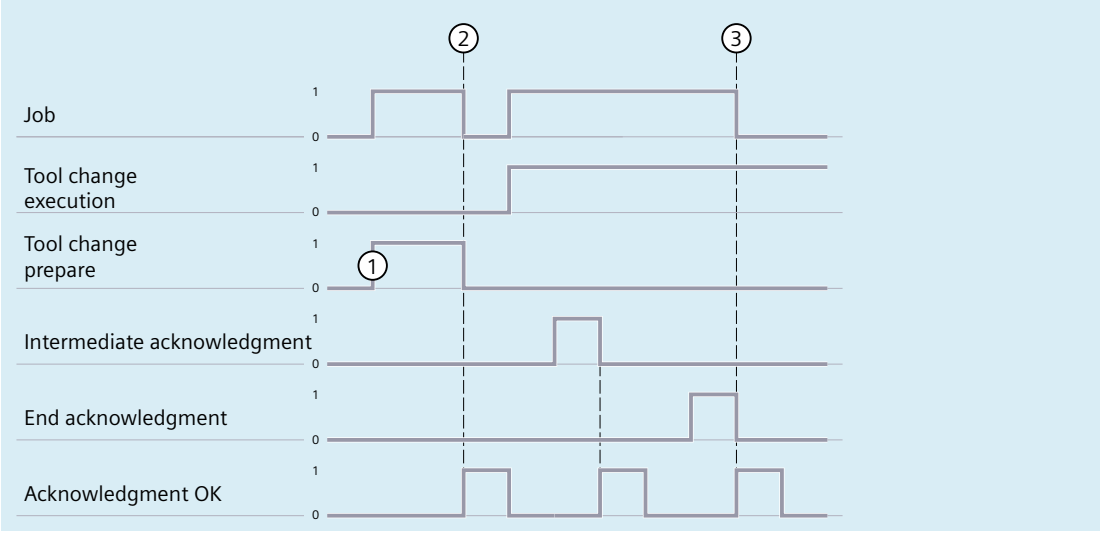
② The PLC program acknowledges the transfer steps configured in DB9900, DB9901, and DB9902 with an acknowledgment status 105 or as an asynchronous transfer. The positions of the tools are updated using the transfer steps of the acknowledgments from the tool management.

③ The execution of the job is acknowledged by the PLC 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 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 influenced by the setting in MD20270: $MC_CUTTING_EDGE_DEFAULT and MD20310: $MC_TOOL_MANAGEMENT_MASK bits 5, 6, 7 and 8.

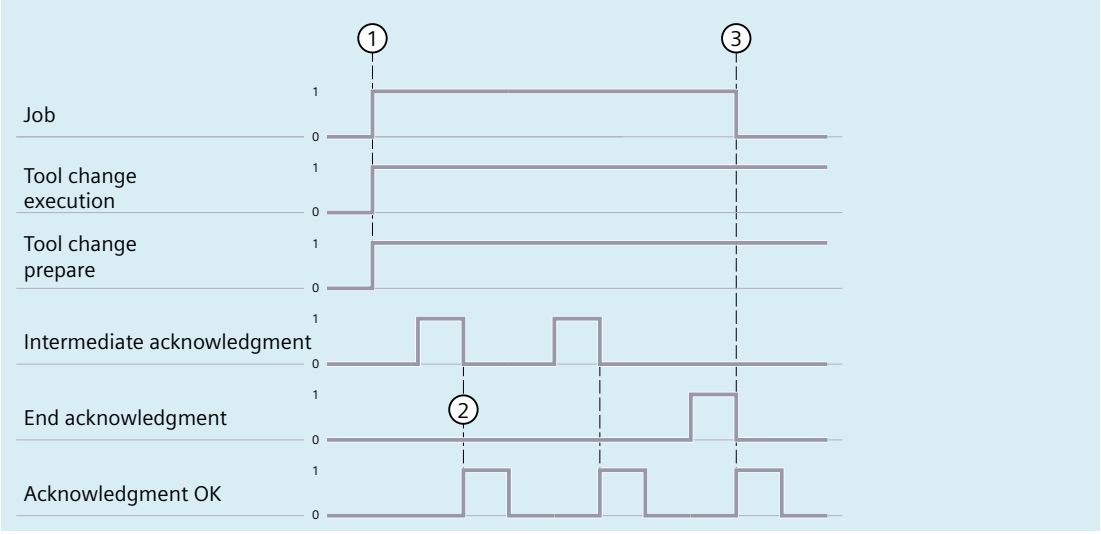
This sequence described here corresponds to the presetting of the machine data.

Sequence of the tool management without block splitting (machine data setting milling)

```
Txx Myy ;
```

Myy is the setting from MD22560: MD22560: $MC_TOOL_CHANGE_M_CODE

This type of programming is not recommended!



Signal description:

① The PLC 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.

6.4 Configuring step tables

Configuring step tables

The transfer step tables (TM_CTS in the DB9900 and TM_VTS in DB9901) and the acknowledgment step table (TM_ACK in DB9902) are in the programming 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 data blocks are not yet filled with the necessary data. The user must edit/process them in the programming tool using the menu command "View" → "Data block". The constant tables (TM_CTS in DB9900 and TM_ACK in DB9902) are configured by writing the initial data block values in the programming tool.

The initial data block values are loaded into the control together 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 program. A table of all mechanical single movements that should be acknowledged supports 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 or 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 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	of		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	of		to		Comment
	Maga-zine	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 program.
102	1	0	9998	2	"Prepare tool change": Tool from maga-zine to buffer. The source position must be entered by the PLC program.

6.5 Configuring acknowledgment steps

Configuring the acknowledgment steps

The PLC provides 31 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 exception of the special case acknowledgment step 9: Total acknowledgment) is stored in the 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

Acknowl-edgment step	Address DB9902	Name	Initial 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 the NC
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 the NC

Example as a complete step

Acknowledgment step	Transfer step		Acknowledgment status	Comment
	New tool	old tool		
1	0	0	3	"Cancel job" 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 in which the acknowledgement was made, the feedback for a PLC cycle is output in bit 100.0 (acknowledgement OK) or as static signal bit 100.1 (acknowledgement faulty), in the case of faulty acknowledgement the error status in byte 104 and the last set acknowledgement bits in bytes DBB108...111. If there is an error, these static bits are used to identify which acknowledgment step triggered the error. If several acknowledgment bits are set in the PLC program, then these are also entered into the image. The error status is reset when the user acknowledges the error in DB40xx.DBX9.0 or DB42xx.DBX9.0.

6.6 Adapting the PLC program

Adapting the PLC 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 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 turret magazine and a milling machine with a chain magazine and dual gripper:

- Application example for milling machine (Page 91)
- Application example for turning machine (Page 79)

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 NC/PLC interface, without previous intermediate acknowledgment.

For example:

- Rotations of a turret 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 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 a pending job.
- Only valid transfer step numbers may be used (1 ... 64, 100 ... 164). For asynchronous messages, at least one transfer step must be specified for status 201, for status 204 a transfer step for the new tool must be specified.
- 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 program. After setting an acknowledgment bit, the NC/PLC interface must not be changed until the feedback message in DB41xx.DBB100/DB43xx.DBB100!
- Asynchronous messages with two transfer steps must be acknowledged at the tool change interface (DB42xx).

6.7 Information on magazine location

Overview

It is possible to read up to eight NC variables in one job via the "NC services" interface of DB1200.

Variable $TC_MPP2 with index 7: Location type

Parameter assignment:

Read variable from the NC	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
	DB120x.DBB3001	Access result	0/3/5/10
	DB120x.DBW3004	Data from NC variable	n

Legend:

n > 0: Location type for virtual location

n = 0: "match all" (buffer)

n = 9999: undefined (no virtual location)

Variable $TC_MPP4 with index 8: Location state

Parameter assignment:

Read variable from the NC	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

Read variable from the NC	Address	Signal	Valid values
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 NC variable	n

Legend:

n = 1 blocked

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 top half location

n = 128 occupied in bottom half location

Variable $TC_MPP6 with index 9: T number of the tool at this location

Parameter assignment:

Read variable from the NC	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 NC 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

6.8 PI service: TMMVTL

Function

With program instance 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

Start PI service in the NC 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.

Examples:

- When using buffers to return the tool (for example tool boy 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.

Example: Loading/unloading

7

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 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:

Acknowledg-ment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
xxx	DB4000.DBX0.0	--	--	(99)

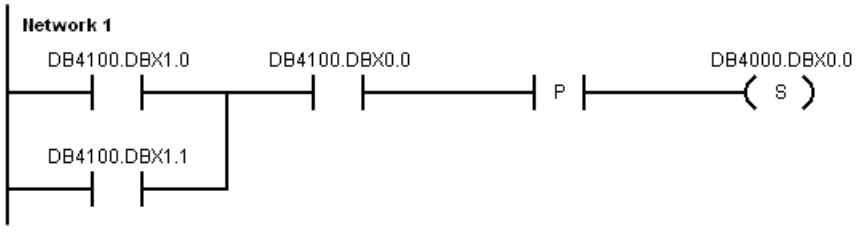


Figure 7-1 Programming in the PLC program

Loading procedures using handling systems or transporting the tool from the spindle into the magazine can be performed using further asynchronous messages.

Note

Manual tools

Please observe the special issues when using manual tools (Page 27).

A tool that was loaded into the spindle or a magazine, is not a manual tool. As a consequence, it is placed in the magazine using T0/M06.

Additional loading processes are described in the following sections.

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 the tool, which is not yet located in the magazine, into the spindle using 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 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.

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.

Example: Change manual tools

8

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 set for the tool on the spindle is in the NCK, or that he puts the appropriate tool onto the spindle for the data set stored in the NC.

NOTICE
Safety regulations for manual tools Maintaining safety regulations must be secured in the PLC program.

The PLC program is informed with DB43xx.DBX1.5..6 whether a manual tool is involved in a tool change job. The operator receives a prompt to carry out the tool change:

17212 "Channel %1: Tool management: Load manual tool %3, duplo No. %2 to tool holder %4"

17214 "Channel %1: Tool management: Remove manual tool %3 from spindle / tool holder %2"

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 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":

Acknowledg-ment 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, .DBX1.1, .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 program (tool change):

DB4300.DBX0.0, .DBX1.1 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, .DBX1.1 and .DBX1.6 ("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 of 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

Acknowledg-ment 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)

Application example for turning machine

9

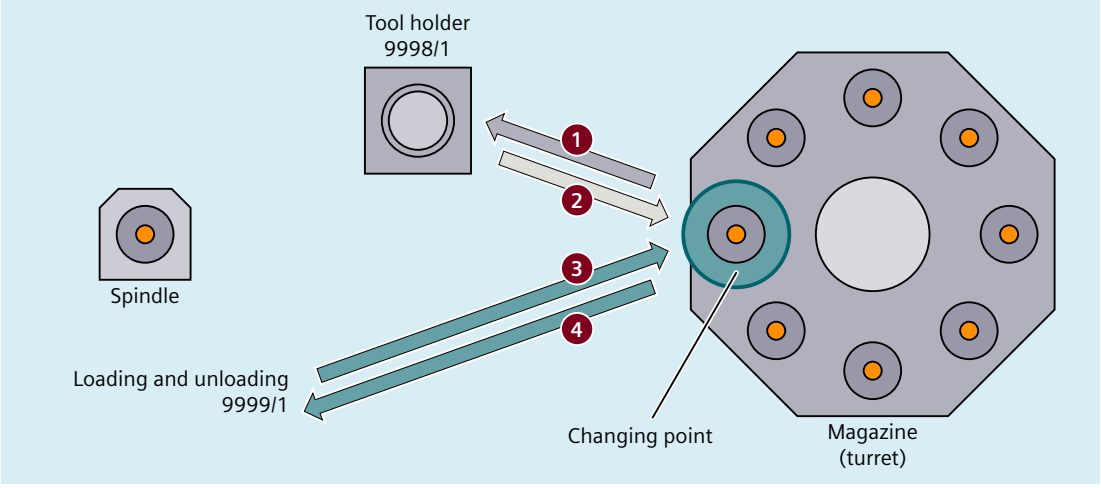
9.1 Example: Turning machine with revolver magazine (MAGKONF_MPF)

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 9-1 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: revolv-
                                               er, 5: box magazine)
N40 LOCATIONS = 8                              ;Number of magazine locations
N50 NUM_BUFFER = 1                             ;Number of buffers (spindle, grip-
                                               per)
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 MAGA-
ZINE"<<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

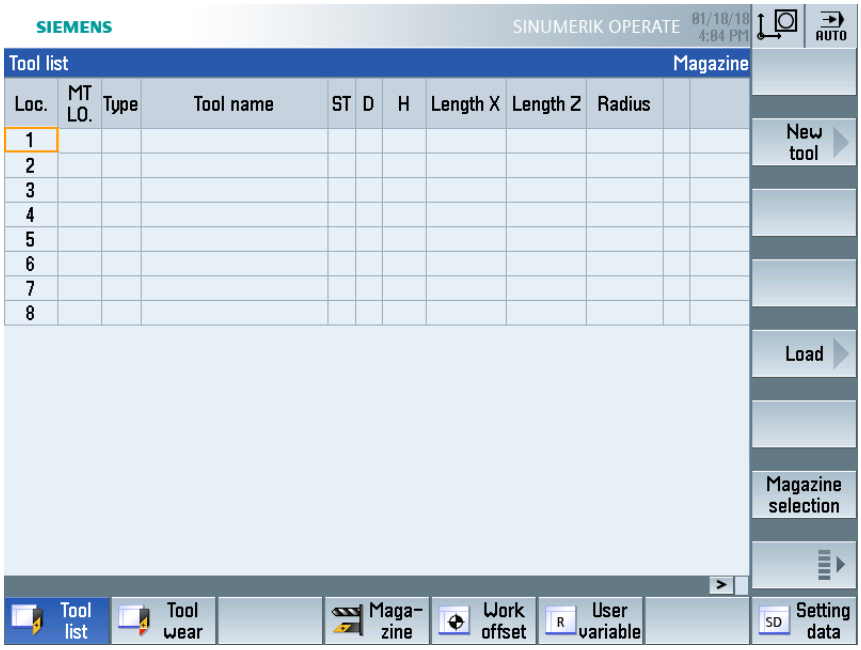


Figure 9-2 Turning machine tool list

9.2 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 program must detect incoming jobs in the job interfaces and execute the mechanical movements.

Total acknowledgment to tool management:

Acknowledg-ment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
--	DB4000.DBX0.0	--	--	(99)
--	DB4200.DBX0.0	--	--	(99)

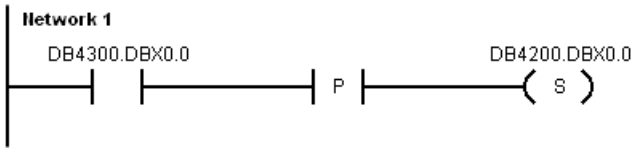


Figure 9-3 Programming in the PLC program

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 program in the variable transfer table.

9.3 Example: Tool change cycle for turning machine

Transfer variables of the T replacement cycle:

Preconditions:

- The "Replacement tools for tool management" option is not active.
- Tool management has been informed of each magazine movement.

In MD10717 $MN_T_NO_FCT_CYCLE_NAME (name of cycle to be executed instead of the T function, e.g. "TCHANGE"), you specify that a cycle is run through 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 turret can be set on the programmed tool in this cycle (POSM). This is necessary when the turret 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 "Change tool" to the NC/PLC interface.

Programming

In the following programming 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 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 (i.e. the 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
```

Tool call for tool measuring

For special applications, such as measuring in JOG, it is necessary to load a specific tool (e.g. miller_15, duplo number 2) - independent of its status - to the spindle or the tool holder; this can be the case, for example, if a tool was being used, but was locked by the tool monitoring function. Now, before being released again, it should be measured. The NC command TCA is available to call a tool independent of the status and by selecting duplo number and tool holder number; this is used when measuring tools in JOG. The NC command TCA ("Tool name", duplo number, tool holder number) behaves like the T command with respect to alarm and command output to the PLC.

Constraints

The following supplementary conditions must be observed for an active T function replacement (default setting for turning):

- The cycle defined in MD10717 $MN_T_NO_FCT_CYCLE_NAME and executed with the T call, is not started with the TCA command.
  To ensure that a tool change cycle can also be started when TCA is programmed, the language command TCA must be redefined in _TCA (default setting NC).
- TCA can be substituted with a replacement cycle. The name of the replacement cycle is entered in MD15710 $MN_TCA_CYCLE_NAME. This replacement cycle is called from the cycle of the redefined TCA function (TCA.SPF from the toolbox).
- The T and TCA replacement can be implemented in a common replacement cycle.

Note

The offset selection, in accordance with 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.

There is a sample cycle in the toolbox for the replacement of both tool calls in a common replacement cycle.

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
```

The function is available irrespective of tool management; refer to the Chapter "Replacement of functions with subprograms" in the Basic Functions (K1) (https://support.industry.siemens.com/cs/document/109803078) Function Manual.

9.4 Example: Block search and MCALL

Requirement

In the tool change subprogram, you must ensure for the block search via modally called cycles (e.g. drilling cycles) that the modal call is retained and ensure that the correct tool is used.

Further information

SINUMERIK Operate (https://support.industry.siemens.com/cs/document/109803077) Commissioning Manual, "Technology cycles for drilling"

After a block search with calculation, the tool change subprogram (L6) is called in the block search ASUB. The modal call may only be deselected there when L6 has not been called from the ASUB.

Programming example

```
L6
N100 IF $P_MC == TRUE   ; MCALL is active
N110   IF $AC_ASUP <> 0  ; Call is performed via search ASUB or
overstore → do not select MCALL
N140   ELSE
N150    MCALL
N160   ENDIF
```

```
N170 ENDIF
```

The machine manufacturer must also ensure that a real tool change with axis movements and a modal cycle call are permitted. Under certain circumstances, the tool change must be prevented when the programmed tool after a block change is not the same as the tool in the spindle.

9.5 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 PLC 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.

9.6 Example: Test for empty buffer

Procedure

Read the T number of a tool in gripper 1 and 2:

1. In the PLC program, enter the parameters in DB1200.
2. In DB1200.DBX0.0, set the start for reading the location states.

Once the job has been successfully executed, the results are entered starting from DB1200.DBB3000:

Read variable from the NC	Address	Signal	Values
Parameter	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
Job	DB1200.DBX0.1	Write variable	0
	DB1200.DBX0.2	PI service	0
	DB1200.DBB1	Number of variables	2
	DB1200.DBX0.0	Start	→ 1

Read variable from the NC	Address	Signal	Values
Result	DB1200.DBB2000.0	Job completed.	1
	DB1200.DBX2000.1	Error in job	0
	DB1200.DBX3000.0	Variable is valid	1
	DB1200.DBB3001	Access result	0
	DB1200.DBW3004	Data from NC variable	n
	DB1201.DBX3000.0	Variable is valid	1
	DB1201.DBB3001	Access result	n
	DB1201.DBW3004	Data from NC variable	0

9.7 Example: Transporting a tool from a buffer into the magazine

Procedure

A tool is to be moved from a buffer (e.g. 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 a job to relocate the tool is generated.

In the PLC program, enter the parameters in DB1200, and in DB1200.DBX4000.0, set the start signal of the PI service.

Start PI services in the NC 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 a job to relocate the tool from the buffer. The target location in magazine 1 is in DB4100.DBW12. The PLC program can therefore execute the necessary sequence.

9.8 Example: Repeat "Prepare tool change" order

Sequence: Repeat the job

For a milling machine with toolboy and shifter, the asynchronous transfer message and job 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 job 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 program transferred the tool from the toolboy to the shifter.
- The PLC program communicates the tool movement from the toolboy to the shifter asynchronously and starts onward transport to the magazine.
- In the next cycle the PLC program recognizes the new job to relocate the tool from the toolboy to the spindle.

There is no longer any tool in the toolboy. The PLC 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 job (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.
- The tool management generates a new preparation job with the source position shifter for the new tool.

Note

The acknowledgment to repeat the job "Prepare tool change" may only be issued **before** the end acknowledgment of the "Prepare tool change" job!

Acknowledgment to tool management:

Acknowledg-ment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
xxx	DB4000.DBXx.x	0	0	7

Application example for milling machine

10

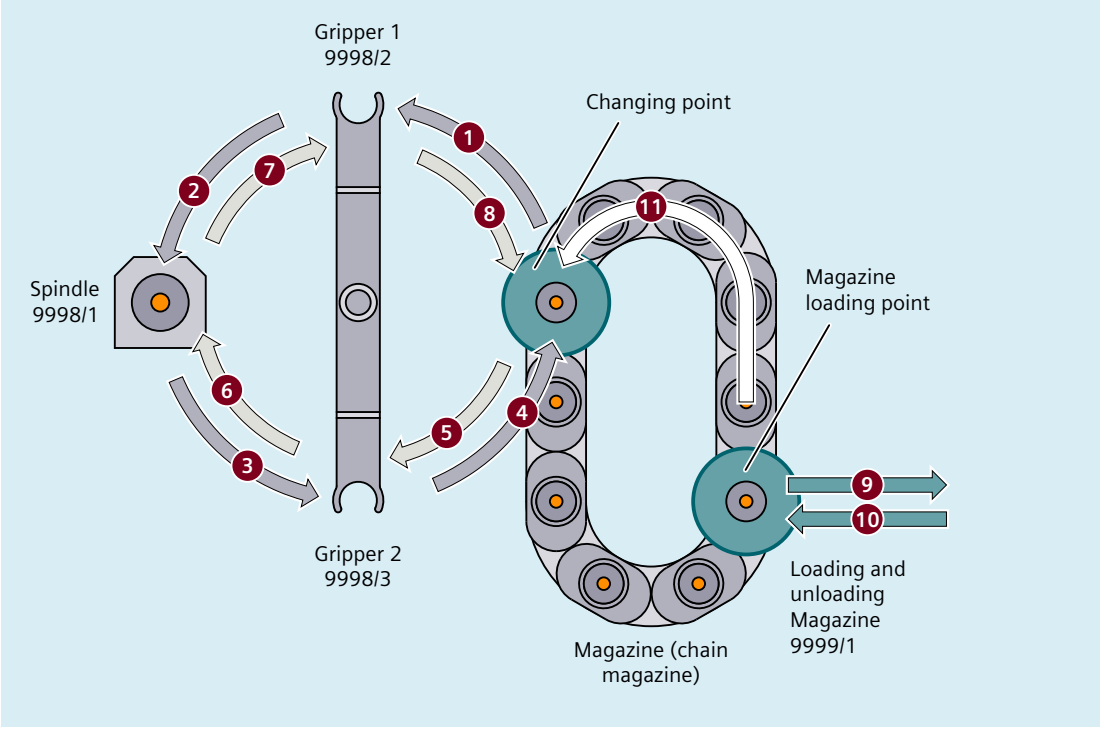
10.1 Example: Milling machine with chain magazine and dual gripper (MAGKONF_EXAMPLE_MPF)

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 10-1 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

Magazine	Location	Meaning
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. Reset (po).

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 79)
- 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: Turret,
                                                5: Flat magazine)
N40 LOCATIONS = 8                               ;Number of magazine locations
N50 NUM_BUFFER = 3                              ;Number of buffers (spindle, grip-
                                                per)
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 MAGA-
ZINE"<<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

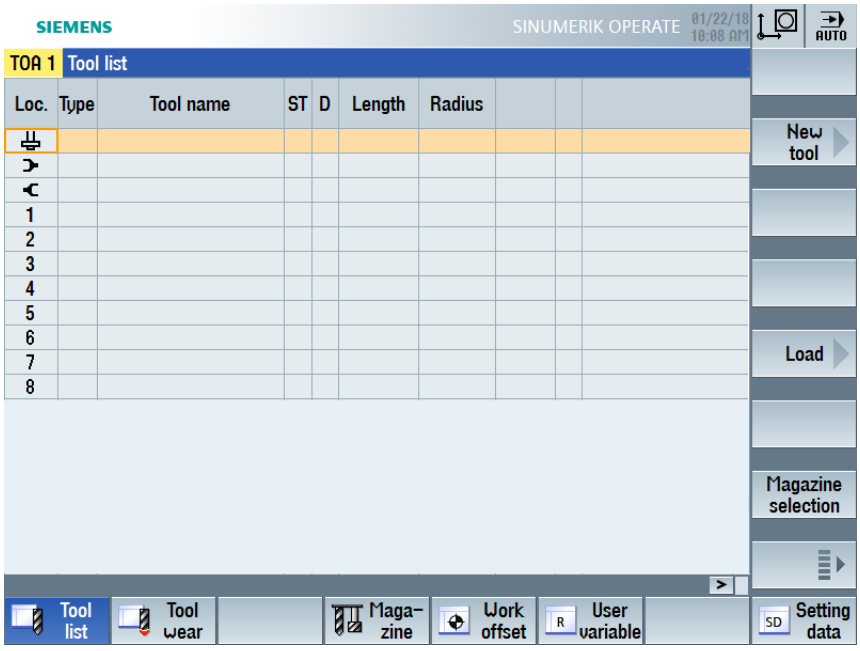


Figure 10-2 Tool list Milling machine

10.2 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. These sequences are described in:

- Example: Loading/unloading (Page 73)
- Example: Change manual tools (Page 75)

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 part program, a tool change is initiated without a new tool. Only the spindle tool is transported to the magazine (empty the spindle).

① ... ⑭ designate the steps in the following sequence chart.

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 Section Example: Acknowledgment steps (milling machine) (Page 105)

"Tool change" flow diagram

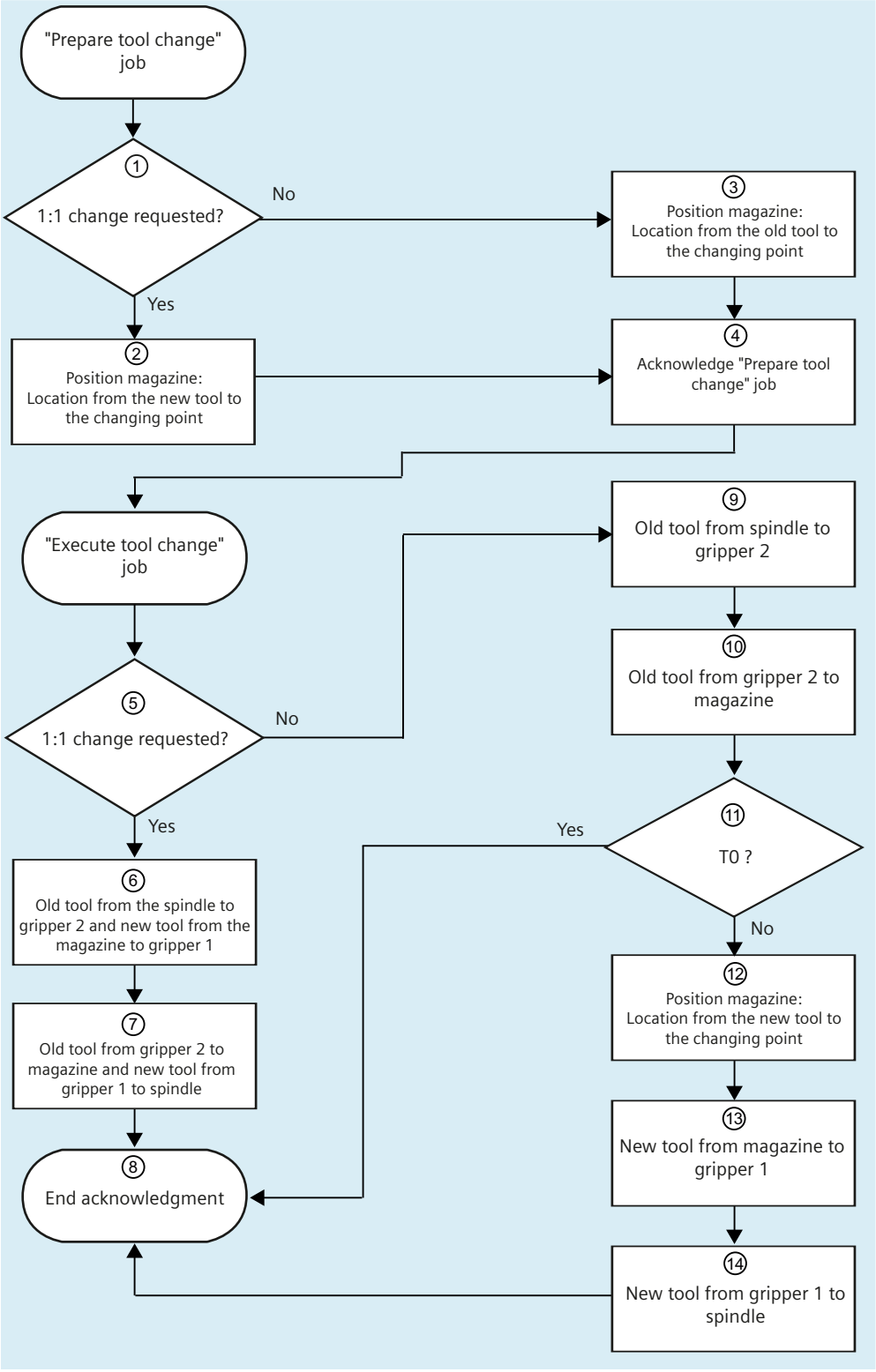


Figure 10-3 Flow diagram

Description of the sequence:

- NC part program:
  T command or T command with simultaneous tool change call (M6)
- Interface signals:
  Job from TOOLMAN: DB43xx.DBX0.0 (job bit) and
  DB43xx.DBB1 (command bits): Prepare change (DB43xx.DBX1.2)
- Magazine number and location number of the tools to be moved:
  DB43xx.DBW6..20: 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 situation: There is a tool in the tool holder and a new tool is 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 tool management is informed via DB43xx.DBX1.3 whether the active tool should be removed from the tool holder (T0). In this case, the magazine positions the storage location of the old tool (that is still 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.

Acknowledgment to TOOLMAN:

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:

Acknowledgment step	**Acknowledgment bit**	**Transfer step new tool**	**Transfer step old tool**	**Status**
4	DB4200.DBX0.4	101	0	204

The variable location in magazine 1 is positioned at the changing point to the spindle.

Transfer step	**From**		**To**	
	Magazine	**Location**	**Magazine**	**Location**

101 DB9901.DBW0	1	n	9998	1
n is the actual location number (n ≠ 0) entered by the PLC program in the variable transfer table.				

→ Continue with Step 4

Step 3: Position magazine, location from old tool to 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 TOOLMAN:				
Acknowledgment step	**Acknowledgment bit**	**Transfer step new tool**	**Transfer step old tool**	**Status**
4	DB4200.DBX0.4	101	0	204
The variable location in magazine 1 is positioned at the changing point to the spindle.				
Transfer step	**From**		**To**	
	Magazine	**Location**	**Magazine**	**Location**
101 DB9901.DBW0	1	n	9998	1

n is the actual location number (n ≠ 0) entered by the PLC program in the variable transfer table.

→ Continue with Step 4

Step 4: Prepare and acknowledge tool change job

← Previous step: Step 2 or Step 3

Acknowledgment to TOOLMAN: • NC program: M206 initiates the job for execute tool change • Interface signals: Job from TOOLMAN: DB43xx.DBX0.0 (job bit) DB43xx.DBB1 (command bits): Execute change (DB43xx.DBX1.1) • Magazine number and location number of the tools to be moved: DB43xx.DBW6 ... DBW20: Source location of the new tool, target location of the old tool				
Acknowledgment step	**Acknowledgment bit**	**Transfer step new tool**	**Transfer step old tool**	**Status**
1	DB4200.DBX0.1	0	0	1
With this, the preparation for the tool change is complete for many systems.				

→ Continue with Step 5

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 spindle to gripper 2 and new tool from magazine to gripper 1

← Previous step: Step 5

The PLC program controls the machine functions used to execute gripper movements, tool clamping, etc. Once the mechanical movements have been completed and acknowledged in the PLC program, acknowledgment of the tool movements is notified to the tool management. Acknowledgment to TOOLMAN:				
Acknowledgment step	**Acknowledgment bit**	**Transfer step new tool**	**Transfer step old tool**	**Status**
5	DB4200.DBX0.5	1	2	105
New tool from magazine to gripper 1 → Step 6 or 13				
Transfer step	**From**		**To**	
	Magazine	**Location**	**Magazine**	**Location**
1 DB9900.DBW0	0	1	9998	2
Old tool from spindle to gripper 2 → Step 6 or 9				
Transfer step	**From**		**To**	
	Magazine	**Location**	**Magazine**	**Location**
2 DB9900.DBW8	9998	1	9998	3

→ 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 program, tool movements are acknowledged to the tool management. Acknowledgment to TOOLMAN:				
Acknowledgment step	**Acknowledgment bit**	**Transfer step new tool**	**Transfer step old tool**	**Status**
7	DB4200.DBX0.7	3	4	105
New tool from gripper 1 to spindle → Step 7 or 14				
Transfer step	**From**		**To**	
	Magazine	**Location**	**Magazine**	**Location**
3 DB9900.DBW16	9998	2	9998	1
Old tool from gripper 2 to magazine → Step 7 or 10				

Transfer step	From		To	
	Magazine	Location	Magazine	Location
4 DB9900.DBW24	9998	3	0	2

→ Continue with Step 8

Step 8: End acknowledgment

← Previous step: Step 7 or Step 14

With the tool changer in the basic position or in a state in which the machine can continue machining, the end acknowledgement takes place. Here it is possible that there are still mechanical movements to be executed before the tool change can be completed.

Acknowledgment to TOOLMAN:

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 used to execute gripper movements, tool clamping, etc. Once the mechanical movements have been completed and acknowledged in the PLC program, acknowledgment of the tool movements is notified to the tool management.

Acknowledgment to TOOLMAN:

Acknowledgment step	Acknowledgment bit	Transfer step new tool	Transfer step old tool	Status
8	DB4200.DBX1.0	0	2	105

Old tool from spindle to gripper 2 → Step 6 or 9

Transfer step	From		To	
	Magazine	Location	Magazine	Location
2 DB9900.DBW8	9998	1	9998	3

→ Continue with Step 10

Step 10: Old tool from gripper 2 to magazine

← Previous step: Step 9

The PLC program controls the machine functions used to execute gripper movements, tool clamping, etc. Once the mechanical movements have been completed and acknowledged in the PLC program, acknowledgment of the tool movements is notified to the tool management. Acknowledgment to TOOLMAN:				
Acknowledgment step	**Acknowledgment bit**	**Transfer step new tool**	**Transfer step old tool**	**Status**
9	DB4200.DBX1.1	0	4	105
Old tool from gripper 2 to magazine → Step 7 or 10				
Transfer step	**From**		**To**	
	Magazine	**Location**	**Magazine**	**Location**
4 DB9900.DBW24	9998	3	0	2

→ Continue with Step 11

Step 11: T0 ?

← Previous step: Step 10

Request: Is T0 set in the tool chains job? → DB43xx.DBX1.3

- If only the tool holder should be emptied, the tool change can be completed. **→ continue with step 8**
- Is a new tool is to be brought into the tool holder? **→ continue with Step 12**

Step 12: Position magazine, location from new tool to changing point

← Previous step: Step 11

Sequence as in Step 2 Acknowledgment to TOOLMAN:				
Acknowledgment step	**Acknowledgment bit**	**Transfer step new tool**	**Transfer step old tool**	**Status**
4	DB4200.DBX0.4	101	0	204
The variable location in magazine 1 is positioned at the changing point to the spindle.				
Transfer step	**From**		**To**	
	Magazine	**Location**	**Magazine**	**Location**
101 DB9901.DBW0	1	n	9998	1

n is the actual location number (n ≠ 0) entered by the PLC 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 TOOLMAN:				
Acknowledgment step	**Acknowledgment bit**	**Transfer step new tool**	**Transfer step old tool**	**Status**
10	DB4200.DBX1.2	1	0	105
New tool from magazine to gripper 1 → Step 6 or 13				
Transfer step	**From**		**To**	
	Magazine	**Location**	**Magazine**	**Location**
1 DB9900.DBW0	0	1	9998	2

→ Continue with Step 14

Step 14: New tool from gripper 1 to spindle

← Previous step: Step 13

The PLC program controls the machine functions used to execute gripper movements, tool clamping, etc. Once the mechanical movements have been completed and acknowledged in the PLC program, acknowledgment of the tool movements is notified to the tool management. The tool change can be ended. Acknowledgment to TOOLMAN:				
Acknowledgment step	**Acknowledgment bit**	**Transfer step new tool**	**Transfer step old tool**	**Status**
11	DB4200.DBX1.3	3	0	105
New tool from gripper 1 to spindle → Step 7 or 14				
Transfer step	**From**		**To**	
	Magazine	**Location**	**Magazine**	**Location**
3 DB9900.DBW16	9998	2	9998	1

→ Continue with Step 8

10.3 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: is the actual location number (n ≠ 0) to be entered by the PLC program in the variable transfer table.

Acknowledgment-step table

Acknowledgment step	Transfer step		Acknowledgment 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 job

Acknowledgment step	Transfer step		Acknowledg-ment status	Comment
	Old tool	New tool		
3 DB9902.DBW8	0	0	105	Intermediate acknowledgment for subsequent job, 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 96)

10.4 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
;-------------------------------------------------------------------
```

10.5 Monitoring the maximum speed of a tool

If the maximum permissible speed for a tool is exceeded, or the rotary acceleration, e.g. of large tools, then this represents a danger for man and machine.

The appropriate limit values can be set for each tool via the following system variables:

- Speed limit value: $TC_TP_MAX_VELO[<internal T number>]
- Rotary acceleration limit value: $TC_TP_MAX_ACC[<internal T number>]

This limits the speed and/or the rotary acceleration of the spindle into which the tool is loaded.

The dynamic response limitation is active for the following spindle functions:

- Spindle control mode
- Tapping with G331/G332
- Synchronous following spindle
- Spindle positioning mode
- Constant cutting speed (G96, G961, G962)
- Constant peripheral grinding speed
- Manual traversing of the spindle in JOG mode

Note

The function is only available when magazine management is active.

Note

If a tool that is in a multitool is loaded into the spindle (with the multitool), the limitations do not take effect. The tool must be located directly on the spindle for this purpose.

Displaying tool parameters on the user interface

In order that the tool parameters $TC_TP_MAX_VELO (speed limit value) and $TC_TP_MAX_ACC (rotary acceleration limit value) are displayed on the user interface (read/write), they must be configured accordingly. A detailed description can be found in:

More information

- Configuring tool lists (Page 300)

The spindle is the tool holder

If the spindle number is also the number of the tool holder ($MC_TOOL_MANAGEMENT_TOOLHOLDER = 0) (this is typical for milling machines), then the following applies:

The extension when programming the tool change also refers to the spindle whose dynamic response should be limited.

Example: Extension = 3

```
T3="Miller_10" M3=6 S3=4000
```

A tool with the identifier "Miller_10" is loaded onto spindle S3. The speed of spindle S3 is limited by the tool that is located on it.

Note

If no extension is programmed, the tool change and the programmed speed refer to the current leading spindle.

The tool holder is not a spindle

If the tool holder is rigid and the tool holder number is not a spindle ($MC_TOOL_MANAGEMENT_TOOLHOLDER > 0) (this is typical for turning machines), then there is no assignment to a spindle.

However, for driven tools, it is necessary to assign the tool holder to a spindle. This is realized using the magazine location parameter $TC_MPP_SP[buffermagazine no., location no.].

If this assignment is not made, then the dynamic response is not monitored and not limited. As a consequence, for example, it is not possible to limit the dynamic response of the spindle of a turning tool with the fixed turning chisel.

Example

Tools on tool holder 2 are driven by spindle 5:

(Buffer magazine no. = 9998, arbitrary location no. therein)

```
$TC_MPP5[9998,location no.]=2
```

```
$TC_MPP_SP[9998,location no.]=5
```

Programming:

```
"T2="Drill_8" M2=6 S5=4000.
```

A tool with the identifier "Drill_8" is changed to tool holder 2. The speed of spindle S5 is now limited by the tool in tool holder 2.

Note

If an extension has not been programmed (e.g. T="Drill" S=4000), then the tool change refers to the current main tool holder and the programmed speed to the current leading spindle.

Limitation becomes effective on tool change

The dynamic response limitation by the tool on the spindle is only active if the tool is on the spindle as far as the NC data is concerned. To do this, the PLC acknowledgements are evaluated.

When the tool has been changed on the spindle or the tool holder, the NC/PLC interface signal "Tool with dynamic response limit active" (DB390x.DBX2003.0) is set.

Note

The NC/PLC interface signal "Tool with dynamic response limit active" (DB390x.DBX2003.0) indicates that a tool with dynamic response limitation is in the spindle. The assignment state of the spindles can be checked, especially if all tools have a maximum tool speed, which under certain circumstances is very high so that there is no effective limiting.

Example: Tool change:

Assumptions:

- The tool with the internal T number = 6 with parameterized dynamic response limitation is in the spindle (old tool, AWZ).
- The tool with the internal T number = 8 with parameterized dynamic response limitation, "Milling cutter" tool group is loaded into the spindle (new tool, NWZ).

Step No.	NC program	NC → PLC	PLC → NC
1	`N100 T1="Mill"`	---	---

2	---	`CMD=2 (tool preparation)`
		`NWZ T-No=8: Mag./ mag.loc.1→ Sp1`
		`AWZ T-No=6: Sp1 → Mag./ mag.loc.2`
			--- 	

Step No.	NC program	NC → PLC	PLC → NC
3	---	---	CMD=2, status=1 NWZ T-No=8: Mag./mag.loc.1→ Mag./mag.loc.1 AWZ T-No=6: Sp1 → Sp1
4	N101 M1=6	---	---
5	---	CMD=3 (tool change) NWZ T-No=8: Mag./mag.loc.1→ Sp1 AWZ T-No=6: Sp1 → Mag./mag.loc.2	---
6	---	---	CMD=3, status=105 NWZ T-No=8: Mag./mag.loc.1→ Mag./mag.loc.1 AWZ T-No=6: Sp1 → Mag./mag.loc.2
7	---	---	CMD=3, status=105 NWZ T-No=8: Mag./mag.loc.1→ Sp1 AWZ T-No=6: Mag./mag.loc.2 → Mag./mag.loc.2
8	---	---	CMD=3, status=1 NWZ T-No=8: Sp1→ Sp1 AWZ T-No=6: Mag./mag.loc.2 → Mag./mag.loc.2

CMD: PLC command
NWZ: New tool
AWZ: Old tool
Mag.: Magazine
Loc.: Location
Sp: Spindle

Up to step 6, the limits of the old tool are active.

In step 6, the PLC acknowledges the NC that the old tool was taken from spindle 1; this means that there are no limits imposed by the tool.

In step 7, the PLC acknowledges the NC that the new tool has been loaded to spindle 1, i.e. the limits of the new tool are active.

In step 8, the tool change is formally completed; no tool is moved.

Behavior when special events occur

Cutting edge activation for tool on the spindle

The limitation of the dynamic response by the tool on the spindle is also effective if one of the cutting edges of the spindle tool is activated or deactivated.
Otherwise, changing the speed limiting of the spindle tool has no effect on the speed limiting of the spindle.

Warm restart or abort of tool movements

The dynamic response limitation of a spindle by a tool depends on the tool, which, from the perspective of the NC machine data, is in the particular spindle. If required, the data in the NC should be aligned with the actual assignment of the spindles. This should be observed for a warm restart or if tool movements are aborted.

Channel interchange

When the spindle channel is interchanged, the limit of the dynamic response if retained.

Note

The channel into which the spindle is changed, must work with the same TO unit.

Diagnostics

Determining the cause of the limitation

The cause of a limitation can be determined using the following system variables:

- Limitation of the spindle speed by the current tool in the spindle:
  $AC_SMAXVELO_INFO[<n>] == 17, where n = spindle number
- Limitation of the spindle rotary acceleration by the current tool in the spindle:
  $AC_SMAXACC_INFO[<n>] == 17, where n = spindle number

Maximum values

The maximum values can be read via the following system variables:

- Limitation of the spindle speed: $AC_SMAXVELO[<n>]
- Limitation of the rotary acceleration of the spindle: $AC_SMAXACCEL[<n>]

n = spindle number

Actual values

The actual value of the spindle speed can be read via the following system variable:

$AA_S[<n>]; n = spindle number

NC/PLC interface signals

The signal is set irrespective of whether the speed or the acceleration of the spindle is limited by the current tool:

DB390x.DBX2003.0 (tool with dynamic response limitation)

Commissioning

Requirements

- The "Tool monitoring for maximum speed/acceleration" option must be set.
- The tool is only loaded into the spindle or its cutting edge activated **after** the speed/acceleration limitation has been set.

Note

- With MD17520 $MN_TOOL_DEFAULT_DATA_MASK Bit 6 = 0 (default):
  If the tool is already in the spindle/tool holder, it is not possible to change the speed or acceleration limit in $TC_TP_MAX_VELO[<T number>] or $TC_TP_MAX_ACC[<T number>].
- With MD17520 $MN_TOOL_DEFAULT_DATA_MASK Bit 6 = 1:
  If the tool is already in the spindle/tool holder, it is possible to change the speed or acceleration limit in $TC_TP_MAX_VELO[<T number>] or $TC_TP_MAX_ACC[<T number>], but the change doesn't take effect immediately.

- One TO unit can be assigned to several channels (MD $MC_TOA_UNIT). The spindle number must also be unique across these channels and not only in the channels in which it is configured for axis interchange. This is realized using the magazine location data
  - $TC_MPP5[<buffer magazine no.>, <location no.>] if the MD $MC_TOOL_MANAGEMENT_TOOLHOLDER = 0.
  - $TC_MPP_SP[<buffer magazine no.>, <location no.>] if the MD $MC_TOOL_MANAGEMENT_TOOLHOLDER > 0.

Note

The application itself must ensure that a spindle interchange in another channel is also correspondingly taken into account for a tool change.

The PLC can check the synchronization of the tool assignment of the spindles with the data management of the NC. Tools with a maximum tool speed set the PLC user interface signal "Tool with dynamic limiting active". This signal can be compared with the assignment sensors of the spindle.

Parameterization

- Speed limit value: $TC_TP_MAX_VELO[<internal T number>] = <value>, with <value> > 0
- Rotary acceleration limit value: $TC_TP_MAX_ACC[<internal T number>] = <value>, with <value> > 0

Note

The set values of $TC_TP_MAX_VELO and $TC_TP_MAX_ACC only take effect at the **next loading** of the tool into the spindle/tool holder, or when activating its cutting edge.

Data backup

After importing a data backup with tool data, from the data perspective, tools can be on spindles or tool holders. The dynamic limit is correspondingly active. It is the task of the PLC user program to adapt the actual assignment of the spindles to the NC data

Interface assignment

Signals from the axis/spindle (NC-> PLC):

Syntax	Description
DB390x.DBX2003.0 = TRUE	"Tool with dynamic response limitation active" =1; when option "Tool monitoring for maximum speed", tool data "TC_TP_MAX_VELO" >0 or "TC_TP_MAX_ACC">0 are set, and the affected spindle is **not** in axis mode.

Program example

Program code	Comment
N10 $TC_TP[10]="Miller"	;T10 = "Miller"
N20 $TC_TP_MAX_VELO[10]=3000	;Speed limiting from T10: 3000 r/min
...	
N100 SETMS(1)	;Spindle 1 = leading spindle
N110 M5	;Spindle stop
N120 T="Miller" M6	;Tool with T10 is loaded into spindle 1 ⇒
	<Axis>.spindle.in.toolWithDynLimitationActive = 1
N125 R1=$AC_SMAXVELO[1]	;Spindle speed limiting: 3000 r/min
N130 S5000 M3	;M3: 5000 r/min but limited to 3000 r/min
N140 $TC_TP_MAX_VELO[10]=2500	;Speed limiting from T10: 2500 r/min
	;Spindle speed limiting remains at 3000 r/min
N150 D1	;Cutting edge activation for tool on the spindle
	;Spindle speed limiting to 2500 r/min becomes effective
	;⇒<Axis>.spindle.in.toolWithDynLimitationActive = 1

NC programming

11

11.1 Overview of OPI and system variables

Fundamentals

All the data required for the purpose of tool management (e.g. to define a magazine or load a tool...) is stored in the NC. This data can be read and written via part programs with system variables and via the PLC with FB2 and FB3. When configuring the machine, the user (machine manufacturer) must determine the most efficient method of reading and writing tool management data, i.e. in the PLC, the NC or in an ASUP.

Read and write access can generally be made to system variables.

When language commands are used, it may be necessary to program the "STOPRE" command.

The $TC variables do not generate a preprocessing stop.

Tool identifiers can consist of the following characters:

a...z

A...Z

0...9

+ - _ . ,

The names are case-sensitive, i.e. uppercase and lowercase characters are considered different characters.

Note

Additional information on OPI variables can be found in the Help file for the NC-VAR selector (own application) and NC-VAR Access (TIA Portal).

Execution of the commands

The initial writing of data for tools or magazines causes the creation of the corresponding objects with their default data. Examples include:

- $TC_TP1 creates a tool.
- $TC_MAP1[x] creates a magazine.

Note

Not enabled/active functions

If data of unreleased functions are written, NO alarm is output and the write command is not executed.

The attempt to read data from inactive functions causes an alarm to be output.

Overview

The following diagram shows an overview of all cutting edge, tool and magazine data ($TC_...) when tool management is active.

Note:

The sequence of system variables shown corresponds to the OPI numbering sequence.

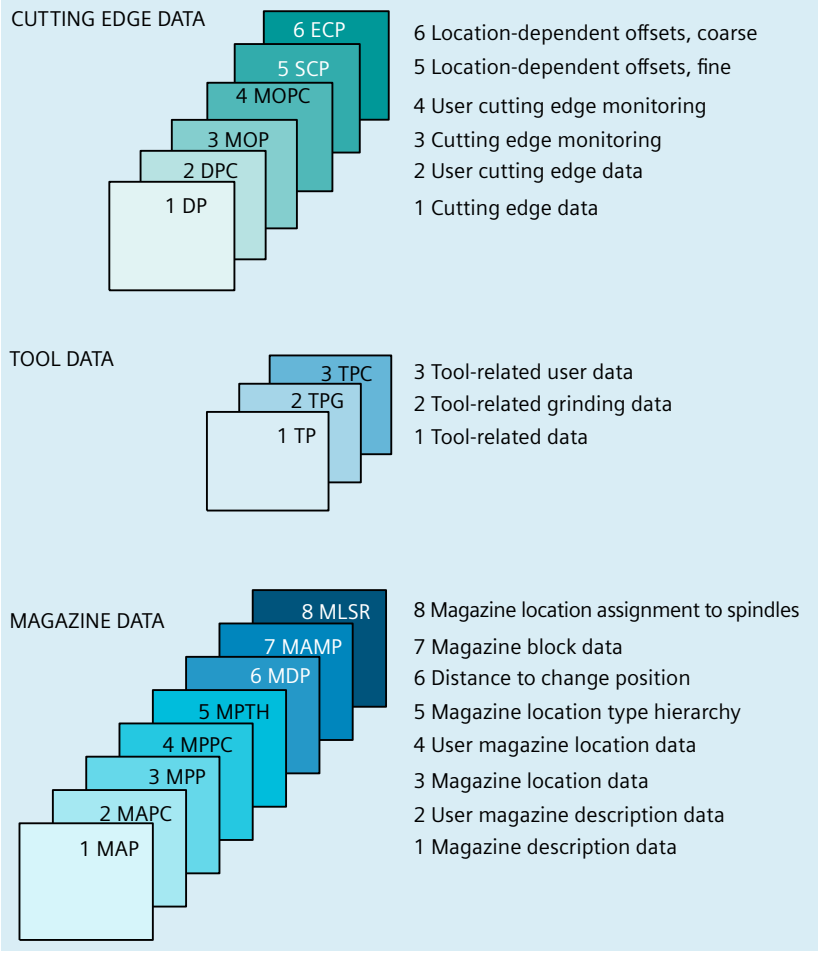


Figure 11-1 Overview of cutting edge, tool and magazine data

The identifiers (DP,...TP,...MOP,...) are taken from the NC language. They are part of the names of the system variables $TC_DP,...

Note

The gray data fields are only available if tool management is active.

Dark gray data fields are also available without tool management, but with monitoring function.

White data fields are available when the TOOLMAN function is not active.

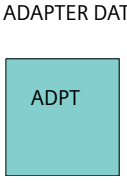


Figure 11-2 Adapter data

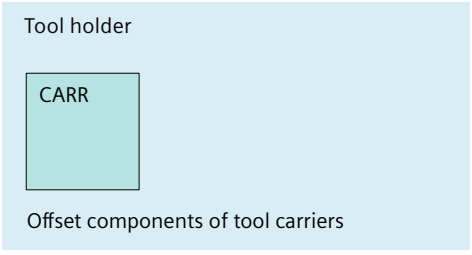


Figure 11-3 Tool holder data

Information on tool carriers can be found in the "Tools" Function Manual in the section "Orientable tool carriers".

11.2 Use of the magazine data

Sequence for defining data

The "Assign tool to a magazine location" sequence creates an interdependency between the tool data and the magazine/magazine location data.

Example:

The tool contains the magazine location type for which it is intended. The magazine type contains its own magazine location type. If the tool is assigned to the magazine location, as a rule the location type cannot be changed again as this can cause inconsistencies.

The resulting requirement is for tools and magazines to be loaded by a special routine into the PLC and that the structure-determining definitions may no longer be changed during the preparation (these are e.g. magazine dimension, magazine location type, duplo no., tool name, ...). They do not include: Cutting edge data, magazine location status, tool status, ...

Use of the magazine data

Action	Program command	Description
Creating a new magazine	`$TC_MAPx[y] = value;`	Value <>0, y = magazine no. of a magazine not yet set up
Delete a magazine	`$TC_MAP1[y] = 0;`	The data of the magazine and its magazine locations as well as any defined distances to change positions are deleted. The associated memory is released.
Delete a magazines and the tools contained in it	`$TC_MAP6[y] = 0;`	The data of the magazine and its magazine locations as well as any defined distances to change positions are deleted. Any tools contained in the magazine are also deleted. The associated memory is released
Delete all magazines	`$TC_MAP1[0] = 0;`	All of the data of all magazines of the selected TO area unit is deleted and the associated memory is released again. The magazine data block is then empty.
Set the magazine location data	`$TC_MPPx[y,z] = value;`	Value <> 0, z = location number Before creating the data of the first location, the associated magazine (magazine type and number of locations) must have been defined. If the number of columns parameter `$TC_MAP7` of the created magazine is written to, then all magazine locations belonging to the magazine are created with their default values according to the data for the number of lines and columns for the magazine.
Set magazine location type hierarchy	`$TC_MPTHx[y] = value;`	
Set magazine distances (distance to change position)	`$TC_MPTHx[y] = value;`	
Delete magazine distances (distance to change position)	`$TC_MDPx[y,0] = 0`	Delete all defined distances of the magazine with the number "y", I.e. the magazine is no longer "seen" during a tool search and an empty location search.
	`$TC_MDPx[0,0] = 0;`	Delete all defined distances of all magazines of the TO unit.
Delete the assignments of the buffer to the spindles	`$TC_MLSR[x,0] = 0;`	Delete all defined assignments of a buffer location with the number "x", I.e. the location "x" is no longer "seen" during the tool search.
	`$TC_MLSR[0,0] = 0;`	Delete all defined assignments of buffer locations of the TO unit to spindles
Set magazine block data	`$TC_MAMPx = value;`	

Loading data

Because tools are linked to magazines via magazine location parameter $TC_MPP6, the following rules for correct definition of tools and magazines must be adhered to:

1. Load tool data
2. Load magazine data
3. Load $TC_MPP6 parameters (⇒ places tool in magazine location)

The same sequence is used for data backup.

The grinding data of a tool cannot be written until tool type = "grinding tool" has been defined for at least one cutting edge.

The distance parameter ($TC_MDPx) and the buffer assignment parameter ($TC_MLSR) cannot be written until the magazines and their locations have been defined.

Delete data

A tool cannot be deleted while it is still contained in a magazine. The following sequence of operations must be followed when deleting:

1. Delete the magazine data (this removes tools from the magazine); or remove the tool explicitly from the magazine.
2. Delete tool data

In addition, a magazine cannot be deleted if it has the state $TC_MAP3[i]= 8 (motion is active). The delete command is rejected for all magazines even if only **one** magazine is preventing the command from being executed.

Note

If a single tool is to be deleted it must first be removed from the magazine location with an unload operation and then it can be deleted.

Tools that are currently selected cannot be deleted! You can ensure that no tool is selected beyond a part program by programming T0 before the end of a part program independently of the settings in the machine data (see MD for selecting tools beyond the end of a program).

11.3 Use of the tool and cutting edge data

Overview

If a parameter for a cutting edge, tool or magazine that does not exist is written, a new cutting edge, tool or magazine is created.

Note

When a tool is created, all the cutting-edge-specific data of cutting edge D1 are created with it. (DP, DPC, MOP, MOPC are preset to "0".) The grinding-specific tool data ($TC_TG1...) is not created until one of the tool types ($TC_DP1) 400-499 has been programmed for one of the cutting edges of the tool.

Use of the tool and cutting edge data with tool management

Magazine management and monitoring are active (TMMG and TMMO)

Action	Program command	Description

Create a tool	`T_NO = NEWT("tool identifier", duplo number)`
	or
	`$TC_TP1[y] = duplo number;`
	`$TC_TP2[y] = "tool identifier"`
		
		`y = tool number`

Create a cutting edge	`$TC_DPx[y,z] = value`	Create cutting edge D = z if D = z does not yet exist!
		`y = tool number`
		`z = cutting edge number`

Set tool data	`$TC_TPx[y] = value;`
	or
	`$TC_TPx[GETT("DRILL",DUPLO NO)] = value;`
	or
	`$TC_TPCx[y] = value;`
	`$TC_TGx[y] = value;`
		
		`y = tool number`
		Write tool-related user data
		
		Write tool-related grinding data

Set data of a cutting edge	`$TC_DPx[y,z] = value`
	`$TC_DPCx[y,z] = value`
	`$TC_MOPx[y,z] = value`
	`$TC_MOPCx[y,z] = value`
		Write offset data
		Write cutting edge-related user data
		Write cutting edge-related monitoring data
		Write CC (OEM) cutting edge monitoring data
		`y = tool number`
		`z = cutting edge number`

Deletion of the tool and cutting edge data	`$TC_TP1[0,0];`	When deleting tools, the entries for the location data must also be corrected.

Deletion of a tool	`$TC_TP1[y] = 0;`
	or
	`$TC_TP1[GETT("tool identifier", duplo number)] = 0;`
	or
	`DELT["tool identifier", duplo number]`
		All tool-related data is set to "0" (user data, hierarchy data, ...). When deleting a tool, the entries for the location data must also be corrected.

Use of the tool and cutting edge data without tool management

Action	Program command	Description

Create a tool	`$TC_DPx[y,z] = value;`
	
	`$TC_TP2[y] = "Number"`
		Create tool T if T does not yet exist!
		`y = tool number`
		`z = cutting edge number`

Create a cutting edge	`$TC_DPx[y,z] = value`	Create cutting edge D = z if D = z does not yet exist!
		`y = tool number`
		`z = cutting edge number`

Action	Program command	Description

Set data of a cutting edge	`$TC_DPx[y,z] = value`
	`$TC_DPCx[y,z] = value`
	`$TC_MOPx[y,z] = value`
	`$TC_MOPCx[y,z] = value`
		Write offset data
		Write cutting edge-related user data
		Write cutting edge-related monitoring data
		Write CC (OEM) cutting edge monitoring data
		`y = tool number`
		`z = cutting edge number`

Deletion of the tool and cutting edge data	`$TC_DP1[0,0] = 0;`	All tools of the channel are deleted, the memory is released.

Deleting data

When deleting, the memory area is deleted and automatically released again.

A tool can only be deleted if it is not active (offset-determining) and is not in a magazine, i.e. it must have been removed or unloaded.

Note

If tool management is active you must ensure that the tool being deleted is not assigned to a magazine location ($TC_MPP6). This assignment must be removed before the tool is deleted.

The grinding-specific tool data ($TC_TG1...) is created as soon as one of the tool types ($TC_DP1) 400-499 has been programmed for any of the cutting edges of the tool.

If the tool type is set from the current value taken from the range 400-499 to a value outside this range, the grinding data memory released again, i.e. the grinding-specific data is lost.

11.4 Cutting edge data

11.4.1 Cutting edge data

Overview of cutting edge data

Number	System variable $TC_...	Description
1	DP	Cutting edge data (Page 122)
2	DPC	User cutting edge data (Page 124)
3	MOP	Cutting edge monitoring (Page 125)
4	MOPC	User cutting edge monitoring (Page 126)
5	SCP	Location-dependent offsets, fine (Page 127)
6	ECP	Location-dependent offsets, coarse (Page 127)

This data exists for each cutting edge that is created (D1-D12). Tool management includes the geometry and user data as well as the optional monitoring data for the cutting edges.

If the cutting edges are created via HMI, the D number is counted up from 1. It is possible to program the D no. with gaps, e.g. D1, D3, D6 if cutting edges are set up using an NC program.

11.4.2 Cutting edge parameter $TC_DP

$TC_DPx[t,D]

Cutting edge parameters for geometry, technology and tool type.

Parameters:

- x: Parameters 1 ... 40
  The number of cutting edge parameters that can be read and written by the OPI is displayed in OPI variable "numCuttEdgeParams" in the Y block. This value is one less than the number of cutting edge data accessible using system variables (see the table below)
- t: T number 1 ... 32000
- D: Cutting edge number 1 ... 12 or D number

Depending on the tool type, up to 40 cutting edge parameters can be programmed for each tool cutting edge.

More information: Function Manual Tools, tool offset

OPI block TO

Address calculations:

- Line = (D - 1) * "numCuttEdgeParams" + parameter number
- Column = T number

Tool offset parameters (system variables)				
NC identifier	**Type**	**Designation**	**OPI variable**	**Type**
$TC_DP1	INT	Tool type	edgeData	DOUBLE
$TC_DP2	Double	Cutting edge position*	edgeData	DOUBLE
$TC_DP3	Double	Geometry length 1	edgeData	DOUBLE
$TC_DP4	Double	Geometry length 2	edgeData	DOUBLE
$TC_DP5	Double	Geometry length 3	edgeData	DOUBLE
$TC_DP6	Double	Geometry radius	edgeData	DOUBLE
$TC_DP7	Double	Geometry - corner radius (tool type 700; slotting saw)	edgeData	DOUBLE
$TC_DP8[1)]	Double	Geometry length 4 (tool type 700; slotting saw)	edgeData	DOUBLE
$TC_DP9[1)]	Double	Geometry length 5	edgeData	DOUBLE
$TC_DP10[1)]	Double	Geometry - angle 1	edgeData	DOUBLE
$TC_DP11[1)]	Double	Geometry – angle 2 for tapered milling tools	edgeData	DOUBLE

Tool offset parameters (system variables)				
NC identifier	**Type**	**Designation**	**OPI variable**	**Type**
$TC_DP12	Double	Wear - length 1	edgeData	DOUBLE
$TC_DP13	Double	Wear - length 2	edgeData	DOUBLE
$TC_DP14	Double	Wear - length 3	edgeData	DOUBLE
$TC_DP15	Double	Wear - radius	edgeData	DOUBLE
$TC_DP16	Double	Wear - slot width b/rounding radius	edgeData	DOUBLE
$TC_DP17	Double	Wear - projection k	edgeData	DOUBLE
$TC_DP18	Double	Wear - length 5	edgeData	DOUBLE
$TC_DP19	Double	Wear - angle 1	edgeData	DOUBLE
$TC_DP20	Double	Wear - angle 2 for conical milling tools	edgeData	DOUBLE
$TC_DP21	Double	Adapter - length 1	edgeData	DOUBLE
$TC_DP22	Double	Adapter - length 2	edgeData	DOUBLE
$TC_DP23	Double	Adapter - length 3	edgeData	DOUBLE
$TC_DP24	Double	Clearance angle	edgeData	DOUBLE
$TC_DP25	Double	**ManualTurn**: Cutting speed **ShopMill**: Status value of tool of type 1xx and 2xx.	edgeData	DOUBLE
$TC_DPCE [t,d]	INT	System variable of an offset data record with T=t and D=d containing cutting edge number CE. (Unique D no. or user assignment of D no. for cutting edge numbers) Value range for permissible cutting edge numbers: 1, 2, 3, ... MD18106.	-	-
$TC_DPH [t,d]	INT	H parameter (Y/extraCuttEdgeParams), Bit0=1	edgeData	DOUBLE
$TC_DPV	Double	Tool cutting edge orientation	edgeData	DOUBLE
$TC_DPV3	Double	L1 component of the tool cutting edge orientation	edgeData	DOUBLE
$TC_DPV4	Double	L2 component of the tool cutting edge orientation	edgeData	DOUBLE
$TC_DPV5	Double	L3 component of the tool cutting edge orientation	edgeData	DOUBLE
$TC_DPVN3	Double	L1 component of the orientation normals	edgeData	DOUBLE
$TC_DPVN4	Double	L2 component of the orientation normals	edgeData	DOUBLE
$TC_DPVN5	Double	L3 component of the orientation normals	edgeData	DOUBLE
$TC_DPNT	INT	Number of teeth, cutting edge parameter 34	edgeData	DOUBLE
$TC_DPROT	Double	Tool clamping angle for turning tools	edgeData	DOUBLE
$TC_DP_RADIUS3[1)]	Double	Geometry - radius 3	edgeData	DOUBLE
$TC_DP_WEAR_RADIUS3[1]	Double	Wear - radius 3	edgeData	DOUBLE
$TC_DP_LEN6[1)]	Double	Geometry - length 6	edgeData	DOUBLE
$TC_DP_WEAR_LEN6[1)]	Double	Wear - length 6	edgeData	DOUBLE
1) The meaning of this data is different depending on the tool type.				

$TC_DP11

$TC_DP11 contains the identification for the main direction of machining as is defined and required by the Siemens Cycle950. $TC_DP11 assumes an intermediate position between tool OEM parameter and NC system variable:

- $TC_DP11 is a tool OEM parameter in so far as NC does not evaluate the contents of the value.
- $TC_DP11 is an NC system variable in so far as when accessing via $P_ADT[n], with n=11, NC is subject to the special values 1, 2, 3, 4 of the tool adapter transformation if TMMG and the subfunction "Tool adapter" are active. This system parameter property is also used with the analog OPI block TOT.

11.4.3 User cutting edge data $TC_DPC

$TC_DPCx[t,D]

Up to 64 additional cutting edge parameters can be programmed for each cutting edge. Setting with MD18096 MM_NUM_CC_TOA_PARAM and enable with MD18080 MM_TOOL_MANAGEMENT_MASK (set bit 2=1)

X = Parameter 1...64

T = T number 1...32000

D = Cutting edge number 1...12

D = D number

OPI block TUE/TUO

Calculation of line:(d-1)*numCuttEdgeParams_tu+parameter no.

Calculation of column: T number

User-related cutting edge data					
NC identifier	Type	Designation	OPI variables	Type	Default setting
$TC_DPC1	Double	CC_Cutting edge parameter 1	edgeData	REAL	0
...	Double	...	edgeData	REAL	0
$TC_DPC64	Double	CC_Cutting edge parameter 64	edgeData	REAL	0

Note

The data is displayed in the tool management. Here you could store "Max. cutting rate", for example, which is then evaluated in the part program.

11.4.4 Cutting edge-related tool monitoring $TC_MOP

$TC_MOPx[t,D]

Tool cutting edges are monitored according to tool life, workpiece count and/or wear.

X = Parameter 1...15

T = T_Number 1...32000

D = Cutting edge number 1...12

D = D number

OPI block TS

Calculation of line: (d-1)*numCuttEdgeParams_ts+parameter no.

Calculation of column: T number

Tool management monitoring data					
NC identifier	**Type**	**Designation**	**OPI variable**	**Type**	**Default setting**
$TC_MOP1	Double	Prewarning limit for tool life in min.	data	REAL	0
$TC_MOP2	Double	Residual tool life in minutes	data	REAL	0
$TC_MOP3	INT	Prewarning limit for count	data	REAL	0
$TC_MOP4	INT	Residual unit quantity	data	REAL	0
$TC_MOP11	Double	Setpoint for tool life	data	REAL	0
$TC_MOP13	INT	Setpoint for workpiece count	data	REAL	0
$TC_MOP5	Double	Wear prewarning limit - or location-dependent offset fine prewarning limit	data	REAL	0
$TC_MOP6	Double	Actual wear or location-dependent offset, fine actual value	data	REAL	0
$TC_MOP15	Double	Wear setpoint - or location-dependent offset, fine setpoint	data	REAL	0
$TC_MOP_SUSPENSION_REASON	INT	Reason why the cutting edge causes the tool to go into 'disabled' state	data	REAL	0

$TC_MOP_SUSPENSION_REASON

This variable describes which disable methods are active for the cutting edge.

Value		Description	
Value 0:		No disable method is active	
Bits 0-2:		Internal disabling reasons (only active if the corresponding monitoring type has been enabled by $TC_TP9)	
	Bit 0:	The value of the tool life has reached the limit value (read-only)	
	Bit 1:	The value of the unit quantity has reached the limit value (read-only)	
	Bit 2:	The tool wear value has reached the limit value (read-only)	
Bit 3:		Reserved	

Value		Description	
Bits 4-9:		External disabling reasons:	
	Bit 4:	OEM disable method 1 is active	
	Bit 5:	OEM disable method 2 is active	
	Bit 6:	OEM disable method 3 is active	
	Bit 7:	OEM disable method 4 is active	
	Bit 8:	OEM disable method 5 is active	
	Bit 9:	OEM disable method 6 is active	
Bits 0 to 3 are read-only; the bits are ignored when a write attempt is made.			

11.4.5 User cutting edge monitoring $TC_MOPC

$TC_MOPCx[t,D]

Tool monitoring user data (edge-specific)

Up to 64 additional tool monitoring parameters can be programmed for each cutting edge. (V4.7 and higher). Setting with MD18098 MM_NUM_CC_MON_PARAM and enable with MD18080 MM_TOOL_MANAGEMENT_MASK (set bit 2)

X = Parameter 1....64

T = T_Number 1....32000

D = Cutting edge number 1....12

D = D number

OPI block TUS

Calculation of line: (d-1)*numCuttEdgeParams_tus+parameter no.

Calculation of column: T number

Tool monitoring user data (edge-specific)					
NC identifier	**Type**	**Designation**	**OPI variables**	**Type**	**Default setting**
$TC_MOPC1	Int	CC monitoring parameters	userdata	REAL	0
...	Int	...	userdata	REAL	0
$TC_MOPC64	Int	CC monitoring parameters	userdata	REAL	0

11.4.6 Location of use-dependent offsets fine (additive offsets) $TC_SCP

$TC_SCPx[t,D]

Location-dependent offsets fine (the term "additive offsets" is also frequently used) comprise all the magnitudes of error which contribute to the total deviation between the actual workpiece and the specified dimensions. The parameters for the location-dependent offsets refer to the geometrical data of a cutting edge. DL: Location-dependent; offsets regarding the location of use.

X = Parameter for DL=1...DL=6

T = T number 1...32000

D = Cutting edge number 1...12

D = D number

OPI block TOS, TOST

Calculation of line: (d-1)*(maxnumEdgeSC*numParams_SC) + ((EdgeSC_1)*numParams_SC) +Parameter no.

Calculation of column: T number

Location-dependent offsets					
Name		**Type**	**Designation**	**OPI variable**	**Type**
$TC_SCPx					
x = 13-21	1_RADIUS3, 1_LEN6	Double	Can be activated with DL=1	edgeSCData	REAL
x = 23-31	2_RADIUS3, 2_LEN6	Double	Can be activated with DL=2	edgeSCData	REAL
x = 33-41	3_RADIUS3, 3_LEN6	Double	Can be activated with DL=3	edgeSCData	REAL
x = 43-51	4_RADIUS3, 4_LEN6	Double	Can be activated with DL=4	edgeSCData	REAL
x = 53-61	5_RADIUS3, 5_LEN6	Double	Can be activated with DL=5	edgeSCData	REAL
x = 63-71	6_RADIUS3, 6_LEN6	Double	Can be activated with DL=6	edgeSCData	REAL
			Transformed location-dependent offsets fine, block TOST	edgeSCData	REAL

11.4.7 Location-dependent offsets coarse (setting-up offsets) $TC_ECP

$TC_ECPx[t,D]

The location-dependent offsets, coarse (also setting-up offsets) can be set by the machine setter before the machining operation (see also $TC_SCP (Page 127)).

X = Parameter for DL=1...DL=6

T = T number 1...32000

D = Cutting edge number 1...12

D = D number

OPI block TOE, TOET

Calculation of line: (d-1)*(maxnumEdge_SC*numParams_SC)+((EdgeSC-1)*numParams_SC) +Parameter no.

Calculation of column: T number

Setting-up offsets					
Name		**Type**	**Designation**	**OPI variable**	**Type**
$TC_ECPx				edgeECData	REAL
x = 13-21	1_RADIUS3, 1_LEN6	Double	Can be activated with DL=1	edgeECData	REAL
x = 23-31	2_RADIUS3, 2_LEN6	Double	Can be activated with DL=2	edgeECData	REAL
x = 33-41	3_RADIUS3, 3_LEN6	Double	Can be activated with DL=3	edgeECData	REAL
x = 43-51	4_RADIUS3, 4_LEN6	Double	Can be activated with DL=4	edgeECData	REAL
x = 53-61	5_RADIUS3, 5_LEN6	Double	Can be activated with DL=5	edgeECData	REAL
x = 63-71	6_RADIUS3, 6_LEN6	Double	Can be activated with DL=6	edgeECData	REAL
			Transformed setting-up offsets, block TOET	edgeECData	REAL

11.5 Tool data

11.5.1 Overview

Overview of tool data

Number	System variable $TC_...	Description
1	TP	Tool-related data (Page 128)
2	TPG	Tool-related grinding data (Page 135)
3	TPC	Tool-related user data (Page 136)

11.5.2 Tool-related data $TC_TP

$TC_TPx[t]

General tool data

This data describes the tool in the magazine.

Programming of general tool data with tool management

x: = Parameter 1 ... 11

t: = T number 1 ... 32000

OPI block TD

Calculation of line: T number

Calculation of column: n.a.

Tool-related data, tool management					
NC identifier	**Type**	**Designation**	**OPI variable**	**Type**	**Default setting**
$TC_TP2	String	Tool identifier	toolIdent	String	"T no."
$TC_TP1	INT	Duplo number	duploNo	WORD	T no.
$TC_TP3	INT	Size on left	toolsize_left	WORD	1
$TC_TP4	INT	Size on right	toolsize_right	WORD	1
$TC_TP5	INT	Size above	toolsize_upper	WORD	1
$TC_TP6	INT	Size below	toolsize_down	WORD	1
$TC_TP7	INT	Magazine location type	toolplace_spec	WORD	9999
$TC_TP8	INT	Status Value 0 not enabled Bit 0 active tool Bit 1 enabled Bit 2 disabled Bit 3 measure Bit 4 prewarning limit reached Bit 5 tool being changed Bit 6 fixed-location coded Bit 7 tool was in use Bit 8 identifier for tools in buffer Bit 9 = 1 ignore disabled state Bit 9 = 0 do not ignore Bit 10 to be unloaded Bit 11 to be loaded Bit 12 reserved Bit 13 reserved Bit 14 1:1 exchange Bit 15, manual tool Bit 17 tool is at a disabled magazine location	toolState	WORD	0 = not enabled
$TC_TP9	INT	Tool monitoring method Value 0 no tool monitoring Bit 0 tool life Bit 1 unit quantity Bit 2 wear monitoring active Bit 3 wear monitoring, location-dependent offset fine active	toolMon	WORD	0

Tool-related data, tool management					
NC identifier	**Type**	**Designation**	**OPI variable**	**Type**	**Default setting**
$TC_TP10	INT	Number for a replacement tool.	toolSearch	WORD	0
$TC_TP11	INT	Tool information: Allows tool groups to be divided into subgroups. Tool selection, only with tools of the lower group	toolInfo	Integer	0
$A_TOOLMN	INT	Magazine assignment tool	toolInMag	WORD	
$A_TOOLMLN	INT	Location assignment tool	toolInPlace	WORD	
$P_TOOLND	INT	Number of cutting edges	numCuttEdges	WORD	
-		Adapter No. Assignment	adaptNo	WORD	
$A_MYMN	INT	Owner magazine for tool	toolMyMag	WORD	
$A_MYMLN	INT	Owner magazine location for tool	toolMyPlace	WORD	
$TC_TP_PROTA	string	Name of the three-dimensional protection area for the tool or name of the file that contains the description of the protection area for the tool.			
$TC_TP_MAX_VELO	real	Maximum speed of the tool, if the value is > 0. If a speed limit is not defined (=0), then no monitoring is performed	toolMaxVelo		
$TC_TP_MAX_ACC	real	Maximum acceleration of the tool, if the value is > 0. If an acceleration speed limit has not been defined (=0), then no monitoring is performed	toolMaxAcc		

$TC_TP1 ... $TC_TP2

Like the T No. is sufficient to identify a tool, a tool is equally unique in terms of its duplo number and its tool name (identifier).

TO units may therefore only contain names that have different duplo numbers. The write operations of $TC_TP1 and $TC_TP2 are checked for the above and rejected if collisions are found.

$TC_TP3 ... TP6

Size in terms of half locations:

Size 1 means that the tool exactly completely occupies its own magazine location. If the size increases in one direction by 1, half an adjacent location is reserved in this direction for the tool.

The maximum programmable size is 11. This means that no more than five whole adjacent locations can be reserved by the tool in one direction.

The tool size (half locations) can only be written or changed if the tool does not have an owner location. This applies to the following situations:

- The tool has yet not been loaded
- During loading, as long as the tool is not at its target location in a real magazine
- If the tool is at a location of the buffer or loading magazine and beforehand, the owner location as well as the reservation in the magazine was deleted (exclusively using the NC commands DELMRES and DELMOWNER).

$TC_TP7

The magazine location type can only be written or changed if the tool does not have an owner location. This applies to the following situations:

- The tool has yet not been loaded
- During loading, as long as the tool is not at its target location in a real magazine
- If the tool is at a location of the buffer or loading magazine and beforehand, the owner location as well as the reservation in the magazine was deleted (exclusively using the NC commands DELMRES and DELMOWNER).

$TC_TP8 (only with TMMO and TMMG)

Status bit 0

Tool state: Bit 0 = 0 Tool is released. Bit 0 = 1 Tool is not released.

Status bit 1

The tool status is described with system variable $TC_TP8. The system variable is bit-coded. In other words, a specific state of the tool is assigned to each bit of this data.

The status of a tool must be **bit 1=1** ("enabled") so that it can be loaded within the scope of a programmed tool change for machining in the tool holder.

During tool selection, the status of a tool that is loaded onto the tool holder (spindle, ...) is set by the NC to **bit 0** ("active").

Status bit 2

A tool cannot be loaded if its status is **bit 2=1** ("disabled"). The status is automatically set by the tool monitoring function when a monitoring value of at least one cutting edge reaches the limit value or one of the external disable methods is activated by the system variable $TC_TP_SUSPENSION_REASON or $TC_MOP_SUSPENSION_REASON. The status bit 2=4 of the tool on the tool holder can or will be ignored when generating the INIT blocks (see MD 20110 and 20112). The PLC also has the option to make NC ignore the status during tool selection.

The reason for the "disabled" state can be determined using system variables $TC_TP_SUSPENSION_REASON and $TC_MOP_SUSPENSION_REASON.

Internal monitoring types and external disabling methods can work simultaneously, see the following example:

Example

```
$TC_TP8[1]=2 ; Tool enabled
$TC_TP9[1]=1 ; Tool monitoring is active
; Tool life expires
$TC_MOP2[1,1]=0
; at this point:
; $TC_TP8[1] == 6
; $TC_TP_SUSPENSION_REASON[1] == 0x1
; $TC_MOP_SUSPENSION_REASON[1,1] == 0x1
; OEM mode 1 is activated
$TC_MOP_SUSPENSION_REASON[1,1] = $TC_MOP_SUSPENSION_REASON[1,1] B_OR 0x10
; at this point:; $TC_TP8[1] == 6
; $TC_TP_SUSPENSION_REASON[1] == 0x11
; $TC_MOP_SUSPENSION_REASON[1,1] == 0x11
; Tool life reset$TC_MOP2[1,1]=1000
; at this point:; $TC_TP8[1] == 6
; $TC_TP_SUSPENSION_REASON[1] == 0x10
; $TC_MOP_SUSPENSION_REASON[1,1] == 0x10
; OEM mode 1 is deactivated
$TC_MOP_SUSPENSION_REASON[1,1] = 0
; at this point:
; $TC_TP8[1] == 2; $TC_TP_SUSPENSION_REASON[1] == 0x0
; $TC_MOP_SUSPENSION_REASON[1,1] == 0x0
```

Status bit 3

The status **bit 3 = 1** indicates the measurement of the tool.

Status bit 4

The status **bit 4=1** ("prewarning limit") is mainly for information purposes. With this status the tool can still be loaded.

Status bit 5

The tool status **bit 5=1** (="W"= is being change) is always reset by the software during buffered booting. A tool receives/loses this status within the scope of a programmed tool change.

The following applies: The new tools involved in the tool change are assigned the status bit 5=1 when the tool is selected. The status is reset again by the end acknowledgment for each tool command.

The following applies in particular:

With the end acknowledgment of the PLC command 3, 4, 5
(programming M06 in a block with $MC_TOOL_CHANGE_MODE=1, T, M06 in a block with $MC_TOOL_CHANGE_MODE=1
T address with $MC_TOOL_CHANGE_MODE=0)
the status bit 5=1 of the old and the new tool is reset.

Tools that are in the buffer can also be used for a new programmed tool command if they have the status bit 5=1 (alternating).

Tools that are in the real magazine and have this status can be used depending on bit 21 of the MD 20310 or cannot be used for a competing tool-change command for another spindle.

The status bit 5=1 is generally not taken into consideration for a tool selection within the scope of a block search or for init block generation.

For a RESET, the status is reset for those tools that are involved in a tool change at that point in time.

The status **bit 5=1** is not evaluated when a manual tool is selected.

Status bit 6 (location coding)

Bit 6 = 1 Tool is fixed-location coded.

Status bit 7 (was in use)

Only with TMMG:

- For MD $MN_TOOL_DEFAULT_DATA_MASK bit 5 = 0:
  The status bit 7=1 ("was in use") is set by the NC if the tool is removed from a magazine location of the type spindle or tool holder.
- For MD $MN_TOOL_DEFAULT_DATA_MASK Bit 5 = 1:
  The status bit 7=1 ("was in use") is set by the NC if the tool is removed from a magazine location of the type spindle or tool holder and a machining feedrate was programmed for the tool (i.e. a non-G0 block was selected, i.e. machining took place).

Status bit 8 (return transport)

The tool status **bit 8=1** ("return transport") ensures that during the next tool change, a tool that is at a buffer location and not intended for the next job in machining is returned to the real magazine.

Status bit 9 (ignore disabled state)

If this bit is set, the disabled state of this tool is ignored. This means the disabled tool can be used (depending on the search strategy).

This state acts independently of the PLC interface signal:

"Tool disable not effective " (" Tool disable not effective" (DB21.DBx29.7).).

Status bit 10 (to be unloaded)

Bit 10 is set for tools that are located in a magazine and are to be unloaded. The following definitions apply:

- The state remains for Power On
- It is included in the data back-up and rewritten when transferred back to the NC.
- Unloading a tool via an unloading location causes the NC to reset the tool status.

Bit	Value	Meaning
10	0	"Not to be unloaded"
	1	"To be unloaded"

For $MC_TOOL_CHANGE_ERROR_MODE, Bit 9 = 1:

A tool that has the status "to be unloaded" ($TC_TP8, Bit 10 = 1) is treated like a disabled tool, i.e. it is no longer taken into consideration during a tool search and can no longer be selected within the scope of T programming.

Status bit 11 (to be loaded)

Bit 11 is set for tools that are not in a magazine and are to be loaded. The following definitions apply:

- The state remains for Power On.
- It is included in the data back-up and rewritten when transferred back to the NC.
- When assigning a tool to a real magazine, the tool status is reset by the NC (applies to locations of location type 1, i.e. not to internal magazines such as the loading magazine, buffer magazine, etc.).

Bit	Value	Meaning
11	0	"Not to be loaded"
	1	"To be loaded"

Status bit 12 (master tool)

Status bit 12 = 0 not allowed, Bit 12 = 1 indicates that the tool is a master tool. Master tools are permanently assigned to a magazine. This status is only set to provide information and has

no effect on the NCK (e.g. does not lock a location). The user defines via the unload program whether the tool can be unloaded.

Status bit 13 (reserved)

Bit 13 is reserved.

State bit 14 (tool marked for 1:1 exchange)

NC-internal state, that indicates (for a selected search strategy, 1:1 interchange), that the new and old tool should be interchanged 1:1.

Note

Take care when "manually" changing the tool status via the OPI during machining. This could undo internal state changes by the NC and result in incorrect machining.

Status bit 15 (manual tool)

Bit 15 = 1 indicates that the tool is a manual tool.

Status bit 16 (reserved)

Bit 16 is reserved.

Status bit 17 (tool is at a disabled magazine location)

Tools in a subsequently disabled magazine location are assigned, with $TC_TP8 Bit 17=1, the identifier indicating that they are located in a disabled magazine location. With instruction $TC_MPP6, tools identified in this way can be placed in disabled magazine locations or disabled adjacent locations. Oversize tools that project beyond the edge of the magazine are not taken into account.

$TC_TP9

If a monitoring type is activated for the tool with $TC_DP9, the current monitoring parameters are evaluated and, if necessary, the tool status set to "disabled" or "prewarning limit reached". An existing tool disable is however not withdrawn. Not even then when the monitoring function for this tool has been deactivated.

$TC_TP11

Tool subgroups

The system variable is bit-coded. Only bits 0 ... 3 are evaluated. A tool group (the same identifier, different duplo No.) can be split into a maximum of 4 subgroups in this way. A tool can also be included in several subgroups.

If no bit is set, so $TC_TP11[x]=0, this means the same as "all bits set", i.e. the tool belongs to all the defined subgroups.

Selection of the tool subgroup

1. With language command **$P_USEKT** (UseKindofTool)
   (only possible when not working with the setting T = location)
   During tool search, only tools that have one of these bits in system variable $TC_TP11, can be found. This means that it is possible to form "Technology Groups", to differentiate between tools with the same identifier and specifically release them for machining.
   Example 1:
   $P_USEKT = 4
   This means that the only tools to be taken into account are those with bit 2 in $TC_TP11 or
   Example 2:
   $P_USEKT = 9
   This means that the only tools to be taken into account are those with bit 3 or 0 in $TC_TP11
2. By programming a tool
   With the function **T = location**, $P_USEKT is set automatically at every tool change, and in fact at the $TC_TP11 value of the loaded tool.
   Example: T3 M06
   The bit value of $TC_TP11 of T3 is now valid (is accepted in "USEKT").
   During the transition to a replacement tool (and only there) the only tools to be taken into account are those with one of these bits set in system variable $TC_TP11.

11.5.3 Tool-related grinding data $TC_TPG

$TC_TPGx[t]

Technology-specific grinding data

The default setting for grinding data is "0". Tools with **tool type 400 ... 499** are always **grinding tools**, i.e. additionally have these data which take up additional memory space. If a tool of type 400 ... 499 is set to a value outside this range, then its loses its grinding-specific data - the associated memory is released again and can be used for other tools.

x: = Parameter 1 ... 9

t: = T number 1 ... 32000

OPI block TG

Calculation of line: T number

Calculation of column: N/A

Tool-related grinding data				
Name	**Type**	**Designation**	**OPI VAR**	**Type**
$TC_TPG 1	INT	Spindle number	spinNoDress	REAL
$TC_TPG 2	INT	Chaining rule	conntectPar	REAL
$TC_TPG 3	Double	Minimum grinding wheel radius	minToolDia	REAL
$TC_TPG 4	Double	Minimum grinding wheel width	minToolWide	REAL
$TC_TPG 5	Double	Current grinding wheel width	actToolWide	REAL
$TC_TPG 6	Double	Max. speed	maxRotSpeed	REAL
$TC_TPG 7	Double	Maximum peripheral speed	maxTipSpeed	REAL
$TC_TPG 8	Double	Inclination angle of inclined wheel	inclAngle	REAL
$TC_TPG 9	INT	Parameter number for radius calculation	paramNrCCV	REAL

11.5.4 Tool-related user data $TC_TPC

$TC_TPCx[t]

User-related tool data

An additional 64 tool-specific parameters can be set up per tool (V4.7 and higher). Setting with MD18094 MM_CC_TDA_PARAM and enable with MD18080 MM_TOOL_MANAGEMENT_MASK (set bit 2)

x: = Parameter 1 ... 64

t: = T number 1 ... 32000

OPI block TU/TUD

Calculation of line: T number

Calculation of column: Parameter number

Tool-related OEM user data				
NC identifier	**Type**	**Designation**	**OPI VAR**	**Type**
$TC_TPC1	Double		data	REAL
...	Double		data	REAL
$TC_TPC64	Double		data	REAL

Note

The data is displayed in the tool management. Additional tool status information, for example, can be stored here.

11.6 Magazine data

11.6.1 Overview of magazine data

Overview of magazine data

Number	System variable $TC_...	Description
1	MAP	Magazine description data (Page 137)
2	MAPC	User magazine description data (Page 140)
3	MPP	Magazine location data (Page 140)
4	MPPC	User magazine location data (Page 144)
5	MPTH	Magazine location type hierarchy (Page 145)
6	MDP	Distance to change position (Page 146)
7	MAMP	Magazine block data (Page 149)
8	MLSR	Magazine location assignment to spindles (Page 157)

11.6.2 Magazine description data $TC_MAP

$TC_MAPx[n]

Magazine description data

These data are used to describe and set the properties of the real magazine. The TMC block is used to configure the magazine data for the entire TO area. The settings of TM overwrite the settings of TMC.

x: = Parameter 1 ... 10

n: = Magazine number 1 ... 3200

OPI block TM

Calculation of line: Magazine number

Calculation of column: N/A

Magazine description data, tool management					
NC identifier	**Type**	**Designation**	**OPI variables**	**Type**	**Default setting**
		Magazine number	magNo	WORD	0
$TC_MAP2	String	Identifier of the magazine	magIdent	String	" "
$TC_MAP1	INT	Type of magazine **1** = Chain **3** = Turret **5** = Box magazine **7** = Tool buffer magazine **9** = Loading magazine	magKind	WORD	0
$TC_MAP3	INT	Status of magazine Bit 0: Active magazine Bit 1: Disabled Bit 2: Magazine is at load position Bit 3: Tool motion is active Bit 4: Magazine or tool may be moved. Enabled for loading Bit 6: Magazine is fixed-location-coded, i.e. tools in this magazine are considered fixed-location-coded tools and treated accordingly Bit 8: Edge location must not be overlapped left Bit 9: Edge location must not be overlapped right Bit 10: Edge location must not be overlapped top Bit 11: Edge location must not be overlapped bottom	magState	WORD	2
$TC_MAP4	INT	Reserved			-1
$TC_MAP5	INT	Reserved			-1
$TC_MAP6	INT	Number of lines (box magazines only)	magDim	WORD	1
$TC_MAP7	INT	Number of columns	magDim2	WORD	0
-	-	Number of locations for the magazine, corresponds to TC_MAP6*TC_MAP7	magNrPlaces	WORD	0
$TC_MAP8	INT	Current magazine position relative to change position	magActPlace	WORD	0
-			magCmd	WORD	
-			magCmdState	WORD	
-			magCmdPar1	WORD	
-			magCmdPar2	WORD	
$TC_MAP9	INT	Current wear grouping number	magWearCompoundNo	DINT	0
$TC_MAP10 (Bit 0 ... 7)	INT	Current tool search strategies of magazine (see $TC_MAMP2)	magToolSearchStrat	WORD	0
$TC_MAP10 (Bit 8 ... 15)	INT	Current empty location search strategy of magazine	magPlaceSearchStrat	WORD	0

$TC_MAP3

The magazine status **bit 3** (tool motion is active) is always reset when the software is booted with backup.

A magazine that has the status "Tool motion is active" cannot be deleted.

Empty locations are not sought in magazines with the "disabled" status. If a disabled magazine is explicitly defined for the empty location search, then the process is aborted with an error message.

A tool that is in a "disabled" magazine cannot be loaded into the spindle or the tool holder.

Overlapping edge locations (bit 8 ... 11=1)

For default setting (Bit 8 ... 11 = 0), an oversized tool extends beyond the magazine edge locations. If this has to be prevented (e.g. due to the mechanical situation at the machine), then one or several of these bits can be set.

Example: Bits 8 and 9 are set, bits 10 and 11 are not set.

This means that an oversized tool must not overlap the magazine (typically a box-type magazine) on its left and right sides, but an oversized tool may overlap / extend beyond the magazine on its top and bottom sides.

The following applies:

Left and top regarding a reference location is where the magazine locations with the lower magazine numbers are.

Right and bottom regarding a reference location is where the magazine locations with the higher magazine numbers are.

$TC_MAP7

The magazine locations are created if $TC_MAP7 is written to.

- The actual number of magazine locations is always equal to TC_MAP6*TC_MAP7.
- $TC_MAP7 can only be written to once, and more precisely from a default value of 0 a higher value. (However, it can always be written to the same value.)
- $TC_MAP7 can only be written to if the magazine type ($TC_MAP1) and the number of lines ($TC_MAP6) have a value greater than 0. A valid number (1, 3, 5, 7, 9) must be written to $TC_MAP1.
- $TC_MAP1 and $TC_MAP6 must be written to before $TC_MAP7 is written to.

$TC_MAP8

The current magazine position $TC_MAP8 is refreshed by the NC every time the magazine is moved.

When the magazine configuration has been loaded, variable $TC_MAP8 is assigned the value zero. The position value is the number of the magazine location that is located at the zero position of the magazine. As a maximum, the magazine position can have the number of magazine locations in the magazine. Larger or negative values are rejected.

$TC_MAP10

Magazine-specific tool search

The bit settings correspond precisely to the system variables $TC_MAMP2.

The default setting "0" always applies to buffer magazines, that is to say the search is made forwards from the 1st magazine location.

11.6.3 Magazine user data $TC_MAPC

$TC_MAPCx[n]

Magazine user data

Up to 64 user data can be additionally created for each magazine (V4.7 and higher).
Setting in the MD18090 MM_NUM_CC_MAGAZINE_PARAM and enable with MD18080 MM_TOOL_MANAGEMENT_MASK (set bit 2)

x: = Parameter 1 ... 64

n: = Magazine number 1 ... 32000

OPI block TUM

Calculation of line: Parameter number

Calculation of column: Magazine number

Magazine description data OEM user				
Name	**Type**	**Designation**	**OPI VAR**	**Type**
$TC_MAPC1			userData	DINT
...			userData	DINT
$TC_MAPC64			userData	DINT

11.6.4 Magazine location data $TC_MPP

$TC_MPPx[n,m]

Magazine location data

The following data describes the magazine location

x: = Parameter 1 ... 7

n: = Physical magazine number 1 ... 32000

m: = Physical location number 1 ... 1500

The maximum value of x is stored in OPI variable numMagPlaceParams in block Y.

OPI block TP

Calculation of line: (magazinLocNo-1)*numMagPlaceParams+parameterNo.

Calculation of column: Magazine number

Magazine location data, tool management					
NC identifier	**Type**	**Designation**	**OPI variables**	**Type**	**Default setting**
$TC_MPP1	INT	Location type 1 = Magazine location 2 = Spindle, tool holder 3 = Gripper 4 = Loader 5 = Transfer location 6 = Loading station 7 = Loading point	placeData	WORD	0
$TC_MPP2	INT	Location type > 0: Location type for virtual location = 0: Every tool fits in this location 9999: Not defined	placeData	WORD	9999
$TC_MPP3	BOOL	Consider adjacent location on/off	placeData	WORD	FALSE
$TC_MPP4	INT	Location state Bit 0: Disabled Bit 1: Free to accept a tool (occupied) Bit 2: Reserved for tool from buffer Bit 3: Reserved for new tool to be loaded Bit 4: Occupied in left half location Bit 5: Occupied in right half location Bit 6: Occupied in upper half location Bit 7: Occupied in lower half location Bit 8: Left half location reserved Bit 9: Right half location reserved Bit 10: Top half location reserved Bit 11: Lower half location reserved Bit 12: Wear group disabled Bit 13: Overlapping permitted. Tools can overlap disabled magazine locations. Only possible if consider adjacent location is active (see also MD 17520).	placeData	WORD	1
-		Reference of phys. magazine (r.t.)	placeData	WORD	0
$TC_MPP5	INT	Location type index (location type numbering) or wear group number	placeData	WORD	0
$TC_MPP6	INT	T no. of tool at this location	placeData	WORD	0
$TC_MPP7	INT	No. of adapter in mag. location	placeData	WORD	0

Magazine location data, tool management					
NC identifier	**Type**	**Designation**	**OPI variables**	**Type**	**Default setting**
$TC_MPP_SP	INT	For tool speed and acceleration monitoring Only relevant when $MC_TOOLHOLDER_MANAGEMENT > 0. Spindle no. which is linked to the tool holder.	placeData		
$TC_MPP66	INT	T no. of the tool, for which the identification "reserved for tool in buffer" is set. If a tool is transported into the spindle within the scope of changing from the magazine, the magazine location is reserved (identifier "reserved for tool in the buffer"), the T no. of this tool is entered into $TC_MPP6 and the tool has entered this location in the variable $A_MYMN/MYMLN.	placeData		

Writing magazine location data

Points to be noted about writing magazine location data:

When writing to a $TC_MPP... parameter for the first time, all of the magazine locations defined by magazine parameters are created with their default values (the memory for the locations is therefore "used up"). I.e. the magazine must have been defined by this time ($TC_MAP... parameter).

$TC_MPP1 (location type)

Only magazine locations of the type "Magazine location" ($TC_MPP1 = 1) may be defined at magazines that are not of the type "internal" ($TC_MAP1 = 7 or = 9).

Location types:

- 1 = Magazine location
  Only locations of type "1" can be defined on real magazines.
- 2 = Spindle / tool holder
- 3 = Gripper
- 4 = Loader
- 5 = Transfer location
  The distinction gripper / loader / transfer location is intended for HMI applications. Here, NC does not make a distinction, loader and transfer location are treated just like a gripper.
- 6 = Loading station
  After the tool moves to this location, the tool stays there. It can only be removed by explicit operation (unloading).
- 7 = Loading point

If a tool is moved from the magazine or buffer to this location, after the PLC acknowledgement of this motion command, the tool is automatically removed from this location.

Please note when writing the location status and number of the tool in this location that the following dependencies on $TC_MPP2 to $TC_MPP4 apply; these are checked during the write operation:

- If the location already contains a tool, the location type to be written must be checked against the tool location type.
- The status "not occupied" may only be written when none of the "assigned" states is set and there is no tool at the location.
- The "Disabled" state can be set irrespective of the other states.
- If there is no tool contained then the state "free" is automatically active; i.e. the state "not free" cannot be set by the NC program or PLC, HMI.
- "Occupied" states can only be set by the NCK within the scope of the adjacent location consideration; i.e. these states are ignored during writing by the NC program or PLC, HMI.
- The state "Reserved for tool from buffer" is set when a tool is removed by the NC from the real magazine during a tool change. This location is then not designated as "free" for tools other than the tool removed.
- The states "reserved for tool from buffer" and "reserved for new tool to be loaded" **of a location** are automatically reset when a tool is placed in this location.
- The states "reserved for tool from buffer" and "reserved for new tool to be loaded" of a **real magazine location** are automatically reset when a tool from this location is placed at a location in the loading/unloading magazine.
- The state "reserved for tool from buffer" is reset during an empty location search if the tool for which the empty location is being sought is assigned a magazine location other than its previous real magazine location. The newly found empty location is assigned the state "Reserved for tool from buffer" and becomes the new owner of the tool being sought.

The magazine location state "Reserved for tool to be loaded" is always reset when the control system is restarted. If "Consider adjacent location" is active, reservations of adjacent locations are also considered.

Users only have to deal with these rules if they wish to define the magazine directly at the NC program level. Data back-up is such that the rules are observed when data is imported to the NC.

$TC_MPP5 (location type index)

This data contains the spindle number for magazine locations of type "spindle" ($TC_MPP1) and is thus made known to the tool management.

The value cannot be changed for location type = 1 ($TC_MPP1; i.e. for all locations of internal magazines) if there is a tool at the location.

$TC_MPP6 (T No.)

- Tools can only be placed in magazine locations when both the tool and the magazine, plus its magazine locations, have been defined.

The tool may occupy only one magazine location!

Procedure:

It first attempts to find the tool associated with the T no.

- If it is already defined, then an attempt is made - subject to appropriate check procedure - to add it to the magazine location.
- If it is not yet defined, then an error has occurred.

Checks:

- The type of the tool to be placed must match the type of the location. If the type has not been set explicitly at the time of writing (default = 9999 = "Not defined"), then the tool is not placed.
- The state of the location must be "free" and must not be "disabled".
- If the value T no. = 0 is programmed, then this means that the present tool will be removed from the magazine location.
  Notice: $TC_MPP6 = 0 also changes the state of the location: a tool can only be placed in a magazine location if the location does not already contain a tool. The old tool might first have to be removed with $TC_MPP6 = 0.

Note

Because of this dependency of the individual data, it is mandatory for the T no. of the tool to be written as the last data in a magazine configuration. If you do not adhere to this sequence, default values might be set which may result in unwanted data.

11.6.5 Magazine location user data $TC_MPPC

$TC_MPPCx[n,m]

Magazine location user data

Up to 64 user data can be additionally created for each magazine (V4.7 and higher). Setting for number of parameters in MD18092 $MN_MM_NUM_CC_MAGLOC_PARAM and enable with MD18080 $MN_MM_TOOL_MANAGEMENT_MASK (set bit 2)

x: = Parameter 1 ... 64

n: = Magazine number 1 ... 30

m:= Magazine location number 1 ... 32000

OPI block TUP

Calculation of line: (m - 1)*numMagLocParams_u+Parameter no.

Calculation of column: Magazine number

OEM user magazine location data					
NC identifier	**Type**	**Designation**	**OPI variable**	**Type**	**Default setting**
$TC_MPPC1	INT		userplaceData	DINT	0
...	INT		userplaceData	DINT	0
$TC_MPPC64	INT		userplaceData	DINT	0

11.6.6 Magazine location type hierarchy $TC_MPTH

$TC_MPTH[n,m]

Magazine location type hierarchy

The location types can be organized in a hierarchy by programming these system variables.

n: = Index of the hierarchy, from 0 ... $MN_MM_MAX_NUM_OF_HIERARCHIES

m: = Index within the hierarchy n, location type 0 ... $MN_MM_MAX_HIERARCHY_ENTRIES

Magazine location types, refer also to $TC_TP7 and $TC_MPP2.

OPI block TT

Calculation of line: Number of location type+1

Calculation of column: Number of the location hierarchy+1

Magazine data: Magazine location type hierarchy					
NC identifier	**Type**	**Designation**	**OPI variable**	**Type**	**Default setting**
$TC_MPTH[n,m]	INT	Location type hierarchy n: Hierarchy 0 ... $MN_MM_MAX_NUM_OF_HIERARCHIES - 1 m: Location type 0 ... $MN_MM_MAX_HIERARCHY_ENTRIES - 1	placeType	WORD	9999

If a tool is to be loaded into the magazine, then the location type determines the availability of locations, i.e. $TC_TP7 and $TC_MPP2 must be defined.

If the location type of the tool is part of the location type hierarchy, then the location assignment is carried out in accordance with this hierarchy.

Several such hierarchies can be set up in one TO-area unit, With a conventional hierarchy, a location type may only be entered in one hierarchy.

Example

A chain magazine is to be split into 6 location types and the following hierarchy defined (the magazine No. is "1", the numbers of the location types are selected at random).

Location type_124 < Location type_3 < Location type_15 < Location type_1080 < Location type_5 <Location type_18

Definitions:

Magazine:

$TC_MPP2[magazine no., location]

$TC_MPP2[1,1..6] = 124

$TC_MPP2[1,7..12] = 3

$TC_MPP2[1,13..18] = 15

$TC_MPP2[1,19..24] = 1080

$TC_MPP2[1,25..30] = 5

$TC_MPP2[1,31..36] = 18

Hierarchy:

$TC_MPTH[0,0] = 124

$TC_MPTH[0,1] = 3

$TC_MPTH[0,2] = 15

$TC_MPTH[0,3] = 1080

$TC_MPTH[0,4] = 5

$TC_MPTH[0,5] = 18

If a tool of type_15 ($TC_TP) is loaded, it is preferable for it to be stored at locations 13 ... 18. If none of these locations are free, the search for an empty location continues, in accordance with the hierarchy, at locations of type_1080.

11.6.7 Distance to change position $TC_MDP

$TC_MDPx[n,m]

Distance from magazine zero

$TC_MDPx[n,m]=value

x: = 1: Loading magazine: Loading points, loading station (1st int. mag.)

x: = 2: Buffer magazine: Spindle, gripper, ...(2. int. mag.)

n: = Magazine no. of real magazine

m: = Location no. of internal magazine (loading point, ...).

Value: = Distance in number of locations

OPI block TPM

Calculation of line:

(LocNo.-1)*numPlaceMulti*numPlaceMultiParams+ParameterNo.

Calculation of column: Magazine number

Magazine data: Distance to change position				
NC identifier	**Type**	**Designation**	**OPI VAR**	**Type**
$TC_MDP1	INT	Distance between tool change position of magazine n and location m of 1st internal magazine (loading magazine, 9999)	multiPlace	WORD
$TC_MDP2	INT	Distance between tool change position of magazine n and location m of 2nd internal magazine (buffer storage magazine, 9998)	multiPlace	WORD

Description

Magazine position

The current magazine position is required for tool change, loading and unloading. This position refers to the magazine zero point established by the machine manufacturer. This is usually at the change position.

The number of the location at the magazine zero point has to be given during initialization. Otherwise, it is assumed that the non-existent location 0 is at the change position.

If the magazine is moved by a task, the current position is changed accordingly. The NC does not know how many positions the magazine has moved but knows the targets of the relevant commands. On the basis of the defined distance between and object (e.g. spindle 2) and the change position, the NC is able to update the current position.

Remark:

The value of the distance and the current magazine position are also evaluated for box magazines.

For empty location searches and tool searches, search strategies based on reference to the current magazine position convert the position contained in system variable $TC_MAP8 to the change position, loading point in each case at which the search is started. With search tasks, the NC always specifies internally which change position, loading point is to be used as reference for the search.

Assignment

By defining $TC_MDPx[n,m], a connection or assignment (distance relationship) is established between the real magazine and the buffers or the loading points. The HMI can only display assigned buffer locations and loading points. A tool transport within the scope of a t preparation or a change can only be realized via an assigned buffer (exception: asynchronous transfers). The tool search, for T preparation and change is also only made in those magazines assigned to a spindle or tool holder.

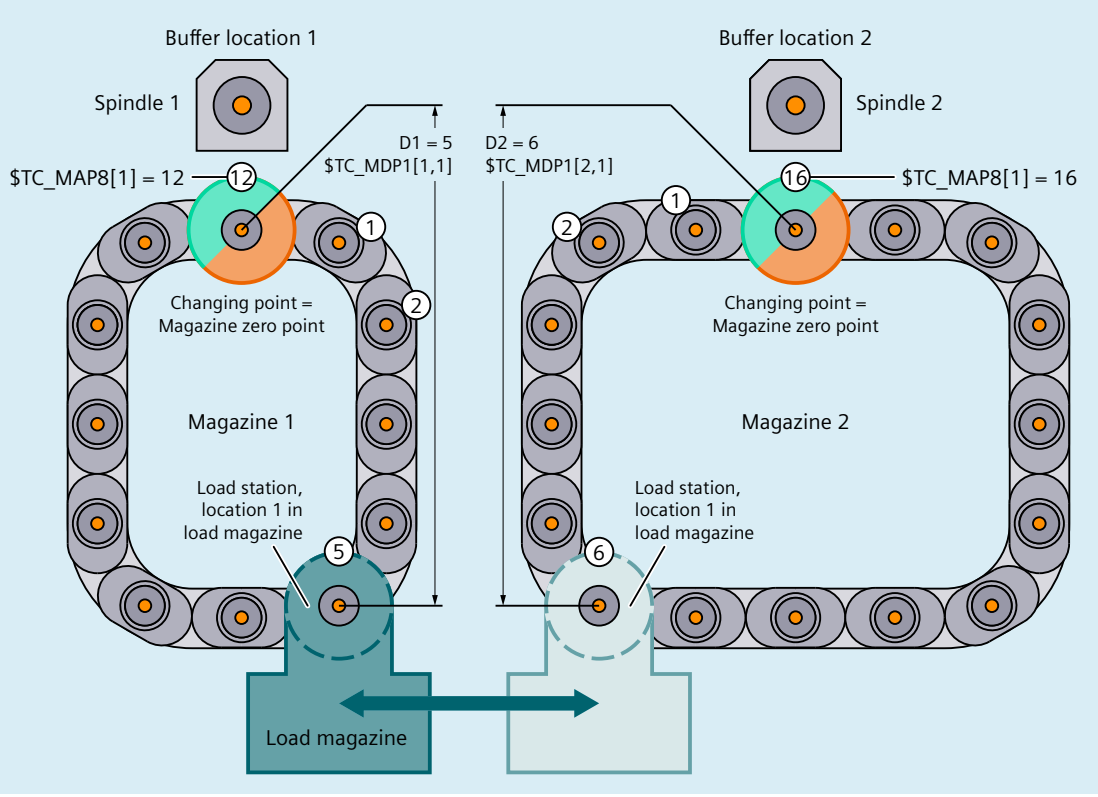
The maximum number of these distance relationships can be defined with machine data $MN_MM_NUM_LOCS_WITH_DISTANCE.

Example: 1st TO unit, 1 magazine, 1x spindle, double gripper, 2x loading point. This is a total of 5 distance relationships (3x $TC_MDP2 to assign the real magazine to the 3 buffers, 2x $TC_MDP1 to assign the real magazine to the two loading points).

Note

Command $TC_MDP2[n,m]=9999 can be used to dissolve a distance relationship.

Example:



D1 = Distance from magazine zero point to loading station -> 5 locations = $TC_MDP1[1,1]

D2 = Distance from magazine zero point to loading station -> 6 locations = $TC_MDP1[2,1]

Figure 11-4 Distance to change position $TC_MDPx[y,z]=value

The magazine zero point is usually the change position of the spindle. Therefore, the following applies:

- If location 1 is located at zero position, the current magazine position = 1 ($TC_MAP8[1]).

Examples for programming the distance to the zero position:

$TC_MDP1[1,1] = 5	Distance between location 1 of the loading station and the zero position of magazine 1
$TC_MDP1[2,1] = 6	Distance between the same location and the zero position of magazine 2

$TC_MDP2[1,1] = 0	Distance between location 1 of 2nd internal magazine and zero position of magazine 1
$TC_MDP2[2,2] = 0	Distance between location 2 of 2nd internal magazine and zero position of magazine 2

11.6.8 Magazine blocks $TC_MAMP

$TC_MAMPx

Magazine block data

x: = Parameter 1, 2, 3

OPI block TMC

The TMC block is used to configure the magazine settings for the entire TO area. The settings may partly be overwritten by the magazine-specific settings of TM.

Calculation of line: N/A

Calculation of column: N/A

Magazine block data, magazine control block					
NC identifier	**Type**	**Designation**	**OPI variables**	**Type**	**Default setting**
$TC_MAMP1	String	Identifier of the configuration magazine	magCBIdent	String	" "
		Number of the loading magazine	magBLMag	WORD	
		Number of the buffer magazine	magZWMag	WORD	

Magazine block data, magazine control block					
NC identifier	**Type**	**Designation**	**OPI variables**	**Type**	**Default setting**
$TC_MAMP2	INT	Type of tool search (bits 0 ... 7) and type of empty location search (bits 8 ... 16) Bit 0=0: Default strategy Take the first available tool found in the tool group. First search in the magazine from which the last change took place. Bit 0=1: Select the "active" tool in the magazine of the previously changed tool; otherwise search for replacement tool with lowest duplo number. If no tool is found in this magazine, the search is continued in the other associated magazines. Bit 1: Search for the next replacement tool at the shortest possible distance from the current magazine position. Bit 2: Select the "active" tool, otherwise the replacement tool with the lowest number in $TC_TP10 (does not work with manual tools). Bit 3: Search for the tool in the group with the **lowest actual value** of the monitored quantity. Bit 4: Search for tool in the group with the **highest actual value** of the monitored quantity.	magSearch		0
$TC_MAMP2	INT	Bit 5: Reserved Bit 6: First search in the magazine currently being considered (this is only effective in conjunction with bit 7=1) Bit 7=0: Start the tool search in the magazine where the tool last changed came from Bit 7=1: Always start the search in the 1st magazine of the distance table Bit 7= 1 + Bit 6= 0 + Bit 0=1 or Bit 2=1, if no "activeTool" is found in the magazine, then, if available, the active tool is selected from another magazine linked with the tool holder	magSearch		0

Magazine block data, magazine control block					
NC identifier	**Type**	**Designation**	**OPI variables**	**Type**	**Default setting**
$TC_MAMP2		Bit 8 Search forwards starting at first location number Bit 9 Search forwards starting at current magazine position Bit 10 Search backwards starting at last location number			
$TC_MAMP2		Bit 11 Search backwards starting at current magazine position Bit 12 Symmetrical search starting at the current magazine position Bit 13 (1:1 exchange) behavior If the 1:1 exchange is not possible, the search strategy for an empty location is considered "symmetrical". The 1:1 exchange acts in addition to other set search strategies. If possible, the 1:1 exchange is treated as a priority. Bit 14 = 0 Search within individual magazines first. If no possible location is found, search in the next magazine for a free location for the tool. Bit 14 = 1 Search in all magazines for the best location according to the hierarchy for the tool.			

Magazine block data, magazine control block					
NC identifier	**Type**	**Designation**	**OPI variables**	**Type**	**Default setting**
		Bit 15 = 0 (conventional hierarchy type): For this type, the location type of the tool conducting the search is searched for in the table of system variable $TC_MPTH. If the location type is found, this hierarchy is taken and evaluated from this level to the end. Bit 15 = 1 (alternative hierarchy type): Location type hierarchies can be defined for the listed location types 1, ..., $MN_MM_MAX_NUM_OF_HIERARCHIES. The hierarchy for location type 1 is defined by $TC_MPTH[0,n], for location type 2 by $TC_MPTH[1,n], and so on. (n: index within a hierarchy). With this setting, a location type can be defined in different hierarchies.			
$TC_MAMP2	INT	Bit16: The mini hierarchy, which consists only of the location type itself and location type 0, cancels consideration of hierarchy. In this case, the empty location search does not distinguish between a suitable location type ($TC_TP7 == $TC_MPP2) and the general location type "0" of the magazine location.			
$TC_MAMP3	INT	Procedure for tools in a wear group (bit 0...7) Search strategies for wear groups (bit 8...15) Bit 0=0: When a wear group is activated, the tool status remains unchanged Bit 0=1: When a wear group is activated, the tool status is changed. One tool from each tool group becomes active	modeWearGroup	WORD	0

Magazine block data, magazine control block					
NC identifier	**Type**	**Designation**	**OPI variables**	**Type**	**Default setting**
$TC_MAMP3		Bit 1=0: When a wear group is disabled, the tool status remains unchanged Bit 1=1: When a wear group is disabled, the tool status is changed Bit 2...7 reserved Search strategy for next wear group Bit 8=0: Find the next possible wear group Bit 8=1: Find the wear group with the next highest group number that can be activated Bit 9...11 reserved			
$TC_MAMP3		Search strategy within the tool group for the tool to be set to active (NC command SETTA or PI_SETTST) Bit 12=0: Lowest possible duplo number Bit 12=1: Lowest possible magazine location number Bit 13=1: Lowest number contained in $TC_TP10 (substitute order)			

Interaction of the bits for the tool search strategy

The following strategies always start the search in the first magazine of the distance table.

$TC_MAMP2 =		**Meaning**
Bit	**Hex value**	
7 + 0	'H80'	Otherwise, the same as value 0
7 + 0	'H81'	Otherwise, the same as value 1 - however, if an "active" tool is not found in the magazine, - if available - the "active" tool is selected from another magazine linked with the tool holder. If no tool with the "active" state is available, a search is made for the replacement tool with the lowest duplo number.
7 + 6 + 0	'HC1'	Otherwise, the same as value 1 - however, if an "active" tool is not found in the magazine, a search is made in the same magazine for any tool that can be used; if no tool is available, a search is made in another magazine linked with the tool holder.
7 + 1 7 + 6 + 1	'H82' 'HC2'	Otherwise, the same as bit 1 =1 ('H2')
7 + 2	'H84'	Otherwise, the same as bit 2 = 1 ('H4') - however, if an "active" tool is not found in the magazine, - if available - the "active" tool is selected from another magazine linked with the tool holder. If no tool with the "active" state is available, a search is made for the replacement tool with the lowest number in $TC_TP10.
7 + 6 + 2	'HC4'	Otherwise, the same as bit 2 =1 ('H4') - however if no "active" tool is found in the magazine, a search is made in the same magazine for a tool that can be used, if available, a search is made for the replacement tool in another magazine linked with the tool holder with the lowest number in $TC_TP10.

$TC_MAMP2 =		Meaning
7 + 3 7 + 6 + 3	'H88' 'HC8'	Otherwise, the same as bit 3 =1 ('H8')
7 + 4 7 + 6 + 4	'H90' 'HD0'	Otherwise, the same as bit 4 =1 ('H10')

Search operation for $TC_MAMP2, bit 0=1, bit 6=0, bit 7=1

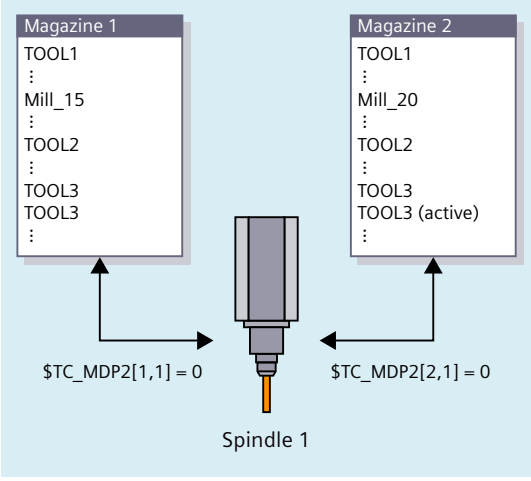
A search is made for the appropriate tool in the magazine according to the tool search strategy that has been set (defined using parameter $TC_MAMP2). The search is considered to have been successful if a tool was found according to the search strategy.

Search operation for $TC_MAMP2, bit 0=1, bit 6=1, bit 7=1

A search is first made for the appropriate tool in the magazine according to the tool search strategy that has been set (defined using parameter $TC_MAMP2). The search is considered to have been successful if a tool according to the search strategy was found - or all tools of the group were checked and no tool was found according to the search strategy - but in spite of this, a tool that can be used was found among the tools that were searched.

If a search is made according to the strategy, bit 2=1 (sequence of use $TC_TP10) then the same principle applies. The criterion is also "active tool" and instead of the duplo number, the number in $TC_TP10.

Example to clarify the principle of operation of bit 6 and bit 7



Assumption:

- All tools can be used
- Basic setting of the search strategy is bit 0 = 1 - search for the active tool, if there is no active tool, take the tool with the lowest duplo No.
- Spindle_1 is linked to both magazines, magazine_1 is the first in the distance table

Example_1:

Bit 6 = 0
Bit 7 = 0

T="Mill_15"	M06	There is only one "Mill_15" tool, it was found in magazine_1 and was loaded
...		
T="Tool3"	M06	The last change was performed from magazine_1. Therefore, the search is first made in magazine_1. There is no active "Tool3" there, but there are two tools in this group that can be used. The tool with the lowest duplo no. is selected. Magazine_2 is no longer considered.

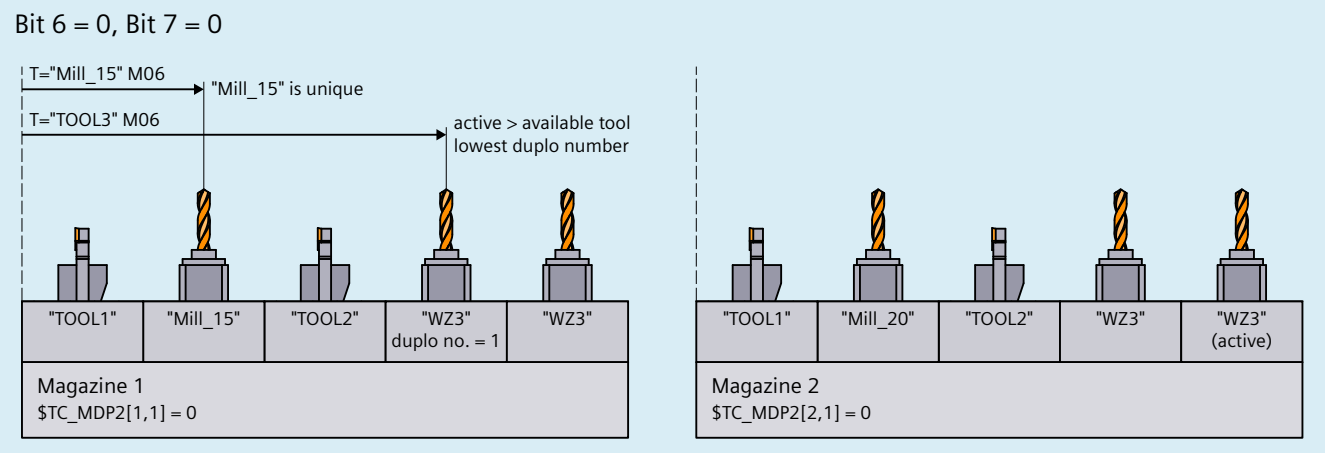


Figure 11-5 Bit 6 = 0, Bit 7 = 0

Example_2:

Bit 6 = 0
Bit 7 = 1

T="Mill_20"	M06	There is only one "Mill_20" tool, it was found in Magazine_2 and was loaded
...		
T="Tool3"	M06	The last change was performed from Magazine_2. As a result of bit 7 = 1, the tool search is however started in Magazine_1. But this setting also means that the search strategy "active tool" has priority over a magazine-specific approach. This means that: No active tool from the group "Tool3" was found in Magazine_1, i.e. the search is continued in the next magazine (corresponding to the sequence in the distance table). An active "Tool3" is there - and this is selected.

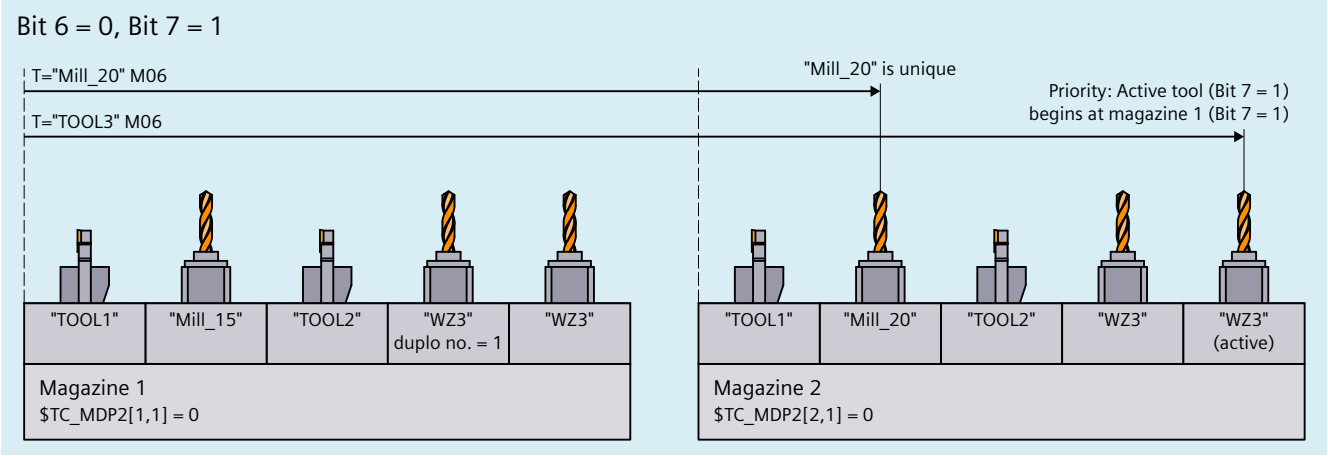


Figure 11-6 Bit 6 = 0, Bit 7 = 1

Example_3:

Bit 6 = 1
Bit 7 = 1

T="Mill_20"	M06	There is only one "Mill_20" tool, it was found in Magazine_2 and was loaded
...		
T="Tool3"	M06	The last change was performed from Magazine_2. As a result of bit 7 = 1, the tool search starts in Magazine_1. However, no tool of this group is present there. As a result of bit 6 = 1 (considers the actual magazine with priority), a search is now made for a "Tool3" that can be used in Magazine_1 (for this search, this is also the actual magazine); this tool is also found.

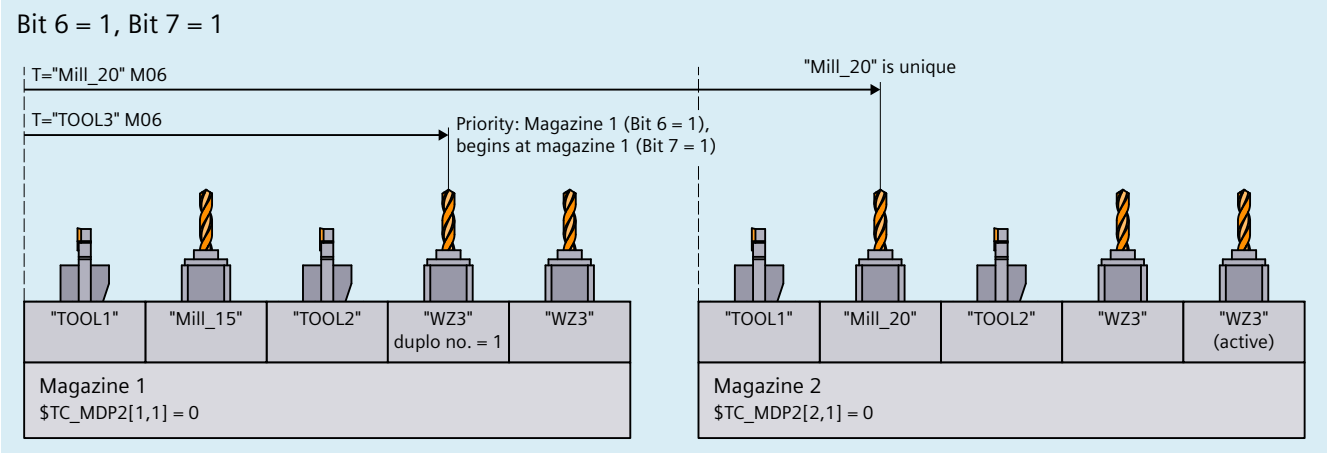


Figure 11-7 Bit 6 = 1, Bit 7 = 1

$TC_MAMP2

By default, the tool search is always performed magazine-specifically, starting in the magazine from which the last change was made.

With bit 7 there is an additional option to select the tool search.

Bit 7 = 1

The search always starts in the 1st magazine of the distance table. If the search for the active tool is set using either bit 0 or bit 2, a search is made for the active tool across all of the magazines linked with the spindle. A search is only made for a replacement tool if no active tool was found in all of these magazines.

Note

The tool sequence in a tool group is not defined (e.g. ascending duplo number). This means, if the default strategy is used for the search (MAMP2=0), any tool that is capable of being used is found.

11.6.9 Assignment of buffers to spindles $TC_MLSR

$TC_MLSR[x,y]

Assignment of buffer locations to spindles - $TC_MLSR[x,y]

x: = Location no. in buffer 1 ... 32000

y: = Location no. of the spindle in buffer magazine 1 ... 32000

OPI block n.a.

Calculation of line: N/A

Calculation of column: N/A

NC identifier	Type	Designation	OPI variable	Type	Default setting
$TC_MLSR[x,y]	INT	System variable for assigning magazine locations of the buffer magazine to the spindle	-	-	0

Description

This assignment is important when searching for tools. The search is always made, viewed from the requesting spindle, initially in the associated buffers, then in the assigned magazines. This means that a tool, which is located in the buffer, can only be found if there is an assignment to the spindle via parameter $TC_MLSR.

The number of possible links between the spindle and the buffer can be set using machine data $MN_MM_DIST_REL_PER_MAGLOC.

Example: 1st TO unit, 1 magazine, 1x spindle, double gripper, 2x loading point. The double gripper is linked to the spindle via $TC_MLSR[2,1]=0 and $TC_MLSR[3,1]=0. The correct value for the MD is "2".

The sequence when programming is decisive for the sequence when automatically returning tools.

Note

The content value of the system variable is not evaluated. The assignment is defined via indices x and y. In order to check via the part program whether a certain assignment exists, a read operation has to return the value zero.

Note

No more than 16 magazines or buffer locations can be assigned to one spindle.

Magazine distance to buffer via tool holder / spindle

$TC_MDP2 and $TC_MLSR establish a relationship between the buffer locations and magazines.

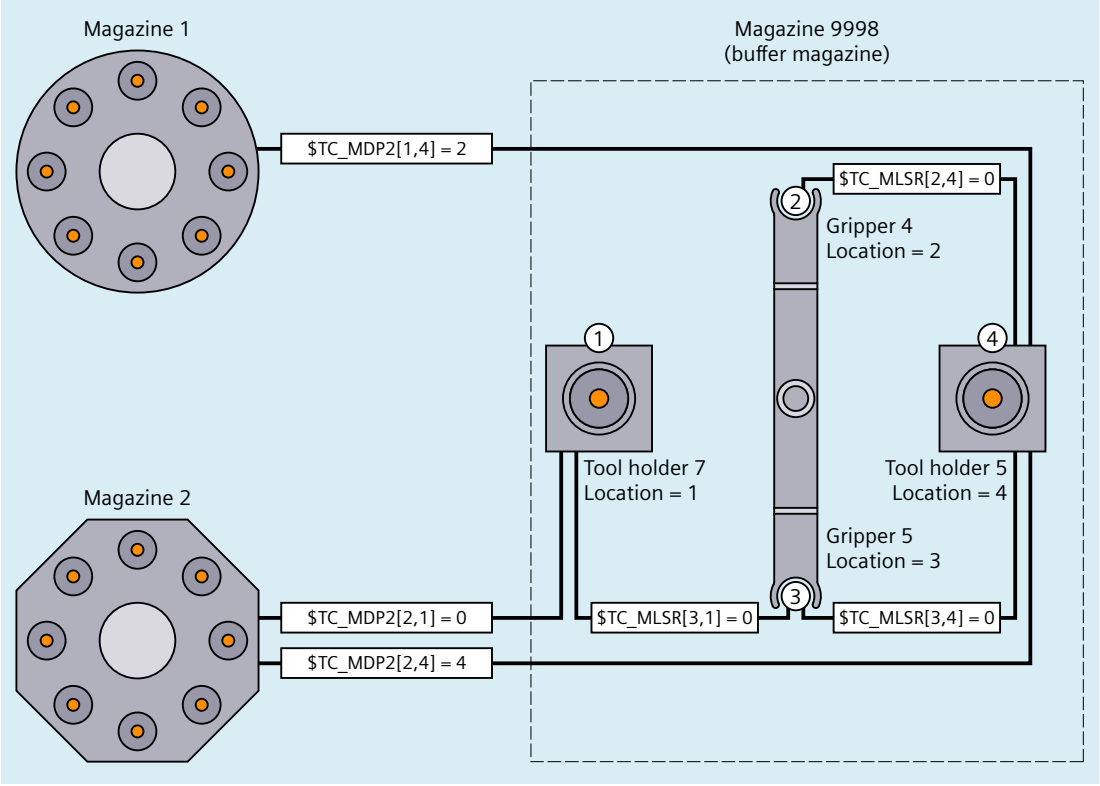


Figure 11-8 Magazine distance to the buffer

Configuration

Two magazines are defined with numbers 1 and 2.

Four locations 1, 2, 3 and 4 are defined in buffer 9998; two tool holders 5, 7 and two grippers 4, 5.

```
$TC_MPP1[9998,1] = 2          ;Location type = spindle or tool holder
$TC_MPP5[9998,1] = 7          ;Tool holder no. = 7
$TC_MPP1[9998,2] = 3          ;Location type = gripper
$TC_MPP5[9998,2] = 4          ;Gripper no. = 4
$TC_MPP1[9998,3] = 3          ;Location type = gripper
$TC_MPP5[9998,3] = 5          ;Gripper no. = 5
$TC_MPP1[9998,4] = 2          ;Location type = spindle or tool holder
$TC_MPP5[9998,4] = 5          ;Tool holder no. = 5
```

Both grippers are linked to tool holder 5 via $TC_MLSR. They do not require their own distance definition for the magazines. They are linked to the magazines via tool holder 5 with the distance relations defined there. However, it is also possible to define separate distance relationships for the grippers.

Tool holder 5 is linked with both magazines via $TC_MDP2.

Tool holder 7 is only linked with magazine 2; gripper 5 is assigned to it.

11.7 Multitool data

11.7.1 $TC_MTP - multitool data

Overview

A multitool is a combination of several tools to form a unit. The following parameters allow a multitool to be described so that it fits into the classification of the tool magazine function.

All of the parameters in the following table have no physical units. The default access right is "keyswitch position 0" - for access operations from the NC program as well as for access operations from the OPI. Access rights can be restricted using the REDEF command.

`$TC_MTPx[y]`

x: Parameter 1 ... 8, ... PROTA

y: = Multitool number MT = 1 ... 32000

NC identifier	Description	Format	Preassignment, limit values
`$TC_MTPN`	The number of locations is defined as delete command	INT	2 2-$MN_MAX_TOOLS_PER_MULTITOOL
`$TC_MTP2`	Identifier	STRING	MT number
`$TC_MTP3`	Size to the left in half locations (= towards a lower location number)	INT	1 1 ... 11

NC identifier	Description	Format	Preassignment, limit values
$TC_MTP4	Size to the right in half locations (= towards a higher location number)	INT	1 1 ... 11
$TC_MTP5	Size upwards in half locations (= towards a lower location number)	INT	1 1 ... 11
$TC_MTP6	Size downwards in half locations (= towards a higher location number)	INT	1 1 ... 11
$TC_MTP7	Magazine location type	INT	9999 (not defined) 0 ... 32000
$TC_MTP8	State; bit-coded	INT	0 (not enabled) 0 ... "0x3FFFF"
$TC_MTP_POS	MT position. The following applies: Position 1 is defined as "location 1 in the machining position"	INT	0 (not defined) 0-$MN_MAX_TOOLS_PER_MULTITOOL
$TC_MTP_KD	Type of distance coding	INT	0 (not defined) 1 (= location number) 2 (= length) 3 (= angle)
$TC_MTP_PROTA	Name of the protection zone or name of the file, which contains the description of the protection zone.	STRING	""(= not defined)

$TC_MTPN - number of MT locations

The multitool does not have a duplo number like the tool. The parameter $TC_MTPN contains the number of locations, which the multitool offers to accept tools. In addition, by programming $TC_MTPN[mtNo] = 0 the multitool is deleted. After MT locations have been generated, the parameter can no longer be changed.

A multitool is similar to a chain magazine with regard to its locations, i.e. if there are tool search strategies or empty location search strategies with results that depend on the type of magazine, the property can be like that of a chain magazine.

$TC_MTP2 - MT name

Multitools can be assigned names so that a multitool can be uniquely defined by its name or by its number. The rules that apply when assigning names are the same as those that apply to tool names. Multitool names must be different from tool names and magazine names within a TO unit.

$TC_MTTP3 $TC_MTP6 - MT size

The size data of a multitool is essentially the same as the size data of a tool - and is used to find an optimally suitable magazine location for the multitool when loading into the magazine (empty location search) when consider adjacent location is active.

The rules for changing the multitool size are the same as the rules for changing the tool size, i.e. the MT size of a multitool can be changed so long as the MT is not assigned to an owner location.

$TC_MTP7

The magazine location type of the multitool is essentially the same as the magazine location type of a simple tool and is required for the empty location search, empty location check (loading operations into the magazine, tool change). The permitted values and rules to change data correspond to those that apply for the corresponding data in the tool. The value cannot be changed if the multitool is loaded into the magazine.

For multitools that are contained in the loading magazines or buffer magazines and still do not have an owner location (i. e. that are logically valid as unloaded – $A_MYMN=0, $A_MYMLN=0), the magazine location type can be changed. The magazine location type of a multitool can be changed, as long as the multitool is still not assigned to an owner location.

$TC_MTP8 - MT state

The state of a multitool corresponds to the state of an individual tool. The defined states are the same as those of an individual tool as defined under $TC_TP.

The following additional definitions apply:

Bit	Meaning	Comment
0	Active	Reserved Multitools do not have the "active" state defined. Only tools contained in them can assume the state.
1	Released	Equipped multitools have this state, so that a tool in them can be selected.
2	Disabled	Multitool can be disabled. Independent of the state of the contained tools, the selection of these tools can be prevented. The PLC signal "ignore disabled state" also ignores the disabled state of the multitool. The tool change command TCA for a tool ignores the disabled state of the multitool.
3	measured	Reserved Multitools cannot be measured. Only tools contained in them can assume the state.
4	Prewarning limit reached	Reserved Multitools cannot be monitored. Only tools contained in them can assume the state.
5	is being changed	Protect the multitool during tool selection from competing additional tool selection/MT motion. This means that it is not possible to select a tool within the MT, as long as this state is set due to the selection of a tool in the MT. If the state is set, it is permitted to select another tool from the MT, if this selection regarding the same toolholder is realized just like the selection, that had set the state of the MT. In this case, there is no competing tool selection, but instead a replacing one. If the state is set, then a tool in this MT also has this state set. The state is reset again together with that of the tool. Bit 21 from $TC_TOOL_MANAGEMENT_MASK refers to the tool and the MT. The state is reset at POWER ON.
6	fixed-location-coded	Multitools can be fixed-location-coded
7	was in use	The state of multitools is defined the same as that of the tool. After unloading a tool from the tool holder, the MT and the old tool are assigned the state.

Bit	Meaning	Comment
8	autom. return transport	Regarding the tool transport, multitools behave just the same as basic tools. $TC_TOOL_MANAGEMENT_MASK, bit15 also refers to this state.
9	ignore locked	The state is internally set and reset. The rules are essentially the same as the rules for the tool.
10	MT is to be unloaded	Multitools can be selected for unloading
11	MT is to be loaded	Multitool can be selected for loading
12	Reserved	Reserved
13	Reserved	Reserved
14	Selected for 1:1 exchange	Multitool is selected for the pending 1:1 exchange. The rules as for the tool apply.
15	Manual tool	The multitool is used as manual tool. The state is set and reset again for the MT as part of the tool selection. The tool to be selected and its MT change the state synchronously. The tools not involved with the tool selection do not experience a state change.
16	MT is locked if a tool in the MT is locked	If a tool contained in the MT has made the state transition to "disabled", then the MT also assumes this state (bit 2). After a tool has made the state transition to "not disabled", and then there is no longer any tool in the multitool that has the "disabled" state, then the MT itself also loses the "disabled" state. Loading/removing a disabled / enabled tool to the MT or from the MT can change the MT state "disabled/enabled". Example: Loading to a disabled tool corresponds to an MT state transition to "disabled".

Note

Reserved or non-defined MT status bits cannot be written to. Writing to such a bit results in Alarm 17050 "illegal value".

Interdependencies between the tool and multitool states

Tools, which are loaded in the multitool, can influence the state of the multitool and for MT transport operations, the MT can influence the tool status of the loaded tool.

Bit	Meaning	Comment
1	Released	Users allocate tools and multitools a status independent of one another. (Enabled tools can be loaded into a multitool that has not been enabled. Tools that have not been enabled can be loaded into an enabled MT.)
2	Disabled	Changing this state is initiated from the tool as part of the tool monitoring and the command RESETMON . Examples 1a, 1b, 2a, 2b at the end of this table show how the state of loaded tools can influence that of the multitool. User-specific changes to the MT state are not transferred to the loaded tools.
5	is being changed	The state is set and reset again for the MT as part of the tool selection. The tool to be selected and its MT change the state synchronously. The tools not involved with the tool selection do not experience a state change. The state is always reset at POWER ON.
6	fixed-location-coded	Tools loaded in the MT are implicitly fixed-location-coded, i.e., the state has no significance for the tool in the MT. For the MT itself, the state is defined; however, this has no influence on the loaded tools. If the tool state changes, this has no effect on the state of the multitool.

Bit	Meaning	Comment
7	was in use	The state is automatically set as part of the tool change for the MT and the tool in the MT involved in the change. Tools in the MT, not involved in the tool change, do not change the state. The user is responsible for the reset. If the last tool contained in the MT loses this state, the MT also loses this state. If the user deletes the multitool state, then the states of all of the tools contained in it are also deleted. If the user sets this MT state, it has no effect on the loaded tools in the MT.
8	autom. return transport	At the tool change, the state is automatically set and reset for the MT only, however not for the tools contained in the MT.
9	ignore locked	The state is set and reset automatically. The state is set and reset for the tool to be loaded and for the MT in which this tool is loaded. For the other tools not involved in the change, the state remains unchanged.
10	MT is to be unloaded	When unloading from the magazine, the MT and the its loaded tools lose this state. If a tool that has this state set is removed from the MT, then the state in the tool that has been removed is reset.
11	MT is to be loaded	When loaded to a magazine location, the MT and the tools loaded in it lose this state. If a tool that has this state set is loaded into an MT, then the tool state: – is not changed, if the MT has the state set – is reset, if the MT does not have this state set.
12	Master tool	The user allocates the state. The tool state does not influence the MT – and vice versa.
14	selected for 1:1 exchange	This state is not required for tools loaded in the MT. As part of the empty location search, it is only automatically set for the MT and reset again at the end of the command.
16	MT is locked if a tool in the MT is locked	See examples 1a, 1b, 2a, 2b at the end of this table. Notice If tools are in the MT at the time of the state change, the following applies: When the state is changed, the state of the loaded tools is checked and the MT state "disabled" set or reset. Notice The loading/removing of a tool to the MT / from the MT causes the states of the tools loaded in the MT to be checked loading/removing has been completed and the MT state "disabled" is set accordingly. See examples 4a, 4b.

Example 1a:

Bit 2 = 0 and **bit 16 = 1**

The MT has three tools that are not disabled and have been enabled. The MT itself is also not disabled and has been enabled. Bit 16 = 1 is set.

If the "disabled" state of any tool in this MT is set (either manually or automatically), the "disabled" state will also be set in the MT, although there are still two tools for which the "disabled" state has not been set (i.e., these tools can still be used).

Example 1b:

Bit 2 = 1 and **bit 16 = 1**

The MT has three tools that are disabled and have been enabled. The MT itself is also disabled and has been enabled. Bit 16 = 1 is set.

If the "disabled" state of two of the three tools is deleted in the MT, then the "disabled" state of the multitool itself remains unchanged. If, for the last tool with the "disabled" state, the

"disabled" state is deleted (either manually or automatically), then the "disabled" state is also deleted in the MT.

Example 2a:

Bit 2 = 0 and **bit 16 = 0**

The MT has three tools that are not disabled and can be used. The MT itself is also not disabled and has been enabled. Bit 16 = 0 is set.

If the "disabled" state of two of the three tools is set in the MT (either manually or automatically), then the status value of bits 2 in the MT remain unchanged. This means that the last "not disabled" tool of the multitool can still be loaded. The MT is only set to the "disabled" state when this last tool also enters the "disabled" state.

Example 2b:

Bit 2 = 1 and **bit 16 = 0**

The MT has three tools that are disabled and can be used. The MT itself is also disabled and has been enabled. Bit 16 = 0 is set.

If the "disabled" state of a tool in the MT is deleted (either manually or automatically), then in addition, the "disabled" state in the MT is also deleted, independent of the state of the other tools in the MT.

Bit 16 = 0 is advantageous, for example, if the MT only contains identical tools. All tools can then be used until their relevant monitoring limit has been reached.

Bit 16 = 1 is advantageous, for example, if an MT is to be loaded and the tools it contains are different and have to perform various machining operations. If one of these tools is now disabled due to the monitoring function, then another MT must be loaded, which contains a complete set of tools that are ready for use for this particular machining operation.

A graphic representation of the "disabled" state coupling is again shown in the following two diagrams:

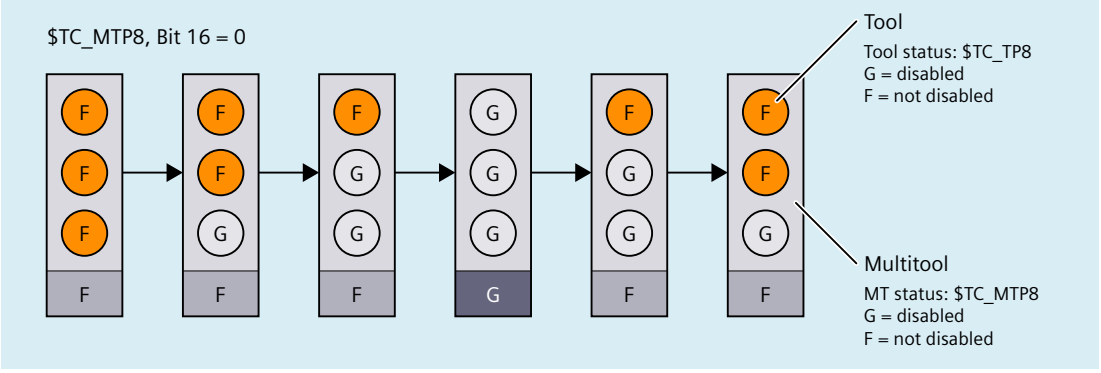


Figure 11-9 Automatic state change of the multitool depending on the state of the loaded tools and bit 16 of $TC_MTP8

Example 3a:

Bit 16 = 0:

If in the figure above bit 16 is changed from 0 to 1 at the time when the state is the same as shown in the fifth MT state figure from the left (MT with F), then the state of the MT changes from F to G because at least one tool with the state G has been loaded after the bit value

change and therefore the MT also assumes the state G. When the bit value changes from 0 to 1, the figure changes to the following.

Example 4a:

Bit 16 = 0:

If in the third MT state figure from the left (MT with F), the tool with F is removed from the multitool, the state of the MT changes from F to G because only two tools with the state G are loaded after the tool removal and therefore the MT also assumes the state G.

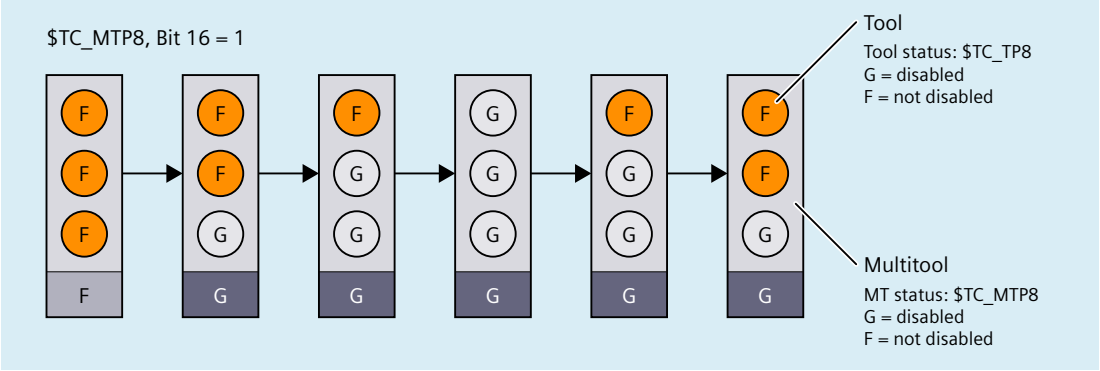


Figure 11-10 Automatic state change of the multitool depending on the state of the loaded tools and bit 16 of $TC_MTP8

Example 4b:

Bit 16 = 1:

If in the second MT state figure from the left (MT with G), the tool with G is removed from the multitool, the state of the MT changes from G to F because only two tools with the state F are loaded after the tool removal and therefore the MT also assumes the state F.

$TC_MTP_POS - MT position

The position of the multitool is the MT location number of the location, which is in the machining position, and which typically contains the machining (active) tool; see the next figure.

With the final acknowledgement (PLC status value = 1) of a tool change command for a tool in the MT, the MT position is set to the value of the MT location number of the MT location, which contains the tool that was loaded. The other PLC status values (e.g. status_105) do not change the MT position value.

With the end acknowledgement (PLC status value = 1) of an MT positioning command (POSMT, _N_POSMT), the MT position is set to the value of the programmed MT location number. The other PLC status values are not permitted for the MT positioning command and are rejected with Alarm 6405 "incorrect acknowledgement data".

With an asynchronous transfer control, the PLC can position the multitool.

If a command output to the PLC is not acknowledged with "end" (e.g. cancel, status_3), then the MT position in the NC is not updated (even if the positioning has already been mechanically carried out).

The MT position of a multitool, which is not located in a toolholder, has no further significance. The exception is programming T="magazine location number", which is typically used for turrets.

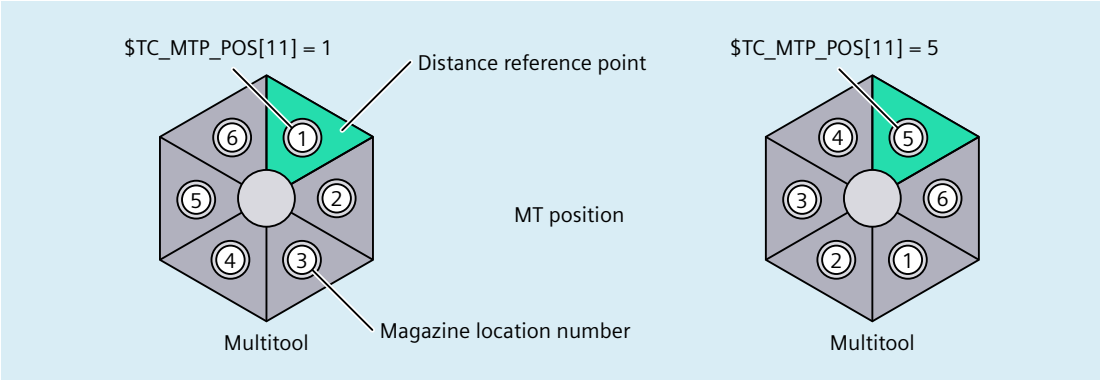


Figure 11-11 The location on the machining position determines the MT position

Example 1:

An MT is moved to the tool holder with a tool motion command (e.g. MVTOOL). The parameters are set according to the current MT position. With the end acknowledgement of the MT motion command, the PLC can inform the NC whether and how the position of the multitool is to be updated.

- The FC6 is parameterized with MultitoolLoc = 0. NC does not perform a position synchronization, the mechanical position can differ from the data position.
- The FC6 is parameterized with MultitoolLoc = 1 (valid MT location number). This means that the NC synchronizes the MT position with the acknowledged position.

If subsequently a tool change is programmed for a tool of this MT on the tool holder, a tool change command is output to the PLC with which the MT positioning data is sent to the PLC. With the end acknowledgement of the tool change, the MT position is updated to the value that belongs to the tool that is to be loaded. The PLC must have performed the MT positioning before the end acknowledgement is sent to the NC. The PLC must not change the MT position for the tool selection, tool change commands specified by the NC in the acknowledgement data. The attempt is rejected with alarm 6405 "Command has invalid PLC acknowledgement parameter".

Example 2:

An MT is moved to the tool holder with a tool motion command (e.g. MVTOOL). As in example 1, the MT positioning/synchronization depends on the PLC application.

If subsequently a RESET is executed, then – when RESET is configured with "Activate tool on the main tool holder" – a tool change command is output to the PLC with which the data of the tool to be activated and the MT positioning data is sent to the PLC. With RESET, the tool to be activated is specified in the NC as the tool at the MT location displayed at the MT position output to the PLC. This means that in this example, which tool is activated at RESET depends on the PLC logic for the tool motion on the main tool holder. If a different tool is to be activated, this must be specified in the application, for example by positioning the MT appropriately first with the NC program command POSMT (or similar PI _N_POSMT by the HMI or PLC). It is also possible to set the MT position in the NC directly to the desired value before the RESET (by programming parameter $TC_MTP8 or by writing the position via

the operator panel interface (by the HMI or PLC), or the PLC positions the multitool with an asynchronous transfer).

$TC_MTP_KD - type of distance coding for the MT locations

The location distances (**K**indOf**D**istance) within the multitool can be defined in a variety of ways:

$TC_MTP_KD[y] = 1	The distance coding uses the MT location number. The location number is automatically allocated when creating the MT locations. $TC_MTPPL, $TC_MTPPA cannot be written except with the value 0.0.
$TC_MTP_KD[y] = 2	The distance coding is via length specifications. $TC_MTPPL can be written, $TC_MTPPA cannot be written, except $TC_MTPPA[...] = 0.0.
$TC_MTP_KD[y] = 3	The distance coding is via angle specifications. $TC_MTPPA can be written, $TC_MTPPL cannot be written, except $TC_MTPPL[...] = 0.0.

The type of distance coding can only be changed as long as no MT locations have been created. Otherwise Alarm 6438 "Inconsistent data modification is not permitted" is generated.

Example:

If, for distance coding, the location number has been defined ($TC_MTP_KD[y] = 1), then writing $TC_MTPPL, $TC_MTPPA with values other than 0.0 results in alarm 6464 "Command cannot be programmed for the current multitool distance coding".

Example:

The multitool is defined. Although the number of MT locations has been defined, they have still not been created.

```
$TC_MTP_KD[200] = 1     ;subsequently writing to $TC_MTPPL or $TC_MTPPA results in
                        ;Alarm "Command cannot be programmed for the defined multi-
                        tool distance coding"
$TC_MTP_KD[300] = 2     ;subsequently writing to $TC_MTPPL is permitted.
                        ;subsequently writing to $TC_MTPPA results in Alarm
                        ;"Command cannot be programmed for the defined
                        ;multitool distance coding"
```

The MT locations are now created, e.g. by writing to the system of parameters:

```
$TC_MTPPL[300, 1] = 30.0      ; all MT locations are implicitly created using the command
```

Then an attempt is made to change the distance type (that was previously defined using $TC_MTP_KD[300] = 2):

```
$TC_MTPPL[300, 1] = 3   ;After the MT locations have been created, the type of
                        ;distance coding must no longer be changed:
                        ;Alarm "Inconsistent data change not permitted"
```

11.7.2 $TC_MTPP - Multitool location data

Overview

The default access right of the following parameters is "keyswitch position 0" (everyone has access), for access operations from the NC program as well as for access operations from the OPI. Access rights can be restricted using the REDEF command.

$TC_MTPPx[y, z]

x:= 2, 4, 6, 7, L, A (see table)

y: = multitool number MT=1 ... 32000

z: = n-th location in the multitool; 1,...$MN_MAX_TOOLS_PER_MULTITOOL

The x parameter numbers are based on the values of the magazine location parameters.

The x numbers of the parameters are based on the numbers of the $TC_MPP1,... magazine location parameters.

NC identifier	Description	Format	Physic. unit	Preassignment limit values
$TC_MTPP2	Multitool location type (corresponds to the significance of the magazine location type)	INT	None	0 (=every tool fits the location) 0 ... 32000
$TC_MTPP4	MT location state	INT	None	0 (=every tool fits the location) 0 ... 3
$TC_MTPP6	T number of the tool contained	INT	None	0 0 ... 32000
$TC_MTPP7	Reserved for magazine location data			
$TC_MTPPL	Distance to the reference location; "length" unit (L stands for length)	REAL	Length [mm/inch]	0 0 ... 1000
$TC_MTPPA	Distance to the reference location; "angle" unit (A stands for angle)	REAL	Angle [degrees]	0 0 ... 360

$TC_MTPP2 - MT location type

The location type of a tool must match the location type of the MT location. It is checked when the multitool is loaded by writing parameter $TC_MTPP6.

The location type may only be changed when the new value is compatible with the value of the location type of a loaded tool.

The default value is selected (each tool fits the MT location) in such a way that, when the "Location type check" function is not required in the specific application for the tool loading operation, the parameter does not have to be considered, and therefore the default setting is already suitable.

$TC_MTPP4 - MT location state

The following location states have been defined (bit numbers and definition based on the state of the magazine location; $TC_MPP4):

0 Disabled

1 Free to accept a tool

$TC_MTPP6 - T number of the loaded tool

The locations of the multitools can be loaded with tools. The loading can only be defined with the aid of parameter $TC_MTPP6. In particular, loading involving the PLC has not been defined. (The OPI PI service _N_TMMVTL and the NC language command MVTOOL are exclusively for the loading/unloading of tools and multitools to and from magazines.) With $TC_MTPP6[MT no., loc. no.] = T number, loading is performed. With $TC_MTPP6[MT no., loc. no.] = 0, a tool is removed from the multitool.

The multitool can contain tools with the same or different names. Each tool fits into each location of a multitool with the default value of $TC_MTPP2. Generally, it is assumed that a multitool is a mechanical entity whose correct loading is the sole responsibility of the user.

If $TC_MTP8, bit 16 = 1 has been set for the MT state, the following must be considered:

When loading a tool to an MT or removing a tool from an MT, the MT state "disabled"/"not disabled" can change. The following applies:

- After loading, the states of the tools in the MT determine whether and how the state of the multitool is changed automatically.

When removing a tool from an MT, the following applies:

- After removing the tool, the states of the remaining tools in the MT determine whether and how the state of the MT is changed automatically.

Formally, a tool can be loaded to an MT or removed from an MT at any time. It is the responsibility of the user to ensure compliance with the application. For example, the user must know what it means to remove an MT loaded on a tool holder from the tool holder, or to remove and active tool from an MT.

A tool can be assigned either to a magazine or to an MT, but not to both at the same time. Such an attempt is rejected with alarm 17260 "Illegal magazine location definition".

When loading a tool to an MT, the magazine location type of the tool and the MT location are checked for compatibility if the MT location type does not have the value "Every tool fits the location" (this is the default value). A tool can only be loaded to an MT location when the state of the MT location is "Free".

Note

Multitools do not have a "Consider adjacent location" function for oversized tools. This means that the tool size is not evaluated when loading a tool to an MT location.

$TC_MTPPL, $TC_MTPPA - Location distance

The distance between the tools in the multitool is defined alternatively with one of these parameters. Which parameter is to be defined for the specific multitool is determined via the MT parameter $TC_MTP_KD. The change rules for the date are those that have also been defined for the multitool sizes. System parameters are not required for the distance coding "MT location number". When creating the locations for an MT, the locations are automatically assigned the numbers 1, 2, etc.

11.7.3 Multitool-related user data

$TC_MTPCx[t]

User-related multitool tool data

An additional 64 multitool-specific parameters can be set up per multitool.
Set with MD18192 MM_NUM_CC_MULTITOOL_PARAM and enable with MD18080 MM_TOOL_MANAGEMENT_MASK (set bit 2)

x: = Parameter 1 ... 64

t: = T number 1 ... 32000

OPI block MTUD

Calculation of line: T number

Calculation of column: Parameter number

Tool-related OEM user data				
NC identifier	**Type**	**Designation**	**OPI VAR**	**Type**
$ TC_MTPC1	Double		data	REAL
...	Double		data	REAL
$TC_MTPC64	Double		data	REAL

Note

The data is displayed in the tool management. Additional tool status information, for example, can be stored here.

11.7.4 Multitool-location-related user data

$TC_MTPPCx[t]

User-related multitool tool data

An additional 64 multitool-location-specific parameters can be set up per multitool. Set with MD18194 $MN_MM_NUM_CC_MTLOC_PARAM and enable with MD18080 MM_TOOL_MANAGEMENT_MASK (set bit 2)

x: = Parameter 1 ... 64

t: = T number 1 ... 32000

OPI block MTUP

Calculation of line: T number

Calculation of column: Parameter number

Tool-related OEM user data				
NC identifier	**Type**	**Designation**	**OPI VAR**	**Type**
$ TC_MTPPC1	Double		data	REAL
...	Double		data	REAL
$TC_MTPPC64	Double		data	REAL

Note

The data is displayed in the tool management. Additional tool status information, for example, can be stored here.

11.8 Adapter data $TC_ADPT

$TC_ADPTx[n]

If machine data $MN_MM_NUM_TOOL_ADAPTER is set to a value = -1 or > 0, the adapter data is defined, deleted, read and written via the following variables

x: = Parameter 1 ... 3, T

n: = Number of the adapter

Note

OPI block AD

Calculation of line: Length 1, 2, 3 = line 1, 2, 3, transformation = line 4

Calculation of column: Adapter number

Adapter data				
Name	**Type**	**Designation**	**OPI VAR**	**Type**
$TC_ADPT1	Double	Adapter geometry: Length 1	adaptData	REAL
$TC_ADPT2	Double	Adapter geometry: Length 2	adaptData	REAL
$TC_ADPT3	Double	Adapter geometry: Length 3	adaptData	REAL
$TC_ADPTT[n]	Double	Adapter transformation number	adaptData	REAL

The adapter geometry values act on the geometry values of the cutting edge in the same way as system variables $TC_DP 21, $TC_DP 22 and $TC_DP 23. The parameters are available only when the tool management is active.

Transformation numbers 1 to 8 can be programmed for the adapter transformation function. The parameter is available only when the tool management is active.

$TC_MPP7[m,p]: Number of the adapter assigned to magazine location

Value=0 : No adapter assigned to location

Value>0 : Number of the assigned magazine

11.9 User-definable variables

User-definable parameters

These programmable variables provide the user with three user-definable parameters. These system variables are transferred via the user interface with the T selection signal and the change command to the PLC. They allow the user to send additional tool management information to the PLC. The parameters can be read and written by the NC program. They are not buffered and are set to "0" on Reset or end of program.

$P_VDITCP[x]

x: = Parameters 0, 1, 2

NC identifier	Description	Format
$TC_VDITCP[0]	Tool management VDI user-definable parameter 0	int
$TC_VDITCP[1]	Tool management VDI user-definable parameter 1	int
$TC_VDITCP[2]	Tool management VDI user-definable parameter 2	int

Programming

The parameters can be programmed as required in the NC program. The output to the PLC is however always realized in conjunction with the tool preparation or change command programmed in the following.

Example:

```
T= "Tool1"
$P_VDITCP[0] = 1
M06
$P_VDITCP[0] = 2
T= "Tool2"
```

Exactly the value = 2 is also given to the PLC with the command output of T = "Tool2" to the PLC and not the value 1 when the M06 command is output to the PLC.

The output of the programmed value also takes place when M6 is programmed, i.e. the output can now also be realized with the command number 3 provided $MC_CHANGE_MODE=1 has been set.

11.10 NC language commands

11.10.1 CHKDNO - Uniqueness check for D number

If tool management is active, the function `CHKDNO` checks the uniqueness of the D number. Uniqueness of the D number here (no replacement tools) means that the D numbers of all tools defined in the TO unit can occur only once. D numbers in the TO unit are unique and absolute.

Syntax

```
State = CHKDNO(T1, T2,D)
```

Meaning

`CHKDNO`	The function returns the value of `state` as the return value.	
	Data type:	Bool
`T1`	T number of tool 1 (optional)	
	Data type:	INT
`T2`	T number of tool 2 (optional)	
	Data type:	INT
`D`	D number (optional); without parameter D, all D numbers of the named tools are checked.	
	Data type:	INT
Return value		
`state`	`state` shows whether a D number collision occurs	
	Data type:	BOOL
	FALSE = D number collision	
	TRUE = D numbers are unique	

D numbers of replacement tools

Replacement tools can be defined and used when tool management is active. The machining part program does not generally indicate whether any replacement tools are available. As a rule, the machining program addresses tools with T="Identifier". (Programming T=location number is referred back internally to T="Identifier"). The program otherwise only contains the actual programming of the offset (the D number). Therefore, the D numbers of the tool and replacement tool must be identical.

Example

Active tool and replacement tools for T="drill_5mm"

- T No. = 10 with D numbers 1, 2, 3 (active)
- T No. = 11 with D numbers 1, 2, 3 (replacement)
- T No. = 12 with D numbers 1, 2, 3 (replacement)

Active tool and replacement tools for T="drill_3mm"

- T No. = 20 with D numbers 1, 2, 3 (active)
- T No. = 21 with D numbers 1, 2, 3 (replacement)
- T No. = 22 with D numbers 1, 2, 3 (replacement)

`CHKDNO` - without parameters specified - detects a collision of D numbers 1, 2 and 3 for "drill_5mm" with D numbers 1, 2 and 3 for "drill_3mm", but not between the D numbers of the active and replacement tools.

The collisions are displayed as individual alarms, e.g.:

- "Channel 1 D number 1 defined for tool T no. 10 and 20"
- "Channel 1 D number 1 defined for tool T no. 10 and 21"

The state = FALSE is also returned in the event that the parameterization is invalid. (The T or D number called is not defined in the channel.)

If the following applies: `MAX_CUTTING_EDGE_NO ≤ MAX_CUTTING_EDGE_PER_TOOL` , `CHKDNO` always returns the TRUE state, irrespective of the parameter settings.

11.10.2 CHKDM - Uniqueness check within a magazine

With active tool management, the command `CHKDM` checks the data in NC for D number uniqueness within one or more magazines. The functionality corresponds to `CHKDNO`. The parameters are optional.

Syntax

```
state = CHKDM (magazine no., D no., tool holder no.)
```

Meaning

CHKDM	Checks the uniqueness of a D number within a magazine	
	Data type:	Bool
MagNo	Magazine number of magazine to be checked	
	Data type:	INT
	Omitting the parameter or programming with a value=0 means that the tools of all the connected magazines of the spindle numbers or tool holder numbers specified in Tool holder no. are checked.	
Dno	D number against which the check will be made.	
	Data type:	INT
	Omitting the parameter or programming with a value=0 means that all D numbers of the specified magazine are checked for uniqueness.	
Tool holder no.	Specifies the spindle number or tool holder number for which the magazines are checked	
	Unit	INT
	Omission of the parameter means that the magazines are obtained for the check from the distance table of the spindle location for the leading spindle or the leading tool holder.	
Return value		
state	Result of the check:	
	Data type	Bool
	TRUE	Checked D numbers are unique
	FALSE	Check not OK

11.10.3 GETACTTD - Determining the tool number of a unique D No.

If tool management is active (e.g. in measuring cycle programs), command GETACTTD provides the associated T number of the active tool in the tool group based on a D number.

Syntax

```
status = GETACTTD (Tno, Dno)
```

Meaning

GETACTTD	returns the T number of the active tool in the tool group	
	Data type:	INT
Dno	D number for which the T number shall be searched. The D numbers are not checked for uniqueness.	
	Data type:	INT
	If the same D numbers are defined in different tool groups of the same TO unit, then the T number of the first tool group that is found determines the tools that have the specified number.	
Tno	T number found	
	Data type:	INT

Return value		
status	Result of search	
	Data type:	INT
	0	T number found, Tno is assigned the value
	-1	No T number exists for the specified D number, Tno assigned the value 0.
	-2	D number is not unique; Tno is assigned the value of the D number that was first determined.
	-3	The tool group does not contain any tools that have the desired status and the specified D number, Tno is assigned the value 0.
	-4	The tool group contains several tools that have the desired status and the D number that has been sought; Tno contains the value of the first tool found with the desired D number.
	-5	Function could not be executed for other reasons.

11.10.4 GETDNO - Read out D numbers

Function GETDNO reads out the offset number d of a tool with T number t and cutting edge ce.

Syntax

```
d = GETDNO(t, ce)
```

Meaning

GETDNO	Reads out offset number d	
	Data type:	INT
t	T number of the tool	
	Data type:	INT
ce	Cutting edge ce of the tools	
	Data type:	INT
Return value		
d	Return value offset number d	
	Data type:	INT
	If t or ce are parameters which have no data record, d=0 is returned. Any parameters violating the syntax rules will generate an alarm.	

Dependencies

- The command is only available if $MN_MAX_CUTTING_EDGE_NO > $MN_MAX_CUTTING_EDGE_PER_TOOL.
- $MN_MAX_CUTTING_EDGE_NO ≤ $MN_MAX_CUTTING_EDGE_PER_TOOL returns GETDNO d=ce as D number.

11.10.5 SETDNO - Set or change D number

With function SETDNO, offset number d of cutting edge ce of tool t can be set or changed.

Syntax

```
state = SETDNO(t, ce, d)
```

Meaning

SETDNO	Function for editing offset number d of cutting edge ce of tool t	
	Data type:	INT
t	T number of the tool	
	Data type:	INT
ce	Cutting edge ce of the tools	
	Data type:	INT
d	Offset number d	
	Data type:	INT
	If t or ce are parameters that have no data record, state = FALSE is returned. Any parameters violating the syntax rules will generate an alarm.	
Return value		
state	Status of the function	
	Data type:	INT
	FALSE (0)	Machining was not successful
	TRUE (1)	Machining was successful

Dependencies

t, ce, d must be specified as > 0; d=0 cannot be set.

See also

DZERO - Invalidate D numbers (Page 177)

11.10.6 DZERO - Invalidate D numbers

Function DZERO identifies all D numbers of the TO unit as invalid. This command is used for support during setup. Offset data sets tagged with this command are no longer checked by the CHKDNO NC command. The D numbers have to be set with SETDNO to make these accessible again, see also SETDNO - Set or change D number (Page 177).

Syntax

```
DZERO
```

11.10.7 DELDL - Delete additive offsets

Function DELDL deletes the additive offsets for the cutting edge of a tool (to release memory space). Both the defined wear values and the setup values are deleted.

Syntax

```
status = DELDL(t, d)
```

Meaning

DELDL	Deletes all sum offsets of cutting edge d of tool t	
	Data type:	INT
t	Number of the tool (optional); if t and d are not specified, all sum offsets of all cutting edges of all tools of the TO unit are deleted.	
	Data type:	INT
d	Number of the cutting edge (optional); if d is not specified, all sum offsets of all cutting edges on tool t are deleted.	
	Data type:	INT
Return value		
status	Delete result	
	Data type:	INT
	0	Offsets have been successfully deleted
	-1	Offsets have not been deleted (if the parameter settings specify exactly one cutting edge), or not deleted completely (if the parameter settings specify multiple cutting edges)

11.10.8 NEWT - Create new tool

With function NEWT, you can create a new tool without specifying a T number.

The function returns the automatically generated tool number, with which the tool can subsequently be addressed. The new tool is automatically given a cutting edge with CE number = 1 and D number = 1. All offsets are set to 0 by default. You can change the CE number with function SETDNO, see SETDNO - Set or change D number (Page 177).

An alarm is output if the attempt to create a new tool fails.

Syntax

```
tno = NEWT ("TO", duplo no.)
```

Meaning

NEWT	Creates a new tool with name "TO" and duplo no. Duplo no. and returns the T number of the new tool as the return value.	
	Data type:	INT

Tool	Tool label	
	Data type:	STRING
Duplo no.	Duplo number of tool (optional); if not otherwise specified, duplo number no. = old duplo no. +1 is used.	
	Data type:	INT
Return value		
tno	T number of new tool	
	Data type:	INT

Examples

Example 1: Create new tool with the function NEWT and CE/D numbers = 2, 47

Program code	Comment
def int tno	
tno = NEWT("tool", 111)	;Tool with Ident/DuploNo.="Steel"/111, T no.=tno=1 in the example and a cutting edge CE=1, D=1 is created
	;the cutting edge is to be changed to CE=2, D=47
$TC_DPCE[tno, 1]=2	;Rename the CE number
SETDNO(tno,2,47)	;Rename the D number
	;Assign the remaining data of the tool/ cutting edge

Example 2: Create tool "Steel"/111, T no.=tno=1 with $TC... and CE numbers = 2 4 (let us assume that T no.=1 does not yet exist)

Program code	Comment
$TC_TP1[1] = 111	;Create new tool with T no.=1, duplo no.=111
$TC_TP2[1] = "Steel"	;Assign tool identifier="Steel"
$TC_TPCE[1,47] = 2	;Create new offset, D=47, assign CE no.=2
	;Assign the remaining data of the tool/ cutting edge

The function is used for creating tools in a loading program (load cycle).

11.10.9 NEWMT Create new multitool

Function NEWMT is available with TMMG.

With function NEWMT, a new multitool with the specified name "name" and the MT number of locations = "number of locations" is created.

An implicit STOPRE is initiated at the end of the block.

Messages:

- If the "Several tools at magazine location" function is not activated, Alarm 6436 "Command cannot be programmed. Function is not activated" is generated.
- If parameter `name` is already in use, alarm 14020 "Illegal parameter value" is output.
- If parameter `Number of locations` is outside the permissible range, alarm 14020 "Illegal parameter value" is output.
- If the function `NEWMT` is interrupted with an alarm, the value "0" is returned for `mtNo`.

Syntax

```
mtNo = NEWMT(name, location number)
```

Meaning

`NEWMT`	Creates a new multitool of name "`name`" and number of locations `Number of locations`.	
	Data type:	INT
`name`	Multitool label	
	Data type:	STRING
`Number of locations`	Number of multitool locations	
	Data type:	INT
Return value		
`mtNo`	MT number of the multitool; the number is created automatically and returned as the result value.	
	Data type:	INT

Examples

A tool with the name "A" and the T number 1 and a magazine with the name "B" and the number 2 already exist. The following is now programmed:

Program code	Comment
`def int mtNo`	`;`
`mtNo = NEWMT("C", 2)`	`;`

The command is executed successfully, `mtNo` has the value 3. The multitool has been created for 2 locations. However, the locations have not been created yet. A call with the name "A" or "B" would have resulted in an alarm, as the names have already been allocated to a tool and a magazine.

After creating using `NEWMT` additional parameters of the multitool can be defined, e.g.

Program code	Comment
`$TC_MTP7[ 3 ] = 12`	`; Multitool can be loaded to magazine location with this location type number`
`$TC_MTP_POS[ 3 ] = 2`	`; Position value = location 2`
`$TC_MTP_KD[ 3 ] = 3`	`; Angle-coded distance`

Program code	Comment
$TC_MTPPA[3, 1] = 0.0	; The two locations are created with this write operation
$TC_MTPPA[3, 2] = 180.0	;
$TC_MTP8[3] = 2	; Released (multitool definition has been completed)

The two parameters $TC_MTPN and $TC_MTP2 were already defined using the NEWMT command. Parameters $TC_MTP3..6 were not programmed. The default values are effective.

11.10.10 DELT Delete tool

With function DELT, a tool can be deleted by specifying the tool identifier and the duplo number. Only tools that have been unloaded can be deleted.

All tool-related data is set to 0 (user data, hierarchy data, ...).

Syntax

```
DELT("TL",DUPLO_NO)
```

Meaning

DELT	Deletes the specified tool.	
	Data type:	-
Tool	Tool label	
	Data type:	STRING
DUPLO_NO	Duplo number of the tool	
	Data type:	INT
Return value	-	

Examples

The function is used for deleting tools in the part program.

Program code	Comment
T=GETT("DRILL", DUPLO_NR)	;
	;

11.10.11 DELMT - Delete multitool

Function DELMT is available with TMMG.

A multitool is deleted with function DELMT.

A multitool can only be deleted if it is not contained in a magazine (is unloaded) and none of the tools contained in the multitool are the active tool for offset selection. If there are

still tools in the multitool at the time of deletion, they are automatically removed from the multitool before the delete operation. The following tools are not deleted:

If a name is specified for which a multitool has not been defined, the command is rejected with Alarm 17220 "Tool does not exist".

Note

After the multitool was unloaded, the tools contained in it are also unloaded. After the multitool has been deleted, the tools contained in it are again available for loading into a magazine - or into another multitool.

Syntax

```
DELMT (name)
```

Meaning

DELMT	Deletes the multitool of name name	
name	Multitool label	
	Data type:	STRING

11.10.12 $TC_MTPN - Delete one or all multitools

It is possible to delete one or several multitools with $TC_MTPN analogously to $TC_TP1[TNo] = 0 The delete command is defined as follows:

- $TC_MTPN[MTNo.] = 0 ; Delete multitool with the number MTNo > 0
- $TC_MTPN[0] = 0 ; Delete all multitools

For MTNo > 0, the multitool with the number MTNo is deleted. For MTNo = 0, all multitools are deleted. The same rules apply as defined for the DELMT command, see also DELMT - Delete multitool (Page 181).

Syntax

```
$TC_MTPN[MTNo]
```

Meaning

$TC_MTPN[MTNo]	Returns the multitool number or deletes a multitool.	
	Data type:	INT
MTNo	Number of the multitool	
	Data type:	INT
	Value range:	0 < x < SLMAXTOOLNUMBER

Note

The delete commands to delete all tools ($TC_TP1[0] = 0 and $TC_DP1[0,0] = 0) also delete all multitools.

Only with the deletion of the multitool, are the MT locations also deleted.

11.10.13 Reading the GETT - T no.

The GETT function returns the T number associated with the tool identifier and its duplo number as a return value.

The command can be used for tools as well as for multitools. A duplo number cannot be programmed for multitools.

If the tool identifier or duplo number cannot be assigned to a tool, value -1 is returned.

This function is used, for example, to retroload tools via the part program.

Syntax

```
tNo = GETT("TL", DUPLO_NO)
```

Meaning

GETT	Returns the T number of a tool or multitool	
	Data type:	INT
Tool	Tool label	
	Data type:	STRING
DUPLO_NO	Duplo No. of the tool (optional)	
	Data type:	INT
	If a duplo number has not been specified, the T number of any tool from the group of tools with the specified identifier or the number of a multitool is returned. (The sequence is not defined within a tool group.)	
Return value		
tNo	T number of the tool	
	Data type:	INT
	> 0	Tool or multitool number associated with the programmed name
	-1	The name is neither a tool nor a multitool name
	-2	A multitool name was programmed with duplo number

Examples

Determine the T number for drill with duplo number.

Program code	Comment
R10=GETT("DRILL", DUPLO_NO)	;The T number is in R10
$TC_TPx,[GETT("DRILL",DUPLO_NO)]=value	;Write tool-related data

11.10.14 SETPIECE - decrementing the workpiece counter

With the SETPIECE function, the user can update the count monitoring data of the tools associated with the machining process. Each tool that has been loaded since the last activation of SETPIECE is acquired. The function serves at the end of the NC part program to decrement the count from all the tools associated with count monitoring.

Note

The command is not active in the block search (with/without calculation). If the value for the count = 0, the internal table for flagged tools/cutting edges is deleted.

Note

SETPIECE works through the table "blindly". This means that every cutting edge entered in the table is recorded. It is irrelevant where the tool is located at the time of the SETPIECE programming, whether it is stored in the magazine, remains on the spindle, or has been unloaded.

If it is set that the tool remains active after a reset and there is a tool in the spindle at the start of the program, this tool is selected internally and an entry is made in the Setpiece table.

Syntax

```
SETPIECE(x,y)
```

Meaning

SETPIECE	Updates the workpiece count monitoring data	
	Data type:	-
x := 0 ... 32000	Value used when decrementing	
	Data type:	INT
y := 0...8	Spindle index, value 0 means index of main spindle (does not have to be programmed)	
	Data type:	INT

Examples

Program code	Comment
SETPIECE(1);	Workpiece counter of main spindle is decremented by 1
SETPIECE(1,1);	Workpiece counter of spindle no. or tool holder no. 1 is decremented by 1
SETPIECE(4,2);	Workpiece counter of spindle no. or tool holder no. 2 is decremented by 4

Example of SETPIECE with M06 tool change command:

The tools involved in a workpiece (program) should be decremented by a value of 1.

Program code	Comment
T1	;T1 is preselected (with regard to main spindle)
M06	;T1 is changed
D1	;D1 becomes active
T2	;T2 is preselected
:	;Machining program
:	
M06	;T2 is changed
D1	;D1 from T2 is activated
T3	;T3 is preselected
:	;Machining program
:	
:	
M06	
T0	;Preparation for clearing the spindle
:	
M06	;Clearing the spindle
SETPIECE(1)	;SETPIECE on all tools
M30	

The counter is to be decremented for each tool

In this example, tools T1, T2 and T3 are to machine a program. All three tools are monitored for workpiece count. The aim is to decrement tool T1 by the value 1, T2 by the value 2 and not to decrement T3.

The following programming is required:

Program code	Comment
N500 T1	
N600 M06	
N700 D1	;With the offset selection, the tool that was loaded is stored in the SETPIECE memory
N900 T2	;Prepare next tool
	;Machining command

```
Program code                Comment
:
N1000 setpiece(1)           ;SETPIECE acts on T1, SETPIECE memory is cleared
N1100 M06
N1200 D1
N1400 T3
:                           ;Machining commands
:
N1500 setpiece(2)           ;Only acts on T2
N1600 M06
N1700 D1
:                           ;Machining commands
:
N1800 setpiece(0)           ;Only acts on T3, not decremented
N1900 T0
N2000 M06
N2100 D0
N2300 M30
```

11.10.15 GETSELT - Read the selected T number

The function GETSELT is available with TMMO and TMMG.

The GETSELT function returns the T number of the selected tool with regard to the individual tool holders or spindles from the NC program viewpoint and is used to access the offset data of the selected tool during T programming and cutting edge selection.

During preprocessing or while the NC program is running, the selected T number is not available until after the next cutting edge selection. The task of the function GETSELT is to establish synchronization with the main run earlier.

NOTICE
Alarm delay If an alarm delay is active ($MC_TOOL_CHANGE_ERROR_MODE, bit 0 = 1) and an alarm occurs during the tool preparation, which is to be delayed until after M06 and programmed before the programming of M06 GETSELT, the T number found by GETSELT has the value = -1.

Syntax

```
GETSELT (Tno, th, ssl)
```

Meaning

GETSELT	Reads out the selected T number	
	Data type:	-

Tno	T number of the tool	
	Data type:	INT
	= 0	There is no selected T number or the tool has been deleted in the meantime (occurs only with ssl="S").
	= -1	If the alarm delay is active and GETSELT is called between the T and the M06 programming.
th= 0,1, ..., 20	Number of spindle or of the tool holder (optional). If th is not specified, this acts as th=0.	
	Data type:	INT
	• If ssl="" or is not set, the command refers to the currently active main spindle/active main tool holder • If ssl="S", during the block search and test mode, the command refers to the tool holder that determined the D offset before the block search	
ssl	Block search (optional)	
	Data type	CHAR
	= ""	Function as previously, i.e. the current state after the NC program
	= "S"	Block search: Returns the tool selected before block search or test mode

If "ssl" is set (="S" for "searchrun"), during the block search or test mode, the GETSELT NC command returns the tool number of the selected tool on the specified tool holder **before the block search** or **before the test mode**. At the end of the block search or test mode, the GETSELT NC command returns the value irrespective of ssl after the first T programming for the specified tool holder.

After program end / abort, the T numbers of all the tool holders that were returned with GETSELT are set to the value = 0. For the leading tool holder, the value is then set to the corresponding setting in $MC_RESET_MODE_MASK, or after a program restart to the corresponding setting in $MC_START_MODE_MASK.

If required, the function makes an implicit preprocessing stop if the main run has not received the tool preparation command.

OPI variables

The values supplied by the "GETSELT" function are available in the OPI variable "toolHolderData" of the C/S block.

Parameter no. 2 returns the value of `GETSELT(tNo, Th)`.

Parameter no. 4 returns the value of `GETSELT(tNo, Th, "S")`.

The OPI variable is available in all versions of the tool management.

Example (1)

There is no tool holder location in the magazine management for the programmed value of tool holder no. = 3.

Program code	Comment
N110 GETSELT (tNo, 3)	; tNo = 0 is the result.

Example (2)

Tool preparation has not been programmed for the programmed tool holder number = 4, and no tool has been loaded.

Program code	Comment
N110 GETSELT (tNo, 4)	; tNo = 0 is the result.

Example (3)

There is a tool with the name "disabled" and the T number = 5.

Program code	Comment
def int tNo	
N100 T=”disabled”	; Alarm 'No tool ready for use' is ; delayed until the associated M06 is programmed ; or until T is programmed again.
N110 GETSELT (tNo)	; tNo = -1 is the result
N120 M06	; The delayed alarm from N100 is output

OPI:

In this case, the OPI variable "progTNumber" in the C/S variable block displays the previously programmed T number or the value = 0 if no other T selection has been programmed before the faulty T selection.

Example (4)

In the reset state, the tool with the T number = 5 is loaded into the main spindle. A block search is started on block "N100" in the following NC program.

Program code	Comment
def int tNo, tNo1	
N70 T7	
N80 GETSELT(tNo)	; tNo = 7
N90 GETSELT(tNo1,,"S")	; tNo1=5
N100 X1 G0	; Target block
N110 GETSELT(tNo)	; tNo = 7
N120 GETSELT(tNo1,,"S")	; tNo1=5, because no new T has been programmed ; after the target block
N130 T8	
N140 GETSELT(tNo)	; tNo = 8
N150 GETSELT(tNo1,,"S")	; tNo1=8, because a new T has been programmed

OPI:

In this case, the OPI variable "progTNumberSSL" in the C/S variable block displays the previously programmed T number or the value = 0 if no other T selection has been programmed before the faulty T selection.

11.10.16 GETEXET - Reading of the loaded T number

The function `GETEXET` is available with TMMO and TMMG.

The function GETEXET is specifically designed for block search. Its parameters are set in the same way as for GETSELT. It returns the T number of the tool that is active from the point of view of the NC program.

Syntax

```
GETEXET(Tno, th, ssl)
```

Meaning

`GETEXET`	Reads out the selected T number	
	Data type:	-
`Tno`	T number of the tool	
	Data type:	INT
	= 0	There is no loaded T number or the tool has been deleted in the meantime (occurs only with ssl="S").
`th= 0,1, ..., 20`	Number of spindle or of the tool holder (optional). If `th` is not specified, GETEXET refers to the current leading spindle.	
	Data type:	INT
	= 0:	If ssl="" or is not set, the command refers to the currently active main spindle / tool holder. If ssl="S", the command refers to the tool holder that determined the D offset before the block search.
`ssl`	Block search (optional)	
	Data type	CHAR
	= "":	Function as previously, i.e. the current state after the NC program
	= "S":	Block search function: Returns the tool loaded before the block search or test mode.

If "ssl" is set (="S" for "searchrun"), during the block search or test mode, the GETEXET NC command returns the T number of the loaded tool on the specified tool holder **before the block search** or **before the test mode**. At the end of the block search or test mode, the GETEXET NC command returns the value independent of ssl after the first programming of a tool change for the specified tool holder.

If required, the function makes an implicit preprocessing stop if the main run has not received the tool change command.

After the program end / abort, the "getexet" T numbers of all tool holders are set to the value = 0. For the leading tool holder, the value is then set to the corresponding setting

in $MC_RESET_MODE_MASK, or after a program restart to the corresponding setting in $MC_START_MODE_MASK.

Note

If the tool change is only programmed with T ($MC_TOOL_CHANGE_MODE), GETSELT and GETEXET return the same T number.

If the tool change is programmed with T (selection) and M06 (change), the two commands have different contents.

OPI variables

The values supplied by the "GETEXET" function are available in the OPI variable "toolHolderData" of the C/S block.

Parameter no. 3 returns the value of `GETEXET(tNo, Th)`.

Parameter no. 5 returns the value of `GETEXET(tNo, Th, "S")`.

The OPI variable is available in all versions of the tool management.

Example (1)

Three tools are defined with Tno. = 5/Name = "Tool5" , Tno. = 7/Name = "Tool7" and Tno. = 9/ Name = "Tool9".

The tool change is programmed with T + M6.

Two tool holders have been defined. Tool holder 2 is the leading tool holder.

At the start, there is neither a tool on the tool holder nor a tool active or programmed:

```
Program code                                    Comment
def int tNo1, tNo2
T="Tool5"   ; tool selection for main
tool holder
getselt ( tNo2, 2 )                             ;tNo2 = 5 preselected tool for main tool
                                                holder
getselt ( tNo1, 1 )                             ;tNo1 = 0

getexet ( tNo2, 2 )                             ;tNo2 = 0 no active tool for main tool
                                                holder
getexet ( tNo1, 1 )                             ;tNo1 = 0

M6   ; tool change for leading tool holder
getselt ( tNo2, 2 )                             ;tNo2 = 5
getexet ( tNo2, 2 )                             ;tNo2 = 5

T="Tool9"   ; tool selection for main
tool holder
getselt ( tNo2, 2 )                             ;tNo2 = 9
getexet ( tNo2, 2 )                             ;tNo2 = 5
```

Program code	Comment
T1="Tool7" ; tool selection for secondary tool holder	
getselt (tNo2, 2)	;tNo2 = 9 preselected tool for main tool holder
getselt (tNo1, 1)	;tNo1 = 7 preselected tool for secondary tool holder
M1=6 ; tool change for secondary tool holder	
getexet (tNo1, 1)	;tNo1 = 7 active tool for secondary tool holder

Example (2)

In the reset state, the tool with the T number = 5 is loaded into the main spindle. A block search is started on block "N100" in the following NC program.

Program code	Comment
def int tNo, tNo1	
N70 T7 M6	
N75 T8	
N80 GETEXET(tNo)	;tNo =7
N90 GETSELT(tNo1,,"S")	;tNo1=5
N100 X1 G0	;target block
N110 GETEXET(tNo)	;tNo =7
N120 GETEXET(tNo1,,"S")	;tNo1=5, because no new tool has been programmed ;after the target block
N130 M6	
N140 GETEXET(tNo)	;tNo=8
N150 GETEXET(tNo1,,"S")	;tNo1=8, because a new tool has been loaded

OPI:
In this case, the OPI variable "progTNumberSSL" in the C/S variable block displays the previously programmed T number or the value = 0 if no other T selection has been programmed before the faulty T selection.

Use of $P_TOOLNO - GETEXET

With active tool management, GETEXET should generally be used.

If the following specification are complied with, $P_TOOLNO can also be used:

After programming D, $P_TOOLNO always returns the correct value of the active T number.

Program code	Comment
T2 M6 D1	;Tool group "2" has precisely one tool with Tno. = 2
r1 = $P_TOOLNO	;r1 = 2

For $MC_CUTTING_EDGE_DEFAULT=-1, >0, $P_TOOLNO always returns the correct value of the active T number.

Program code	Comment
T2 M6	;Tool group "2" has precisely one tool with Tno. = 2
r1 = $P_TOOLNO	;r1 = 2

There are exceptions for $MC_CUTTING_EDGE_DEFAULT=-2 and 0.

a) $MC_CUTTING_EDGE_DEFAULT = 0

Program code	Comment
T2 M6	;Tool group "2" has precisely one tool with Tno. = 2
r1 = $P_TOOLNO	;!!!r1 = ? - the T no. is not defined until the main run. ;!!!This means that it is generally not available here.
def int tNo	
T2 M6	;Tool group "2" has precisely one tool with Tno. = 2
GETEXET(tNo, $PMTHSDC)	;Safe determination of the active Tno. tNo = 2

b) $MC_CUTTING_EDGE_DEFAULT = -2

Program code	Comment
N10 T2 M6	;Tool group "2" has precisely one tool with Tno. = 2
r1 = $P_TOOLNO	;!!!r1 = 22 - if T no.=22 was the active tool before N10
def int tNo	
N10 T2 M6	;Tool group "2" has precisely one tool with Tno. = 2
GETEXET(tNo, $P_MTHSDC)	;Safe determination of the active Tno. to be loaded ;tNo =2 - the Tno. becomes the active Tno. at the ;next programming of D>0.

This means that for $MC_CUTTING_EDGE_DEFAULT=-2, $P_TOOLNO and also GETEXET can be used, and with a different meaning.

11.10.17 $P_MTHSDC - leading tool holder with regard to the D offset selection

System variable $P_MTHSDC is provided with TMMG.

The system variable $P_MTHSDC (**M**ain **t**ool **h**older regarding selected **D**–Correction) returns the number of the tool holder/spindle on which the last tool change was performed on a leading tool holder/spindle. This means that this is the tool holder on which the tool is located that determines the next D offset and therefore becomes the active tool (if this tool was not already active).

Syntax

```
$P_MTHSDC
```

Meaning

$P_MTHSDC	Supplies the number of the tool holder/spindle.	
	Data type:	INT
Return value		
	> 0	Successful read access
	= 0	No tool holder or spindle available
	-1	TMMG not available

Examples

Settings:

$MC_CUTTING_EDGE_DEFAULT = -2,
i.e. the old tool offset remains active with M06 if D has not been explicitly programmed.

$MN_TOOL_CHANGE_MODE = 0

$MC_TOOL_MANAGEMENT_TOOLHOLDER = 2

```
Program code                Comment
N10 SETMTH ( 1)
N20 T="Tool5"               ;T no. = 5 is the only tool in the tool group
                            ;tool change to the current main tool holder (tool holder
                            1)
N30 D5                      ;Offset of the tool last changed on a main tool holder
                            ;(T no. 5 in this case)
N100 SETMTH ( 2)            ;Change of the main tool holder
N110 T="Tool7"              ;T no. = 7 is the only tool in the tool group
                            ;tool change to the current main tool holder (tool holder
                            2)
N120 SETMTH ( 1)            ;Change of the main tool holder
...                         ;Further blocks without T, M6, D
N130 D5                     ;Offset with D5 of the tool that was loaded last on to a
                            ;main tool holder (T=7 on tool holder 2
                            ;in this case). Only now is a new active tool defined
```

To ensure that after a block search on a block between N120 and N130 the D selection in N130 refers to the correct tool ("Tool7" here), the tool change for "Tool7" on tool holder 2 must be performed as main tool holder and must be performed as the last tool change. This may have to be ensured via ASUBs. This information cannot be determined with the previously available NC commands and variables. The variable contains the information on which tool holder the last tool change was performed as this was main tool holder.

In the above case, the variable $P_MTHSDC contains the value 2, because the last change to a leading tool holder was made on this tool holder. With the aid of GETEXET(TNo,2) or if ($P_MTHSDC > 0) GETEXET(TNo,$P_MTHSDC), the tool can be determined that belongs on this tool holder so that the program can be continued correctly.

$P_MTHSDC = 0 means that currently no tool holder has been defined that has the active tool after D programming.

11.10.18 $P_TH_OF_D - leading tool holder with regard to the current D offset

System variable $P_TH_OF_D is available with TMMO and TMMG.

The system variable $P_TH_OF_D (Tool holder of active D correction) returns the number of the tool holder/spindle on which the tool is or was located that determines the active D offset.

Syntax

```
$P_TH_OF_D
```

Meaning

$P_TH_OF_D	Supplies the number of the tool holder/spindle	
	Data type:	INT
	> 0	Successful read access
	0	No tool holder available
	-1	Function not available

Examples

Settings:

$MC_CUTTINGEDGE_DEFAULT = -2, i.e. the old tool offset remains active with M06 if D has not been explicitly programmed.

$MN_TOOLCHANGE_MODE = 0

$MC_TOOL_MANAGEMENT_TOOLHOLDER = 2

Program code	Comment
N00	;No tool active → $P_TH_OF_D =
N10 SETMTH(1)	
N20 T="Tool5"	;T no. = 5 is the only tool in the tool group;tool change to the current main tool holder (tool holder 1)
N30 D5	;Offset is of the tool that is located on tool holder 1;→ $P_TH_OF_D = 1
N100 SETMTH(2)	;Change of the main tool holder
N110 T="Tool7"	;T no. = 7 is the only tool in the tool group ;tool change to the current main tool holder (tool holder 2)
N120 SETMTH(1)	;Change of the main tool holder
...	;Further blocks without T, M6, D
N130 D5	;Offset with D5 of the tool that was loaded last on to a;main tool holder (here T=7 on tool;holder 2) → $P_TH_OF_D = 2

In a ProgEvent program, after a block search, the situation is to be established with regard to tools and tool offset that would have existed without a block search.

After a block search on a block between N100 and N130, it is determined how the currently active "D5" from block N30 is established. For this purpose, it must be recognized that "D5"

comes from the tool on tool holder 1. The last tool change on a main tool holder was performed on block N20 on tool holder 1. This information is provided by system variable $P_TH_OF_D.

The variable $P_TH_OF_D specifies on which tool holder the tool is located that determines the **currently active D offset**.

The variable $P_MTHSDC specifies on which tool holder the tool is located that determines the offset at the **next D programming**.

$P_TH_OF_D = 0 means that currently no tool holder is defined to which the current tool offset "D" refers. The active tool is then also T=0 and the active D no. =0.

11.10.19 GETACTT - Read the active internal T number

Function GETACTT returns the T number of the tool with status "active" and "was in use" via parameter "TNo" from a tool group "name" (a tool becomes "active" immediately before it is loaded into the tool holder)

Note

GETACCT cannot detect a tool that is positioned in the spindle for its first use.

The tool sequence in a group is not defined. GETACCT will read any random tool in the group where the status bits "active" and "was in use" are set.

Syntax

```
status=GETACTT(Tno,name)
```

Meaning

GETACTT	Returns the T number of the active tool	
	Data type:	INT
Tno	T number of the active tool	
	Data type:	INT
name	Name of the tool group	
	Data type:	STRING
Return value		
status	Shows readout status	
	Data type:	INT
	0	Function successful; T No. contains the desired value
	-1	No tool matching the specified identifier exists; T No. contains value = 0
	-2	The tool group does not contain a tool with the desired status; T no. contains value = 0
	-3	There are several tools with the desired status in the tool group; T No. contains the value of the first tool with the desired status

GETACTT can have several meanings! It is always conceivable that there may be several tools in the tool group that have the same status. The command will only then meaningfully function if the user ensures there is exactly one tool with the desired status in the tool group.

The command does not initiate a main run synchronization. It may be necessary to enter STOPRE before the call.

Example

Tool group "Drills" contains three tools with the duplo numbers 1, 2, 3, and the T numbers 1, 2, 3:

Program code	Comment
def int tNo, status	;Initially, there is no active tool in the tool group "Drill"
status=GETACTT(Tno, "Drill")	;status=-2, Tno=0
T="Drill"	;Preparation sets tool status to "active"
status=GETACTT(Tno, "Drill")	;status=0, Tno=0
	;the tool is active, but the identifier "was in use" is not yet applied
M06	;Change
T="Hugo"	;Preparation
status=GETACTT(Tno, "Drill")	;status=-2, Tno=0
	;the tool is active, but the identifier "was in use" is still not yet applied
M06	;Change
status=GETACTT(Tno, "Drill")	;status=0, Tno=1
	;Read job is performed
	;Tool "Drill" is now assigned the status "was in use" as it was removed and the "active" status remains unchanged

11.10.20 SETMS - Set leading spindle

Available with TMMO and TMMG.

Function SETMS declares the spindle specified in n to be the leading spindle. Definition as leading spindle is also possible via machine data.

When SETMS is programmed without a spindle name, the spindle programmed in the machine data used instead.

Syntax

```
SETMS(n)
```

Meaning

SETMS	Declares the spindle specified in n as the leading spindle	
	Data type:	-
n	Number of spindle	
	Data type:	INT

Reset behavior

The programmed values from SETMS can remain active beyond program end/RESET/START.

11.10.21 SETMTH - Set leading tool holder number

The function SETMTH is made available with TMMG.

With the SETMTH function, the tool holder with the number n becomes the leading tool holder.

The machine data **MD20124 TOOL_MANAGEMENT_TOOLHOLDER** can be set to determine whether a tool holder number can be assigned to define the location of the tool to be loaded instead of a spindle number. It only makes sense to use this NC command if MD is > 0.

Note

SETMTH does not change the active tool. The new main tool holder definition cannot be taken into account until the tool change is programmed.

Syntax

```
SETMTH(n)
```

Meaning

SETMSH	Declares the tool holder in n as the main tool holder	
	Data type:	-
n	Number of the tool holder	
	Data type:	INT

Example (1)

Programming example

Program code	Comment
T="Mill" M06	;No address extension programmed → this refers to the leading tool holder; i.e. tool holder 1 (value of machine data TOOL_MANAGEMENT_TOOLHOLDER). The tool change is performed at the buffer location with $TC_MPP5=1. The path is corrected with the tool offsets.
...	
T2="Drill" ..M2=6	;Address extension for the secondary tool holder has been programmed. The tool change is performed at buffer location 2. The path is not corrected.
...	
SETMTH (2)	;Declare tool holder 2 to be the leading tool holder
T="Mill_2""M06	;No address extension programmed → this refers to the leading tool holder; i.e. tool holder 2. The tool change is performed at buffer location 2. The path is corrected with the tool offset.
...	
T1="Drill_1" M1=6	;Address extension for the secondary tool holder has been programmed. The tool change is performed at the buffer location with $TC_MPP5=1. The path is not corrected.
...	
SETMTH	;Declare the tool holder given by TOOL_MANAGEMENT_TOOLHOLDER to be the leading tool holder
T="Mill_3" M06	;No address extension programmed → this refers to the leading tool holder; i.e. tool holder 1 (value of the MD TOOL_MANAGEMENT_TOOLHOLDER). Tool change is performed at buffer location 1. The path is corrected with the tool offset.

Example (1)

The following applies:

```
$MC_RESET_MODE_MASK = "H18041"
$MC_SPIND_DEF_MASTER_SPIND = 1
$MC_TOOL_MANAGEMENT_TOOLHOLDER = 2
```

After the end of program/RESET both the active tool offset and the programmed values for SETMTH and SETMS remain active. The tool change still does not take place at the spindle, but rather at the tool holder instead.

Program code	Comment
T="Drill" M06 D2	;Tool change to leading tool holder = 2
SETMS(3)	;New leading spindle = 3
SETMTH(1)	;New leading tool holder = 1
T="Mill" M06 D1	;Tool change to leading tool holder = 1
M17	

After end of program or RESET, the following applies:

- Spindle no. = 3 is the leading spindle
- Tool holder no. =1 of the leading tool holder
- Tool = "Milling tool" with offset D1 determines the path offset

After Power On, the settings for the machine data take effect:

- Spindle no. = 1 is the leading spindle
- Tool holder no. = 2 is the leading tool holder.

The tool offset is derived from the lowest D number of the tool that is located in the main tool holder; i.e. which is T="Drill" with D1

(Assuming that the tool has two D offsets D1, D2).

Example (2)

The following applies:

```
$MC_RESET_MODE_MASK = "H41"
$MC_SPIND_DEF_MASTER_SPIND = 1
$MC_TOOL_MANAGEMENT_TOOLHOLDER = 0
```

After the end of program/RESET both the active tool offset and the programmed value for SETMS remain active. The tool change takes place at the spindle which now becomes the tool holder.

Program code	Comment
T="Drill" M06 D2	;Tool change to leading tool holder = 1
SETMS(3)	;New leading spindle = leading tool holder = 3
T="Mill" M06 D1	;Tool change to leading tool holder = 3
M17	

After end of program or RESET, the following applies:

- Spindle no. = 1 is the leading spindle
- Tool = "Milling tool" with offset D1 (on spindle with no. = 3) determines the path offset.

After Power On, the settings for the machine data take effect:

- Spindle no. = 1 is the leading spindle/leading tool holder.
- The tool offset is derived from the lowest D number of the tool that is located in the leading tool holder; i.e. which is T="Drill" with D1 (assuming that the tool has two D offsets D1, D2).

Reset behavior

The programmed values from SETMTH can remain active beyond program end/RESET/START.

11.10.22 POSM - Positioning the magazine

Function POSM positions a magazine to a particular location in an internal magazine (e.g. spindle, tool holder, loading magazine), irrespective of the current assignment of the location or the status of the tool it contains. The NC command includes parts of the OPI PI service _N_TMPOSM.

Note

The NC command POSM(...) is terminated without waiting for acknowledgement from the PLC.

Syntax

```
POSM (p, m, ip, im)
```

Meaning

POSM	Positions a magazine at a particular location	
	Data type:	-
p	Location number to which the magazine is to be positioned	
	Data type:	INT
m	Magazine number of the magazine to be moved (optional). If m is not set, the location number refers to the magazine contained in the distance table as the first magazine for the specified internal location. The magazine (number m) must be linked by a distance relationship with the selected loading or buffer-magazine location. Alarms are generated when incorrect parameters are specified (e.g. undefined location numbers).	
	Data type:	INT
ip	Location number for the specified internal magazine (spindle location, loading magazine, etc.) This parameter is optional. If ip is not specified, the positioning operation refers to the main spindle location or the main tool holder location.	
	Data type:	INT

in	Magazine number of internal magazine in relation to location number ip to which the magazine must be moved. An internal magazine is either a loading or a buffer magazine. This parameter is optional. If im is not specified, the command refers to the buffer magazine.	
	Data type	INT

Example

Specified configuration:

- Magazine (magazine number = 1),
- Spindle (buffer magazine = 9998, location 1),
- Loading magazine (loading magazine = 9999, location 2).

Program code	Comment
N100 POSM(4, 1, 1, 9998)	; It is moved from magazine 1 of location number 4 to the spindle.
N100 POSM(4, 1, 1, 9999)	; Command for traversing to loading magazine

Example with result check

A magazine is specified as shown in the following diagram.

Location 12 is to be positioned at the change position and the program must not be continued until positioning has been successfully completed (simplest case with only one magazine and one defined change position).

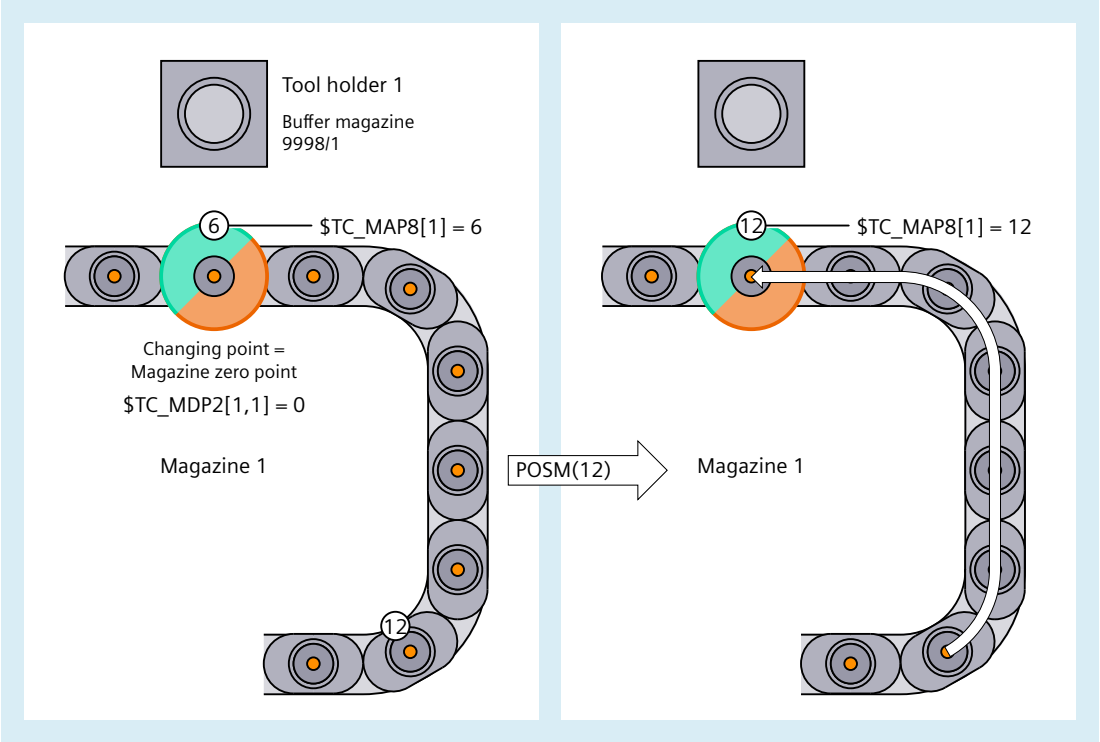


Figure 11-12 Magazine positioning with a check of the positioning operation result

In this example, the magazine zero point is the location in front of tool holder 1. It is defined by system variable $TC_MDP2. Tool holder 1 is assigned to the leading spindle of the channel.

Program code	Comment
N100 POSM(12)	;Moves location 12 to the change position, any non-programmed parameters are set internally to POSM (12, 1, 1, 9998)
N200 wait:	
N300 G4 F1	;Waiting time according to the conditions prevailing at the machine (exit possibly necessary if reaction is required to positioning errors)
N400 if ($TC_MAP8[1] <> 12)	;goto wait
	;After POSM(12) is executed, the current magazine position must be equal to 12.

Multitool

The definition of function POSM remains unchanged. The magazine location to be positioned to is programmed irrespective of whether the location is empty, contains a tool or a multitool.

The PI service _N_TMPOSM offers various options for programming magazine positions; for instance, a tool can be programmed using its tool number or its name and duplo number. If this tool is at a magazine location, the system positions to this magazine location. If the tool is in a multitool, which in turn is at a magazine location, the system positions to this magazine location. If, instead of a tool number, an MT number is programmed and the multitool is at a magazine location, the system positions to this magazine location.

11.10.23 POSMT - position multitool on toolholder to location number

The function POSMT is available for TMMG.

Function POSMT positions a multitool that is located on a tool holder to the programmed MT location number. At the time that the command is executed, it is not permissible that any tool offset is active for the programmed tool holder, i.e. D0 must have been previously executed.

The complete command is: POSMT(state, MTlocno, THno)

Note

Selecting an offset or deselecting a possibly active tool offset is not linked with the command.

The command initiates an implicit STOPRE.

Syntax

```
state = POSMT(state, MTlocno, THno)
```

Meaning

POSMT	Positions a multitool to a programmed MT location number	
	Data type:	INT
MTlocnox	MT location number of the MT, which is located on the programmed tool holder.	
	Data type:	INT
THno	Tool holder number on which the MT to be positioned is located. This parameter is optional. If THno is not programmed, the number of the master tool holder is automatically used.	
	Data type:	INT
Return value		
state	Success status of command	
	Data type:	INT
	0	Successful execution of the command. Command successfully completed at the PLC. (PLC acknowledgement may still be pending).
	-1	Command cannot be used because TMMG and/or multitool function is not active.
	-2	Function is not executed because of block search, program test.
	-3	MT cannot be positioned, because a tool offset associated with the programmed tool holder is (still) active.
	-4	MT cannot be positioned, because the programmed tool holder does not contain an MT, but a tool.
	-5	MT cannot be positioned, because the programmed tool holder neither contains an MT nor a tool.

	-6	MTlocno has an invalid value.
	-7	THno has an invalid value.
	-8	Multitool is presently being changed
	-9	Type of distance coding ($TC_MTP_KD) is not defined.

Example 1

The tool with the name "Tool1" / duplo number = 5 or T number = 33 is loaded in multitool 555 at MT location = 2. The MT has been created with 6 locations.

Program code	Comment
DEF INT state = 0	;
T="Tool1" M06 D1	
	;The PLC changes MT 555 on tool holder 1 and positions it on MT location 2
	;The PLC acknowledges the change, axis movements are programmed ...
D0	;Deselect tool offset
POSMT(state, 5, 1)	
	;Position the MT on location 5
	;The previously active tool with T number 33 is still active (however, due to the POSMT command is no longer in the machining position)
M17	

The following is still configured - "activate tool of the tool holder and activate its offset D1".

With RESET (end of program) a tool change command is generated for the PLC, which has the following initial data:

- Bring the multitool from tool holder 1 from the tool holder to tool holder 1 (i.e. the MT is already located on the tool holder)
- Position the multitool at location 5

In the NC, after the input, the PLC end acknowledgement is initiated:

- Activate the tool with tool number 34 and select its offset D1

The multitool remains on the tool holder - and neither a tool nor D offset are active.

Example (2)

If, in example 1, POSMT is not programmed before the end of the program, but only the MT position is set to the value 5 in the NC with $TC_MTP8, the command to the PLC - generated in the init block - would specify this MT position and the PLC would have to position the MT according to the position data. When acknowledging the MT position specified by the NC in the init block, the PLC must not change this.

11.10.24 MVTOOL - NC command to move tool

Function MVTOOL is available for TMMG.

Function MVTOOL loads and unloads tools by NC programming only. It can also be used to transport a tool from one magazine location to any other magazine location.

It is mandatory for a tool to be positioned at the source magazine location.

This NC command does not generate an alarm.

Whether MVTOOL was carried out with or without error(s) must be checked via the return value of parameter "state".

The command can also be applied to multitools.

Syntax

```
MVTOOL (state, magFrom, locFrom, magTo, locTo)
```

Meaning

MVTOOL	Loads and unloads tools	
	Data type:	-
magFrom	Magazine number of the magazine in which the tool to be moved is located	
	Data type:	INT
locFrom	Location number of location in which the tool to be moved is located	
	Data type:	INT
magTo	Target magazine number of magazine to which the tool is to be moved. This can be a loading magazine, buffer magazine or another real magazine . This parameter is optional. If magTo is not programmed, the value from magFrom is used.	
	Data type:	INT
locTo	Target location number of location to which the tool is to be moved. This parameter is optional. If locTo is not programmed or the value = 0 is programmed, an empty location search is conducted in magazine magTo - if this is a real magazine . A location search cannot be performed in an internal magazine. If magTo is the number of an internal magazine and locTo is not programmed or programmed with the value=0, status=-12 is set.	
	Data type:	INT
state	Success status of command	
	Data type:	INT
State number	$MC_TOOL_CHANGE_ERROR_MODE Bit 10	

0	= 0	Execution was successful (PLC acknowledgment may still be pending)
	= 1	
-1	= 0	Command cannot be used because TMMG is not active.
	= 1	
-2	= 0	Function is not performed because of block search, program testing
	= 1	
-3	= 0	Tool cannot be moved (because e.g. tool status "being changed" is set)
	= 1	
-4	= 0	No tool located in source location
	= 1	
-5	= 0	`magFrom` has invalid value
	= 1	
-6	= 0	`locFrom` has invalid value
	= 1	
-7	= 0	`magTo` has invalid value
	= 1	`magTo` has an invalid value (or magTo is a locked magazine)
-8	= 0	`locTo` has an invalid value
	= 1	
-9	= 0	No distance relation defined (if exactly one magazine is an internal magazine).
	= 1	
-10	= 0	No empty location found (if parameter locTo is not programmed) `magTo` is a locked magazine, statuses from -13 to -17 also lead to this status
	= 1	No empty location found (if parameter locTo is not programmed)
-11	= 0	The target location is occupied by another tool
	= 1	
-12	= 0	The parameter `locTo` must be programmed, as `magTo` is an internal magazine.
	= 1	
-13	= 0	Not available
	= 1	The target location is reserved for another tool from the temporary storage.
-14	= 0	Not available
	= 1	The target location is reserved for another tool to be newly loaded.
-15	= 0	Not available
	= 1	The target location is locked.
-16	= 0	Not available
	= 1	The type of the target location does not match the tool to be moved, or the type of the tool is not defined.

-17	= 0	Not available
	= 1	The target location is occupied or reserved in a half-space for another oversized tool.
-18	= 0	Not available
	= 1	The function tries to move an oversized tool, which would result in a tool collision on a secondary station.

Note

The command MVTOOL always reserves the target location with "reserved for tool to be loaded".

The command sets the tool status of the tool to be moved to "tool is being changed", bit value "H20" from the command preparation to the command completion (acknowledgment with status 1 or 3).

Command completion means the successful completion with PLC status = 1 or 10 - or the command is interrupted with PLC status = 3.

If a tool is moved from a real magazine to an internal magazine (or vice versa), the respective magazine distance relation must be defined.

If the programmed parameters are invalid or if a tool is not located at the source magazine location, or if the programmed target location cannot be occupied by the tool, or if the empty location search does not find a location because parameter locTo has not been programmed, or if a magazine distance relation that is required has not been defined, the appropriate error code is returned via state.

Deselecting an active tool offset is not linked with the command.

PLC

In the NC, the NC command generates the command for the PLC, outputs the command within the scope of executing the active block to the PLC. The active block is only considered to have been executed if the end acknowledgment of the command is available from the PLC. This can be the PLC status = 1, 3, or 5. Only then can a new block be loaded for execution.

Examples

The following configuration exists:

- Magazine (Magazine no. = 5, locations 1, ...10),
- a spindle (buffer magazine no. = 9998, location 1),
- a loading point (loading magazine no. = 9999, location 1)

The magazine is linked to the spindle and the loading point via a distance relation (see $TC_MDP1/$TC_MDP2).

Example 1

The tool from loading location 9999/1 is to be loaded in magazine 5. The following is programmed:

Program code	Comment
def int state	
$TC_MPP6[9999, 1] = 123	; place tool with internal T no. = 123 onto loading location
MVTOOL(state, 9999, 1, 5)	; searching for a suitable empty location in magazine 5
;If the tool is to be loaded onto precisely location 7:	
MVTOOL(state, 9999, 1, 5, 7)	; before loading, a check is performed to ensure location 7 is empty

Example 2

The tool with T No. = 123 is to be loaded from the loading location to spindle 1 (magazine No. = 9998, location No. = 1):

Program code	Comment
$TC_MPP6[9999, 1] = 123	
MVTOOL(state, 9999, 1, 9998, 1)	

Example 3

With the same configuration, a tool that is loaded on location 7 is to be moved to another suitable location in the same magazine.

Program code	Comment
MVTOOL(state, 5, 7, 5)	
;or to location 3 in the same magazine	
MVTOOL(state, 5, 7, 5, 3)	
;to any location in magazine 11	
MVTOOL(state, 5, 7, 11)	

11.10.25 SETTA - Activate tool from wear group

Procedure SETTA sets all tools belonging to the wear group that are not disabled to active. However, from a tool group, no more than one tool, with reference to a spindle or tool holder, becomes active.

If, at the call instant, no wear groups have been defined or if the function "wear group" via machine data has not been activated, the command can still be used. The call parameters must then be set accordingly.

If the selection criteria of SETTA match several tools within a tool group, selection criterion $TC_MAMP3, bits 12-13 is applied. If there is already an active tool that fulfills the criteria of SETTA, no additional tool (relating to the same spindle) is activated.

If SETTA is not parameterized, the active setting refers to all tools that are loaded, released, and not blocked in the TO area.

Syntax

```
SETTA (STATUS, MNO, VNO, USEKT)
```

Meaning

SETTA	Sets all tools belonging to the wear group that are not disabled to active	
	Data type:	-
MNR	Magazine number (optional)	
	Data type:	INT
	0	or Not specifying the parameter means that active setting applies to all magazines - irrespective of whether a distance relation to a spindle location exists.
	-1	All magazines with a distance relation to a spindle or tool holder.
	-2	An active tool from every tool group represented in the magazine is set in the magazine of each spindle or each tool holder (equivalent to calling SETTA several times with the specific magazine numbers that are involved). This means there can be, for example, 2 tool holders with 1 magazine assigned to each. The tools of a group can be distributed in any configuration among the two magazines. If both magazines include tools of a group, this means that two tools from the tool group are allocated the active state.
If several magazines are addressed (MNR ≤ 0), the magazines are regarded as a unit and a tool from the tool group is activated across all of the magazines that are being considered. Blocked magazines ($TC_MPP3, Bit 1=1) are not taken into account.		
VNR	Wear group number (optional)	
	Data type:	INT
	0	The active setting applies to the magazine locations that are not assigned to a wear group.
	-1	Active wear group ($TC_MAP9). If there is no active wear group in any of the magazines, status -1 is returned and the status of the tools is not changed.
	-2	or if the parameter is not specified, this means that active setting applies to the whole magazine. Especially parameter VNR is not specified or set to -2 if the system does not use the "wear group" function.
USEKT	Tool subgroup (optional)	
	Data type:	INT
	0	or If the parameter is not specified, all tools in the group are considered.
	-1	The currently programmed value of $P_USEKT is used.

STATUS	Return parameter (optional)	
	Data type:	REF INT
	0	Procedure was correctly executed.
	1	Procedure was executed, but another tool is active in the group (e.g. an unloaded tool).
	-1	Procedure was not executed because there is no active wear group in the selected magazines.
	-2	Procedure was not executed because the programmed wear group number does not exist.
	-3	Procedure was not executed because the programmed magazine number does not exist.
	-4	Procedure was not executed because the function "wear group" is not enabled (MN_TOOL_MANAGE-MENT_MASK).
	-5	Procedure was not executed for other reasons.

A search strategy can be set in parameter $TC_MAMP3 for the tool to be activated by SETTA.

Bit 12 = 0	Smallest possible duplo number (default)
Bit 12 = 1	Smallest possible magazine location
Bit 13 = 1	Smallest possible number contained in parameter $TC_TP10 (sequence of use)

Note

It is mandatory to set the wear group for the SETTA procedure.

Multitool

If SETTA is programmed for a multitool, instead of for a magazine, the following alarm is generated: Alarm 6462 "[Channel %1:] block %2 command %3' can only be programmed for magazines. '%4' means no magazine.

See also

SETTIA - Deactivate tool from wear group (Page 210)

11.10.26 SETTIA - Deactivate tool from wear group

Procedure SETTIA resets the status "active" in all tools. By parameterizing the NC command, this can be either magazine-specific or wear group-specific.

Note

It is mandatory to set the wear group for the SETTIA procedure.

Syntax

```
SETTIA(STATUS, MNO, VNO,USEKT)
```

Meaning

SETTIA:	Sets all tools belonging to the wear group that are not disabled to inactive	
	Data type:	INT
MNR	Magazine number (optional)	
	Data type:	INT
	0	The inactive setting is applied to all magazines irrespective of any assignment to a spindle. In this case, the tools in the buffer are also considered as well as the tool holder.
	> 0	Magazine number in which the inactive setting will be applied. Tools in this magazine that are in the buffer are not considered. This means that if these tools are placed back into the magazine, they still have "active" status.
	-1	All magazines with a distance relation to a spindle or tool holder.
VNR	Wear group number (optional)	
	0	The inactive setting applies to all tools that are not assigned to a wear group. If no wear group is defined, the inactive setting is applied to all tools in the magazine.
	> 0	Wear group number in which the active setting will be applied.
	-1	Active wear group ($TC_MAP9).
USEKT	Tool subgroup (optional)	
	0	All tools in the group are considered.
	> 0	The tools are considered that have a bit set in the value programmed in USEKT in parameter $TC_TP11.
	-1	The currently programmed value of USEKT is used.
STATUS	Return parameter (optional)	
	Data type:	REF INT
	0	procedure was able to be executed correctly.
	-1	Procedure was not executed because there is no active wear group in the selected magazines.
	-2	Procedure was not executed because the programmed wear group number does not exist.
	-3	Procedure was not executed because the programmed magazine number does not exist.
	-4	Procedure was not executed because the function "wear group" is not enabled (MN_TOOL_MANAGEMENT_MASK).
	-5	Procedure was not executed.

A search strategy can be set in parameter $TC_MAMP3 for the tool to be activated by SETTIA.

Multitool

If SETTIA is programmed for a multitool, instead of for a magazine, the following alarm is generated: Alarm 6462 "[Channel %1:] block %2 command %3' can only be programmed for magazines. '%4' means no magazine".

See also

SETTA - Activate tool from wear group (Page 208)

11.10.27 RESETMON - NC command for setpoint activation

Procedure RESETMON sets the actual value of the tool to the setpoint.

The original maximum values, such as target tool life and target workpiece unit quantity, tool states such as "Tool disabled" (prewarning limit reached) are reactivated and the external disable methods (the OEM bits of $TC_TP_SUSPENSION_REASON) are reset accordingly.

This command is used for tools and multitools.

Syntax

```
RESETMON (state, t, d, mon, resetStates)
```

Meaning

`RESETMON`	Sets the actual value of the tool to the setpoint	
	Data type:	...
`state`	Return parameter, which can have the following values:	
	Data type:	REF INT
	0	Command was successfully executed
	-1	The cutting edge with the specified D number d does not exist.
	-2	The tool with the specified T number t does not exist.
	-3	There is no monitoring function defined for the specified tool. This status is only possible if t has been specified explicitly.
	-4	Monitoring function is not active in the NC, i.e. the command has not been executed.
`t`	Internal T number	
	Data type:	INT
	t = 0	All tools are handled.
	t > 0	Precisely this tool is handled.
	t < 0	The absolute value of t is formed and all replacement tools of this tool are handled.
`d`	D number of the tool (optional). If the parameter is not specified at all or is assigned the value 0, all D numbers or all cutting edges of the tool are handled.	
	Data type:	INT
	d > 0	The command applies specifically to the specified D number.

`mon`	Bit-coded parameters (optional) If the parameter is either not specified at all or assigned the value 0, all actual values of the active, tool-specific monitoring functions for the designated cutting edge(s) are set to the setpoints. In addition, all external disable methods are reset.	
	mon > 0	Precisely the actual value of the specified monitoring type and/or external disable methods are handled. Possible values for the 4 bits are the positive values of the system parameter **$TC_TP9** (1, 2, 4, 8) or the corresponding bit combinations when several monitoring types are activated. The next 12 bits of the parameter apply to the OEM bits, which represent the external disable methods (see the definitions of the $TC_TP_SUSPENSION_REASON system variable). By definition, they are always active, so the behavior of these bits is not tied to the current value of $TC_TP9.
	mon < 0	Precisely the actual value in the monitoring type specified in "absolute value mon" is handled. There is no restriction due to the system variable values $TC_TP9. The values of non-activated monitoring types can also be reset in this way too. In particular, it is also possible to simultaneously reset the actual values of wear and additive offset monitoring values.
`resetStates`	Bit-coded parameters (optional)	
	Bit 0	Tool status "active" is deleted
	Bit 1	Tool status "enabled" is set
	Bit 2	Tool status "disabled" is reset if • a) the monitoring data permit this • b) parameter "mon" is set accordingly
	Bit 3	Tool status "measure" is set
	Bit 4	Tool status "prewarning limit" is reset if • c) the monitoring data permit this • d) parameter "mon" is set accordingly
	Bit 5	Not permitted
	Bit 6	Not permitted
	Bit 7	Tool status "was in use" is deleted
	Bit 8	Not permitted
	Bit 9	Not permitted
	Bit 10	Tool status "to unload" is deleted
	Bit 11	Not permitted
	Bit 12	Not permitted

The parameter `resetStates` allows selective modification of the tool status in addition to the monitoring parameters. The bit coding for `resetStates` corresponds to that for the tool status parameter $TC_TP8[x].

If this parameter is not specified, machine data $MN_TOOL_RESETMON_MASK is accessed. The bit coding for this data is identical to that for parameter "resetStates". With the analogous PI service PI_TRESMO, this machine data is also effective.

RESETMON for multitools

Multitool

The parameters of RESETMON are extended for a multitool.

Syntax for multitools

```
RESETMON (state, MTno, Dno, mon, resetStates)
```

Meaning of the parameters for multitools

`state`	Return parameter, which can have the following values:		
	Data type:		REF INT
	0		Command was successfully executed
	-1		The cutting edge with the specified number `Dno` does not exist.
	-2		The multitool with the specified number `MTno` does not exist.
`MTno`	Internal T number		
	Data type:		INT
	t = 0		All tools are handled.
	t > 0		Precisely this tool is handled.
	t < 0		The absolute value of t is formed and all replacement tools of this tool are handled.
Dno	D number of the tool (optional). If the parameter is not specified at all or is assigned the value 0, all D numbers or all cutting edges of the tool are handled.		
	Data type:		INT
	Dno = 0		If the parameter is not specified, or if the tool is programmed with 0, all cutting edges of the tools are handled in the multitool.
	Dno > 0		The command refers precisely to the cutting edges of the tools in the multitool with the specified D number.
`mon`	mon applies to the tools in the multitool - see above		
	Data type:		INT
`resetStates`	`resetStates` applies, on one hand, to states of the individual tool, which when reset to the setpoints, must be additionally changed, but here, for this particular programming, also to the corresponding MT states, which are changed if at least the same state has been changed in one of the tools loaded in the multitool.		
	Bit	Value	Meaning

	1 =	0 1	Multitool status "enabled" remains unchanged Multitool status "enabled" is set
	2 =	0 1	Multitool status "disabled" remains unchanged Multitool status "disabled" is deleted if permitted by the monitoring data and the 4th parameter is set accordingly

Note

If the optional parameter `resetstates` is not programmed, the value of MD $MN_TOOL_RESETMON_MASK is considered an implicit value.

Analogously to this, the MT number can be programmed for the parameter T number of the corresponding PI service _N_TRESMO.

The MD $MN_TOOL_RESETMON_MASK available to parameterize the RESETMON command, with the description "behavior of the tool data for RESETMON", retains its effect on the tools - even if they are located in a multitool and are reset using RESETMON.

The selectable state changes also affects the multitool, if this has been programmed and the appropriate state has been defined.

		Tool	MT
Bit 0	Delete "active" status	X	-
Bit 1	Set the "enabled" status	X	X
Bit 2	Conditionally delete "disabled" status if monitoring data permit this	X	X
Bit 3	Set the "measured" status	X	-
Bit 4	Conditionally delete "Prewarning limit" status if monitoring data permit this	X	-
Bit 7	Delete "was in use" status	X	-
Bit 10	Delete "to unload" status	x	x

Multitool example (1)

The actual time monitoring value (4th parameter = 1) of the cutting edges with the D number 2 (3rd parameter) of all tools that have been loaded to the MT with the number 500, is reset to the setpoint. As part of this, the state value of the tools whose time monitoring has been changed, is changed in accordance with the settings of the last parameter.

As 500 is the number of a multitool, the state of this multitool is also changed in accordance with the settings of the last parameter. Note that only the programmed bits 1, 2, 10 are relevant for the MT state change.

The success of the command is indicated in the "state" reference parameter.

```
int state = 0

RESETMON(state, 500, 2, 1, "H49F")
```

MT 500 in the example is equipped with three tools. Each tool has two cutting edges with D no. = 1 and 2. Two are time-monitored and are disabled due to the time limit in the cutting edges with D number = 2; the third tool is workpiece count-monitored and is also disabled because the workpiece count limit has been reached in the cutting edge with the

D number = 2. The two time-monitored tools are now reset with RESETMON, the tool state "disabled" is canceled and the MT state "disabled" is also canceled (although the workpiece count-monitored tool is still disabled). The other programmed tool and MT states except "disabled" are changed according to the definition.

If the following had been programmed:

```
RESETMON(state, 500, 2, 0, "H49F")
```

all the monitoring actual values of all tools in the MT would have been reset to their setpoints (4th parameter = 0) and all tools would have reset the "disabled" state and the MT would have reset its "disabled" state. The other programmed states would also have been changed.

Multitool example (2)

The actual values of the time monitoring of all cutting edges and all tools are set to the setpoints of the time monitoring.

The states of the tools are changed in accordance with the programming (last parameter = "H405"; reset "disabled" state conditionally and reset states "active" and "to be unloaded" unconditionally). The state of the multitool is also changed in accordance with the programming (the MT state "disabled" conditionally, the state "to be unloaded" unconditionally).

```
int state = 0
RESETMON(state, 0, 0, 1, "H405")
```

11.10.28 DELTC - Delete toolholder data record

The function "Orientable tool carrier" must be active. The function can be additively superimposed over the functions TMMO and TMMG.

The function `DELTC` deletes a tool holder data set.

Syntax

```
DELTC(n,m)
```

Meaning

`DELTC`	Deletes a toolholder data set	
	Data type:	-
`n`	First number of the tool holder data area whose values are to be set to zero (optional). If `n` is not specified, all tool holder data sets are set to zero starting at the lowest through to the highest number.	
	Data type:	INT

m	Last number of the tool holder data area the values of which are to be set to zero (optional). If m is not specified, the tool holder data set identified by n is set to zero. If m is greater than the highest number of a tool holder data record in this channel, those data sets up to the highest number are set to zero.	
	Data type:	INT

The tool holder data sets are defined by the system variables $TC_CARRx. System variable $TC_CARR1[0] is provided for setting all data sets to zero. With DELTC, a range of numbers for the tool holder data from n to m can now be set to zero.

In particular, the content of DELTC is equal to $TC_CARR1[0]=0= set all data sets to zero.

Parameters n, m have to be programmed with values larger than zero. Other values result in an alarm.

Parameter n must be less than m. Other values result in an alarm.

Also, n must lie in the range of numbers permitted for tool holder data.

The selected range of numbers must include the range of numbers for the tool holder data sets on the channel. Programming is otherwise rejected and an alarm is issued.

If the function "Tool holder data" is not activated ($MN_MM_NUM_TOOL_CARRIER 0 0), DELTC will also generate an alarm.

Example

In the TO unit, 14 tool holder sets with the numbers 1 to 14 are defined.

Program code	Comment
DELTC(5,8)	;set the values of the data sets 5, 6, 7, 8 to zero
DELTC(5,20)	;Set the values of the data sets 5, 6, 7, ..., 14 to zero
DELTC(9)	;set the values of the data set 9 to zero
DELTC()	;set the values of the data sets 1, ..., 14 to zero
DELTC(0,1)	;error → alarm - n, m must be greater than zero
DELTC(0,-2)	;error → alarm - n, m must be greater than zero
DELTC(0)	;error → alarm - n must be greater than zero
DELTC(15,20)	;error → alarm - n must not be more than 14
DELTC(20)	;error → alarm - n must not be more than 14

11.10.29 TCA - Tool selection/tool change irrespective of tool status

Function TCA is available for TMMO and TMMG.

The function TCA loads a tool irrespective of its "disabled" or "to be unloaded" status.

For certain routines (e. g. measuring cycles), it is necessary to load a specific tool onto the spindle / the tool holder for tool change regardless of its status (e. g. a disabled tool).

Syntax

```
TCA("Tool name", duplo No., toolholder No.)
```

Meaning

`TCA`	Loads a tool irrespective of its status.	
	Data type:	-
`Tool name`	Identifier of the tool to be loaded	
	Data type:	STRING
`Duplo no.`	Duplo number of tool to be loaded (optional). If `Duplono.` is not specified, the tool with the lowest duplo number is loaded.	
	Data type:	INT
`Tool holder no.`	Tool holder or spindle on which the tool change is to be performed (optional). If `Tool holder no.` is not specified, the tool change refers to the currently set or programmed leading spindle or leading tool holder. The following applies for TMMO without TMMG: The parameter corresponds to the address extension of the T command. (The setting for MD $MC_T_M_ADDRESS_EXT_IS_SPINO is taken into account.)	
	Data type:	INT

TCA behaves like the T command in its alarm and command output.

If neither TMMG nor TMMO are active, an alarm is generated.

Any alarms occurring during programming are handled in the same way as the alarms during T programming.

Note

Offset selection, in accordance with $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.

Examples

1. Preparation and change with T command (i. e. $MC_TOOL_CHANGE_MODE=0)
   Configuration 1x turret, 1x tool holder
   There are 2 tools with the identifier "Finish cutter" and duplo numbers 1 and 2.
   ```
   TCA("Finish cutter", 1.1)
   ```
   The tool "Finish cutter" with duplo number 1 is loaded onto tool holder 1.
   With the machine configuration assumed above, the following programming would have the same result:
   ```
   TCA("Finish cutter")
   ```
   The duplo number is not specified, this means that the tool with the lowest duplo number is changed, i.e. duplo "1".
   Tool holder not specified. Therefore, the change is effective for the current main tool holder, i.e. "1".

2. Change with M06 ($MC_TOOL_CHANGE_MODE=1)
   Configuration: 1x chain magazine, 2x spindles, spindle_1 is the leading spindle. 4 tools are loaded, "MILL_20MM", with duplo numbers 4, 5, 8 and 15.
   "MILL_20MM", duplo "8" was disabled and must be measured. Measurement is taken on spindle 2.
   ```
   TCA("MILL_20MM",8,2)
   M2=6
   ```
   The tool "MILL_20MM", duplo "8" is prepared for spindle "2" and changed.
   In this case, the following programming would lead to a different result:
   ```
   TCA("MILL_20MM")
   M06
   ```
   Tool "MILL_20MM" with duplo "4" (lowest duplo number) is prepared for spindle "1" (this is the leading spindle) and changed with M06.

Note

The following special issues apply for TCA when compared to T commands:

- TCA and D cannot be programmed in the same block.
- TCA renders the set search strategies ($TC_MAMP2 and/or $TC_MAP10) ineffective and ignores the programmed values of $P_USEKT.
- The tool must have status "enabled".
- The PLC interface signals "Transition to new replacement tool" and "Last replacement tool of group" are not set.

PLC

The PLC is not allowed to reject a tool prepared with "TCA". Acknowledgement with Status_2, Status_7 or Status_103 is not permissible.

Notice: Currently the interface does not have any criteria as to whether a tool may be rejected or not.

If this function is being used, an additional identifier must be used to indicate this to the PLC.

Program code	Comment
`$TC_VDITCP[2] = 101`	`("101" - identifier that the PLC is not allowed to reject)`
`TCA("Mill",1)`	

Supplement to the definition if the programmed tool is a multitool

With the function TCA, a specific tool with "tool name"/"duplo number" can be loaded into the tool holder irrespective of the tool state. If this tool is a multitool, the state of the multitool as well as the state of the tool is ignored for the command. This especially allows a tool to be selected and/or swapped out from a disabled multitool.

Note

The command can only be programmed for tools - also those that are loaded in a multitool; however it cannot be programmed for a multitool. If the programmed name is an MT name for example, Alarm 6460 "[Channel%1:] block %2 command %3 can only be programmed for tools. %4 means no tool." is generated.

Note

A tool from a disabled magazine or magazine location cannot be loaded into the spindle or the tool holder even with TCA.

11.10.30 TCI - Load tool from buffer into magazine

The function TCI is available for TMMG.

The function TCI loads the tools from the buffer locations back into the magazine. Tool holder locations are however excepted from this. Generally, applications are for complex machines with several grippers.

The necessary empty location search is performed in the same way as for a programmed tool change with T.

Note

TCI cannot be programmed together with M06 in one NC block. Tool change preparation and execution are performed in one operation.

The TCI command cannot be substituted (T function replacement).

TCI(locNo, tool holder no.)

locNo	Number of the buffer whose tool is to be returned to the magazine. Since the locNo cannot be the location number of a tool holder, returning the tool has no effect on active tool offset.
Tool holder no.	This specifies the number of the tool holder from where the tool is to be removed. This parameter is optional. If this parameters is not specified, the current main tool holder is automatically selected.

Alarm 6403 is generated if an invalid location number is programmed.

Syntax

```
TCI(locNo, tool holder no.)
```

Meaning

TCI	Loads the tools from the buffer locations back into the magazine.	
	Data type:	-
locNo	Number of the buffer whose tool is to be returned to the magazine. Since the locNo cannot be the location number of a tool holder, returning the tool has no effect on active tool offset.	
	Data type:	INT
Tool holder no.	Specifies the number of the tool holder from where the tool is to be removed (optional). If tool holder no. is not specified, the current main tool holder is automatically selected.	
	Data type:	INT

Alarm 6403 is generated if an invalid location number is programmed.

The location number locNo is invalid

- if locNo indicates a tool holder / spindle (alarm 6450)
- if locNo indicates a non-defined buffer location (alarm 6403)
- if locNo is not linked to the programmed tool holder or main tool holder by $TC_MLSR (alarm 6454)
- if no distance table is defined either for the buffer locNo or for the tool holder / spindle (alarm 6454)

Alarm 6451 is generated if no buffer magazine has been defined.

Alarm 6452 is generated if the specified tool holder is not defined.

Alarm 6431 is generated if TMMG is not active.

To successfully program TCI, it is necessary for the specified location number locNo to be assigned to the tool holder via $TC_MLSR. Empty locations are searched for in the magazines defined in the distance table (defined by $TC_MDP2) of the buffer locNo or of the tool holder. If both the buffer locNo and the tool holder have a distance table, the buffer distance table is the one that is used. Alarm 6454 is generated if neither has a distance table.

Note

The command TCI contains as parameter, the number of a location (gripper, loader, transfer point) of the buffer magazine. To use this NC command in your own cycle programs, the location number can be identified using the system variable $P_MAGNREL, $P_MAGREL.

Examples

The following magazine configuration is present:

- Magazine 1

The following is defined in the buffer magazine with 5 locations:

- Spindle 2 (location 1) with grippers 1 and 2 (locations 3 and 4 coupled with the spindle through $TC_MLSR[3,1] = 0 and $TC_MLSR[4,1] = 0)
- Spindle 1 (location 2) with gripper 3 (location 5 coupled with the spindle through $TC_MLSR[5,2] = 0)

The following is programmed:

Program code	Comment
`TCI(2)`	`;generates alarm 6450`
`TCI(5)`	`;returns the tool from location 5 (gripper 3) back to the magazine`
`TCI(9)`	`;Alarm 6403 (buffer only has the numbers 1 to 5)`

The user determines the sequence in which the buffer locations are cleared by programming.

PLC

TCI is executed in the PLC like the programming of T0 M06.

Multitool

The TCI command also changes multitools from internal magazines into the magazine.

11.10.31 GETFREELOC - Search for empty location

Function `GETFREELOC` is available for TMMG.

The function `GETFREELOC` searches for an empty location for a given tool in those magazines assigned to the specified loading location or the specified spindle / tool holder location by an entry in the distance table. The strategy set by $TC_MAMP2 or $TC_MAP10 is used as the search strategy. The search strategy can be programmed with:

Defined location type hierarchies are taken into consideration during a search for an empty location by the PI service or in a programmed tool change.

Alarm 14020 is generated if fewer than three parameters are programmed.

Note

GETFREELOC only reserves the empty location found if the optional parameter `withReserv` is programmed with "L" or "S".

GETFREELOC(magNo&, locNo&, T no., refMag, refLoc, withReserv)

Find/check empty location for specified tool regarding specified loading / buffer magazine and location number. Reserve the location found corresponding to the programming of the parameter withReserve.

Syntax

```
GETFREELOC(magNo, locNo, T no., refMag, refLoc, withReserv)
```

Meaning

GETFREELOC	Searches for a free space in the magazine for a given tool	
	Data type:	-
mgNo	Magazine number of the magazine; the parameter is both an input parameter and a result parameter.	
	Data type:	REF INT
	Input value	
	> 0	Magazine number of the magazine in which the search is to be made. Note that only magazine numbers are valid that can be reached from the programmed reference location (parameter refMag/refLoc).
	0	The magazine in which the search is to be made is not specified. Search starts in the magazine that is the first in accordance with the selected search strategy.
	Output value	
	> 0	Magazine number of the magazine where the empty location is found.
	0	If no empty location is found.
	-1	TMMG not active
	-2	Invalid magazine number specified
	-3	Invalid magazine location number specified. The location number is also regarded as invalid when the magazine number is invalid.
	-4	Invalid T number tNo specified.
	-5	Invalid letter for "refMag".
	-6	If refMag = = "S", invalid tool holder number refLoc specified. If regMag = = "L", invalid load location number refLoc specified.
	-7	Parameter "withReserv" has an illegal value.
	-8	Parameter withReserv = "reserved for new tool to be loaded" cannot be programmed for tools that are in the buffer (whatever the value of the owner magazine for this tool).
	-9	Parameter withReserv = "reserved for tools in the buffer" is only possible for tools that are in the buffer.
	-10	The value of parameter "withReserv" is not compatible with the programmed value of parameter refMag.
	-11	First parameter has the value of a multitool number (i.e. empty location search in the MT). Only three parameters can be programmed.
	-99	Other (unexpected) problem in the programming.

locNo	Magazine number of the location; The parameter both an input parameter and a result parameter.	
	Data type:	REF INT
	Input value	
	> 0	Magazine location number of the location that is to be checked for receiving the specified tool. If magNo=0 is programmed, a value locNo > 0 is ignored.
	0	No specification of the magazine location. Search starts with the location that is the first according to the set search strategy.
	Result value	
	> 0	Magazine location number of the empty location that was found or checked.
	0	If no empty location is found.
	-1, -2, ... -99	Have the same meaning as the error status values of parameter magNo.
tNo	T number of the tool for which an empty location is to be searched. The searched location must be suitable for the tool size and the type of magazine location defined in the tool. If an invalid T number is programmed, parameters magNo, locNo each return the value -4.	
refMag	Reference magazine referred to for the empty location search (optional parameter).	
	Data type:	CHAR
	"S" = buffer magazine	
	"L" = load magazine	
	"-" = no reference magazine. Is used if a magazine definitely has to be specified.	
	If a value not equal to "S", "L" is programmed, parameters magNo, locNo each return the value -5. If the specified reference magazine is not yet defined, parameters magNo, locNo also return the value -5.	
refLoc	Spindle number/tool holder number;	
	Data type:	INT
	If refMag equals "S", then the spindle number / tool holder number is specified here for empty location search. If an invalid tool holder number is programmed, parameters magNo, locNo each return the value -6. If refMag equals "L", the number of the location in the loading magazine is specified for which a search will be made for an empty location. If an invalid number is programmed, parameters magNo, locNo each return the value -6. This parameter is optional. If it is not programmed, the search for the main tool holder is performed for refMag = "S". When refMag = "L", the search is performed for location number = 1 in the loading magazine. When refMag = "-", the parameter is not taken into account.	

withReserv	Reserved for the found/checked empty location in the real magazine.	
	Data type:	CHAR
	"L" = reserves the found/checked empty location in the real magazine with "reserved for tool to be loaded" (this is in system parameter $TC_MPP4, the bit value "H8). The reservation is only set when the programmed tool has not been loaded yet, i.e. the owner address of the tool is zero. Otherwise, the status is set to the value = -8.	
	"S" = if the tool for the programmed T number is already located at a location in the buffer: Reserve the found/checked empty location in the real magazine with "reserved for tool in the buffer" (in system parameter $TC_MPP4, the bit value "H4"; this is generally automatically set if the tool change is programmed with T/M06). If the owner magazine address of this tool is = 0 and the tool is located in a magazine location (i.e. the current magazine address of the tool is not 0): Also set this owner address to the value of the found/checked empty location. If the tool already has an owner magazine address, this is replaced by the new magazine address if the new address is different from the old address. If the location of the old owner magazine address is reserved, the reservation is canceled. If a location is already reserved for the programmed tool, this reservation is canceled before the new reservation is set. "-" = leaves the reserved status of the found/checked empty location unchanged. This parameter is optional. If it is not programmed, "-" is implicitly interpreted as programmed; i.e. the reserved status remains unchanged.	

Note

If several parameters are incorrect, the value of magNo, locNo will depend on which parameter NC checks first.

If an empty location is sought for a tool with fixed coding, GETFREELOC always returns the owner location of this tool.

Locations on internal magazines (loading magazine, buffer magazine) cannot be reserved.

If TMMG is not active, an alarm is generated.

Examples

There are two magazines with the magazine numbers 1 and 2 - each with five locations. There are also two spindles (spindle numbers 1 and 2 with location numbers 1 an 2) and a gripper (assigned location number 3, spindle number 1) in the buffer magazine and two loading locations (location numbers 1 and 2) in the loading magazine.

Both magazines are linked to the locations of the buffer and the locations of the loading magazine (via the distance relations). The following examples are independent of one another.

Empty location search / reservation for loading via loading location

```
def int magNo=0, locNo=0
```

```
def int tNo=44     ;Tool with T no. = 44 is defined

GETFREELOC  ( magNo, locNo, tNo, "L", 2 )
     ; for the defined tool with T no. = 44 an empty location is sought for loading location 2.
     ; The suitable empty location is in magazine 2, location 5. The references magNo/locNo
return
     ; the values 2/5.
     ; The state of the magazine location is not changed.

def int magNo=0, locNo=0

def int tNo=44     ;Tool with T no. = 44 is defined

GETFREELOC  ( magNo, locNo, tNo, "L", 2, "L")
     ; for the defined tool with T no. = 44 and empty location is sought for loading location 2.
     ; The suitable empty location is in magazine 2, location 5. The references magNo/locNo
     ; return the values 2/5.
     ; The location 2/5 is reserved "for the tool to be loaded".

def int magNo=0, locNo=0

def int tNo=44      ;Tool with T no. = 44 is defined

GETFREELOC  ( magNo, locNo, tNo, "L", 2, "S" )
     ; for the defined tool with T no. = 44 regarding loading location 2, a search is made for
an empty location.
     ; The programming is illegal because for refMag = "L", the programming
     ; with Reserv = "S" is not defined
     ; The references magNo/locNo return the status values = -10  /-10 = the value of
     ; parameter "withReserv" is not compatible with the value of
     ; parameter "refMag".

def int magNo=0, locNo=0
def int tNo = 44     ;Tool with T no. = 44 is defined

GETFREELOC  ( magNo, locNo, tNo, "L", 2, "e" )
     ; for the defined tool with T no. = 44 regarding the tool holder with location no. = 1,
     ;a search is made for an empty location.
     ; The programming is illegal because parameter "withReserv" has been programmed
     ; with the wrong value.
     ; The references magNo/locNo ; return the status values = -7 / -7 = parameter "withReserv"
     ;  has an illegal value.
```

Empty location search / reservation for loading via tool holder / spindle

```
def int magNo = 0, locNo = 0

def int tNo = 4   ;tool with T no. = 4 is defined

GETFREELOC  ( magNo, locNo, tNo, "S" )
```

or with the same function

```
GETFREELOC  ( magNo, locNo, tNo, "S", "-" )
     ; for the defined tool with T no. = 4, regarding the leading spindle, a search is made for
an empty location.
     ; The suitable empty location is in magazine 1, location 3. The references
     ; magNo/locNo return the values 1/3.
     ; The state of the magazine location is not changed.
```

```
def int magNo = 2, locNo = 0

def int tNo = 44

; Tool with T no. = 44 is defined, is on the tool holder and does not have an
; owner location in the magazine yet (is therefore not considered as loaded),
; or already has an owner location in a magazine (loaded)

GETFREELOC( magNo, locNo, tNo, "S", 1, "L" )
     ; for the tool with T no. = 44, regarding the tool holder with location number1, a
     ; search is made for an empty location. The programming is illegal because an attempt
is made
     ; to set the reservation "reserved for new tool to be loaded" for a tool in the
     ; buffer. The references magNo/locNo return the status values -8 / -8 = "Parameter value
     : cannot be programmed for tools that are in the buffer".

def int magNo, locNo

def int tNo = 44  ; Tool with T no. = 44 is defined, is on the tool holder

GETFREELOC( magNo, locNo, tNo, "S", 1, "S" )
     ; the same starting position applies as for the programming of the previous
     ; command. The suitable empty location is in magazine 2, location 5. The references
magNo/locNo
     ; return the values 2/5. Location 2/5 is reserved "for tool in the buffer".

def int magNo = 0, locNo = 0

def int tNo = 44
      ; Tool with T no. = 44 is defined and is in the gripper which is connected via $TC_MLSR
with
      ; the tool holder of the location with number 1

GETFREELOC( magNo, locNo, tNo, "S", 3, "S" )
     ; for the defined tool with T no. = 44, regarding the tool holder with location number 1, a
     ; search is made for an empty location. The suitable empty location is in magazine 2,
location 5. The references magNo/locNo
     ; return the values 2/5.
     ; The location 2/5 is reserved "for tool in the buffer".
     ; (This state is retained during NC run-up if the reserving tool is still in the
     ; buffer at this time)

def int magNo = 0, locNo = 0

def int tNo = 44   ; Tool with T no. = 44 is defined and is not in the buffer

GETFREELOC( magNo, locNo, tNo, "S", 2, "S" )
     ; for the defined tool with T no.  = 44, regarding the buffer location 2 a search is made
for an empty location
     ;.
     ; The programming is impermissible because an attempt is made to set the reservation
"reserved for
     : tool in the buffer" for a tool search of the buffer.
     ; The references magNo/locNo return the status values -9 / -9 = "Parameter value is only
     ; possible for tools that are in the buffer".

def int magNo = 0, locNo = 0

def int tNo = 44  ; Tool with T no. = 44 is defined, but not (yet) loaded.
```

```
GETFREELOC( magNo, locNo, tNo, "S", 1, "S" )
    ; An empty location
 is sought for the defined tool with T no. = 44 with tool holder no. 1.
; The suitable empty location is in magazine 2, location 5.
    ; But because the tool cannot be loaded,
    ; the last parameter cannot be programmed with the value "S".
    ; The reference magNo/locNo return the status values -8/-8 = "Parameter value cannot be
    ; programmed for tools that are in the buffer".

def int magNo = 0, locNo = 0

def int tNo = 44  ; Tool with T no. = 44 is defined, but not (yet) loaded.

GETFREELOC( magNo, locNo, tNo, "S", 1, "L" )
    ; for the defined tool with T no. = 44, regarding the tool holder with location number 1, a
    ; search is made for an empty location. The suitable empty location is in magazine 2,
location 5. The references
    ; magNo/locNo return the values 2/5.
```

11.10.32 DELMLRES - Delete location status "reserved for tool in the buffer"

Function `DELMLRES` is available for TMMG.

The function `DELMLRES` cancels the location reservation, type `resType` of the location with the number `magNo`.

Syntax

```
state = DELMLRES (magNo, locNo, resType)
```

Meaning

`delmlres`	Cancels location reservation	
	Data type:	INT
`magNo`	Magazine number of the magazine that contains the location.	
	Data type:	INT
	Unit:	
	Value range:	*MIN* < x < *MAX*
`locNo`	Magazine location number of the location whose reservation should be deleted.	
	Data type:	INT
`resType`	Type of reservation to be deleted (optional). • "L" Reservation for "tool to be loaded" ("L "= "Tool to be Loaded") • "S" Reservation for "tool in the buffer" ("S" = tool in spindle" or generally tool in buffer). If `resType` is not programmed, the reservation that is set is deleted - either "L" or "S". If no reservation is set, the command has no effect.	
Return value		
`state`	Result of the command execution	

	Data type:	INT
	0	Command successfully executed (even if the location was not reserved).
	-1	TMMG is not active
	-2	Invalid magazine number specified
	-3	Invalid magazine location number specified
	-4	Invalid letter for `resType`

Example 1

For the oversize tool located in gripper 1 (magazine No. 9998, location No. = 4), the location reservation, for tool in the buffer, is to be deleted.

```
Program code                                    Comment
def int tNo, magNo, locNo                       ;
tNo = $TC_MPP6[ 9998, 4 ]                       ;
magNo = $A_MYMN[ tNo ]                          ; Owner magazine
locNo = $A_MYLN[ tNo ]                          ; Owner magazine location
                                                that                                ; was
                                                reserved
state = DELMLRES ( magNo, locNo, "s" )          ; the same meaning would be "DELMLRES(mag-
                                                No, locNo)"
if (state < 0 ) gotof error
;To ensure that the location is also reserved for the tool that is intended to be
loaded on a tool change, the following additional check can be made:
if ( $TC_MPP66[ magNo, locNo ] GOTOF ERROR
```

Example 2

Load programmed tool via tool holder (magazine No. 9998, location No. = 2). The loading operation is programmed using the commands GETFREELOC, MVTOOL.

```
Program code                                    Comment
def int tNo, magNo, locNo                       ;
tNo = 3                                         ; this tool should be loaded
GETFREELOC (magNo, locNo, tNo, "S", 2, "L")     ; Empty location search with reser-
                                                vation
if ( (magNo > 0) ans (locNo > 0)
  and ( "something not as wanted") )
  if ( state < 0 ) GOTOF ERROR
endif
MVTOOL( state, 9998, 2, magNo, locNo )
```

11.10.33 DELMLOWNER - Delete owner magazine location of the tool

Function `DELMLOWNER` is available for TMMG.

The function DELMLOWNER deletes the owner magazine location for the tool or for the multitool in the buffer magazine.

Syntax

```
state = DELMLOWNER(t)
```

Meaning

DELMLOWNER>:	Deletes the owner magazine location for the tool or for the multitool in the buffer magazine	
	Data type:	INT
t	T number of the tool in which the owner location is deleted	
	Data type:	INT
Return value		
state	Result of the command execution	
	0	Command was successfully executed.
	-1	TMMG is not active
	-2	Tool with the T no. = t does not exist - not for t = 0 either
	-3	The tool is not loaded (does not have an owner location)
	-4	The tool is not in the buffer magazine

Examples

If the owner location is reserved for this tool, implicitly, the location reservation "for tool in buffer" is also deleted.

The system parameters $A_MYMN [t] and $A_MYMLN [t] read, after the delete operation, the value = 0.

Program code	**Comment**
`def int tNo`	`;`
`tNo = GETT( name, duplo )`	`; if only the name and duplo number of the tool are known`
`tNo = $TC_MPP6[ magNo, locNo ]`	`; if only the magazine location containing the tool is known`
`tNo = $TC_MPP66[ magNo, locNo ]`	`; if only the magazine location whose reservation is to be deleted is known`

Multitool

The function DELMLOWNER is also available for multitools.

```
state = DELMLOWNER(INT MTno)
```

- In multitool with number MTno, delete the owner magazine location in the buffer magazine
- Therefore, also delete the owner location of the tools - contained in the multitool - with numbers Tnoi; i=1, ..., $P_MTOOLNT.

After the deletion operation, system parameters $A_MYMN [MTno] / $A_MYMLN [MTno] and $A_MYMN [Tnoi] / $A_MYMLN [Tnoi] read the value = 0.

Note

After the deletion, the system parameters $A_TOOLMTN [Tnoi], $A_TOOLMTLN [Tnoi] and $A_MYMTN [Tnoi], $A_MYMTLN [Tnoi] still read the values of the multitool, of the MT location on which the tool with T number Tnoi has been loaded.

Note

The command cannot be programmed with the T number of a tool, which is loaded in a multitool. This programming is rejected with state = -5.

11.10.34 $P_USEKT - Tool change only with tools of subgroup

Overview

This function is available for TMMO or TMMG.

This command selects a subset of tools of a tool group which is then taken into account for the subsequent tool changes (tool technology group).

The subgroups are set via system variables $TC_TP11[t].

Name	**$P_USEKT**			
Meaning	$P_USEKT is a bit-coded value. Only the contents of bits 0 ... 15 are of significance. All tools having the parameter $TC_TP11 that have set one of the bits of $P_USEKT are available in the following tool changes. The value 0 means that "all bits are set". Alarm 22069 is generated if there is no such tool in a tool group for which a tool change was programmed.			
Data type	INT			
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	x	x	-	-
Implicit preprocessing stop	-	-		

Machine data $MC_RESET_MODE_MASK, bit 20 sets whether after a reset or end of program, the last programmed value of $P_USEKT is retained or whether $P_USEKT is set to the value entered in $MC_USEKT_RESET_VALUE.

Example

The tool group "Mill_25" comprises four tools.

(The following applies: Tool_Change_Mode=1)

```
Mill_25          Duplo 1          T_No. 1          TP11=1
Mill_25          Duplo 2          T_No. 2          TP11=2
Mill_25          Duplo 3          T_No. 3          TP11=4
Mill_25          Duplo 4          T_No. 4          TP11=8
%MPF
...
T="Mill_25"
```

`M06`	Every tool in this group can be loaded, as no selection has been made. The search strategy that has been set prevails.

`...`
`$P_USEKT=2`
`...`
`T="Mill_25"`
`M06`
	"Mill_25", duplo 2 loaded at change

`...`
`$P_USEKT=9`
`...`
`T="Mill_25"`
`M06`
	"Mill_25", duplo 1 or duplo 4 loaded at change (depending on the selected search strategy)

`...`
`$P_USEKT=0`
`...`
`T="Mill_25"`
`M06`
	Every tool in this group can be loaded, as USEKT=0 has canceled the selection. The search strategy that has been set prevails.

`...`
`$P_USEKT=65535`
`...`
`T="Mill_25"`
`M06`
	Every tool in this group can be loaded, as all bits are set. The search strategy that has been set prevails.

Behavior at NC start

The last set $P_USEKT value remains effective. As $P_USEKT can only be changed by an NC program, the $P_USEKT that was set at the last NC reset is effective.

It is possible, for example, to program a desired value n in $P_USEKT by using ASUPs at the program start (ProgEvents).

Behavior at NC reset

For a more precise definition of the behavior at NC reset and NC start, a further bit is defined in $MC_RESET_MODE_MASK:

The newly defined bit 20 = 1 together with bit 0 = 1 of the same machine data specifies that the programmed $P_USEKT value is retained at NC reset.

In the other three possible combinations of bit 0 and bit 20, the value is taken from MD $MC_USEKT_RESET_VALUE.

Overview of the behavior for various settings:

The value from `$MN_USEKT_RESET_VALUE` is 8.

The values from `$P_USEKT` results in the following programs

1st program:

```
N10 $P_USEKT=2
N20 M00
N30 M02
M02
```

2nd program:

```
N100 R0=$P_USEKT
```

Setting of $MC_RESET_MODE_MASK	Bit 0 = 0 Bit 20 = 0	Bit 0 = 1 Bit 20 = 0	Bit 0 = 0 Bit 20 = 1	Bit 0 = 1 Bit 20 = 1
Sequence of actions				
After run-up	8	8	8	8
For "N20 M00" (after setting P_USEKT)	2	2	2	2
After end of program	8	8	8	2
For "N100 M00" (after program start)	8	8	8	2

Bit coding makes it possible for a tool to belong to several tool subgroups. A maximum of 16 different tool subgroups can be implemented through the configuration of NC; i.e. only bits 0 ... 15 are taken into account – not only during the programming of $P_USEKT, but also for $TC_TP11. Tools with $TC_TP11[t] = 0 do not count as a tool subgroup.

Note

The programming $P_USEKT = 0 means that all tools of the tool group are considered in the tool selection.

The value $TC_TP11[t] = 0 means "the tool belongs to all defined tool groups". This ensures compatibility with existing data records.

If the function T=location is used with automatic setting of $P_USEKT, then $P_USEKT cannot be programmed, alarm 6441 is output.

T=location, automatic tool selection

First the attempt is made to load the tool from the programmed magazine location, independently of the value in $TC_TP11. With the selection of the tool, the TP11 value of this tool is taken into $P_USEKT and is considered as programmed. This means that only tools of this subgroup can be selected.

However, if the tool at the programmed location is disabled, then the $TC_TP11 value of the tool at the programmed magazine location is considered in order to access the sister tool. Only tools that have one of the bits of the disabled tool in $TC_TP11 can be replacement tools.

MD setting: $MC_TOOL_MANAGEMENT_MASK bit 16=1, bit 22=1

Behavior with the function T=location

If the tool change is programmed with the function T=location, then the $P_USEKT command can also be used. However, the tool on the programmed magazine location in parameter $TC_TP11 then has to have at least one bit of the programmed $P_USEKT value set. Otherwise, the tool of another magazine location is selected.

MD setting: $MC_TOOL_MANAGEMENT_MASK bit 16=1, bit 22=0

11.10.35 TOOLGNT/TOOLGT - Tool groups

Functions `TOOLGNT` and `TOLLGT` are available for TMMO and TMMG.

The abbreviation `TOOLGNT` means: **TOOLG**roup**N**umberOf**T**ools - number of tools of the group.

The abbreviation `TOOLGT` means: **TOOLG**roup**T**oolNumber - T no. of the i-th tool of tool group with i = 1, ...TOOLNTG.

The NC commands allow information to be obtained about the tools in the tool group.

TOOLGNT("identifier")

TOOLGT("identifier", i)

Syntax

```
result = TOOLGNT("identifier")
result = TOOLGT("identifier", i)
```

Meaning

`TOOLGNT`	Number of tools of the group	
`TOOLGT`	T number of the i-th tool of the tool group	
	Data type:	INT
`Identifier`	Name of tool	
	Data type:	STRING

i	Tool group; i = 1 TOOL0TNG	
	Data type	STRING
Return value		
result	Result value	
	Data type:	INT
	> 0	successful read access
	0	i is a value outside of the permitted range
	-1	Neither function TMMG nor function TMMO active
	-2	No tool exists for the name "identifier"

The NC defines the sequence of the tools in the tool group and this changes during the programmed tool change with tools from this group.

Example

The tools "Drill_6mm"/Duplo_1, "Drill_6mm"/Duplo_2 and "Drill_6mm"/Duplo_3 are available.

The number of tools of the group "Drill_6mm" is read-out first.

```
Program code                                    Comment
R1=TOOLGNT("Bohrer_6mm")                        ; R1=3
;The T number of these tools are then determined.
R11=TOOLGT("Drill_6mm,1))
R12=TOOLGT("Drill_6mm,2)
R13=TOOLGT("Drill_6mm,3)
```

11.10.36 $P_TMNOIS - is number, T number, magazine number, or MT number

The system variable $P_TMNOIS is available with TMMO and TMMG.

The system variable $P_TMNOIS returns the number property. Individual tools, magazines as well as multitools use the number range 1 ... 32000 for numbering. As a consequence, a distinction must be made as to what the number actually refers to. This system parameter does this.

Syntax

```
result = $P_TMNOIS[ nr ]
```

Meaning

$P_TMNOIS	Returns the number property of the specified number.	
	Data type:	INT
nr	Number to be checked	
	Data type:	INT
Return value		

result	Property of the number	
	Data type:	INT
	3	no is the number of a defined tool and the number of a defined magazine
	2	no is the magazine number of a defined magazine
	1	no is the T number of a defined tool
	0	no is the MT number of a defined multitool
	-3	Invalid number. no is neither the number of a tool, a magazine nor a multitool.

Examples

Checks what the number stands for, e. g. a T number, MT number, or an invalid number

Program code	Comment
def int result	;
def int no=4711	;

11.10.37 $P_TOOLEXIST - Determine existence of a tool

System variable $P_TOOLEXIST returns the information whether a tool exists for a T number.

$P_TOOLEXIST is parameterized with the internal T number. Either true or false is returned. If the T no. that returns a multitool is $P_TOOLEXIST=false, the T no. that returns an existing tool is $P_TOOLEXIST=true. $P_TMNOIS (Page 235) can be used as an alternative.

Syntax

```
$P_TOOLEXIST[t]
```

Meaning

$P_TOOLEXIST	Returns the information as to whether a tool exists for a particular number.	
	Data type:	BOOL
	TRUE	A tool with this number exists
	FALSE	A tool with this number does not exist.
t	T number of the tool	
	Data type:	INT

The two commands are each available in all function versions of the tool management. The following still applies: the result value True from $P_TOOLEXIST is equivalent to the result value 1 from $P_TMNOIS. Settings for the following examples:

Examples

Example 1: A new tool is to be created (with a T number which is not identical to that of a multitool or an already existing tool).

Program code	Comment
N1 DEF INT no=2	; number to be checked
N5 if (($P_TMNOIS[no] != 0) and ($P_TMNOIS[no] != 1))	;
N6 $TC_TP2[no] = "Tool_2"	;generate new TL
N7 endif	

Example 2: Example 2: A new MT is to be created (with a T number that is not identical to that of an already existing multitool, or an already existing tool, or an already existing magazine):

Program code	Comment
N1 DEF INT no=2	; number to be checked
N5 if (($P_TMNOIS[no] != 0) and ($P_TMNOIS[no] != 1) and ($P_TMNOIS[no] != 2))	;
N6 $TC_MTPN[no] = 5	; new MT with 5 locations
N7 endif	

Example 3: An explicit number is to be checked as to whether it belongs to a defined tool:

Program code	Comment
N1 DEF INT no=2	; number to be checked
N5 if ($P_TOOLEXIST[no] == true)	;
N6 $TC_TP9[no] = 1	; define the type of tool monitoring
N7 endif	

11.10.38 $A_TOOLMN - read magazine no. from tool

System variable $A_TOOLMN returns the magazine number of the tool with T number t.

TOOLMN stands for "**toolm**agazine **n**umber".

Syntax

```
$A_TOOLMN[t]
```

Meaning

$A_TOOLMN:	Returns the magazine number	
	Data type:	INT
	0	The tool is not assigned to any magazine.
	-1	The tool management function is not active.
	-2	No tool with the specified T number exists.
	If the T number is outside the value range, an alarm is output.	
t	T number of the tool	
	Data type:	INT

11.10.39 $P_MTOOLN/$P_MTOOLMT- Determine the number of multitools/MT numbers

System variables $P_MTOOLN and $P_MTOOLMT are provided with TMMG.

The following system variables provide an overview of the defined multitools.

Syntax

```
result = $P_MTOOLN
result = $P_MTOOLMT[i]
```

Meaning

$P_MTOOLN	Returns the number of multitools in this channel.	
	Data type:	INT
Return value		
result	Result value	
	> 0	Number of defined multitools
	0	No multitool defined
	-1	TMMG function not active
	-2	Multitool function not active
$P_MTOOLMT	Returns the multitool number of the i-th multitool	
	Data type:	INT
i	Number of the multitool	
Return value		
result	Multitool number of the multitool	
	Data type:	INT

Example

Program code	Comment
def int number = 0	
def int i = 0	
def string[32] mtName	
number = $P_MTOOLN	
for i = 1 to number	
r[i] = $P_MTOOLMT[i]	;write all MT numbers of the defined multitool :in consecutive R parameters
endfor	
mtName = $P_MTP2[$P_MTOOLMT[i]]	;write the name of the last multitool after mtName

11.10.40 $P_MTOOLNT/$P_MTOOLT - Number of tools in the multitool

System variables $P_MTOOLNT and $P_MTOOLT are made available with TMMG.

Note

Also refer to the analog tool-specific system parameters $P_TOOLNT, $P_TOOLT.

System variable $P_MTOOLNT returns the number of tools in the multitool.

System variable $P_MTOOLT returns the T number of the i-th tool of the multitool.

Syntax

```
result = $P_MTOOLNT[mt]
result = $P_MTOOLT[mt, i]
```

Meaning

$P_MTOOLNT	Returns the number of tools in the multitool	
	Data type:	INT
mt	Multitool number	
	Data type:	INT
Return value		
result	Result value	
	Data type:	INT
	> 0	Number of tools in the multitool
	0	No tool contained in the multitool with the number mt
	-1	TMMG function not active
	-2	Multitool function not active
	-3	mt is not the number of a defined multitool

$P_MTOOLT[mt, i]	Returns the T number of the i-th tool	
	Data type:	INT
mt	Multitool number	
	Data type:	INT
i	Index of multitools with 0 < i < $P_MTOOLNT	
	Data type:	INT
Return value		
result	Result value	
	Data type:	INT
	> 0	Number of the i-th tool in the multitool
	0	i is a value outside of the permitted range
	-1	TMMG function not active
	-2	Multitool function not active
	-3	mt is not the number of a defined multitool

Example

Multitool with number 500 has 6 locations. 3 tools with the T numbers 11, 22, and 33 are loaded to the locations 1, 2, and 5.

```
Program code                    Comment
def int number = 0
def int i = 0
number = $P_MTOOLNT[500]        ;the number of tools is 3
for i = 1 to number             ;Loop over 3 tools and not over 6 MT locations
r[i] = $P_MTOOLT [500, i]       ;Write the T numbers of the loaded tools to succes-
                                sive R parameters
                                ;r1=11, r1=22, r3=33
endfor
```

11.10.41 $A_TOOLMLN - Read magazine location no. from tool

Remark

TOOLMLN stands for = "**toolm**agazine **l**ocation **n**umber".

Name	$A_TOOLMLN[t]	
Meaning	Returns the magazine location number of the tool with T no.=t. If not assigned to a magazine, 0 is returned. If the tool management function is not active, -1 is returned. If there is no tool with T no.=t, -2 is returned. An alarm is issued if the value range for the T number was violated.	
Data type	INT	
Range of values	1 ... 32000	
Indices	Meaning	Range of values

Name	$A_TOOLMLN[t]			
	The index specifies the T number			1 ... 32000
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	x	-	x	-
Implicit preprocessing stop	x	-		

Remark: It is not possible that the following applies $A_TOOLMLN[t]==0 and $A_TOOLMN[t]>0, or $A_TOOLMLN[t]>0 and $A_TOOLMN[t]==0.

11.10.42 $P_TOOLND - Read number of cutting edges for tool

Remark

TOOLND stands for = "**tool n**umber of **D**s".

Name	$P_TOOLND[t]			
Meaning	Returns the number of cutting edges for tool with T no.=t. A tool always has at least one cutting edge. Default: If there is no tool with T no.=t, -1 is returned. The value 0 is rejected as index error.			
Data type	INT			
Range of values	Default: 1 ... 9 1 - "Machine data value for the maximum number of D numbers"			
Indices	Meaning			Range of values
	The index specifies the T number			1 ... 32000
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	x	-	-	-
Implicit preprocessing stop	-	-		

11.10.43 $A_MONIFACT - factor for tool life monitoring

Overview

If different tool materials are to be machined with the same tool, it may be necessary to increase or reduce the time intervals for monitoring in order to detect the varying degrees of tool wear. The factor is set accordingly before the tool is used. The write operation is performed synchronously with the main run.

A channel-specific parameter, used to multiply the actual time measurement, has been defined.

Setting a value = 0 deactivates the time monitoring function for all tools used on the channel via the part program.

Name	**$A_MONIFACT**			
Meaning	Only relevant when time monitoring is active in the tool management. Factor for influencing the time measurement for tracking time for time-monitored tools. Values < 1 and > 0 slow down time measurement (the clock "runs slower"). Values > 1 speed up the time measurement (clock "runs faster"). Value 1 is active after the control has been powered up, after Reset and M30 (default) and corresponds to real time. Value 0 is also permitted and disables time measurement of all time-monitored tools that are operated on a time-monitored spindle on this channel. Comment: You can get the monitoring time to "run backwards" by using negative values.			
Data type	REAL			
Value range	Value range of type REAL			
Indices	Meaning			Value range
				-
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronized action
	x	x	x	x
Implicit preprocessing stop	-	x		

Tool life counter on monitor

If system variable $A_MONIFACT is set accordingly, the tool life counter on the monitor can run at a speed other than real time. The time values of OPI module TS and the internal NC time values remain the same. Both are realtime values.

Read and write OPI

The time values are transferred without modification irrespective of the current value of $A_MONIFACT.

Note

Tool life value on HMI

The real-time values are displayed on the HMI, not the converted tool life values.

Example

The current values are specified (units in real time).

Planned tool life: 10 minutes

Actual tool life: 2 minutes - in **one minute**, the pre-warning limit is reached

Prewarning limit: 1 minute

The values 10, 2, 1 are displayed on the screen.

$A_MONIFACT = 2 is programmed in the part program (clock runs faster). The planned tool life, the actual tool life, and the prewarning limit displayed on the monitor are still shown in real time.

Planned tool life: 10 minutes

Actual tool life: 2 minutes - but the prewarning limit is reached in **half a minute** because the clock is now running twice as fast.

Prewarning limit: 1 minute

11.10.44 $AC_MONMIN - Factor for tool search

General

The following is defined by the variable $AC_MOMIN:

Only consider those tools whose actual value is at least a factor $AC_MONMIN (0, ... 1) of the setpoint away from the limiting value.

Definition of lowest/highest actual value

Absolute lowest/highest **actual values** are, in accordance with the tool-search strategy "Search for the tool with the lowest/highest actual value" used for the tool search exactly then when **all tools of a tool group have the same monitoring type defined** (via $TC_TP9).

This means all tools of the tool group are either time-monitored or count-monitored, or are wear or alternatively additive offset monitored.

Relative lowest/highest **actual values** are, in accordance with the tool-search strategies "Search for the tool with the lowest/highest actual value" used for the tool search exactly then when **the tools of a tool group have different monitoring types defined** in $TC_TP9.

This means one tool can be time-monitored, the other tool can be count-monitored. A third tool could be both wear as well as time-monitored.

Lowest/highest actual value for exactly one monitoring type

This is the standard application.

Each lowest/highest actual value here of the monitored variable ($TC_MOP2, $TC_MOP4, $TC_MOP6 for time, count, wear or additive offset) corresponds to lowest/highest actual value of the tools in the tool group.

Example:

A tool group "Tool1" is defined. For example, $TC_MAMP2 = "H108" applies - lowest actual value:

Duplo no. $TC_TP1	Actual value $TC_MOP2	Setpoint $TC_MOP11	Absolute Lowest actual value = $TC_MOP2
1	9	10	9

2	8	10	8
3	6	6	6 lowest actual value in the tool group

Therefore the order of tools for use is: Duplo no.= 3 → 2 → 1.

Lowest/highest actual value with several parallel monitoring types

Tools in a tool group can be monitored in different ways.

Or different types of tool monitoring can be defined for a tool. These situations are detected by the NC and handled accordingly:

The definition of the lowest/highest actual value is determined for these cases by the quotient of dividing actual value and setpoint; i.e.

quotient (Q)= actual value/setpoint value

The tool with the lowest quotient has the **lowest actual value** of the tools in the tool group.

The tool with the highest quotient has the **highest actual value** of the tools in the tool group.

Example 1:

Tool group "millers" has two tools with T Nos. =1 and 2 each with a cutting edge D1.

Time monitoring is active for T1; $TC_TP9[1] = 1.

Workpiece count monitoring is active for T2; $TC_TP9[2] = 2.

Q(T1) = $TC_MOP2[1,1]/$TC_MOP11[1,1] is = 0.5

Q(T2) = $TC_MOP4[2,1]/$TC_MOP13[2,1] is = 0.9

Therefore, T1 has the lower actual value.

Example 2:

A tool group "Tool1" is defined. For example, $TC_MAMP2="H108" applies - lowest actual value:

Duplo no. $TC_TP1	Actual value $TC_MOP2	Setpoint $TC_MOP11	Absolute Lowest actual value = $TC_MOP2
1	9	10	0.9
2	8	10	0.8 lowest actual value
3	6	6	1

Therefore the order of tools for use is: Duplo no.=2 → 1 → 3.

$AC_MONMIN

The above definition of the actual value applies for the actual value that is checked against the setpoint where a factor of $AC_MONMIN has been applied.

The following check is made for the absolute actual-value comparison (time monitoring taken here as the example):

$TC_MOP2 = $AC_MONMIN * $TC_MOP11.

This is the criterion for the usability of the tool.

The following check is made for the relative actual value comparison (time monitoring taken here as the example):

$TC_MOP2 / $TC_MOP11 = $AC_MONMIN

This is the criterion for the usability of the tool.

The result is the same in each case.

Note

The lowest of the actual values (both absolute as well as relative) of the cutting edges of a tool is used for the comparison with the actual values of other tools.

Name	**$AC_MONMIN**			
Meaning	Only when tool management is active Gives the factor for the tool-search strategy "Only consider those tools whose actual value is at least a factor $AC_MONMIN* of the setpoint away from the limiting value". The programmed value is ignored if the tool status "disabled" shall be ignored during the tool search. This can be initiated either by the command TCA, PLC signal or machine data for start/reset. See also the system variables $TC_MOPx, $TC_MAMP2.			
Data type	REAL			
Range of values	0 ... 1			
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	X	X	X
Implicit preprocessing stop	-	X	-	

Constraints

If different types of tool monitoring are selected for the tools of a tool group, then the decision has to be made whether it is meaningful for the specific application to use the tool-search strategy "Search tool with lowest or highest actual value" in this tool group.

Similar conditions apply when working with multiple-edge tools. Also, it is necessary to consider whether it is meaningful to apply the tool search strategy "Search tool with lowest or highest actual value" in this tool group.

Note

As for the other tool search strategies, that tool is preferred for use that is on the spindle or in one of the assigned buffer at the time of the tool search; i.e. the tool search strategy is not applied.

The PLC signal "Do not disable tool" renders the tool search strategy ineffective in accordance with $AC_MONMIN.

Activation

The following must apply so that the tool-monitoring-specific tool search strategies can be effective:

- The sub-function "Tool-monitoring function" must be active within the tool management function
- The appropriate monitoring parameter values ($TC_MOP1, ...) must have been set for the cutting edges of the tools
- The monitoring must be activated for the appropriate tool (system variable $TC_TP9)
- $AC_MONMIN can be programmed in the part program as well. The programmed value is only meaningful if points 1, 2, and 3 are met.

11.10.45 $P_TOOLNG - Number of tool groups

This function is only available for TMMO and TMMG.

Name	**$P_TOOLNG**			
Meaning	Number of defined tool groups that are assigned to the channel. > 0: Successful read access 0: No tool group defined (tool group is defined by writing the tool name) -1: Neither function TMMG nor TMMO active			
Data type	INT			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

11.10.46 $A_MYMN/$A_MYMLN - Owner magazine/location of the tool

This function is available for TMMG.

System variables $A_TOOLMN and $A_TOOLMLN contain definitions which magazine/ magazine location the specified tool is currently located at. This can be a real or an internal magazine.

The system variables $A_MYMN and $A_MYMLN indicate the magazine/magazine location (real magazine only), at which the specified tool was loaded or from which a tool contained in an internal magazine was loaded.

Name	**$A_MYMN[t] / $A_MYMLN[t]**			
Meaning	Use: $A_MYMN[t] - number of the owner magazine of the tool with the T no. = t > 0: Tool is loaded 0: Tool is not loaded -1: Function TMMG is not active -2: Tool with the T no. = t does not exist - not for t = 0 either $A_MYMLN[t] - number of the owner-magazine location of the tool with the T no. = t > 0: Tool is loaded 0: Tool is not loaded -1: Function TMMG is not active -2: Tool with the T no. = t does not exist - not for t = 0 either			
Data type	INT			
Range of values	1... 32000			
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	X	-
Implicit preprocessing stop	-	-	-	

1. The following applies for tools that are not loaded:
   $A_MYMN = $A_MYMLN = $A_TOOLMN = $A_TOOLMLN = 0
2. The following applies for manual tools that were loaded or for tools newly loaded on tool holders:
   $A_MYMN = $A_MYMLN = 0
   $A_TOOLMN 1 = 0, $A_TOOLMLN 1 = 0
3. The following applies for tools that have been loaded but are not in an internal magazine:
   $A_MYMN = $A_TOOLMN > 0
   $A_MYMLN = $A_TOOLMLN >0
   For fixed-location-coded tools in buffers, the two parameters indicate where the respective tool shall be brought when returning to the magazine.

11.10.47 $A_MYMTN/$A_MYMTLN - $A_TOOLMTN/$A_TOOLMTLN - tools in the multitool

The system functions $A_MYMTN/$A_MYMTLN or $A_TOOLMTN/$A_TOOLMTLN (with same content) specify in which multitool and location within the multitool the tool is located.

The following applies:

$A_MYMTN = $A_TOOLMTN

and

$A_MYMTLN = $A_TOOLMTLN.

This expresses that the tool remains permanently in the MT.

Syntax

```
result = $A_MYMTN [t]
result = $A_TOOLMTN [t]
result = $A_MYMTLN [t]
result = $A_TOOLMTLN [t]
```

Meaning

$A_MYMTN $A_TOOLMTN	Returns the multitool number of the owner multitool of the tool with T number t.	
	Data type:	INT
t	T number of the tool	
	Data type:	INT
	Tool Tno. t = 1, ..., 32000 MYMultiToolNumber	
	Tool Tno. t = 1, ... 32000 TOOLisOnMultiToolNumber	
Return value		
result	Result value	
	Data type:	INT
	>0	The tool with the T number t is on the multitool with the number result
	0	The tool with Tno.= t is not in a multitool
	-1	TMMG function not active
	-2	Multitool function not active
	-3	Is not a tool T no. (outside of range, not defined, or it is an MT no. or a magazine no.)
$A_MYMTLN $A_TOOLMTLN	Multitool number of tool t.	
	Data type	INT
t	T number of the tool	
	Data type:	INT
	Tool Tno. t = 1, ... 32000 MYMultiToolLocationNumber	
	Tool Tno. t = 1, ... 32000 TOOLisOnMultiToolLocationNumber	
Return value		

result	Result value	
	>0	The tool with the T number t is on the location with the number result within a multitool
	0	The tool with T number = t is not in a multitool
	-1	TMMG function not active
	-2	Multitool function not active
	-3	Is not a tool T no. (outside of range, not defined, or it is an MT no. or a magazine no.)

11.10.48 $P_TOOLNT/$P_TOOLT - T numbers

This function is available for TMMG and TMMO.

Name	**$P_TOOLNT / $P_TOOLT[i]**			
Meaning	These system variables enables an overview of the tools defined in NC. $P_TOOLNT Number of defined tools that are assigned to the channel. > 0:Successful read access 0: No tool defined $P_TOOLT[i] i-th tool number T > 0: T number 0: i is a value outside of the permitted range			
Data type	INT			
Range of values	1... 32000			
Indices	Meaning			Range of values
	N = number of tools that are assigned to the channel i = i-th T no.; i=1, ..., $P_TOOLNT			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

The system variable $P_TOOLT returns value 1 for index i = 1 if at least one D offset has been defined and returns value 0 for other values of i.

11.10.49 $P_TOOLD - D numbers

This function is available for TMMG and TMMO.

Name	$P_TOOLND/$P_TOOLD[t,i]			
Meaning	Determine the defined D numbers of a tool. The command can generally be programmed. i-th number of tool compensations D of the tool with the T no. = t > 0: D number 0: i is a value outside of the permitted range -2: t is the value of a non-defined tool			
Data type	INT			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
	t = T number i = i-th T no.; i=1, ... $P_TOOLND			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

11.10.50 $P_TOOLNDL - Number of defined DL offsets

This function is available for TMMG and TMMO. The function "additive offset" must have been activated via MD.

Name	$P_TOOLNDL[t,d]			
Meaning	Determine the number of defined DL numbers of a D offset. The command can generally be programmed. Number of DL offsets for D offset provided by T no. = t, D no. = d > 0: Number of DL offsets 0: No DL offsets for this D offset -1: Additive offset function not active -2: t is the value of a non-defined tool -3: d is the value of a non-defined D offset			
Data type	INT			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
	t = T number d = D number			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action

Name	$P_TOOLNDL[t,d]			
	X	-	-	-
Implicit pre-processing stop	-	-	-	

11.10.51 $A_USEDND - Workpiece counting

This function is available for TMMO.

Name	$A_USEDND[s]			
Meaning	Number of different cutting edges used on tool holder s since the last workpiece count including the currently used active cutting edge on s. Each tool used is included at least once. Index s means: *TMMG + TMMO* Spindle number / tool holder number s = 0 means that the currently active main tool holder is selected. *TMMO active without TMMG* a) $MC_T_M_ADDRESS_EXT_IS_SPINO = FALSE: s is not evaluated. It is not possible to count the workpieces separately according to tool holders. b) $MC_T_M_ADDRESS_EXT_IS_SPINO = TRUE: Spindle number. s = 0 means that the currently active leading spindle is selected. > 0: Number of cutting edges used 0: No more tools used since the last workpiece count -1: TMMO is not active -2: s is the value of a non-defined tool holder			
Data type	INT			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
	s = 1, ... MAXNUM_AXES_PER_CHAN s = 0 indicates the leading tool holder			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Example, see "$A_USEDT - Workpiece counting (Page 252)".

11.10.52 $A_USEDT - Workpiece counting

Overview

This function is available for TMMO.

Name	$A_USEDT[i,s]			
Meaning	T number of the tool of the i-th cutting edge that has been used on tool holder s since the last workpiece count or is still being used. Example: For i=$A_USEDND, the T number of the first cutting edge or D offset is obtained, which was selected on the programmed tool holder s since the last workpiece count. Index s means: *TMMG + TMMO* Spindle number / tool holder number s = 0 means that the currently active main tool holder is selected. *TMMO active without TMMG* a) $MC_T_M_ADDRESS_EXT_IS_SPINO = FALSE: s is not evaluated. It is not possible to count the workpieces separately according to tool holders. b) $MC_T_M_ADDRESS_EXT_IS_SPINO = TRUE: Spindle number. s = 0 means that the currently active leading spindle is selected. >0: T number (can also exist multiple times if different D offsets of the tool were in use) 0: No more cutting edges used since the last workpiece count -1: TMMO is not active -2: s is the value of a non-defined tool holder			
Data type	INT			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
	S = 1, ... MAXNUM_AXES_PER_CHAN I = 1, ... $A_USEDND			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Example

Two tool holders are defined with numbers 1 and 2. Tool holder no. 1 is the leading tool holder. On tool holder 1, previously three tools with T numbers 10, 20, 30 were used; on tool holder 2, one tool was used with T number 666. Each tool only has offset D1 defined.

The following program section is run in the status:

```
def int n1, n2, i, tNo
n1 = $A_USEDND[1]          ;n1 = 3 same content would have been
                           $A_USEDND[0]
n2 = $A_USEDND[2]          ;n2 = 1
```

```
for i = 1 to n1
tNo = $A_USEDT[1,i]
MSG ("T no. participating in workpiece machining =" << tNo
endfor
                          ;The loop displays T numbers 10, 20, 30
T2=0                      ;Bits 7, 8, 19 are set for synchronization
                          in $MC_TOOL_MANAGEMENT_MASK. (Automatic
                          read-in disable until tool change is
                          acknowledged with "End".)
setpiece(5,2)
if (n2 == 1) tNo = $A_USEDT[1,1]
                          ;Sets tNo to value 0. Setpiece was
                          programmed since determination of n2. This
                          deletes the list of tools used and there is
                          currently no entry for the specified
                          Index1 in the list of tools used.
```

11.10.53 $A_USEDD - Workpiece count

This function is available for TMMO.

Name	**$A_USEDD[i,s]**	
Meaning	D number of the i-th cutting edge that has been used on the tool holder s since the last workpiece count or is still being used. Example: For i=$A_USEDND, the D number is obtained for the first cutting edge or D offset that was selected after the last workpiece count on the programmed tool holder. Index s means: *TMMG + TMMO* Spindle number / tool holder number s = 0 means that the currently active main tool holder is selected. *TMMO active without TMMG* a) $MC_T_M_ADDRESS_EXT_IS_SPINO = FALSE: s is not evaluated. It is not possible to count the workpieces separately according to tool holders. b) $MC_T_M_ADDRESS_EXT_IS_SPINO = TRUE: Spindle number. s = 0 means that the currently active leading spindle is selected. >0: D number 0: No more tools used since the last workpiece count -1: TMMO is not active -2: s is the value of a non-defined tool holder	
Data type	INT	
Range of values	1 ... 32000	
Indices	Meaning	Range of values
	S = 1, ... MAXNUM_AXES_PER_CHAN I = 1, ... $A_USEDND	

Name	$A_USEDD[i,s]			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

11.10.54 $P_MAGN/$P_MAG - magazine

This function is available for TMMG.

Name	$P_MAGN / $P_MAG[i]			
Meaning	$P_MAGN Number of defined magazines that are assigned to the channel. > 0: Successful read access 0: No magazine defined -1:TMMG is not active $P_MAG i-th magazine number > 0: Successful read access 0: i is outside of the permitted range -1: TMMG is not active			
Data type	INT			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
	i = 1, ... $P_MAGN			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Example, see "Example of magazine configuration system variables (Page 258)".

11.10.55 $P_MAGNDIS/$P_MAGDISS/$P_MAGDISL - magazine distance tables

This function is available for TMMG.

Name	**$P_MAGNDIS[n,m] / $P_MAGDISS[l,i] / $P_MAGDISL[l,i]**			
Meaning	$P_MAGNDIS**[n,m]** Number of magazines that are linked to the internal magazine n by the location m > 0: Successful read access 0: No magazine linked with the buffer -1: TMMG is not active -2: n is not the number of an internal magazine -3: m is not the number of an internal magazine location $P_MAGDISS[l,i] Number of the i-th magazine that is linked with location l of the buffer magazine. > 0: Successful read access 0: i is outside of the permitted range -1: TMMG is not active -2: m is not the number of a buffer location -3: No buffer location defined $P_MAGDISL[l,i] Number of the i-th magazine that is linked with location l of the loading magazine. > 0: Successful read access 0: i is outside of the permitted range -1: TMMG is not active -2: m is not the number of a loading magazine location -3: No buffer location defined			
Data type	INT			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
	i = 1, ... $P_MAGNDIS			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Example, see "Example of magazine configuration system variables (Page 258)".

11.10.56 $P_MAGNS/$P_MAGS - tool holder

This function is available for TMMG.

Name	$P_MAGNS/$P_MAGS[n]			
Meaning	$P_MAGNS Number of spindle locations/tool holder locations in the buffer assigned to the channel. > 0: Successful read access 0: no spindle location defined -1: TMMG is not active -2: No buffer magazine defined $P_MAGS[n] n-th number of the spindle/tool holder in the buffer > 0: Successful read access 0: n is outside of the permitted range -1: TMMG is not active -3: No buffer magazine defined			
Data type	INT			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
	n = 1, ... max. tool holder number			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Example, see "Example of magazine configuration system variables (Page 258)".

11.10.57 $P_MAGNREL/$P_MAGREL - Assigned buffer

This function is available for TMMG.

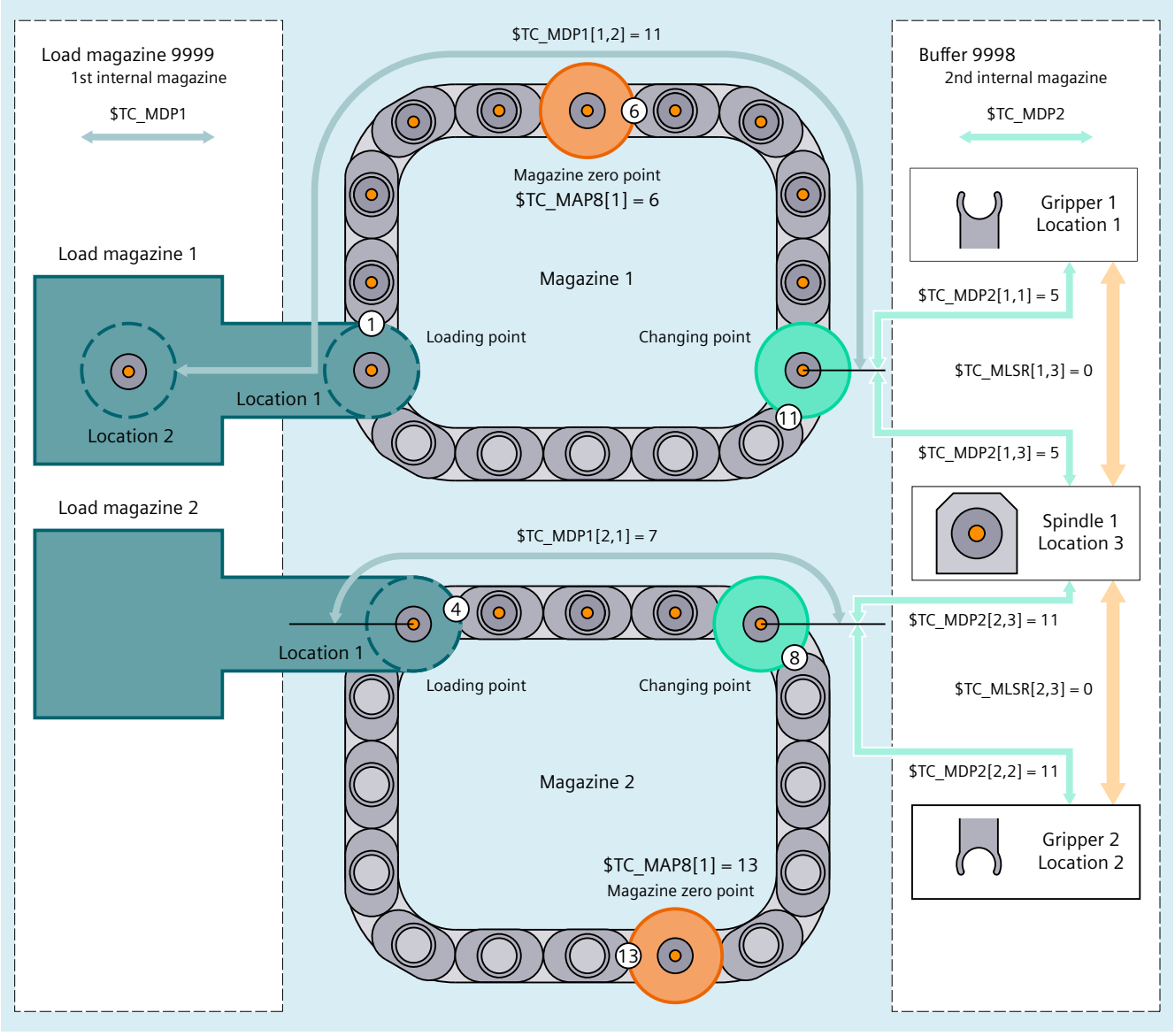
Name	**$P_MAGNREL[n]/$P_MAGREL[n,m]**			
Meaning	$P_MAGNREL[n] Number of buffers assigned to the spindle no. / tool holder no. > 0: Successful read access 0: Spindle location has not been assigned buffer location -1: TMMG is not active -2: n is not the number of a spindle location -3: No buffer magazine defined $P_MAGREL[n,m] m-th buffer number of the n-th spindle number/tool holder number > 0: Successful read access 0: m is outside of the permitted range -1: TMMG is not active -2: n is not the number of a spindle location -3: No buffer magazine defined			
Data type	INT			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
	m = 1, ... $P_MAGNREL n = 1, ... max. tool holder number			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Example, see "Example of magazine configuration system variables (Page 258)".

11.10.58 Example of magazine configuration system variables

Specifications

The magazine configuration selected in the following example is selected. Information about the current magazine configuration can be obtained by reading the system variables described here.



$TC_MDP1[n,m]	Distance between tool change position of magazine n and location m of the 1st internal magazine
$TC_MDP2[n,m]	Distance between tool change position of magazine n and location m of the 2nd internal magazine
$TC_MLSR[n,m]	Assignment of buffer location n to buffer location m
$TC_MAP8[n]	Current magazine position in relation to tool change position

Figure 11-13 Magazine configuration

```
N10 def int noOfMag=0, noOfLoc=0, noOfDist=0, noOfRel=0,
```

```
noOfSpindles=0, spindeleNo=0
N20 def int i=0
; Total number of defined magazines
N100 noOfMag = $P_MAGN      ;noOfMag is assigned value =4 - 2 real magazines
                            ;1, 2+2 internal magazines 9998, 9999
; Display all magazine numbers
N200 for i=1 to noOfMag
N220 MDG ("magazine no.="<<$P_MAG[i])
                            ;Display numbers 1, 2, 9998, 9999
N240 endfor
; Total number of defined magazine locations
N300 for i=1 to noOfMag
N320 noOfLoc=noOfLoc + $TC_MAP7[$P_MAG[i]]
N340 endfor                 ;noOfLoc now contains value 16+16+3+2=37
; Number of magazines linked with Spindle 1
N400 noOfDist=$P_MAGNDIS[9998,3]
                            ;; noOfDist is assigned value=2 - Mag.1, 2 are linked with
                               the spindle location
; Display the magazine numbers linked with Spindle 1 (=location 3)
N500 for i=1 to noOfDist
N520 MSG ("magazine no.=" << $P_MAGDISS[ i ] )
                            ;Display numbers 1, 2
N540 endfor
; Number of magazines linked with Loading station 2
N410 noOfDist = $P_MAGNDIS[9999,1]
                            ;noOfDist is assigned value=1 - Mag. 2 is linked with load-
                            ing station 2
; Display the magazine numbers linked with loading station 2 (=location 1)
N510 for i=1 to noOfDist
N530 MSG ("magazine no.="<< $P_MAGDISL[i] )
                            ;Display number 2
N550 endfor
; Total number of defined spindles
N600                        ;noOfSpindles contains value = 1
noOfSpindles=$P_MAGNS       ;- one spindle location is defined
```

; Display the numbers of the spindles defined in the magazine configuration

```
N620 for i=1 to noOfSpindles
N640 MSG ("magazine no.="<< $P_MAGS[i])
                              ;Display number 1
N660 endfor
```

: Total number of buffer locations assigned to spindle 1

(=gripper in example)

```
N700 noOfRel=$P_MAGNREL[1]
                              ;noOfRel contains value=2, grippers 1 and 2 are assigned
                              to the spindle
```

; Display the numbers of the grippers of spindle no. 1 defined in the magazine configuration

```
N720 for i=1 to noOfRel
N740 MSG ("magazine no.="<< $P_MAGREL[1,i] )
                              ;Display numbers 1, 2
N760 endfor
```

11.10.59 $P_MAGNH/$P_MAGNHLT/$P_MAGHLT - Location type hierarchies

Overview

This function is available for TMMG.

The system variables return the following information:

- Number of hierarchies
- Number of magazine location types each hierarchy.

Name	$P_MAGNH / $P_MAGNHLT[n] / $P_MAGHLTn,m]			
Meaning	$P_MAGNH Number of defined magazine location type hierarchies that are assigned to the channel. > 0; Successful read access 0; No location type hierarchies are defined. -1; TMMG is not active $P_MAGNHLT[n] Number of the defined location types in the n-the defined hierarchy > 0; Successful read access 0; Hierarchy "n" is outside of the defined range. -1; TMMG is not active. -2; n is outside of the defined range. $P_MAGHLTn,m] m-th location type of hierarchy n > 0; Successful read access -1; TMMG is not active. -2; Hierarchy n does not have any defined location types. -3; m is outside of the defined range.			
Data type	INT			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
	n = 1, ... $P_MAGNH m = 1, ... $P_MAGNHLT			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Example

The following three hierarchies are defined:

Hierarchy 1:

```
$TC_MPTH[0,0] = 12
$TC_MPTH[0,1] = 88
$TC_MPTH[0,2] = 9999
$TC_MPTH[0,3] = 9999
$TC_MPTH[0,4] = 9999
```

Hierarchy 2:

```
$TC_MPTH[1,0] = 55
$TC_MPTH[1,1] = 808
```

```
$TC_MPTH[1,2] = 45
$TC_MPTH[1,3] = 9999
$TC_MPTH[1,4] = 9999
```

Hierarchy 3:

```
$TC_MPTH[2,0] = 87
$TC_MPTH[2,1] = 21
$TC_MPTH[2,2] = 3
$TC_MPTH[2,3] = 9999
$TC_MPTH[2,4] = 62
```

Hierarchy 4:

```
$TC_MPTH[3,0] = 9999
$TC_MPTH[3,1] = 9999
$TC_MPTH[3,2] = 9999
$TC_MPTH[3,3] = 9999
$TC_MPTH[3,4] = 9999
```

Hierarchy 5:

```
$TC_MPTH[4,0] = 99
$TC_MPTH[4,1] = 124
$TC_MPTH[4,2] = 32
$TC_MPTH[4,3] = 7
$TC_MPTH[4,4] = 9999
```

The following example returns *the total number of hierarchies* that are defined and *the number of magazine location types in each hierarchy*.

```
N10 def int noOfH, noOfTypes[8], locTypeNo
N100 noOfH=$P_MAGNH                ;noOfH is assigned value = 5
N220 for i=1 to noOfH
N240 noOfTypes[i - 1]=$P_MAGNHLT[ i ]
                                   ;sets the values 2, 3, 5, 0, 4 in
                                   the array
N260 endfor
```

The following example returns *which magazine location types* are defined in the 2nd hierarchy.

```
N310 for i=1 to noOfTypes[1]
N340 MSG ("magazine no.="<<$P_MAGHLT[2, i])
                                   ;shows values 55, 808, 45
N330 endfor
```

The following example returns which magazine location types are defined in the 3rd hierarchy.

```
N350 for i=1 to
noOfTypes[2]
N360 MSG ("magazine no.="<<$P_MAGHLT[3, i])
                          ;shows the values 87, 21, 3, 9999, 62
N370 endfor
```

The following example returns which magazine location types are defined in the 5th hierarchy.

```
N400 for i=1 to noOfTypes[4]
N410 MSG ("magazine no.="<<$P_MAGHLT[5, i])
                                        ;shows values 99, 124, 32, 7
N420 endfor
```

The following example returns the location type of an undefined hierarchy level (beyond the end of hierarchy 2).

```
N500 MSG ( " Magazine no. = " << $P_MAGHLT[ 2, 4 ] )
                             ; shows the value –3
```

11.10.60 $P_MAGNA/$P_MAGA - tool adapter

This function is available for TMMG.

Name	**$P_MAGNA / $P_MAGA[i]**			
Meaning	$P_MAGNA Number of defined adapters that are assigned to the channel. > 0 Successful read access 0 No adapter defined -1 Function "Adapter" or TMMG is not active $P_MAGA[i] i-th adapter number > 0 Successful read access 0 n is outside of the defined range -1 Function "Adapter" or TMMG is not active			
Data type	INT			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
	i = 1, ... $P_MAGNA			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action

Name	$P_MAGNA / $P_MAGA[i]			
	X	-	-	-
Implicit prepro-cessing stop	-	-	-	

$P_MAGNA1, $P_MAGNA2

Name	$P_MAGNA1			
Meaning	$P_MAGNA1 Number of defined adapters of type 1 in the TO unit of the channel. • >0 Successful read access • 0 No adapters of type 1 defined • -1 Function 'Adapter' or TMMG is not active OPI block= T/TMV			
Data type	INT			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
	i = 1, ... $P_MAGNA1			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_MAGNA 2			
Meaning	$P_MAGNA2 Number of defined adapters of type 2 in the TO unit of the channel. • >0 Successful read access • 0 No adapters of type 2 defined • -1 Function 'Adapter' or TMMG is not active OPI block= T/TMV			
Data type	INT			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
	i = 1, ... $P_MAGNA2			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

11.10.61 Additional language commands

Name	$P_TOOLNO			
Meaning	Active tool number T0 to T32000. Generally, the command should not be used when magazine management is active. If magazine management is active, GETTEXET (only with 1 "T") should be used instead. For $MC_CUTTING_EDGE_DEFAULT = 0, the incorrect T number can be determined. If the programming was implemented after programming D > 0, then it is also reliable			
Data type	Integer			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_TOOLP			
Meaning	Tool number last programmed Command is available for TMMO. It is analogous to the TMMG-specific command GETSELT.			
Data type	Integer			
Range of values	1 ... 32000			
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_TOOL			
Meaning	Active tool cutting edge (Dx)			
Data type	Integer			
Range of values				
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_DLNO			
Meaning	Active additive offset number DL=0-DL=max; max=value of $MN_MM_MAX_SUMCORR_PER_CUTTEDGE $P_DLNO is analogous to the already existing parameters $P_TOOL, $P_TOOLNP and active D and T numbers.			
Data type	Integer			
Range of values	0 ... 6			
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-		

Name	$P_TOOLL[n]			
Meaning	Active tool total length; n = 1 ... 3 `$P_TOOLL[1]` to `$P_TOOLL[3]` correspond to the values in `$TC_DP3[]` to `$TC_DP3[]`, including the active tool offsets irrespective of the plane selection and irrespective of setting data such as `$SC_TOOL_LENGTH_CONST` etc.			
Data type	REAL			
Range of values				
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_TOOLR			
Meaning	Active radius			
Data type	REAL			
Range of values				
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_TC		
Meaning	Active tool holder		
Data type	Integer		
Range of values			

Name	**$P_TC**			
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	**$P_TCANG[n]**			
Meaning	Active angle of a toolholder axis; n = 1 ... 2			
Data type	REAL			
Range of values				
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	**$P_TCDIFF[n]**			
Meaning	Difference between calculated and used angle of a toolholder axis for grid (Hirth joint) of the angle			
Data type	REAL			
Range of values				
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	**$P_AD[n]**			
Meaning	Active tool offset; n = 1 ... 40 n = 1 ... 25 $TC_DP1 ... $TC_DP25 n = 26 $TC_DPCE (optional) n = 27 $TC_DPH (optional) n = 28 $TC_DPV (optional) n = 29 $TC_DPV3 (optional) n = 30 $TC_DPV4 (optional) n = 31 $TC_DPV5 (optional) n = 32 $TC_DPVN3 (optional) n = 33 $TC_DPVN4 (optional) n = 34 $TC_DPVN5 (optional) n = 35 $TC_DPNT n = 36 $TC_DPROT n = 37 $TC_DP_RADIUS3 n = 38 $TC_DP_WEAR_RADIUS3 n = 39 $TC_DP_LEN6 n = 40 $TC_DP_WEAR_LEN6			
Data type	DOUBLE			
Range of values				
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	X	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_ADT[n]			
Meaning	$P_ADT[n] - transformed data of the active tool When compensation parameters are read, this parameter returns transformed values of the parameters controlled by the tool adapter transformation - if the active tool is attached to an adapter. n = 1 ... 25 $TC_DP1 ... $TC_DP25 n = 26 $TC_DPCE (optional) n = 27 $TC_DPH (optional) n = 28 $TC_DPV (optional) n = 29 $TC_DPV3 (optional) n = 30 $TC_DPV4 (optional) n = 31 $TC_DPV5 (optional) n = 32 $TC_DPVN3 (optional) n = 33 $TC_DPVN4 (optional) n = 34 $TC_DPVN5 (optional) n = 35 $TC_DPNT n = 36 $TC_DPROT n = 37 $TC_DP_RADIUS3 n = 38 $TC_DP_WEAR_RADIUS3 n = 39 $TC_DP_LEN6 n = 40 $TC_DP_WEAR_LEN6			
Data type	DOUBLE			
Range of values				
Indices	Meaning			Range of values
	n: Parameter number 1 ... 40			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	**$P_ATDT[n, dNo]**			
Meaning	$P_ATDT[n, dNo] - active tool data Active tool offsets of the cutting edge dNo. transformed. n: Parameter number 1 ... 40 n = 1...25 $TC_DP1 bis $TC_DP25 n = 26 $TC_DPCE CuttingEdge number of the cutting edge (function: unique D number) n = 27 $TC_DPH H number of the cutting edge (function: ISO mode) n = 28 $TC_DPV Tool orientation (function: tool orientation) n = 29 $TC_DPV3 Component 1 of tool orientation (function: tool orientation) n = 30 $TC_DPV4 Component 2 of tool orientation (function: tool orientation) n = 31 $TC_DPV5 Component 3 of tool orientation (function: tool orientation) n = 32 $TC_DPVN3 Normal vector component 1 (function: tool orientation) n = 33 $TC_DPVN4 Normal vector component 2 (function: tool orientation) n = 34 $TC_DPVN5 Normal vector component 3 (function: tool orientation) n = 35 $TC_DPNT Number of teeth of cutting edge n = 36 $TC_DPROT Base angle of rotation of cutting edge If an offset parameter belongs to a function that is not active, an alarm is output.			
Data type	DOUBLE			
Range of values				
Indices	Meaning			Range of values
	n: Parameter number 1 ... 40			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_ATD[n, dNo]			
Meaning	$P_ATDT[n, dNo] - active tool data Active tool offsets of the cutting edge dNo. n: Parameter number 1 ... 40 n = 1...25 $TC_DP1 bis $TC_DP25 n = 26 $TC_DPCE CuttingEdge number of the cutting edge (function: unique D number) n = 27 $TC_DPH H number of the cutting edge (function: ISO mode) n = 28 $TC_DPV Tool orientation (function: tool orientation) n = 29 $TC_DPV3 Component 1 of tool orientation (function: tool orientation) n = 30 $TC_DPV4 Component 2 of tool orientation (function: tool orientation) n = 31 $TC_DPV5 Component 3 of tool orientation (function: tool orientation) n = 32 $TC_DPVN3 Normal vector component 1 (function: tool orientation) n = 33 $TC_DPVN4 Normal vector component 2 (function: tool orientation) n = 34 $TC_DPVN5 Normal vector component 3 (function: tool orientation) n = 35 $TC_DPNT Number of teeth of cutting edge n = 36 $TC_DPROT Base angle of rotation of cutting edge n = 37 $TC_DP_RADIUS3 radius 3 n = 38 $TC_DP_WEAR_RADIUS3 radius 3 - wear n = 39 $TC_DP_LEN6 length 6 n = 40 $TC_DP_WEAR_LEN6 length 6 - wear If an offset parameter belongs to a function that is not active, an alarm is output.			
Data type	DOUBLE			
Range of values				
Indices	Meaning			Range of values
	n: Parameter number 1 ... 40			
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$AC_MSNUM			
Meaning	Leading spindle, return value 0: No spindle configured 1 ... n: Number of the leading spindle			
Data type	Integer			
Range of values				
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action

Name	$AC_MSNUM			
	X	-	X	-
Implicit preprocessing stop	X	-	-	

Name	$P_MSNUM			
Meaning	Leading spindle 0: No spindle configured 1 ... n: Number of the leading spindle			
Data type	Integer			
Range of values				
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$AC_MTHNUM			
Meaning	Main tool holder Value = 0 no main tool holder defined Value > 0 number of the main tool holder			
Data type	Integer			
Range of values				
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	X	-
Implicit preprocessing stop	X	-	X	-

Name	$P_MTHNUM			
Meaning	Main tool holder Value = 0 no main tool holder defined Value > 0 number of the main tool holder			
Data type	Integer			
Range of values				
Indices	Meaning			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-		

Name	$TC_TP_MAX_VELO			
Meaning	Maximum tool speed			
Data type	REAL			
Range of values	[0, DBL_MAX]			
Indices	Internal T number			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	X	-	-
Implicit preprocessing stop	-	-	-	-

Name	$TC_TP_MAX_ACC			
Meaning	Maximum rotary acceleration of the tool			
Data type	REAL			
Range of values	[0, DBL_MAX]			
Indices	Internal T number			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	X	-	-
Implicit preprocessing stop	-	-	-	-

Name	$TC_MPP_SP			
Meaning	Only relevant, if $MC_TOOLHOLDER_MANAGEMENT>0: Spindle no. which is linked to the tool holder.			
Data type	Integer			
Range of values	[0, MAXNUM_SPIND_PER_CHAN]			
Indices	Magazine no., magazine location no.			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	X	-	-
Implicit preprocessing stop	-	-	-	-

Name	$P_MTHSDC	
Meaning	Leading tool holder number or leading spindle number with regard to which the active tool is determined for the next D offset selection. > 0 Successful read access 0 No leading tool holder or no leading spindle available. The next D offset works with T0. -1 TMMG not available	
Data type	Integer	
Range of values	[-1, MAXNUM_AXES_PER_CHAN]	

Name	$P_MTHSDC			
Indices	Magazine no., magazine location no.			Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	-

Name	$P_TH_OF_D			
Meaning	Tool holder or spindle on which the active tool is located that contains the active D offset. > 0 Successful read access 0 No tool holder or spindle available as reference because, for example, no D offset is active. -1 Function is not available If read as an OPI variable, this applies to the status in the current main run block.			
Data type	Integer			
Range of values	[-1, MAXNUM_AXES_PER_CHAN]			
Indices				Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	-

Name	$P_MTHNUM_BEFORE_SEARCH			
Meaning	Leading tool holder or leading spindle before the block search or test mode was started. > 0: Successful read access 0: No tool holder or spindle available as reference because, for example, no D offset is active. -1: Function is not available If the block search or test mode is completed, this variable contains the same value as $P_MTHNUM, as of the next D programming. If read as an OPI variable, this applies for the status in the current main run block.			
Data type	Integer			
Range of values	[-1, MAXNUM_AXES_PER_CHAN]			
Indices				Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	-

Name	$P_D_BEFORE_SEARCH			
Meaning	The active D offset before the block search or the test mode was started. > 0: Successful read access 0: No tool holder or spindle available as reference because, for example, no D offset was or is active. -1: Function is not available If the block search or test mode is completed, this variable contains the same value as $P_TOOL, as of the next D programming. If read as an OPI variable, this applies for the status in the current main run block.			
Data type	Integer			
Range of values	[-1, 32000]			
Indices				Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	-

Name	$P_DL_BEFORE_SEARCH			
Meaning	The active DL offset before the block search or the test mode was started. > 0: Successful read access 0: No tool holder or spindle available as reference because, for example, no D offset was or is active -1: Function is not available If the block search or test mode is completed, this variable contains the same value as $P_DLNO, as of the next D or DL programming. If read as an OPI variable, this applies for the status in the current main run block.			
Data type	Integer			
Range of values	[-1, 6]			
Indices				Range of values
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	-

11.10.62 Variables for subroutine replacement technique

Tool management NC command	Functions
$C_T	Value of the programmed address T for cycle parameterization and T function replacement
$C_T_PROG	Address T is programmed in a block with cycle call of T function replacement

Tool management NC command	Functions
$C_TS	For T or TCA replacement, returns the programmed tool identifier ($C_TS_PROG == TRUE).
$C_TS_PROG	Supplies TRUE, if a tool identifier has been programmed for the T or TCA replacement.
$C_TE	Address extension for address T for subprogram call per T function
$C_D	Value of the programmed address D in the ISO mode for cycle parameterization
$C_D_PROG	Address is programmed in a block with cycle call
$C_DL	Value of the programmed address DL (additive tool offset) for a subprogram call per M/T function replacement
$C_DL_PROG	Queries whether, for a subprogram call per M/T function replacement, address DL (additive tool offset) was programmed
$C_TCA	Supplies TRUE, if the TCA replacement is active
$C_DUPLO_PROG	Supplies TRUE, if the duplo number for the TCA replacement has been programmed
$C_DUPLO	Supplies the value of the programmed duplo number for TCA replacement ($C_DUPLO_PROG == TRUE)
$C_THNO_PROG	Supplies TRUE, if the tool holder / spindle number for the TCA replacement has been programmed
$C_THNO	Supplies the value of the programmed tool holder / spindle number for TCA replacement ($C_THNO_PROG == TRUE)

11.10.63 Variables for tool change in synchronized action

Tool management NC command	Functions
$AC_TC_FCT	Command number -1: At the instant of reading, no tool management command is active 1: Move (load, unload,...) 2: Prepare change 3: Change ON 4: Change ON (turret, without M06) 5: Prepare change and change ON (with M06)
$AC_TC_STATUS	Acknowledgement status of PLC FC8/FC6
$AC_TC_THNO	Number of the tool holder or the spindle where the new tool shall be loaded
$AC_TC_TNO	The internal T number of the tool to be loaded at change 0: There is no new tool
$AC_TC_MFN	Source magazine number of the new tool 0: There is no new tool
$AC_TC_LFN	Source location number of the new tool 0: There is no new tool

Tool management NC command	Functions
$AC_TC_MTN	Target magazine number of new tool 0: There is no new tool
$AC_TC_LTN	Target location number of the new tool 0: There is no new tool
$AC_TC_MMYN	Owner magazine of new tool 0: There is no new tool
$AC_TC_LMYN	Owner location magazine of new tool 0: There is no new tool
$AC_TC_MFO	Source magazine number of the old tool 0: There is no old tool
$AC_TC_LFO	Source location number of the old tool 0: There is no old tool
$AC_TC_MTO	Target magazine number of old tool 0: There is no old tool
$AC_TC_LTO	Target location number of the old tool 0: There is no old tool
$AC_TC_CMDT	Trigger variable to NC command output Set for one interpolation cycle when NC outputs a new command.
$AC_TC_CMDC	Counter for NC command output This variable is incremented by 1 at each NC command output. Can also be written (zero setting).
$AC_TC_ACKT	Trigger variable to PLC command Set for one interpolation cycle when PLC outputs a command to NC. Command acknowledgement or independent message (asynchronous transfer).
$AC_TC_ACKC	Counter for PLC commands This variable is incremented by 1 at each PLC command. Can also be written (zero setting).

Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	X	-
Implicit preprocessing stop	X	-		

11.11 Definitions when programming data

11.11.1 Tool change

Programming the tool selection

The tool change can be divided into 2 different steps:

1. Tool change preparation
2. Tool change execution

Steps 1-2 can be programmed separately or together in the NC program (see MD22550 TOOL_CHANGE_MODE).

Examples

- Tool change in one step: (turret)
  - Tx; Make new tool x available and execute tool change.
- Tool change in two steps:
  - Tx; Prepare tool change (select the tool)
  - M06; Execute tool change

Note

If tool management is active, a tool can only be selected with the tool identifier (name). If a T number is now programmed, then the number is used as the identifier (name). The tool must then receive the T number as a name during loading.

- Tool change with identifier:
  - T="DRILL"; A search is performed for tool with identifier "DRILL".
- Tool change with number as identifier:
  - T="123"; A search is performed for tool with identifier "123". Alternatively, T123 can also be programmed.

11.11.2 Cutting edge selection

Cutting edge selection after tool change

When a tool change has been completed, the tool cutting edge can be selected in one of the following ways:

1. The offset number D is programmed.
2. The offset number D is not programmed and is specified by MD20270 $MC_CUTTING_EDGE_DEFAULT:
   = 0: No automatic cutting edge selection after M06.
   > 0: Number of the cutting edge, which is selected in accordance with M06.
   = -1 The cutting edge no. of the old tool remains valid and also selected for the new tool after M06.
   = -2 The offset of the old tool remains valid and also selected for the new tool after M06.

Examples:

Tool selection with the following cutting edge selection

Cutting edge selection always refers to the tool that is changed with command M06.

`T1 M06`	Tool change - no D programmed; therefore offset selection according to MD 20270
`T5`	Tool preselection
X .. Y.. Z...	Working with T1 and the offset from MD 20270
`D2`	Offset D2 from T1 !!!
`M06`	Tool change; T5 is loaded at change - offset selection according to MD 20270
`T1`	Tool preselection
`X.., Y...`	Working with T5 and the offset from MD 20270

When programming tool commands, main spindles and secondary spindles are programmed differently. Only tool offset values of the main spindle tool are taken into account by the geometry because only one active offset can be processed per channel. Processing of tool commands for a secondary spindle is only relevant for signal output to the PLC and the function GETSELT(...).

Spindle no. 2 = main spindle:

```
T2 = "DRILL"
M2 = 06
T1 = "MILL"          Select tool for secondary spindle
M1 = 06              Tool change: load tool in the secondary spindle
D1                   Select cutting edge of "DRILL" (main spindle)
```

Spindle no. 2 = main spindle:

```
T2 = "DRILL"         Select a tool for the main spindle.
                     As an alternative, T= "DRILL" could also be specified.
T1 = x;              Select a tool for a secondary spindle
```

`M2 = 06`	Tool change As an alternative, M06 could also be specified.
`D1`	Select cutting edge of a tool with identifier "DRILL"

11.11.3 Tool transfer from program test

MD20110 $MC_RESET_MODE_MASK, **bit 3** can be used to define that the active tool and the tool offset are imported as follows:

- Bit 3 = 1; from the test program which was last terminated in test mode.
  or
- Bit 3 = 0; from the program which was last terminated before the program test was activated.

Precondition: Bits 0 and 6 must be set in MD20110.

$P_ISTEST

The system variable $P_ISTEST is for checking from the part program whether a program test is active. The system variable returns the value TRUE when program testing is active.

11.12 Programming T=location number

The function "Programming T=location number" is only available when tool management is active. This type of programming is not only suitable for turrets, but for all magazine types.

The programming type for tool selection is set using machine data MD20310 $MC_TOOL_MANAGEMENT_MASK:

- MD20310 $MC_TOOL_MANAGEMENT_MASK, bit 16=0 (default setting) - programming via T = "Identifier".
  - T = "x" with x as tool identifier, e.g. "drill"
- MD20310 $MC_TOOL_MANAGEMENT_MASK, bit 16=1 - Programming via T = "Location number".
  - Tx, with x as location number of the magazine containing the tool used for machining, e.g. location number 4

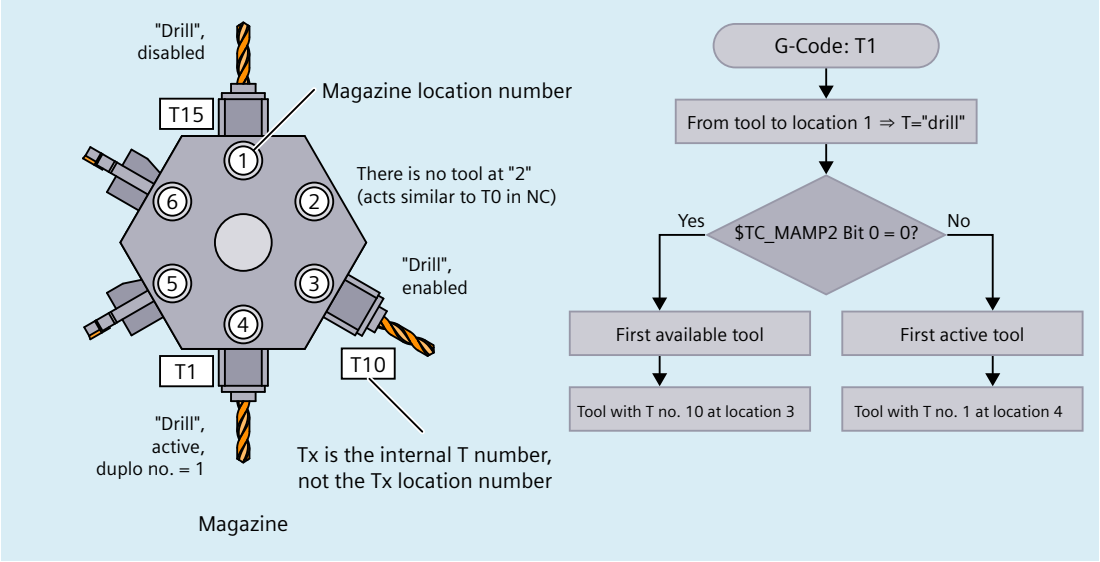


Figure 11-14 Programming of T=location number

Explanation of the illustration:

When the function is active, T1 selects the tool in location number 1 instead of the tool with identifier "1". The first magazine linked with the tool holder is accessed. The identifier of the tool in this location is then determined ("Drill").

The subsequent procedure is as if T="Drill" had been programmed. During the tool change process it is determined which of the three tools from the "Drills" group shall be loaded first.

The set tool search strategy is taken into consideration:

- When the strategy "Take the first available tool from the group" is applied, T10 from location 3 is loaded ($TC_MAMP2 Bit0 = 0).
- When the strategy "Take the first tool with *active* status from the group" is applied, T1 is loaded ($TC_MAMP2 bit0 = 1).
  In so doing, it is assumed that tool T1 is also the replacement tool with the lowest duplo number. If this is not the case, then for $TC_MAMP2 bit0 = 1, for a tool change, the tool with the lowest duplo number is loaded.

The search strategy is set using system variables $TC_MAP10 (current search strategy of the magazine) or $TC_MAMP2 (search strategy type). As preliminary setting, $TC_MAP10 uses the values from $TC_MAMP2.

T15 on location no. 1 cannot be used, because it is disabled.

No alarm is generated if the programmed location does not contain a tool. The change is output as usual, with T no.=0. In this way, for example, the turret can be positioned to an empty location.

If more than one magazine is assigned to the tool holder, then the programmed location number refers to the magazine that is the first magazine defined in the distance table.

If the tools of the tool group are located in different magazines of the tool holder, the search procedure is the same as with the standard tool management.

Note

With the T=location function, alternatively T= "Drill" can also be programmed.

T=1; tool

T="Drill"; tool with identifier Drill

11.13 Call multiple turrets with "T=location number"

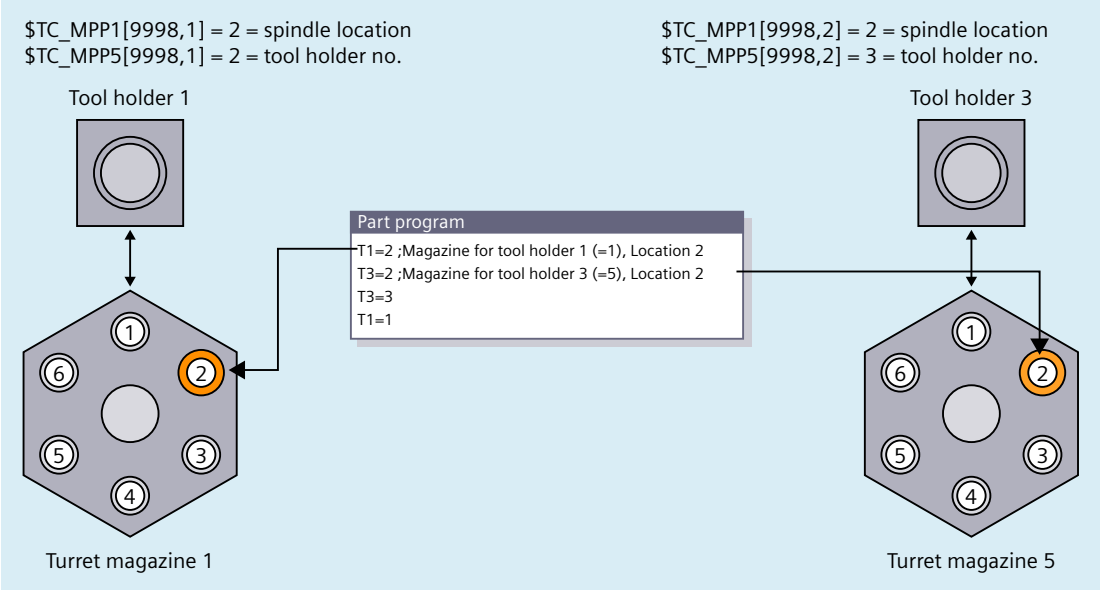


Figure 11-15 T=location number as tool management function on turning machines

The programming option "T = location number" and several magazines can be used to work in one channel or in one TO unit.

- The NC address T can be programmed with an address expansion T1 =...
- Tool management then interprets this as the spindle number or as the toolholder number.
- T without address extension then refers to the main spindle.

11.14 Programming examples

Action	Program command	Description
Create tool	DEF INT DUPLO_NO DEF INT T_NO DUPLO_NO = 7 T_NO=NEWT("DRILL",DUPLO_NO)	Create new tool called Drill with duplo no.= 7. The automatically generated T no. is stored in "T_NO".
	T_NO = GETT("DRILL", DUPLO_NO) or $TC_TP2[1] = "DRILL" ; $TC_TP1[1] = DUPLO_NO	Determine the T number of tool "DRILL" with duplo no. 7 that has already been created. The T No is, however, also assigned here by the programming.
Tool data read / write	$TC_DP1[GETT("DRILL", DUPLO_NO), 2] = 210	Write tool type for the 2nd cutting edge of the tool "Drill"/ DUPLO_NO
	$TC_DP1[T_NO, 2] = 210	Write tool type for the 2nd cutting edge of the tool "T number"
Tool selecting	T="DRILL" or: T=GETT("DRILL", DUPLO_NO) or Tx	If there are several tools with this identifier, then the T no. is returned for the first tool possible. Determines the T number for "DRILL" with duplo number = DUPLO_NO and selects it. Call with T No. e.g. T1,T2,T3,....
Delete tool	$TC_TP1[T_NO,0]=0 or DELT ("DRILL", DUPLO_NO) $TC_TP1[GETT("DRILL"),0]=0 or alternatively: DELT("DRILL")	Tool with T_NO is deleted Tool "DRILL" DUPLO_NO is deleted

11.15 Overview of the remaining OPI blocks of tool management

11.15.1 Magazine data, directory

OPI block TMV

Calculation of line: Magazine number, if a field[ ] is present

Calculation of column: n.a.

NC identifier	Designation	OPI variable	Type
None	Number of magazines	numActMags	WORD
	Number of the magazine	magVNo[ ]	WORD

NC identifier	Designation	OPI variable	Type
	Identifier of the magazine	magVIdent[ ]	String
	Number of adapters	numActAdapter	WORD
	Number of the adapter	adaptVNo	WORD
	Adapter type	AdaptVType	WORD

11.15.2 Tool data, directory

OPI block TV

Calculation of line: Sequential no. of the tools, if [ ] present

Calculation of column: n.a.

NC identifier	Designation	OPI variables	Type
None	Duplo number	nrDuplo[ ]	WORD
	Number of cutting edges	numCuttEdges[ ]	WORD
	Number of tools in TO area	numTools	WORD
	Last T number assigned for tool management	TnumWZV	WORD
	Tool identifier	toolIdent[ ]	String
	Current magazine	toolInMag[ ]	WORD
	Actual location	toolInPlace[ ]	WORD
	T number	toolNo[ ]	WORD
	Number of tool groups	numToolsGroups	WORD

11.15.3 Parameterization, return parameters _N_TMGETT, _N_TSEARC

OPI block TF

Calculation of line: Refer to table

Calculation of column: n.a.

Designation	OPI variables	Calculation of line	Type
Comparison value for PI TSEARCH of variables of the OPI block TD	parDataTD	Parameter index of block TD	WORD
Comparison value for PI TSEARCH of variables of the OPI block TO	parDataTO	Parameter index of block TO	DOUBLE
Comparison value for PI TSEARCH of variables of the OPI block TD	parDataToolIdentTD	Parameter index of block TD	String
Comparison value for PI TSEARCH of variables of the OPI block TS	parDataTS	Parameter index of block TS	DOUBLE

Designation	OPI variables	Calculation of line	Type
Comparison value for PI TSEARCH of variables of the OPI block TU	parDataTU	Parameter index of block TU	DOUBLE
Comparison value for PI TSEARCH of variables of the OPI block TUE	parDataTUE	Parameter index of block TUE	DOUBLE
Comparison value for PI TSEARCH of variables of the OPI block TUS	parDataTUS	Parameter index of block TUS	DOUBLE
Mask for search criterion of PI TSEARCH (OPI block TD)	parMasksTD	Parameter index of block TD	WORD
Comparison value for PI TSEARCH of variables of the OPI block TO	parMasksTO	Parameter index of block TO	WORD
Comparison value for PI TSEARCH of variables of the OPI block TS	parMasksTS	Parameter index of block TS	WORD
Comparison value for PI TSEARCH of variables of the OPI block TU	parMasksTU	Parameter index of block TU	WORD
Comparison value for PI TSEARCH of variables of the OPI block TUE	parMasksTUE	Parameter index of block TUE	WORD
Comparison value for PI TSEARCH of variables of the OPI block TUS	parMasksTUS	Parameter index of block TUS	WORD
D number of cutting edge used	resultCutting EdgeNrUsed	Number of cutting edges of tool	WORD
Number of cutting edges used	resultNrOfCut EdgesUsed	1	WORD
Return: found tools	resultNrOfTools	1	WORD
Return: T numbers of found tools	resultToolNr[ ]	1... resultNrOfTools	WORD
T number of cutting edge used	resultToolNr Used	Number of cutting edges of tool	WORD

11.15.4 Working offsets

OPI block AEV

Calculation of line: Cutting edge number if [ ] available

Calculation of column: n.a.

NC identifier	Designation	OPI VAR	Type
None	Number of D numbers in block	numActDEdges	WORD
	D numbers	Dno[...]	WORD
	Internal T number	toolNo[...]	WORD
	Cutting edge number	cuttEdgeNo[...]	WORD
	Tool identifier	toolIdent[...]	STRING
	Duplo number	duploNo[...]	WORD
	Magazine	toolInMag[...]	WORD
	Location	toolInPlace[...]	WORD

11.15.5 PI services and language commands for tool management

Overview

FB4 (PI_SERV) or FB7 can be used to start program invocation services (PI services) in the NC area. A program section, which carries out a particular function (e.g. with tool management, search for empty location in a magazine), is executed in the NC by making a request via the PI service.

PI service	Functions	NC command
MMCSEM	Semaphores for various PI services	
DELETO	Delete tool	DELT("Tool", Duplo)
DELECE	Delete tool cutting edge	$TC_DP1[t,d]=0
CREATO	Generate tool	NEWT("Tool", Duplo)
CRTOCE	Create tool with specification of cutting edge no.	$TC_DPx[t,D] $TC_DPCx[t,D] $TC_DPCSx[t,D] $TC_MOPx[t,D] $TC_MOPCx[t,D]
TMCRTO	Create tool	$TC_TPx[t]
TMCRTC (not available in PLC)	Create tool by specifying cutting edge no.	$TC_DPx[t,d]
CREACE	Create cutting edge	$TC_DP[t,d]=value
CRCEDN	Create new cutting edge	$TC_DPx[t,d]
TMFDPL	Empty location search for loading	GETFREELOC(magNo, locNo, T no., refMag, refLoc, withReserv)
TMMVTL	Prepare magazine location for loading, unload tool	
TMPCIT	Set incremental value for workpiece counter, decrement count by y	SETPIECE(SpinNo,y)
TMPOSM	Position magazine location or tool	POSM(p,m,ip,im)
TMFPBP	Empty location search acc. to properties	
TSEARC	Complex search using search screen forms	User cycle program
TMRASS	Reset active status	
TMGETT	Specify T number for specified tool identifier with duplo number	GETT("Tool", Duplo)
	Read the preset T number	GETSELT(SpinNo)
CHKDNO (not available in PLC) TMCHKD (not available in PLC)	Check the uniqueness of D numbers of the tool data of the TO unit assigned to the executing channel. Parameters t1, t2, d are optional.	status=CHKDNO (t1,t2,d)
DZERO (not available in PLC)	Set D numbers for all tools of the TO unit assigned to the channel to "invalid". On the OPI, such D numbers are displayed with the value 0. The invalid D number is generated NC-internally by assigning the value "old D number" + 32000 to the D number.	DZERO
	For the offset number D=d, specify the associated internal T no.=t of the tool. The tool that has the status "active" and "was in use" is taken from the tool group.	status=GETACTTD (t,d)

PI service	Functions	NC command
	Specify the D no. for tool t and its cutting edge ce	d=GETDNO(t,ce)
	Set the D no. of tool t and its cutting edge ce to value d	status=SETDNO (t,ce,d)
	Read the active T no. and status	status=GETACTT (Tno,"Tool")
	Delete command for all location-dependent/setup offsets of a cutting edge or tool if d is not specified	status=DELDL(t,d)
SETTST (not available in PLC)	Set tool status to "active"	SETTA(Stat,m,vnr)
SETTST (not available in PLC)	Set tool status to "not active"	SETTIAStat,m,vnr)
CHKDM (not available in PLC)	Check uniqueness of D nos. in magazine; m=Magazine	status=CHKDM(m)
	Set tool holder number (h=holder no.)	SETMTH(h)
	Set leading spindle (s=spindle no.)	SETMS(s)
TRESMO	Tool life / quantity / wear setpoint activation	RESETMON
TMAWCO (not available in PLC)	Set a wear group to active	$TC_MAP9
	Location-dependent offsets, coarse	$TC_EPx[t,d]
	Technological grinding data	$TC_TPGx[t]
	Read the loaded T number	GETEXET
	NC command to move tool	MVTOOL
	Delete tool holder block	DELTC
	Tool deselection/tool change irrespective of tool status	TCA
	Delete the location status "reserved for tool in buffer"	DELMLRES
	Delete owner magazine location of the tool	DELMLOWNER

NC language commands

NC states are read with the following NC commands.

Functions	NC command
Active tool number T	$P_TOOLNO
Last programmed tool number (without magazine management)	$P_TOOLP
Active tool cutting edge	$P_TOOL
Active total tool length; n=1-3	$P_TOOLL[n]
Active tool holder	$P_TC
Active angle of a tool holder axis; n=1-2	$P_TCANG[n]
Difference between calculated and used angle of a tool holder axis for a Hirth joint of the angle	$P_TCDIFF[n]
Active tool radius; n=1-2	$P_TOOLR
Number of cutting edges for tool t; t=T number 1 32000	$P_TOOLND[t]
Tool with T number t exists	$P_TOOLEXIST[t]
Active tool offsets, n=1...40	$P_AD[n]
Active tool offsets transformed; n=1-40	$P_ADT[n]
Active additive offset number	$P_DLNO
T no. of the tool (numerical)	$C_T
Tool identifier (string)	$C_TS
Bool variable that shows whether a T word is available in $C_T	$C_T_PROG

Functions	NC command
Parameter for tool name as string	$C_TS_PROG
Address extension of the T word	$C_TE
Programmed D no.	$C_D
Bool variable that shows whether an offset number is available in $C_D	$C_D_PROG
Programmed additive/setup offset	$C_DL
Bool variable that shows whether an offset number is available in $C_DL	$C_DL_PROG
Command number, specifies the required operation: 1: Move (load/unload, relocate...); 2: Prepare change; 3: Change On; 4: Change ON (turret, without M06); 5: Prepare change and change ON (with M06); -1 At the read instant, no tool management command is active	$AC_TC_FCT
Acknowledgement status of PLC FC8/FC6	$AC_TC_STATUS
Tool holder or spindle number onto which the new tool should be loaded	$AC_TC_THNO
Source magazine number of the new tool	$AC_TC_MFN
Source location number of the new tool	$AC_TC_LFN
Target magazine number of new tool	$AC_TC_MTN
Target location number of the new tool	$AC_TC_LTN
Source magazine number of the old tool (to be removed)	$AC_TC_MFO
Source location number of the old tool (to be removed)	$AC_TC_LFO
Target magazine number of the old tool (to be removed)	$AC_TC_MTO
Target location number of the old tool (to be removed)	$AC_TC_LTO
Returns the number of the actual leading spindle	$AC_MSNUM
Returns the number of the leading spindle	$P_MSNUM
Returns the number of the main tool holder	$AC_MTHNUM
Trigger variable to output command of the NC (is set for one IPO)	$AC_TC_CMDT
Trigger variable for acknowledgement of the PLC (is set for one IPO)	$AC_TC_ACKT
Counter for the command output	$AC_TC_CMDC
Counter for the acknowledgements	$AC_TC_ACKC
Owner magazine of the new tool	$AC_TC_MMYN
Owner location of the new tool	$AC_TC_LMYN
Returns the number of the main tool holder	$P_MTHNUM
Magazine number of tool t	$A_TOOLMN[t]
Magazine location of tool t	$A_TOOLMLN[t]
Number of the owner magazine	$A_MYMN
Number of the owner magazine location	$A_MYMLN
Number of defined magazines that are assigned to the channel	$P_MAGN
Number of defined magazines, i-th magazine number	$P_MAG[i]
Number of defined adapters that are assigned to the channel	$P_MAGNA
Number of defined adapters, i-th adapter number	$P_MAGA[i]

Functions	NC command
Number of magazines n that are linked to location n	$P_MAGNDIS[m,n]
Number of the i-th magazine that is linked with location l of the buffer magazine	$P_MAGNDISS[l,i]
Number of the i-th magazine that is linked with location l of the loading magazine	$P_MAGDISL[l,i]
Number of defined magazine location type hierarchies	$P_MAGNH
Number of the defined location types in the n-the defined hierarchy	$P_MAGNHLT[n]
m-th location type of hierarchy n	$P_MAGHLT[n,m]
Number of buffers assigned to the spindle numbers, tool holder numbers n	$P_MAGNREL[n]
m-th buffer number of the n-th spindle no., tool holder no.	$P_MAGREL[n,m]
Number of spindle locations / tool holder locations in the buffer magazine that are assigned to the channel	$P_MAGNS
n-th number of the spindle / tool holder in the buffer	$P_MAGS[n]
Determine the defined D number of a tool	$P_TOOLD
Determine existence of a tool	$P_TOOLEXIST
Number of DL offsets of the D offset given by T number T and D number D	$P_TOOLNDL[t,d]
Number of defined tool groups that are assigned to the channel	$P_TOOLNG
Number of defined tools that are assigned to the channel	$P_TOOLNT
i-th tool number T	$P_TOOLT[i]
The following tool changes are available for all tools, whose parameter $TC_TP11 have set one of the bits of $P_USEKT. The value zero has the same contents as "all bits are set"	$P_USEKT, $TC_TP11
Factor for tool search	$AC_MONMIN
Workpiece count	$A_USEDND
Workpiece count	$A_USEDT
Workpiece count	$A_USEDD
Read factor for tool life monitoring	$A_MONIFACT

Multitool and tool specifications in PI services

The following PI services refer to tools and multitools:

PI service	Meaning
TMFDPL	Empty location search for specified tool or multitool for loading or equipping. The empty location search in the MT only makes sense for tools that have not been loaded, i.e. as part of the equipping of the MT with tools.
TMFPBP	Empty location search for specified tool or multitool. This service searches the magazine(s) specified in the relevant parameters for an empty location, which meets the specified criteria (tool size and location type).
TMMVTL	Load or unload tool or multitool. It is not possible to program tools of a multitool for the move command. It is also not possible to move a tool into an MT with the command.
TMGETT	T number of tool with the specified tool identifier/duplo number or MT number of the multitool with the specified identifier (multitool: the duplo number is programmed with the parameter value 00000)
TRESMO	Reset monitoring values. If an MT number is programmed, only the tools of the multitool treated.
TMPOSM	Position magazine on the specified magazine location, specified tool or multitool.

The following PI services are available functionally for tools and multitools, but with different names:

PI service	Meaning
Create	
CREATO	Create tool with specified T number
CRTOCE	Create tool with specified T number and D number
TMCRTO	Create tool with specified identifier/duplo number
TMCRTC	Create tool with specified identifier/duplo number and D number
Delete	
DELETO	Delete tool with specified T number
TMDLTO	Delete tool with specified identifier / duplo number

Note

If these tool-specific PI services are called with MT-specific names or MT numbers, the respective PI service is aborted with error status.

The following PI services are defined functionally only for tools (i.e. there are no similar PI services for multitools):

PI service	Meaning
CHEKDM	Check the D numbers for uniqueness
CHKDNO	Check whether the tools have unique D numbers
DZERO	Set all D numbers invalid with the "Unique D number" function
CRCEDN	Create new tool cutting edge with specification of the D number
CREACE	Create tool cutting edge
DELECE	Delete tool cutting edge
SETTST	Activate tool from tool group (set tool state "active")
TMRASS	Reset active status
TMAWCO	Set active wear group in a magazine
TMPCIT	Set increment value for workpiece counter
TSEARC	Complex search using search screen forms

The following PI services are defined for adapters:

PI service	Meaning
TMCRAD	Creating adapters with a given type
TMDLAD	Deleting adapters with a given number

Note

During the search, the PI service only considers tools, also those contained in a multitool, i.e. the PI service does not consider multitools. A "Complex multitool search" PI service is not defined.

If the tool-specific PI services are called with MT-specific names or MT numbers, the PI service is aborted with error status.

SINUMERIK Operate user interface

12

12.1 Configuring the user interface

Configurable settings

The following settings can be configured:

- General settings
- Configure the tool lists
- Configure the list parameters
- Configure the tool types
- Configure the "Additional Data" window
- Configure the "New Tool - Favorites" window
- Configure the "New Tool" window
- Configuring the "New Multitool" window
- Configuring the "New angle head" window
- Configuring the "New Tool from Code Carrier" window
- Configuring the "New Tool from File" window
- Configure the default values for new tools
- Configure the "Details" window
- Assign names for magazines and magazine locations
- Configuring the "Assign Tooltips for Multitool Locations" window
- Assign names for magazine location types
- Assignment of magazines and magazine locations to channels
- Assignment of magazines and magazine locations to coordinate systems
- Disabling the softkey depending on filters
- Configuring the "Load Station Selection" window
- Coolant and tool-specific functions
- Reason for the tool change when reactivating
- Configure the code carrier connection
- Configuring store to file

The following sections explain how you configure these adaptations.

Procedure

1. Open the directory: /**siemens**/sinumerik/hmi/template/cfg.
2. Copy the "oem_sltmlistconfig.xml" template file
3. Copy the file to the directory /**oem**/sinumerik/hmi/cfg
   OR: /**user**/sinumerik/hmi/cfg.
4. Change the file name according to the set technology:

File name	MD52200 technology	MD52201 technology extension
sltmlistconfig.xml	Milling technology	
sltmturninglistconfig.xml	Turning technology	
sltmmillturnlistconfig.xml	Milling technology	Turning technology
sltmcirculargrindinglistconfig.xml	Cylindrical grinding technology	
sltmsurfacegrindinglistconfig.xml	Surface grinding technology	
sltmplclistconfig.xml	PLC tool management "TRANSLINE 2000"	

5. Further technologies are activated with the machine data MD 52202 $MCS_TECHNOLOGY_ADDON.

Bit	Tools
0	Grinding tools (types 40x), types 40x also include dresser types 490, 494 and 496
1	Grinding tools (types 41x), types 41x also include dresser types 490, 495 and 497

6. Remove the comments of the example in the template to create your specific configuration.

Configuration examples

Examples for all possible configurations are provided in the template oem_sltmlistconfig.xml, which are described individually in the following sections:

```
 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
- <CONFIGURATION>
<!--  ************************************************************-->
<!--  enter your configuration behind this comment -->
<!--  ************************************************************ -->


 Configuration examples


<!--  ************************************************************ -->
<!--  enter your configuration ahead this comment -->
<!--  ************************************************************ -->
 </CONFIGURATION>
```

Enter your configuration using the following xml identifiers:

Identifier <tag>	For the following setting:
<SETTINGS>	General settings (Page 294)
<LISTCONFIGURATION>	Configuring tool lists (Page 300)
<PARAMETERCONFIGURATION>	Configuring list parameters (Page 320)
<TOOLTYPECONFIGURATION>	Configure tool types (Page 327)
<MOREDATACONFIGURATION>	Configuring the "Additional Data" window (Page 331)
<NEWTOOLFAVORITECONFIGURATION>	Configure the "New tool - favorites" window (Page 333)
<NEWTOOLCONFIGURATION>	Configuring the "New Tool" window (Page 334)
<NEWMULTITOOLCONFIGURATION>	Configuring the "New Multitool" window (Page 335)
<NEWCROSSHEADCONFIGURATION>	Configuring the "New angle head" window (Page 335)
<NEWTOOLIDENTCONNECTIONCONFIGURATION>	Configuring the "New Tool from Code Carrier" window (Page 336)
<NEWTOOLSAVETOOLDATACONFIGURATION>	Configuring the "New Tool from File" window (Page 336)
<NEWTOOLDEFAULTVALUECONFIGURATION>	Configuring default values for new tools (Page 337)
<DETAILSCONFIGURATION>	Configuring the "Details" window (Page 344)
<MAGAZINEPLACENAMECONFIGURATION>	Assigning a name for magazines and magazine locations (Page 346)
<MULTITOOLPLACETOOLTIPCONFIGURATION>	Configuring the "Assign Tooltips for Multitool Locations" window (Page 348)
<PLACETYPECONFIGURATION>	Assigning a name for magazine location types (Page 348)
<CHANNELMAGAZINEASSIGNMENT>	Assignment of magazines and magazine locations to channels (Page 350)
<COORDSYSMAGAZINEASSIGNMENT>	Assignment between magazines and magazine locations and coordinate systems (Page 351)
<FILTERCONFIGURATION>	Disabling the softkey depending on filters (Page 353)
<LOADSTATIONSHANDLEALLTOOLS>	Configuring the "Load Station Selection" window (Page 354)
<TOOLCHANGEREASON>	Reason for the tool change when reactivating (Page 358)
<IDENTCONNECTIONCONFIGURATION>	Configure the code carrier connection (Page 361)
<SAVETOOLDATACONFIGURATION>	Configuring Save to file (Page 364)

12.1.1 General settings

<SETTINGS> tag

The general settings for the tool management user interface are stored in the <SETTINGS> tag.

Entries	Meaning
ReactivateWithMagPos	True - reactivate with positioning of the tool to the tool magazine loading point. False - default setting, reactivate without positioning the tool magazine.
ReactivateAllMonitorModes	True - the actual values of all the monitoring types set in the NC are reset. False - default setting, actual value of the active monitoring type is reset.
ReactivateEnabled	True - default setting, reactivation is enabled. False - reactivation is disabled.
CreateNewToolDialog	True - the "New Tool" window is displayed. In this window, you can specify the tool name and the size of the tool (oversized tool) before actually creating the tool. Other input options can be defined by configuring the "New tool" window. False - default setting, the "New Tool" window is not displayed. Once the name has been entered, the tool will be created directly in the tool list.
CreateNewMultiToolDialog	True - default setting, the "New Multitool" window is displayed. In this window, you can configure the following before actually creating the multitool: • Multitool names • Number of multitool locations • Type of clearance input • Value of the location clearance for each multitool location False - the "New Multitool" window is not displayed. Once the name has been entered, the multitool will be created directly in the tool list.
CreateNewCrossHeadDialog	True - the "New angle head" window is shown. In this window you can enter the angle head name and the angle head size (angle head oversized) before actually creating the angle head. Other input options can be defined by configuring the "New angle head" window. False - default setting, the "New angle head" window is not displayed. Once the name has been entered, the angle head will be created directly in the tool list

Entries	Meaning
MagazineSelectionDialog	True - a dialog box displaying the buffer, tool magazine and NC memory is displayed. A mark can be added to or removed from each of these elements to indicate whether that element appears in the list display. The "Go to" softkey can be used to select an element directly in the list display. False - no dialog box for magazine selection. The system switches between the buffer, the tool magazines and the NC memory. If there is no MagazineSelectionDialog entry in the <Settings> tag, the dialog box is automatically displayed with more than one tool magazine (excluding the system magazine) when magazine configuration is taking place. If there is only one tool magazine, the switchover function is used.
NewToolFavoritesOnly	Refers to the "New tool" function. True - only those tool types that are configured as favorites are listed. False - default setting, all tool types are listed.
SortationInFirstEtcLevel	True - the "Sorting" function is placed at the first softkey level. The "Continue" function is omitted. False - default setting, sorting at all softkey levels.
ToolBufferOnceOnTop	True - default setting, buffer is displayed at the start of the list. False - buffer is displayed for each magazine. Only those locations are displayed that are assigned to the magazine.
UnloadToolFromLoadPlaceEnabled	True - default setting. A tool at the loading point can be removed from the loading point using the "Unload" softkey. False - the "Unload" softkey is not displayed for a tool at the loading point.
CreateNewToolWithCuttPos9Enabled	True - cutting edge position 9 is offered in the cutting edge position selection. True - default setting, cutting edge position 9 is not offered in the cutting edge position selection.
AutoHNumberDisplayEnabled	Column of the cutting edge parameter "IsoHNoDPH - ISO H number" depending on the machine configuration. True - default setting, the column is automatically displayed. False - the column is not automatically displayed.
AutoUniqueDNumberDisplayEnabled	Column of the cutting edge parameter "UniqueDNo" depending on the machine configuration. True - default setting, the column is automatically displayed. False - the column is not automatically displayed.
AccessLevelDeleteAllDNos	The following entry specifies the access level as of which the "Delete all D numbers delete" function is offered. Default setting: Access level 3 (user) - relevant only when the "Unique D number" setting has been made.
MagazineNumberOfDCheckAndActivate	Default setting: -2 - the tools of all magazines are considered by the function "Check for unique D number and activate tool." >0 - The above function only considers the tools of the named magazine.

Entries	Meaning
AutoFixedPlaceDisplayEnabled	Column of the tool parameter "ToolStateFixed" - tool status - tool is fixed-location coded and acts dependent on the magazine configuration. True - default setting, the column is automatically hidden, if all of the configured magazines are fixed location coded. False - the column is not automatically hidden.
MagPlaceAdapterDisplayEnabled	Columns of the magazine location adapter parameters are displayed in the "Magazine" window. False - default setting, parameters are not displayed.
AdaptTransDataViewEnabled	The magazine location adapter transformed display of the cutting edge parameters is enabled. It can be switched on and off in the settings dialog of the tool lists. False - default setting, the magazine location adapter transformed display of the cutting edge parameters is not enabled.
ActiveCompensationsEnabled	The "Working offsets" function is enabled. This function depends on the machine configuration. Unique D number, magazine location adapter and magazine management must be set. False - default setting, the "Working offsets" function is not enabled.
M2NLockEnabled	Locking of an M to N switchover while a function of the tool management is being executed. True - default setting, switchover is locked.
SDToolLengthInListEnabled	True - default setting, evaluation of SD42940, SD42942, and SD42950 in axis name display in the tool lists. False - the setting data are not evaluated in the axis name display.
DisableMultiTools	True - even though the Multitools function is set in the NC, they are not offered in the tool lists. False - default setting, if the Multitools function is set in the NC, they are offered in the tool lists.
CursorPositionInfoEnabled	True - information about the tool on which the cursor is positioned is written to the SlCap local variable "/Hmi/TMHMICurData" when exiting the tool list. False - Default setting, no information on the tool is provided when exiting the tool list.
ContinuousCursorPositionInfoEnabled	True - information about the tool on which the cursor is positioned is written to the SlCap local variable "/Hmi/TMHMICurData". False - Default setting, no information on the tool is provided
SelectionInfoEnabled	True - Information on the selected tools is written to the SlCap-local variable "/Hmi/TMHMICurData". The data sets for the individual tools are separated by a line break '\n'. False - Default setting, no information on the marked tools is provided.

Entries	Meaning
SelectionInfoOnLeavingEnabled	true - Information on the selected tools is written to the SlCap-local variable '/Hmi/TMHMICurData' when exiting the tool list. The data sets for the individual tools are separated by a line break '\n'. false - Default setting, no information on the selected tools is provided when exiting the tool list.
EnableMonAndBasisGrindingToolTypes	True - Enable switchover of grinding tool types in respect of "with/without grinding-specific tool monitoring", "with/without consideration of the base dimension when calculating the grinding wheel radius". False - Default setting, switchover not enabled.
ErrorMessageWithoutProgressBar	True - Error messages are output without a progress indicator and must be acknowledged via softkey. False - default setting, error message is automatically closed when the progress indicator reaches the end.
ColumnZoomByResolution	True - column width is adapted to the resolution of the HMI. False - default setting, the column width is adapted to the font size in the HMI. With this setting, with higher resolutions more columns can be displayed next to each other without needing to scroll.
MarkingInListEnabled	True - Default setting, "Marking" function enabled in the tool list False - Function enabled
AccessLevelReadDetailsAllParameter	The following entry specifies the access level as of which the "Further details" function is offered in the "Details" dialog. Default setting: Access level 7 (keyswitch position 0) - can be overlaid through the setting in MD51198.
FixNumberOfMultiToolPlaces	Multitools are created with a fixed number of multitool locations. There is no prompt when you create the multitool.
FixKindOfDistInMultiTools	Multitools are created with a fixed type of clearance input of the multitool locations. There is no prompt when you create the multitool. Possible input value: 3 - clearance of the multitool locations as an angle.
DlDisplayMode	• DisplayDistributed - default setting, the setup offsets EC are displayed in the tool list and the additive offsets SC are displayed in the tool wear list. • EcInWearList - the setup offsets EC are displayed in the tool wear list and the sum offsets SC are not displayed.

Entries	Meaning
CombinedToolMonitoring	Specifies the combined monitoring types offered in the toggle field: • Time/Count - tool life and workpiece count • Time/Wear - tool life and wear monitoring • Count/Wear - workpiece count and wear monitoring • Time/WearDlSc - tool life and wear monitoring for additive and setup offsets • Count/WearDlSc - workpiece count and wear monitoring for additive and setup offsets • Time/Count/Wear - tool life, workpiece count and wear monitoring • Time/Count/WearDlSc - tool life, workpiece count and wear monitoring for additive and setup offsets
FiltersCombinedWithOr	Defines the filter criteria, which are linked in the filter window using a logical OR. The default setting for all the filter criteria is a logical AND. The possible entries are: • "StandBy" - for tools that are ready for use • "Active" - for tools with active identifier • "Locked" - for locked tools • "Prewarn" - for the prewarning limit • "RemainingPieces" - for the remaining unit quantity • "RemainingTime" - for the remaining tool time • "MarkedToUnload" - for the unload identifier • "MarkedToLoad" - for the load identifier
AddToGrindingSpindles	Defines the spindle numbers that are to be offered in the "GrindingSpindleNo" ($TC_TPG1) parameter when selecting the grinding spindles. The numbers of the grinding spindles must be separated by a space, e.g. -1 8 9. Default display: real NC spindles and 0
HideGrindingSpindles	Defines the spindle numbers of the real NC spindles that are not to be offered in the "GrindingSpindleNo" ($TC_TPG1) parameter when selecting the grinding spindles. The numbers of the grinding spindles must be separated by a space, e.g. 1 3.
MagazineMoveMessage	True - "Magazine movement running" or "Magazine movement ended" message is displayed. False - default setting, message is not displayed.
ToolLoadErrorMessage	Refers to the output of the fault message due to a negative PLC. True - default setting, the fault message is displayed. False - the fault message is not displayed.

The following settings refer to markings in the lists:

True - marking is displayed.

False - marking is not displayed (default setting).

If several markings apply to one location, the most important information is displayed. For instance, if a tool is locked, and at the same time, the "Tool position" and the "Reference direction holder angle" for this tool do not match, then a red cross is displayed for locked. The yellow triangle for the conflict cannot be seen.

Entries	Meaning
ShowSymbolActivePlace	Marking of the magazine location at the machining position. True - default setting applies for the "Turret" magazine type. False - setting for all other magazine types.
ShowSymbolActiveMultiToolPlace	Marking of the multitool location at the machining position. Is only displayed when the multitool is located in the spindle or the tool holder. True - default setting.
ShowSymbolLoadPlace	Selection of the magazine location at the loading position. True - default setting applies for all magazine types (except Turret is true). False - setting for the "Turret" magazine type.
ShowSymbolChangeInSpindlePlace	Marking of the magazine location at the change position. True - default setting applies for all magazine types (except Revolver is true). False - setting for the "Turret" magazine type.
ShowSymbolActiveTool	Marking of the active tool. False - default setting.
ShowSymbolProgrammedTool	Marking of the follow-on tool. True - default setting.
ShowSymbolActiveDNo	Marking of the active cutting edge. False - default setting.
ShowSymbolActiveDNoInTool	Marking of the active tool by marking the active cutting edge. True - default setting for configuration without magazine management. False - setting for configuration with magazine management.
ShowSymbolAcitveDlNo	Marking of the active DL number. True - default setting.
ShowConflictPositionAndDirection	Marking of a conflict; tool position and cutting direction do not match. True - default setting.
ShowConflictActiveToolWarning	Marking of a conflict; turret magazine type and the active tool are not at the machining position. True - default setting.

Example

```
<SETTINGS>
     <ReactivateWithMagPos value="false" type="bool" />
     <MagazineMoveMessage value="false" type="bool" />
     <CreateNewToolDialog value="true" type="bool" />
```

```
    <MagazineSelectionDialog value="false" type="bool" />
    <AccessLevelWriteDetailsAllParameter value="4" type="int" />
</SETTINGS>
```

12.1.2 Configuring tool lists

<LISTCONFIGURATION> tag

The entries used for configuring the following list views are located in the <LISTCONFIGURATION> tag:

- "Tool list" window, identifier <SlTmToollistForm>, see the 1st example.
- "Tool Wear" window, <SlTmToolwearForm> identifier
- "Magazine" window, <SlTmToolmagazinForm> identifier
- "Tool Data OEM" window, <SlTmTooloemForm>identifier
  This window is hidden in the default configuration. The window is only displayed when you enable it with <Enabled>, see the 3rd example.
- "Tool Selection" window, <SlTmToolselectionForm> identifier: In order to select a tool this window can be displayed in the program input screen.
- "Tools for program ... " window identifier <SlTmTTDForm>: This window is used to display the tools stored for a program (tool demand file *.TTD).

Entry	Meaning
Enabled	True - window is displayed. False - window is hidden.
CAPTION	Definition of the list name.
Text	Used as label/item below the "CAPTION" to define the heading text.
COLUMNX	Column selection: X stands for the column number.
TOOLTYPE_ALL	The column setting applies to all tool types.
TOOLTYPE_ALL_XXX	The column setting applies to all tool types of a group. XXX stands for the group number of the tool types. The hundreds position is used as the group number, for example, TOOLTYPE_ALL_100 or TOOLTYPE_ALL_200 etc.
TOOLTYPE_XXX	The column setting applies to a particular tool type. XXX stands for the number of the tool type.
EMPTY_MAG_PLACE	The column setting applies to an empty magazine location. Empty magazine locations appear in a list sorted according to magazine.
Item	Identifier of the list parameter. See example: The column in which the magazine location type is displayed is created with the identifier "MagPlaceType." Tool parameter identifiers (Page 304) Cutting edge parameter identifiers (Page 306) Monitoring parameter identifiers (Page 311) Magazine location parameter identifiers (Page 312)

Entry	Meaning
DELETE_COLUMN	Delete column.
INSERT_COLUMN	Insert column.

Note

Independent of their programming, the entries are effective in the following sequence:

1. COLUMNX
2. DELETE_COLUMN
3. INSERT_COLUMN

Note

The Enabled entry does not apply for the "Tool Selection" window. The other entries apply.

Example 1: Configuring the magazine list - magazine location and tool location type

Note

The columns of the list configuration cannot be taken from the display in the user interface in all cases because some columns are automatically inserted and therefore not included in the count of the configuration.

Variant 1

Column 10 and column 11 are overwritten. If column 10 and column 11 do not exist, the new columns will be added to the existing columns.

```
<LISTCONFIGURATION>
    <!-- **** Magazine list *** -->
    <SlTmToolmagazinForm>
      <!-- **** Column 10 *** -->
      <COLUMN10>
        <!-- **** For all tool types *** -->
        <TOOLTYPE_ALL>
          <!-- **** Display magazine location type *** -->
          <Item value="MagPlaceType" type="QString" />
        </TOOLTYPE_ALL>
      </COLUMN10>
      <!-- **** Column 11 *** -->
      <COLUMN11>
        <!-- **** For all tool types *** -->
        <TOOLTYPE_ALL>
          <!-- **** Display tool location type *** -->
          <Item value="ToolPlaceSpec" type="QString" />
```

```
        </TOOLTYPE_ALL>
        <!-- **** For all multitools *** -->
        <TOOLTYPE_9997>
          <!-- **** Display for all multitool location types *** -->
          <Item value="MultiToolPlaceSpec" type="QString" />
        </TOOLTYPE_9997>
      </COLUMN11>
    </SlTmToolmagazinForm>
</LISTCONFIGURATION>
```

Variant 2

The new columns are inserted in the standard configuration between the "Magazine location locked" and "Oversized tool" columns.

```
<LISTCONFIGURATION>
    <!-- **** Magazine list *** -->
    <SlTmToolmagazinForm>
      <!-- **** Insert column *** -->
      <INSERT_COLUMN>
        <!-- **** Column 7 *** -->
        <COLUMN7>
          <!-- **** For all tool types *** -->
          <TOOLTYPE_ALL>
            <!-- **** Display magazine location type *** -->
            <Item value="MagPlaceType" type="QString" />
          </TOOLTYPE_ALL>
        </COLUMN7>
        <!-- **** Column 8 *** -->
        <COLUMN8>
          <!-- **** For all tool types *** -->
          <TOOLTYPE_ALL>
            <!-- **** Display tool location type *** -->
            <Item value="ToolPlaceSpec" type="QString" />
          </TOOLTYPE_ALL>
          <!-- **** For all multitools *** -->
          <TOOLTYPE_9997>
            <!-- **** Display for all multitool location types *** -->
            <Item value="MultiToolPlaceSpec" type="QString" />
          </TOOLTYPE_9997>
        </COLUMN8>
      </INSERT_COLUMN>
    </SlTmToolmagazinForm>
</LISTCONFIGURATION>
```

Example 2: Configuring the magazine list - "Oversized tool" column is deleted

Note

The columns of the list configuration cannot be taken from the display in the user interface in all cases because some columns are automatically inserted and therefore not included in the count of the configuration.

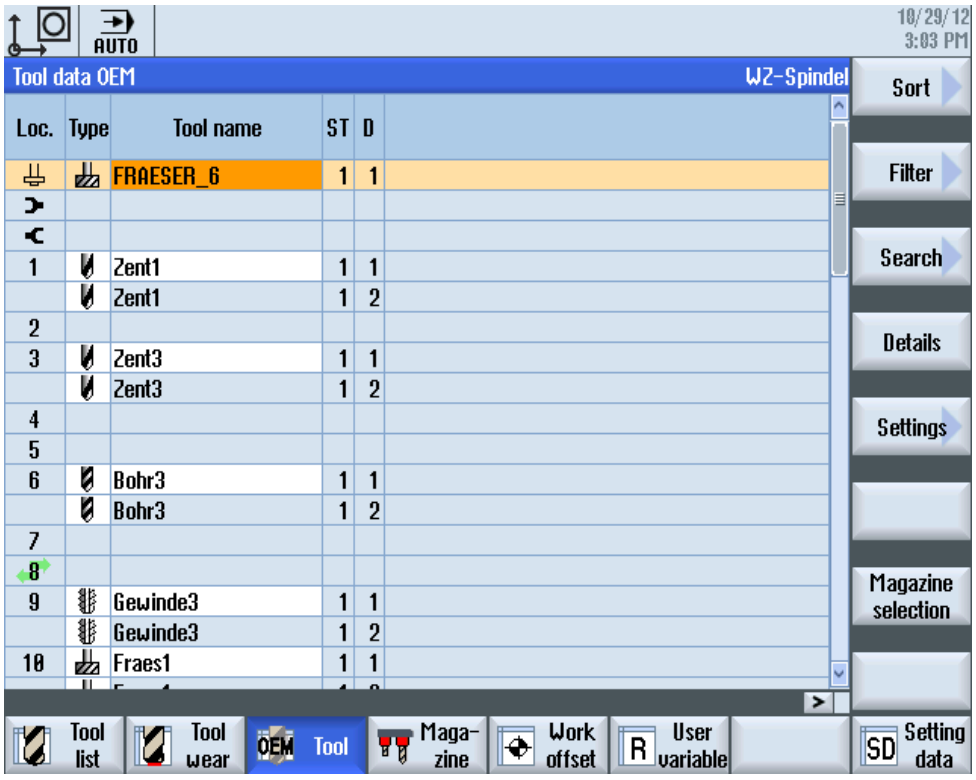
```
<LISTCONFIGURATION>
    <!-- **** Magazine list *** -->
    <SlTmToolmagazinForm>
      <!-- **** Delete column 7 *** -->
      <DELETE_COLUMN>
        <COLUMN7 />
      </DELETE_COLUMN>
    </SlTmToolmagazinForm>
</LISTCONFIGURATION>
```

Example 3: Enabling the OEM tool data list

```
<SlTmTooloemForm>
      <Enabled value="true" type="bool" />
</SlTmTooloemForm>
```

Result

- The "OEM Tool Data" window is displayed. It automatically displays the columns shown in the figure.
- The new "OEM tools" softkey appears and can be operated.



12.1.2.1 Tool parameter identifiers

Tool parameters

For the tool parameters, the following identifiers are used in the configuration file:

Identifier	Parameter or property	System variable *)
Empty	Empty field	-
ToolNo	Number of the tool	-
AdaptNo	Adapter number assignment	$TC_MPP7
DuploNo	Duplo number	$TC_TP1
DuploNoRW	Duplo number, read and write access	$TC_TP1
NumCuttEdges	Number of cutting edges of tool	$P_TOOLND
ToolIdent	Tool name	$TC_TP2
ToolIdentRO	Tool name, read only access	$TC_TP2
ToolInfo	Tool information	$TC_TP11
ToolInMag	Magazine in which the tool is located	$A_TOOLMN
ToolInPlace	Magazine location containing the tool	$A_TOOLMLN

Identifier	Parameter or property	System variable *)
ToolInMagInPlace	Magazine number/magazine location	$A_TOOLMN / $A_TOOL_MLN
ToolMonTCW	Type of tool monitoring, tool life, workpiece count and wear. If wear monitoring has not been enabled via the machine data, this parameter is handled like ToolMonTC.	$TC_TP9
ToolMonTC	Type of tool monitoring, tool life and workpiece count	$TC_TP9
ToolPlaceSpec	Location type	$TC_TP7
ToolPlaceSpecIdent	Location type, if it is to be displayed as text. The precondition is that the text for the location type was configured.	$TC_TP7
ToolSearch	Tool search type for replacement tools	$TC_TP10
ToolMyMag	Owner magazine of the tool	$A_MYMN
ToolMyPlace	Owner magazine location of the tool	$A_MYMLN
ToolSizeLeft	Tool size left in half locations	$TC_TP3
ToolSizeRight	Tool size right in half locations	$TC_TP4
ToolSizeUpper	Tool size top in half locations	$TC_TP5
ToolSizeDown	Tool size bottom in half locations	$TC_TP6
ToolOverSize	Tool size as fixed setting - two half locations left, two half locations right, one half location top, one half location bottom	$TC_TP3 ... $TC_TP6
ToolState	Tool status in the hexadecimal notation	$TC_TP8
ToolStateActiv	Tool status - tool active	$TC_TP8, Bit 0
ToolStateEnabled	Tool status - tool released	$TC_TP8, Bit 1
ToolStateLocked	Tool status - tool locked	$TC_TP8, Bit 2
ToolStateLockedRO	Tool status - tool locked, only read access	$TC_TP8, Bit 2
ToolStateMeasured	Tool status - tool measured	$TC_TP8, Bit 3
ToolStatePrewarn	Tool status - tool has reached prewarning limit	$TC_TP8, Bit 4
ToolStatePrewarnRO	Tool status - tool has reached prewarning limit, only read access	$TC_TP8, Bit 4
ToolStateInChange	Tool status - tool is being changed	$TC_TP8, Bit 5
ToolStateFixed	Tool status - tool fixed place coded	$TC_TP8, Bit 6
ToolStateUsed	Tool status - tool was in use	$TC_TP8, Bit 7
ToolStateAutoReturn	Tool status - automatic return	$TC_TP8
ToolStateIgnoreLocked	Tool status - do not observe locked	$TC_TP8, Bit 9
ToolStateMarkedToUnload	Tool status - tool is marked to unload	$TC_TP8, Bit 10
ToolStateMarkedToLoad	Tool status - tool is marked to load	$TC_TP8, Bit 11
ToolStatePermanent	Tool status - tool is permanent	$TC_TP8, Bit 12
ToolState1To1Exchange	Tool status - one to one replacement	$TC_TP8, Bit 14
ToolStateHandTool	Tool status - manual tool	$TC_TP8, Bit 15
ToolProtAreaFile	File name of the tool description which is required to define the protection zone	$TC_TP_PROTA
ToolMaxVelo	Maximum speed	$TC_TP_MAX_VELO
ToolMaxAcc	Maximum acceleration	$TC_TP_MAX_ACC

Identifier	Parameter or property	System variable *)
ToolInMultiTool	Multitool in which the tool is located	$A_TOOLMTN
ToolInCrossHead	Angle head in which the tool is located	$A_TOOLMTN
ToolInMultiToolPlace	Multitool location containing the tool	$A_TOOLMTLN
ToolInCrossHeadPlace	Angle head location where the tool is located	$A_TOOLMTLN
ToolMyMultiTool	Owner multitool of the tool	$A_MYMTN
ToolMyCrossHead	Owner angle head of the tool	$A_MYMTN
ToolMultiToolPlaceAdaptNo	Adapter number assignment with respect to the multitool/angle head location	$TC_MTPP7
ToolMyMultiToolPlace	Owner multitool location of the tool	$A_MYMTLN
ToolMyCrossHeadPlace	Owner angle head location of the tool	$A_MYMTLN
ToolAlarmIsExtended	PLC tool status - function "extended alarm" active	-
ToolAlarmLimit	PLC tool status - alarm limit reached	-
ToolExtAlarmLimit	PLC tool status - limit "extended alarm" reached	-
ToolUser_1, ... ToolUser_64	OEM tool parameters 1 ... 64	$TC_TPC1, ... $TC_TPC64
ToolAppl_1, ... ToolAppl_10	Siemens application tool parameters 1 ... 10	$TC_TPCS1, ... $TC_TPCS10

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

Note

Tool name "ToolIdent"

The maximum length of tool names is 31 ASCII characters.

The number of characters is reduced for Asian characters or Unicode characters.

The following special characters are not permitted: | # "

12.1.2.2 Cutting edge parameter identifiers

For the cutting edge parameters, the following identifiers are used in the configuration file:

Identifier	Parameters or properties	System variable *)
EdgeNo	D number	-
EdgeNoIdx	Cutting edge number	-
ToolType	Tool type, cutting edge parameter 1	$TC_DP1
CuttEdgePos	Cutting edge position, cutting edge parameter 2	$TC_DP2
GeoLength1	Length 1, cutting edge parameter 3	$TC_DP3
GeoLengthGeoAx1	Length of geometry axis 1, cutting edge parameter 3, e.g. length X	$TC_DP3
GeoLengthGeoAx1DiscDiameter	Grinding wheel diameter in geometry axis 1, cutting edge parameter 3	$TC_DP3
GeoLengthGeoAx1DresserDiameter	Dressing tool diameter in geometry axis 1, cutting edge parameter 3	$TC_DP3
GeoLength	Length, cutting edge parameter 3	$TC_DP3

Identifier	Parameters or properties	System variable *)
GeoLength2	Length 2, cutting edge parameter 4	$TC_DP4
GeoLengthGeoAx3	Length of geometry axis 3, cutting edge parameter 4, e.g. length Z	$TC_DP4
GeoLengthGeoAx3DiscDiameter	Grinding wheel diameter in geometry axis 3, cutting edge parameter 4	$TC_DP4
GeoLengthGeoAx3DresserDiameter	Dressing tool diameter in geometry axis 3, cutting edge parameter 4	$TC_DP4
GeoLength3	Length 3, cutting edge parameter 5	$TC_DP5
GeoLengthGeoAx2	Length of geometry axis 2, cutting edge parameter 5, e.g. length Y	$TC_DP5
GeoLengthGeoAx2DiscDiameter	Grinding wheel diameter in geometry axis 2, cutting edge parameter 5	$TC_DP5
GeoLengthGeoAx2DresserDiameter	Dressing tool diameter in geometry axis 2, cutting edge parameter 5	$TC_DP5
GeoRadius	Radius, cutting edge parameter 6	$TC_DP6
GeoOutsideRadius714	External radius for the star probe, tool type 714, cutting edge parameter 6	$TC_DP6
GeoHeadRadius	Head radius for the ball end mill (tapered die-sinking cutter), tool type 111, cutting edge parameter 7	$TC_DP7
GeoCuttEdgeRadius	Tool nose radius, cutting edge parameter 6	$TC_DP6
GeoCornerRadius	Corner radius, cutting edge parameter 7	$TC_DP7
GeoOutsideRadius	Outside radius, cutting edge parameter 7	$TC_DP7
GeoBoomLength	Cantilever length, cutting edge parameter 7	$TC_DP7
GeoSlotWidth	Slot width, for example, for slotting saw, cutting edge parameter 7	$TC_DP7
GeoRadius714	Ball radius for the star probe, tool type 714, cutting edge parameter 7	$TC_DP7
GeoLength4	Length 4, cutting edge parameter 8	$TC_DP8
PlateLength	Cutting tip length, cutting edge parameter 8	$TC_DP8
GeoProjection	Projection, e.g. for slotting saw, cutting edge parameter 8	$TC_DP8
Incircle	Inner circle for Prime A and Prime B, tool types 570 and 571, cutting edge parameter 8	$TC_DP8
GeoLength5	Length 5, cutting edge parameter 9	$TC_DP9
GeoWidth	Width, cutting edge parameter 9	$TC_DP9
PlateWidth	Cutting tip width, cutting edge parameter 9	$TC_DP9
GeoPitch	Thread pitch, cutting edge parameter 9	$TC_DP9
BoreRadius	Hole radius, cutting edge parameter 9	$TC_DP9
TopConeDiameter	The upper taper diameter for tool types 155, 156 and 157, cutting edge parameter 9	$TC_DP9
ReferenceLengthToConvexRadius	Reference length for radius convexity, e.g. for the barrel cutter, tool type 114, cutting edge parameter 9	$TC_DP9
HolderAngle	Holder angle, cutting edge parameter 10	$TC_DP10

Identifier	Parameters or properties	System variable *)
CorrectionAngle	Correction angle to align the probe, cutting edge parameter 10	$TC_DP10
GeoAngle2	Angle 2, cutting edge parameter 11	$TC_DP11
AngleConicalMillTool	Angle conical mill tools, cutting edge parameter 11	$TC_DP11
CuttDirection	Reference direction holder angle, cutting edge parameter 11	$TC_DP11
WearLength1	Wear length 1, cutting edge parameter 12	$TC_DP12
WearLengthGeoAx1	Wear length of geometry axis 1, cutting edge parameter 12, e.g. Δ length X	$TC_DP12
WearLength	Wear length, cutting edge parameter 12	$TC_DP12
WearLength2	Wear length 2, cutting edge parameter 13	$TC_DP13
WearLengthGeoAx3	Wear length of geometry axis 3, cutting edge parameter 13, e.g. Δ length Z	$TC_DP13
WearLength3	Wear length 3, cutting edge parameter 14	$TC_DP14
WearLengthGeoAx2	Wear length of geometry axis 2, cutting edge parameter 14, e.g. Δ length Y	$TC_DP14
WearRadius	Wear radius, cutting edge parameter 15	$TC_DP15
WearOutsideRadius714	Wear external radius for the star probe, tool type 714, cutting edge parameter 15	$TC_DP15
WearCuttEdgeRadius	Wear tool nose radius, cutting edge parameter 15	$TC_DP15
WearCornerRadius	Wear corner radius, cutting edge parameter 16	$TC_DP16
WearSlotWidth	Wear slot width, for example, for slotting saw, cutting edge parameter 16	$TC_DP16
WearHeadRadius	Wear head radius for the ball end mill (tapered die-sinking cutter), tool type 111, cutting edge parameter 16	$TC_DP16
WearLength4	Wear length 4, cutting edge parameter 17	$TC_DP17
WearProjection	Wear projection, for example, for slotting saw, cutting edge parameter 17	$TC_DP17
WearLength5	Wear length 5, cutting edge parameter 18	$TC_DP18
WearAngle1	Wear angle 1, cutting edge parameter 19	$TC_DP19
Incline	Inclined position for inclined turning tools, tool types 521, 531 and 541, cutting edge parameter 19	$TC_DP19
WearAngle2	Wear angle 2, cutting edge parameter 20	$TC_DP20
AdaptLength1	Adapter length 1, cutting edge parameter 21	$TC_DP21
AdaptLengthGeoAx1	Adapter length of geometry axis 1, cutting edge parameter 21, e.g. adapter length X	$TC_DP21
AdaptLength	Adapter length, cutting edge parameter 21	$TC_DP21
AdaptLength2	Adapter length 2, cutting edge parameter 22	$TC_DP22
AdaptLengthGeoAx3	Adapter length of geometry axis 3, cutting edge parameter 22, e.g. adapter length Z	$TC_DP22

Identifier	Parameters or properties	System variable *)
AdaptLength3	Adapter length 3, cutting edge parameter 23	$TC_DP23
AdaptLengthGeoAx2	Adapter length of geometry axis 2, cutting edge parameter 23, e.g. adapter length Y	$TC_DP23
ReliefAngle	Relief angle, cutting edge parameter 24	$TC_DP24
PlateAngle	Cutting tip angle, combination of cutting parameter 24 and cutting edge parameter 10	$TC_DP24 and $TC_DP10
PlateAngleRO	Plate angle, read-only access, for Prime A and Prime B, tool types 570 and 571, combination of cutting edge parameter 24 and cutting edge parameter 10	$TC_DP24 and $TC_DP10
NoseAngle	Nose angle, cutting edge parameter 24	$TC_DP24
CuttRate	Cutting rate, cutting edge parameter 25	$TC_DP25
SpindleDirection	For rotating tools, direction of rotation of the tool spindle. For turning tools, direction of rotation of the main spindle	$TC_DP25 bit8 and bit9
Coolant1	Coolant 1	$TC_DP25 Bit10
Coolant2	Coolant 2	$TC_DP25 Bit11
MFunction1	M function 1	$TC_DP25 Bit0
MFunction2	M function 2	$TC_DP25 Bit1
MFunction3	M function 3	$TC_DP25 Bit2
MFunction4	M function 4	$TC_DP25 Bit3
IsoHNoDPH	ISO H number	$TC_DPH
OrientNo	Cutting edge orientation	$TC_DPV
OrientV1	Cutting edge orientation vector 1	$TC_DPV3
OrientV2	Cutting edge orientation vector 2	$TC_DPV4
OrientV3	Cutting edge orientation vector 3	$TC_DPV5
OrientVGeoAx1	Cutting edge orientation vector of geometry axis 1, cutting edge orientation parameter 3, e.g. vector X	$TC_DPV3
OrientVGeoAx3	Cutting edge orientation vector of geometry axis 3, cutting edge orientation parameter 4, e.g. vector Z	$TC_DPV4
OrientVGeoAx2	Cutting edge orientation vector of geometry axis 2, cutting edge orientation parameter 5, e.g. vector Y	$TC_DPV5
OrientNormalV1	Cutting edge normal vector 1	$TC_DPVN3
OrientNormalV2	Cutting edge normal vector 2	$TC_DPVN4
OrientNormalV3	Cutting edge normal vector 3	$TC_DPVN5
OrientNormalVGeoAx1	Cutting edge normal vector geometry axis 1, cutting edge normal parameter 3, e.g. normal vector X	$TC_DPVN3
OrientNormalVGeoAx3	Cutting edge normal vector geometry axis 3, cutting edge normal parameter 4, e.g. normal vector Z	$TC_DPVN4

Identifier	Parameters or properties	System variable *)
OrientNormalVGeoAx2	Cutting edge normal vector geometry axis 2, cutting edge normal parameter 5, e.g. normal vector Y	$TC_DPVN5
TeethCount	Number of teeth, cutting edge parameter 34	$TC_DPNT
CalibrationDataSetNo	Calibration data set for measuring input, cutting edge parameter 34	$TC_DPNT
BasicRotationAngle	Clamping angle of tool, cutting edge parameter 35	$TC_DPROT
GeoConvexRadius	Radius convexity, cutting edge parameter 36	$TC_DP_RADIUS3
GeoInsideRadius	Inside radius, cutting edge parameter 36	$TC_DP_RADIUS3
GeoCornerRadius2	Corner radius 2 for the ball end mill (tapered die-sinking cutter), tool type 111, cutting edge parameter 36	$TC_DP_RADIUS3
WearConvexRadius	Wear radius convexity, cutting edge parameter 37	$TC_DP_WEAR_RADIUS3
WearInsideRadius	Wear inside radius, cutting edge parameter 37	$TC_DP_WEAR_RADIUS3
WearCornerRadius2	Wear corner radius 2 for the ball end mill (tapered die-sinking cutter), tool type 111, cutting edge parameter 37	$TC_DP_WEAR_RADIUS3
GeoInsideLength	Cutting edge length inside, cutting edge parameter 38	$TC_DP_LEN6
GeoMaxCuttDepth	Maximum cutting depth, cutting edge parameter 38	$TC_DP_LEN6
GeoPilotPinLength	Length pilot pin cutting edge parameter 38	$TC_DP_LEN6
WearInsideLength	Wear cutting edge length inside, cutting edge parameter 39	$TC_DP_WEAR_LEN6
WearMaxCuttDepth	Wear maximum cutting depth, cutting edge parameter 39	$TC_DP_WEAR_LEN6
WearPilotPinLength	Wear length pilot pin cutting edge parameter 39	$TC_DP_WEAR_LEN6
UniqueDNo	Unique D number	$TC_DPCE
EdgeUser_1, ... EdgeUser_64	OEM cutting edge parameters 1 to 64	$TC_DPC1, ... $TC_DPC64
EdgeAppl_1, ... EdgeAppl_10x	Siemens application cutting edge parameters 1 to 10	$TC_DPCS1, ... $TC_DPCS10

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

12.1.2.3 Monitoring parameter identifiers

For the monitoring parameters, the following identifiers are used in the configuration file:

Identifier	Parameters or properties	System variable *)
SupWarning	Prewarning limits, taking the current monitoring type into consideration	-
SupRemaining	Actual value, taking the current monitoring type into consideration	-
SupDesired	Setpoint, taking the current monitoring type into consideration	-
SupWarningTime	Prewarning limit service life	$TC_MOP1
SupRemainingTime	Tool life actual value	$TC_MOP2
SupWarningPieces	Prewarning limit for workpiece count	$TC_MOP3
SupRemainingPieces	Actual value for workpiece count	$TC_MOP4
SupDesiredTime	Setpoint for tool life	$TC_MOP11
SupDesiredPieces	Setpoint for workpiece count	$TC_MOP13
SupWarningWear	Prewarning limit for wear	$TC_MOP5
SupRemainingWear	Actual value for wear	$TC_MOP6
SupDesiredWear	Setpoint for wear	$TC_MOP15
SupExtendedAlarm	PLC tool management: Extended alarm	-
SupWarningPiecesIncremental	PLC tool management: Prewarning limit for workpiece count	$TC_MOP13 ... $TC_MOP3
SupActualPieces	PLC tool management: Actual value for workpiece count	$TC_MOP13 ... $TC_MOP4
EdgeSupUser_1, ... EdgeSupUser_64	OEM cutting edge monitoring parameters 1 ... 64	$TC_MOPC1, ... $TC_MOPC64
EdgeSupAppl_1, ... EdgeSupAppl_10	Siemens application cutting monitoring parameters 1 ... 10	$TC_MOPCS1, ... $TC_MOPCS10

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

12.1.2.4 Grinding parameter identifiers

For grinding parameters, the following identifiers are used in the configuration file:

Identifier/parameter	Meaning	System variable *)
GrindingSpindleNo	Spindle number	$TC_TPG1
GrindingConnectionRule	Chaining rule	$TC_TPG2
MinimalDiscRadius	Minimum grinding wheel radius	$TC_TPG3
ActualDiscRadius	Actual grinding wheel radius (read access only)	-
MinimalDiscWidth	Minimum grinding wheel width	$TC_TPG4
ActualDiscWidth	Current grinding wheel width	$TC_TPG5
MaximalDiscSpeed	Maximum grinding wheel speed	$TC_TPG6
MaximalDiscPeripheralVelocity	Maximum grinding wheel peripheral velocity	$TC_TPG7
AngleBevelDisc	Angle of inclined wheel	$TC_TPG8

Identifier/parameter	Meaning	System variable *)
ParamForRadiusCalculation	Parameters to calculate the radius	$TC_TPG9
DressingPath	Directory path to the dressing program	$TC_TPG_DRSPATH
DressingProgram	Dressing program name	$TC_TPG_DRSPROG

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

The following parameters are assigned to the cutting edges of the grinding tool:

Identifier/parameter	Meaning
GrindingMonitoring	Grinding-specific tool monitoring
GrindingUseBaseLength	Taking into account the base dimension when calculating the grinding wheel radius

12.1.2.5 Magazine location parameter identifiers

For the magazine location parameters, the following identifiers are used in the configuration file:

Identifier	Parameters or properties	System variable *)
MagPlaceKind	Location type	$TC_MPP1
MagPlaceType	Location type	$TC_MPP2
MagPlaceTypeIdent	Location type as text	$TC_MPP2
MagPlaceTNo	Tool number of the tool in this location	$TC_MPP6
MagPlaceWatchNeighbour	Consider adjacent location	$TC_MPP3
MagPlaceStateLocked	Magazine location locked	$TC_MPP4, Bit 0
MagPlaceStateEmpty	Magazine location free	$TC_MPP4, Bit 1
MagPlaceStateResInterMag	Reserved for tool in buffer	$TC_MPP4, Bit 2
MagPlaceStateResLoadTool	Reserved for tool to be loaded	$TC_MPP4, Bit 3
MagPlaceStateOccupiedLeft	Magazine location occupied, left half location	$TC_MPP4, Bit 4
MagPlaceStateOccupiedRight	Magazine location occupied, right half location	$TC_MPP4, Bit 5
MagPlaceStateOccupiedUpper	Magazine location occupied, upper half location	$TC_MPP4, Bit 6
MagPlaceStateOccupiedDown	Magazine location occupied, lower half location	$TC_MPP4, Bit 7
MagPlaceStateReservedLeft	Magazine location reserved, left half location	$TC_MPP4, Bit 8
MagPlaceStateReservedRight	Magazine location reserved, right half location	$TC_MPP4, Bit 9
MagPlaceStateReservedUpper	Magazine location reserved, upper half location	$TC_MPP4, Bit 10
MagPlaceStateReservedDown	Magazine location reserved, lower half location	$TC_MPP4, Bit 11
MagPlaceStateWearGroupLocked	Wear group disabled	$TC_MPP4, Bit 12
MagPlaceStateOverlapLocked-Place	Overlapping locked magazine locations permitted	$TC_MPP4, Bit 13
MagPlaceMagazineNo	Magazine number	-
MagPlaceTypeIdx	Type index	$TC_MPP5

Identifier	Parameters or properties	System variable *)
MagPlaceWearGroup	Wear group number	$TC_MPP5
MagPlaceAdaptNo	Adapter number	$TC_MPP7
MagPlaceNo	Magazine location number	-
MagNoMagPlaceNo	Magazine number/magazine location number	-
MagPlaceUser_1, ... MagPlaceUser_64	OEM magazine location parameters 1 ... 64	$TC_MPPC1, ... $TC_MPPC64
MagPlaceAppl_1, ... MagPlaceAppl_10	Siemens magazine location parameters 1 ... 10	$TC_MPPCS1, ... $TC_MPPCS10

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

12.1.2.6 Magazine location adapter parameter identifiers

The magazine location adapter parameters have been reorganized. The previous magazine location adapter parameters are still available (see 1st generation = type 1). However, they should not be used in the future and should be replaced by the magazine location adapter parameters (2nd generation = type 2).

The following identifiers are used in the configuration file for the magazine location adapter parameters (1st generation):

Identifier	Parameters or properties	System variable *)
MagPlaceAdapterLength1	Magazine location adapter length 1	$TC_ADPT1
MagPlaceAdapterLengthGeoAx1	Magazine location adapter geometry axis 1	$TC_ADPT1
MagPlaceAdapterLength2	Magazine location adapter length 2	$TC_ADPT2
MagPlaceAdapterLengthGeoAx3	Magazine location adapter geometry axis 3	$TC_ADPT2
MagPlaceAdapterLength3	Magazine location adapter length 3	$TC_ADPT3
MagPlaceAdapterLengthGeoAx2	Magazine location adapter geometry axis 2	$TC_ADPT3
MagPlaceAdapterTrafoNo	Magazine location adapter transformation number	$TC_ADPTT

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

The following identifiers are used in the configuration file for the magazine location adapter parameters (2nd generation):

Identifier	Parameters or properties	System variable *)
MagPlaceAdapterNo	Number of the magazine location adapter	-
MagPlaceAdapterType	Type of magazine location adapter: - 1 Adapter for magazine locations or - 2 Adapter for angle head/multitool locations -	$TC_ADPT_TYPE[n]
MagPlaceAdapterT1Length1	Length 1 - Type 1	$TC_ADPT1[n]
MagPlaceAdapterT1LengthGeoAx1	Length geometry axis 1 - Type 1	$TC_ADPT1[n]
MagPlaceAdapterT1Length2	Length 2 - Type 1	$TC_ADPT2[n]
MagPlaceAdapterT1LengthGeoAx3	Length geometry axis 3 - Type 1	$TC_ADPT2[n]
MagPlaceAdapterT1Length3	Length 3 - Type 1	$TC_ADPT3[n]

Identifier	Parameters or properties	System variable *)
MagPlaceAdapterT1LengthGeoAx2	Length geometry axis 2 - Type 1	$TC_ADPT3[n]
MagPlaceAdapterT1TrafoNo	Transformation number Type 1	$TC_ADPTT[n]

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

1st element of the angle head adapter

Identifier	Parameters or properties	System variable *)
MagPlaceAdapterL1_1	Offset – 1st element, 1st vector component - Type 2	$TC_ADPT_OFF[n, 0, 0]
MagPlaceAdapterL1GeoAx1	Offset – 1st element, geometry axis 1 - Type 2	$TC_ADPT_OFF[n, 0, 0]
MagPlaceAdapterL1_2	Offset – 1st element, 2nd vector component - Type 2	$TC_ADPT_OFF[n, 0, 1]
MagPlaceAdapterL1GeoAx3	Offset – 1st element, geometry axis 3 - Type 2	$TC_ADPT_OFF[n, 0, 1]
MagPlaceAdapterL1_3	Offset – 1st element, 3rd vector component - Type 2	$TC_ADPT_OFF[n, 0, 2]
MagPlaceAdapterL1GeoAx2	Offset – 1st element, geometry axis 2 - Type 2	$TC_ADPT_OFF[n, 0, 2]
MagPlaceAdapterDir1_1	Orientation – 1st rotary axis, 1st vector component - Type 2	Typ 2 - $TC_ADPT_DIR[n, 0, 0]
MagPlaceAdapterDir1GeoAx1	Orientation – 1st rotary axis, geometry axis 1 - Type 2	Typ 2 - $TC_ADPT_DIR[n, 0, 0]
MagPlaceAdapterDir1_2	Orientation – 1st rotary axis, 2nd vector component - Type 2	Typ 2 - $TC_ADPT_DIR[n, 0, 1]
MagPlaceAdapterDir1GeoAx3	Orientation – 1st rotary axis, geometry axis 3 - Type 2	Typ 2 - $TC_ADPT_DIR[n, 0, 1]
MagPlaceAdapterDir1_3	Orientation – 1st Rotary axis, 3rd vector component - Type 2	Typ 2 - $TC_ADPT_DIR[n, 0, 2
MagPlaceAdapterDir1GeoAx2	Orientation – 1st rotary axis, geometry axis 2 - Type 2	Typ 2 - $TC_ADPT_DIR[n, 0, 2]
MagPlaceAdapterAng1	Angle 1 - Type 2	$TC_ADPT_ANG[n, 0]
CrossHeadAng1	Angle 1 at angle head - Type 2	$TC_ADPT_ANG[n, 0]
MagPlaceAdapterAngConst1	Angle offset 1 - Type 2	$TC_ADPT_ANG_CONST[n, 0]

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

2nd element of the angle head adapter

Identifier	Parameters or properties	System variable *)
MagPlaceAdapterL2_1	Offset – 2nd element, 1st vector component - Type 2	$TC_ADPT_OFF[n, 1, 0]
MagPlaceAdapterL2GeoAx1	Offset – 2nd element, geometry axis 1 - Type 2	$TC_ADPT_OFF[n, 1, 0]
MagPlaceAdapterL2_2	Offset – 2nd element, 2nd vector component - Type 2	$TC_ADPT_OFF[n, 1, 1]
MagPlaceAdapterL2GeoAx3	Offset – 2nd element, geometry axis 3 - Type 2	$TC_ADPT_OFF[n, 1, 1]
MagPlaceAdapterL2_3	Offset – 2nd element, 3rd vector component - Type 2	$TC_ADPT_OFF[n, 1, 2]
MagPlaceAdapterL2GeoAx2	Offset – 2nd element, geometry axis 2 - Type 2	$TC_ADPT_OFF[n, 1, 2]
MagPlaceAdapterDir2_1	Orientation – 2nd rotary axis, 1st vector component - Type 2	Typ 2 - $TC_ADPT_DIR[n, 1, 0]
MagPlaceAdapterDir2GeoAx1	Orientation – 2nd rotary axis, geometry axis 1 - Type 2	Typ 2 - $TC_ADPT_DIR[n, 1, 0]
MagPlaceAdapterDir2_2	Orientation – 2nd rotary axis, 2nd vector component - Type 2	Typ 2 - $TC_ADPT_DIR[n, 1, 1]

Identifier	Parameters or properties	System variable *)
MagPlaceAdapterDir2GeoAx3	Orientation – 2nd rotary axis, geometry axis 3 - Type 2	Typ 2 - $TC_ADPT_DIR[n, 1, 1]
MagPlaceAdapterDir2_3	Orientation – 2nd Rotary axis, 3rd vector component - Type 2	Typ 2 - $TC_ADPT_DIR[n, 1, 2
MagPlaceAdapterDir2GeoAx2	Orientation – 2nd rotary axis, geometry axis 2 - Type 2	Typ 2 - $TC_ADPT_DIR[n, 1, 2]
MagPlaceAdapterAng2	Angle 2 - Type 2	$TC_ADPT_ANG[n, 1]
CrossHeadAng2	Angle 2 at angle head - Type 2	$TC_ADPT_ANG[n, 1]
MagPlaceAdapterAngConst2	Angle offset 2 - Type 2	$TC_ADPT_ANG_CONST[n, 1]

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

3rd element of the angle head adapter

Identifier	Parameters or properties	System variable *)
MagPlaceAdapterL3_1	Offset – 3rd element, 1st vector component - Type 2	$TC_ADPT_OFF[n, 2, 0]
MagPlaceAdapterL3GeoAx1	Offset – 3rd element, geometry axis 1 - Type 2	$TC_ADPT_OFF[n, 2, 0]
MagPlaceAdapterL3_2	Offset – 3rd element, 2nd vector component - Type 2	$TC_ADPT_OFF[n, 2, 1]
MagPlaceAdapterL3GeoAx3	Offset – 3rd element, geometry axis 3 - Type 2	$TC_ADPT_OFF[n, 2, 1]
MagPlaceAdapterL3_3	Offset – 3rd element, 3rd vector component - Type 2	$TC_ADPT_OFF[n, 2, 2]
MagPlaceAdapterL3GeoAx2	Offset – 3rd element, geometry axis 2 - Type 2	$TC_ADPT_OFF[n, 2, 2]
MagPlaceAdapterDir3_1	Orientation – 3rd rotary axis, 1st vector component - Type 2	Typ 2 - $TC_ADPT_DIR[n, 2, 0]
MagPlaceAdapterDir3GeoAx1	Orientation – 3rd rotary axis, geometry axis 1 - Type 2	Typ 2 - $TC_ADPT_DIR[n, 2, 0]
MagPlaceAdapterDir3_2	Orientation – 3rd rotary axis, 2nd vector component - Type 2	Typ 2 - $TC_ADPT_DIR[n, 2, 1]
MagPlaceAdapterDir3GeoAx3	Orientation – 3rd rotary axis, geometry axis 3 - Type 2	Typ 2 - $TC_ADPT_DIR[n, 2, 1]
MagPlaceAdapterDir3_3	Orientation – 3rd Rotary axis, 3rd vector component - Type 2	Typ 2 - $TC_ADPT_DIR[n, 2, 2
MagPlaceAdapterDir3GeoAx2	Orientation – 3rd rotary axis, geometry axis 2 - Type 2	Typ 2 - $TC_ADPT_DIR[n, 2, 2]
MagPlaceAdapterAng3	Angle 3 - Type 2	$TC_ADPT_ANG[n, 2]
MagPlaceAdapterAngConst3	Angle offset 3 - Type 2	$TC_ADPT_ANG_CONST[n, 2]

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

12.1.2.7 Multitool parameter identifiers

For the multitool parameters, the following identifiers are used in the configuration file:

Identifier	Parameters or properties	System variable *)
MultiToolNo	Number of the multitool	-
MultiToolNumberOfPlaces	Number of multitool locations	$TC_MTPN
MultiToolNumberOfPlacesRO	Number of multitool locations, only read access	$TC_MTPN
MultiToolIdent	Name of the multitool	$TC_MTP2
MultiToolInMag	Magazine in which the multitool is located	$A_TOOLMN
MultiToolInPlace	Magazine location containing the multitool	$A_TOOLMLN
MultiToolInMagInPlace	Magazine number/magazine location	$A_TOOLMN / $A_TOOLMLN
MultiToolMyMag	Owner magazine of the multitool	$A_MYMN
MultiToolMyPlace	Owner magazine location of the multitool	$A_MYMLN
MultiToolPlaceSpec	Location type	$TC_MTP7
MultiToolPlaceSpecIdent	Location type as text	$TC_MTP7
MultiToolSizeLeft	Multitool size to the left in half locations	$TC_MTP3
MultiToolSizeRight	Multitool size to the right in half locations	$TC_MTP4
MultiToolSizeUpper	Multitool size to the top in half locations	$TC_MTP5
MultiToolSizeDown	Multitool size to the bottom in half locations	$TC_MTP6
MultiToolOverSize	Multitool size as fixed setting - two half locations left, two half locations right, one half location top, one half location bottom	$TC_MTP3 - $TC_MTP6
MultiToolPosition	Multitool position	$TC_MTP_POS
MultiToolProtAreaFile	File name of the multitool description which is required to determine the protection zone	$TC_MTP_PROTA
MultiToolKindOfDist	Type of distance input of the multitool locations	$TC_MTP_KD
MultiToolKindOfDistRO	Type of distance input of the multitool locations, only read access	$TC_MTP_KD
MultiToolDistAngle	Type of distance input of the multitool locations as an angle	$TC_MTP_KD
MultiToolDistAngleRO	Type of distance input of the multitool locations as an angle, read only access	$TC_MTP_KD
MultiToolState	Multitool status in the hexadecimal notation	$TC_MTP8
MultiToolStateEnabled	Multitool status - multitool released	$TC_MTP8, Bit 1
MultiToolStateLocked	Multitool status - multitool locked	$TC_MTP8, Bit 2
MultiToolStateLockedRO	Multitool status - multitool locked, only read access	$TC_MTP8, Bit 2
MultiToolStateInChange	Multitool status - multitool is being changed	$TC_MTP8, Bit 5
MultiToolStateFixed	Multitool status - multitool fixed location coded	$TC_MTP8, Bit 6
MultiToolStateUsed	Multitool status - multitool was in use	$TC_MTP8, Bit 7
MultiToolStateAutoReturn	Multitool status - automatic return	$TC_MTP8, Bit 8

Identifier	Parameters or properties	System variable *)
MultiToolStateIgnoreLocked	Multitool status - do not observe locked status	$TC_MTP8, Bit 9
MultiToolStateMarkedToUnload	Multitool status - multitool is selected to be unloaded	$TC_MTP8, Bit 10
MultiToolStateMarkedToLoad	Multitool status - multitool is selected to be loaded	$TC_MTP8, Bit 11
MultiToolStatePermanent	Multitool status - multitool is a master tool	$TC_MTP8, Bit 12
MultiToolState1To1Exchange	Multitool status - one to one replacement	$TC_MTP8, Bit 14
MultiToolStateHandTool	Multitool status - manual tool	$TC_MTP8, Bit 15
MultiToolStateLockMtIfToolLock	Multitool status - multitool is locked if a tool is locked in multitool	$TC_MTP8, Bit 16
MultiToolUser_1, ... MultiToolUser_64	OEM multitool parameters 1 ... 64	$TC_MTPC1, ... $TC_MTPC64
MultiToolAppl_1, ... MultiToolAppl_10	Siemens application multitool parameters 1 ... 10	$TC_MTPCS1, ... $TC_MTPCS10

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

12.1.2.8 Multitool location parameter identifiers

For the multitool location parameters, the following identifiers are used in the configuration file:

Identifier	Parameters or properties	System variable *)
MultiToolDist	Multitool location distance taking into account the actual type of distance input of the multitool	-
MultiToolPlaceDistLength	Multitool location distance, length	$TC_MTPPL
MultiToolPlaceDistAngle	Multitool location distance angle, angle	$TC_MTPPA
MultiToolPlaceType	Location type	$TC_MTPP2
MultiToolPlaceTypeIdent	Location type as text	$TC_MTPP2
MultiToolPlaceState	Multitool location status in hexadecimal notation	$TC_MTPP4
MultiToolPlaceStateLocked	Multitool location locked	$TC_MTPP4, Bit0
MultiToolPlaceStateEmpty	Multitool location empty	$TC_MTPP4, Bit1
MultiToolPlaceTNo	Tool number of the tool in this multitool location	$TC_MTPP6
MultiToolPlaceAdaptNo	Adapter number	$TC_MTPP7
MultiToolPlaceNo	Multitool location number	-
MultiToolPlaceUser_1, ... MultiToolPlaceUser_64	OEM multitool location parameters 1 ... 64	$TC_MTPPC1, ... $TC_MTPPC64
MultiToolPlaceAppl_1, ... MultiToolPlaceAppl_10	Siemens application multitool location parameters 1 ... 10	$TC_MTPPCS1, ... $TC_MTPPCS10

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

12.1.2.9 Identifiers of the angle head parameters

For the angle head parameters, the following identifiers are used in the configuration file:

Identifier	Parameters or properties	System variable *)
CrossHeadNo	Number of the angle head	-
CrossHeadNumberOfPlaces	Number of angle head locations	$TC_MTPN
CrossHeadNumberOfPlacesRO	Number of angle head locations, only read access	$TC_MTPN
CrossHeadIdent	Name of the angle head	$TC_MTP2
CrossHeadInMag	Magazine in which the angle head is located	$A_TOOLMN
CrossHeadInPlace	Magazine location where the angle head is located	$A_TOOLMLN
CrossHeadInMagInPlace	Magazine number/magazine location	$A_TOOLMN/ $A_TOOLMLN
CrossHeadMyMag	Owner magazine of the angle head	$A_MYMN
CrossHeadMyPlace	Owner magazine location of the angle head	$A_MYMLN
CrossHeadPlaceSpec	Location type	$TC_MTP7
CrossHeadPlaceSpecIdent	Location type as text	$TC_MTP7
CrossHeadSizeLeft	Angle head size to the left in half locations	$TC_MTP3
CrossHeadSizeRight	Angle head size to the right in half locations	$TC_MTP4
CrossHeadSizeUpper	Angle head size upwards in half locations	$TC_MTP5
CrossHeadSizeDown	Angle head size downwards in half locations	$TC_MTP6
CrossHeadOverSize	Angle head size as fixed setting - two half locations left, two half locations right, one half location top, one half location bottom	$TC_MTP3 - $TC_MTP6
CrossHeadPosition	Angle head position	$TC_MTP_POS
CrossHeadProtAreaFile	File name of the angle head description which is required to determine the protection area	$TC_MTP_PROTA
CrossHeadKindOfDist	Type of distance input of the angle head locations	$TC_MTP_KD
CrossHeadKindOfDistRO	Type of distance input of the angle head locations, read-only access	$TC_MTP_KD
CrossHeadDistAngle	Type of distance input of the angle head locations as angle	$TC_MTP_KD
CrossHeadDistAngleRO	Type of distance input of the angle head locations as angle, read-only access	$TC_MTP_KD
CrossHeadState	Angle head status in hexadecimal notation	$TC_MTP8
CrossHeadStateEnabled	Angle head status – angle head enabled	$TC_MTP8, Bit 1
CrossHeadStateLocked	Angle head status – angle head locked	$TC_MTP8, Bit 2
CrossHeadStateLockedRO	Angle head status – angle head locked, only read access	$TC_MTP8, Bit 2
CrossHeadStateInChange	Angle head status – angle head is being changed	$TC_MTP8, Bit 5
CrossHeadStateFixed	Angle head status – angle head is fixed-location-coded	$TC_MTP8, Bit 6
CrossHeadStateUsed	Angle head status – angle head was in use	$TC_MTP8, Bit 7

Identifier	Parameters or properties	System variable *)
CrossHeadStateAutoReturn	Angle head status – automatic retraction	$TC_MTP8, Bit 8
CrossHeadStateIgnoreLocked	Angle head status – ignore locked status	$TC_MTP8, Bit 9
CrossHeadStateMarkedToUnload	Angle head status – angle head is marked to be unloaded	$TC_MTP8, Bit 10
CrossHeadStateMarkedToLoad	Angle head status – angle head is marked to be loaded	$TC_MTP8, Bit 11
CrossHeadStatePermanent	Angle head status – angle head is a master tool	$TC_MTP8, Bit 12
CrossHeadState1To1Exchange	Angle head status – one to one exchange	$TC_MTP8, Bit 14
CrossHeadStateHandTool	Angle head status – manual tool	$TC_MTP8, Bit 15
CrossHeadStateLockMtIfToolLock	Angle head status – angle head is locked if a tool in the angle head is locked	$TC_MTP8, Bit 16
CrossHeadUser_1, ... CrossHeadUser_64	OEM angle head parameters 1 ... 64	$TC_MTPC1, ... $TC_MTPC64
CrossHeadAppl_1, ... CrossHeadAppl_10	Siemens application angle head parameters 1 ... 10	$TC_MTPCS1, ... $TC_MTPCS10

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

12.1.2.10 Identifiers of the angle head location parameters

For the angle head location parameters, the following identifiers are used in the configuration file:

Identifier	Parameters or properties	System variable *)
CrossHeadDist	Angle head location distance taking into account the actual type of distance input of the angle head	-
CrossHeadPlaceDistLength	Angle head location distance length	$TC_MTPPL
CrossHeadPlaceDistAngle	Angle head location distance angle	$TC_MTPPA
CrossHeadPlaceType	Location type	$TC_MTPP2
CrossHeadPlaceTypeIdent	Location type as text	$TC_MTPP2
CrossHeadPlaceState	Angle head location status in hexadecimal notation	$TC_MTPP4
CrossHeadPlaceStateLocked	Angle head location locked	$TC_MTPP4, Bit 0
CrossHeadPlaceStateEmpty	Angle head location empty	$TC_MTPP4, Bit 1
CrossHeadPlaceTNo	Tool number of the tool in this angle head location	$TC_MTPP6
CrossHeadPlaceAdaptNo	Adapter number	$TC_MTPP7
CrossHeadPlaceNo	Angle head location number	-
CrossHeadPlaceUser_1, ... CrossHeadPlaceUser_64	OEM angle head location parameters 1 ... 64	$TC_MTPPC1, ... $TC_MTPPC64
CrossHeadPlaceAppl_1, ... CrossHeadPlaceAppl_10	Siemens application angle head location parameters 1 to	$TC_MTPPCS1, ... $TC_MTPPCS10

*) Parameter corresponds to the system variable or the system variable is the basis for this parameter.

12.1.3 Configuring list parameters

<PARAMETERCONFIGURATION> tag

In the <PARAMETERCONFIGURATION> tag, you have the following options:

1. Changing list parameters.
2. Creating new list parameters based on an existing parameter.

Changing list parameters

You can change all list parameters included in the system.

Tool parameter identifiers (Page 304)
Cutting edge parameter identifiers (Page 306)
Monitoring parameter identifiers (Page 311)
Grinding parameter identifiers (Page 311)
Magazine location parameter identifiers (Page 312)
Magazine location adapter parameter identifiers (Page 313)
Multitool parameter identifiers (Page 316)
Multitool location parameter identifiers (Page 317)
Identifiers of the angle head parameters (Page 318)
Identifiers of the angle head location parameters (Page 319)

Entries	Meaning
HeadLine	Column header. Text entered is displayed as a header. Example: Text "Radius" is displayed as a header (default).
Tooltip	Text that is displayed in the tooltip. Example: Text "Geometry radius" is displayed in the tooltip (default).
ShortText	Text if the parameter is displayed in the "More Data" window. Example: Text "Rad." is displayed in the "More Data" window (default).
DetailsText	Text, if the parameter is displayed in the "Details" window. Example: Text "Radius" is displayed in the "Details" window (default).
Width	Column width in pixels in relation to a 640 x 480 resolution. See example below: The default column width is changed to 53 pixels.

Entries	Meaning	
DisplayMode	Values that the parameter is displayed with. See example below: The default setting is changed to "DoubleMode". The following values can also be accepted:	
	AnyMode	All characters
	IntegerMode	Integers
	UnsignedIntegerMode	Unsigned integers
	DoubleMode	Values with decimal places The number of decimal places is specified in "DecimalPlaces"
	UnsignedDoubleMode	Unsigned values with decimal places The number of decimal places is specified in "DecimalPlaces"
	Length	Specified length
	Angle	Specified angle
	LinearFeedPerTime	Linear feed in mm/minute
	LinearFeedPerRevolution	Linear feed in mm/revolution
	LinearFeedPerTooth	Linear feed in mm/tooth
	RevolutionSpeed	Speed
	ConstantCuttingSpeed	Constant cutting speed
DecimalPlaces	Number of decimal places if "DoubleMode" or "UnsignedDoubleMode" is selected in the DisplayMode. See example: Two decimal places are entered.	
ItemType	Field types in which a value is displayed.	
	TextField	Input/output field for values and texts
	TextFieldReadOnly	Output field for values and texts
	CheckBox	Input/output field for states
	CheckBoxReadOnly	Output field for states
BitMask	Bit mask for the display of a bit from a value. The bit mask is entered as integer value. Bit 0 → 1, Bit 1 → 2, Bit 2 → 4	
AccessLevel	Value for the access level that is used for these parameters in the user interface. Value range 1 ... 7	
UpperLimit	Upper input limit for these parameters in the user interface. Only applies for numerically entered parameters.	
LowerLimit	Lower input limit for these parameters in the user interface. Only applies for numerically entered parameters.	
Attributes	Special property of the parameter	
	EmptyInMultiTool	Parameter is hidden on a multitool or angle head location

Example

In the following example, the `<GeoRadius>` parameter is used.

Only enter the modified data.

```
<PARAMETERCONFIGURATION>
     <GeoRadius>
          <Width value="53" type="int" />
          <DisplayMode value="DoubleMode" type="QString" />
          <DecimalPlaces value="2" type="int" />
     </GeoRadius>
</PARAMETERCONFIGURATION>
```

Configuring new list parameters based on an existing list parameter.

Assign a new parameter name and enter only the changed data:

Entries	Meaning
	Assign new parameter name See example 1: Text "NewGeoRadius"
Base	Name of the parameter on which the new parameter is based. See example 1: "GeoRadius" parameter is used as a template.
	Also enter only the changed data here. All other data is taken from the existing parameter. See example 1: Column width changed to 46 pixels. Number of decimal places changed to 1.

Example 1

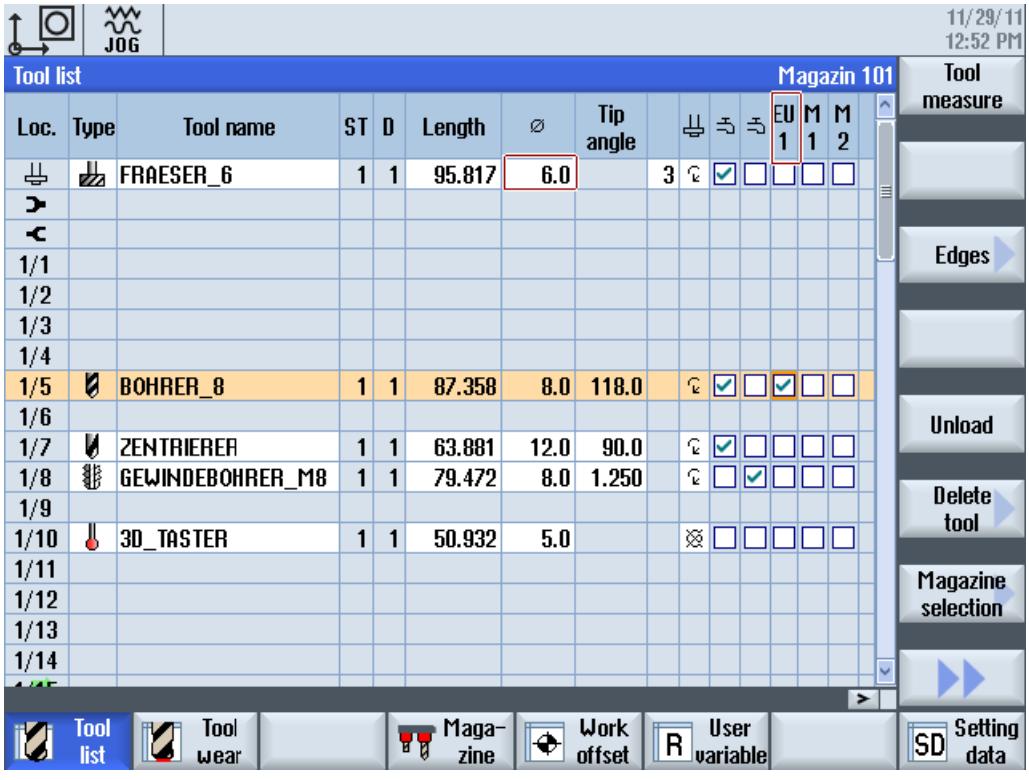
```
<PARAMETERCONFIGURATION>
     <NewGeoRadius>
          <Base value="GeoRadius" type="QString" />
          <Width value="46" type="int" />
          <DisplayMode value="DoubleMode" type="QString" />
          <DecimalPlaces value="1" type="int" />
     </NewGeoRadius>
</PARAMETERCONFIGURATION>
```

Example 2

Entries	Meaning
	Assign new cutting edge parameter name The name is "EdgeUser_1_Bit0" in the following example.
Base	Name of the parameter on which the new parameter is based. In the following example, the "EdgeUser_1" parameter is used as template.

Entries	Meaning
	Also enter only the changed data here. All other data is taken from the existing cutting edge parameter.
	The following entries are changed: Field type: Input and output field for states Value output: All characters Bit mask: Bit 0 Column width changed to 17 pixels. Header changed to "TM_HL_EDGE_USER_1_Bit0" Text in the tooltip changed to "TM_TT_EDGE_USER_1_Bit0"

```
<PARAMETERCONFIGURATION>
    <EdgeUser_1_Bit0>
        <Base value="EdgeUser_1" type="QString" />
        <ItemType value="CheckBox" type="QString" />
        <DisplayMode value="AnyMode" type="QString" />
        <BitMask value="1" type="int" />
        <Width value="17" type="int" />
        <HeadLine value="TM_HL_EDGE_USER_1_Bit0" type="QString" />
        <ToolTip value="TM_TT_EDGE_USER_1_Bit0" type="QString" />
    </EdgeUser_1_Bit0>
</PARAMETERCONFIGURATION>
```



Dynamic headers for columns in lists

There are columns in the tool management lists in which values with different meanings can be shown in one column.

In these cases, the column header always refers to the line in which the cursor is located.

The column header changes accordingly depending on the content displayed.

Example

Depending on the selected monitoring type, the values for tool life monitoring or unit quantity monitoring are shown in the same column.

Depending on the line in which the cursor is located, the header is either "Tool life" or "Unit quantity".

If no monitoring type has been selected in the line, then no heading is displayed.

The objective of this concept is to show many of the parameters and data in the most compact way possible.

12.1.4 List of tool types

Coding of tool types for milling tools

Group with type 1xy (milling tool):

100	Milling tool according to CLDATA (DIN 66215)
110	Ball end mill (cylindrical die-sinking cutter)
111	Ball end mill (tapered die-sinking cutter)
114	Barrel milling cutter
115	Tapered milling cutter
116	Tapered barrel milling cutter
120	End mill (without corner rounding)
121	End mill (with corner rounding)
130	Angle head cutter (without corner rounding)
131	Angle head cutter (with corner rounding)
140	Facing cutter
145	Thread cutter
150	Side mill
151	Saw
155	Bevel cutter (without corner rounding)
156	Bevel cutter (with corner rounding)
157	Tapered die-sinking cutter
160	Drill and thread milling cutter
190 ... 199	OEM - milling tools

Coding of tool types for drills

Group type 2xy (drills):

200	Twist drill
205	Solid drill
210	Boring bar
220	Center drill
230	Countersink
231	Counterbore
240	Tap regular thread
241	Tap fine thread
242	Tap Whitworth thread
250	Reamer
290 ... 299	OEM - drill bits

Coding of tool types for grinding tools

Group type 4xy (grinding tools):

400	Grinding wheel circumference
410	Face grinding wheel
480 ... 489	OEM - grinding tools
490	Dresser
494	Dressing roller for cylindrical grinding
495	Dressing roller for surface grinding
496	Dressing wheel for cylindrical grinding
497	Dressing wheel for surface grinding

Grinding tool types 401, 402, 403 as well as 411, 412, 413 are not listed as tool types that can be selected. The grinding-specific tool monitoring - taking into account the basic dimension when calculating the grinding wheel radius - is set using dedicated parameters. These parameters act on the tool types in the usual form.

Coding of tool types for turning tools

Group type 5xy (turning tools):

500	Roughing tool
505	Y roughing tool[1]
510	Finishing tool
515	Y finishing tool[1]
520	Plunge cutter
521	Inclined plunge cutter
525	Y plunge cutter[1]

530	Cutting tool
531	Inclined cutting tool
535	Y cutting tool[1]
540	Threading tool
541	Inclined threading tool
550	Steel profile
560	Rotary drill (ECOCUT)
570	Prime A
571	Prime B
580	Orientated probe
585	Calibration tool
590 ... 599	OEM - turning tools

[1] Tool types are optional

Coding of tool types for special tools

Group type 7xy (special tools):

700	Slotting saw
710	3D probe
711	Edge probe
712	Mono probe
713	L probe
714	Star probe
725	Calibration tool
730	Stop
731	Spindle sleeves
732	End support
790 ... 799	OEM - turning tools
900	Auxiliary tools

Coding of tool types for crosshead and multitool

The tool types for crosshead and multitool are not real tool types in the sense of the NCK. The identifiers are only used as an aid to be able to address the crosshead an the multitool when configuring the tool lists.

9994	Crosshead
9997	Multitool

OEM tool types

Each group of tool types contains a number range reserved for its own tool types.

By assigning the OEM tool types in the individual groups, it is possible that the OEM tool types are treated almost identically to the standard tool types from the point of view of the cycles and the NCK.

12.1.5 Configure tool types

<TOOLTYPECONFIGURATION> tag

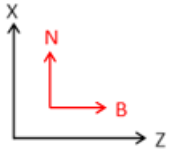
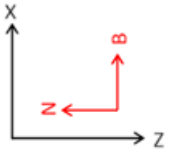
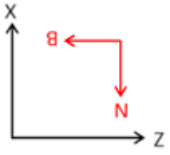
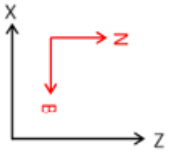
The entries used for configuring the tool types are located in the <TOOLTYPECONFIGURATION> tag:

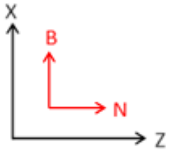
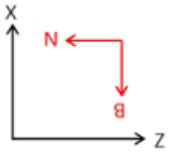
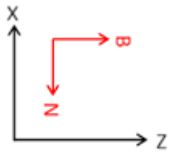
Entries	Meaning
TOOLTYPE_XXX	XXX stands for the number of the tool type. For the assignment of the tool type and number, please refer to Section "List of tool types (Page 324)".
TOANO_X	X stands for the number of the TO area. This entry can be used to make TO area-specific entries. This entry is only evaluated for the Iconorder and IconTrafoYTurning entries.
Tooltype	Number (XXX) of the tool type
Name	Text identifier of the tool name. The tool name is displayed in the following windows: • "New Tool - Favorites" • "New tool - milling cutter 100 ... 199" • "New tool - drill 200 ... 299" • "New tool - grinding tool 400 ... 499" • "New Tool - special and auxiliary tools 700 ... 900" See example: For the text identifier "`TM_PAR_SHANK_END_CUTTER`", the name "End mill" appears on the user interface.
Shortname	Text identifier of the tool name. Tool name is displayed in the tool list. See example: For the text identifier "`TM_PPTT_SHANK_END_CUTTER`", the name "MILLING CUTTER" appears on the user interface.
Tooltip	Text identifier of the tool name. This tool name is displayed in the tooltip. See example: For the text identifier "`TM_TTTT_SHANK_END_CUTTER`", the name "End mill" appears on the user interface.
Icon9 - for milling technology	Icon representing the tool type. The icons are displayed in the following windows: • "Tool list" in the "Type" column • "New tool - favorites" in the "Tool position" column • "New tool - milling cutter 100 ... 199" in the "Tool position" column • "New tool - drill 200 ... 299" in the "Tool position" column • "New tool - special tool 700 ... 900 in the "Tool position" column The icons are in ".png" format. They are stored according to image resolution in the following directories: /**oem**/sinumerik/hmi/ico/ico640, or ico800, or ico1024 /**user**/sinumerik/hmi/ico/ico640, or ico800, or ico1024

Entries	Meaning
IconX - for the turning and grinding technologies	Icon representing the tool type (see Icon9 - for milling technology) Special feature - tool positions are supported in the turning and grinding technologies. X is for the tool position represented by the icon. Position 9 is an undefined position represented by a cross.
Icon9_2	Icon representing the tool type in cutting edge position 9. It is used when cutting edge position 9 is to be treated as a valid cutting edge position. (see <SETTINGS> - CreateNewToolWithCuttPos9Enabled, Chapter "General settings (Page 294)")
Iconorder	Order in which the tool positions, represented by icons, are switched.
DisableInNewToolDialog	With this entry, the tool type in question is not offered when new tools are created. If the tool list finds a tool of this type that already exists, it is correspondingly displayed.
DELETE_TOOLTYPE	All predefined tool types that are not to be supported in the tool lists.
GrindingTooltypes	All tool types that are to be used as grinding tool types.
IconTrafoYTurning	Adaptation of the representation of the cutting edge positions and the reference directions for Y turning tools (see table below)

IconTrafoYTurning

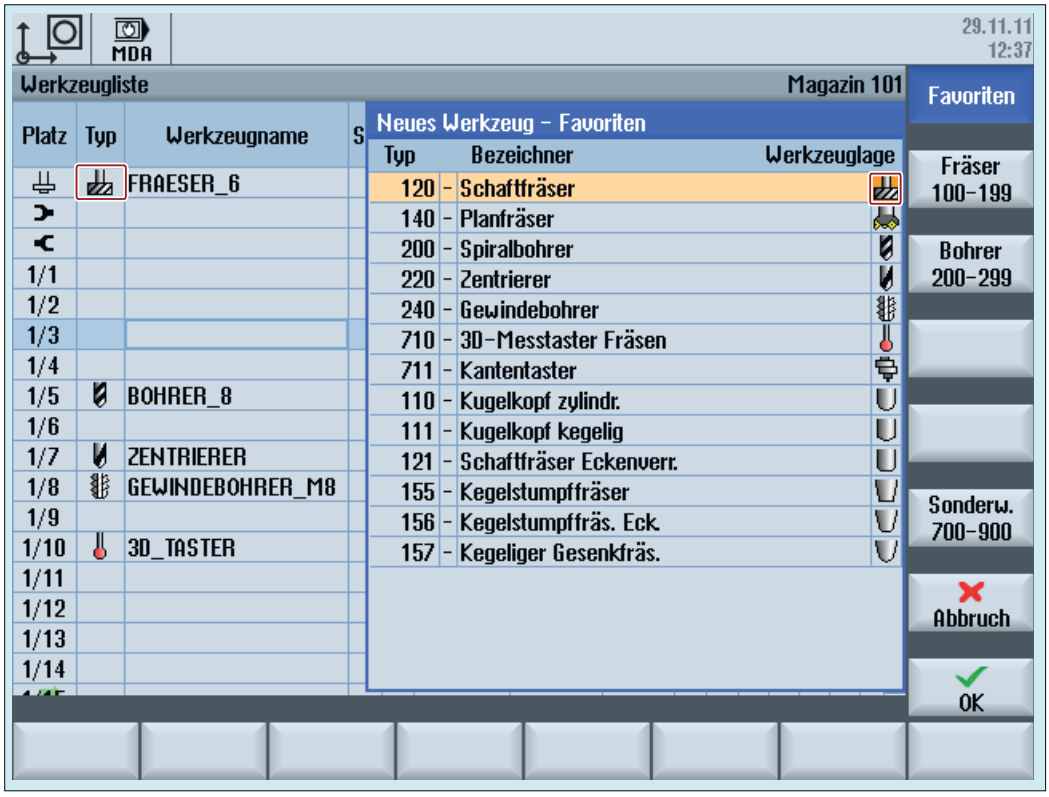
If the tool coordinate system (B, N, O) does not correspond to the WCS (Z, X, Y) during machining, an adjustment for the representation of the cutting edge positions, reference directions and the tool representation can be configured in tool management.

IconTrafoYTurning	Transformation	Icon assignment	Reference direction
0		505, 515: 1,2,3,4,5,6,7,8 525, 535: 1,2,3,4,5,6,7,8	505, 515: 1,2,3,4 525, 535: 1,2,3,4
1		505, 515: 2,3,4,1,6,7,8,5 525, 535: 3,5,4,6,7,1,8,2	505, 515: 4,3,1,2 525, 535: 4,3,1,2
2		505, 515: 3,4,1,2,7,8,5,6 525, 535: 4,7,6,1,8,3,2,5	505, 515: 2,1,4,3 525, 535: 2,1,4,3
3		505, 515: 4,1,2,3,8,5,6,7 525, 535: 6,8,1,3,2,4,5,7	505, 515: 3,4,2,1 525, 535: 3,4,2,1

IconTrafoYTurning	Transformation	Icon assignment	Reference direction
4		505, 515: 1,4,3,2,6,5,8,7 525, 535: 2,1,8,7,6,5,4,3	505, 515: 3,4,1,2 525, 535: 3,4,1,2
5		505, 515: 4,3,2,1,5,8,7,6 525, 535: 6,8,1,3,2,4,5,7	505, 515: 1,4,2,3 525, 535: 1,4,2,3
6		505, 515: 3,2,1,4,8,7,6,5 525, 535: 4,7,6,1,8,3,2,5	505, 515: 4,2,3,1 525, 535: 4,2,3,1
7		505, 515: 2,1,4,3,7,6,5,8 525, 535: 3,5,4,6,7,1,8,2	505, 515: 2,1,3,4 525, 535: 2,1,3,4

Example 1

```
<TOOLTYPECONFIGURATION>
     <TOOLTYPE_120>
       <Tooltype value="120" type="uint" />
       <Name value="TM_PAR_SHANK_END_CUTTER" type="QString" />
       <Shortname value="TM_PPTT_SHANK_END_CUTTER" type="QString" />
       <Tooltip value="TM_TTTT_SHANK_END_CUTTER" type="QString" />
       <Icon9 value="to_poly_shank_end_cutter_down.png" type="QString" />
     </TOOLTYPE_120>
</TOOLTYPECONFIGURATION>
```



Example 2

```
<TOOLTYPECONFIGURATION>
    <DELETE_TOOLTYPE>
        <TOOLTYPE_110/>
    </DELETE_TOOLTYPE>
</TOOLTYPECONFIGURATION>
```

Example 3 - Specifically defining the Iconorder TO area for the Y roughing tool

```
<TOOLTYPECONFIGURATION>
    <TOOLTYPE_505>
        <TOANO_1>
            <Iconorder value="8 7 6 5 3 4 2 1" type="QString" />
        </TOANO_1>
        <TOANO_2>
            <Iconorder value="3 4 2 1 8 7 6 5" type="QString" />
        </TOANO_2>
    </TOOLTYPE_505>
    <TOANO_1>
        <IconTrafoYTurning value="2" type="uint" />
```

```
      </TOANO_1>
</TOOLTYPECONFIGURATION>
```

12.1.6 Configuring the "Additional Data" window

<MOREDATACONFIGURATION> tag

The entries used for configuring the "Additional Data" window are located in the <MOREDATACONFIGURATION> tag. Various data items can be displayed for each tool type in a number of rows and columns. Each data item to be displayed is specified by entering a list parameter:

Tool parameter identifiers (Page 304)

Cutting edge parameter identifiers (Page 306)

Monitoring parameter identifiers (Page 311)

A brief description of the parameter (ShortText) then appears in the window with the value itself beside it. Configuring list parameters (Page 320).

If another description text is required in the "More Data" window, a text can also be entered in a row or column. In the window, the width of the text is made to conform to that of the brief description and value of a parameter.

Any number of rows and columns can be created. A scroll bar appears when the window reaches a certain size.

Entries	Meaning
TOOLTYPE_XXX	Number of the tool type. See example: Tool type 111 = Ball end mill (tapered die-sinking cutter).
TOANO_X	X stands for the number of the TO area. This entry can be used to make TO area-specific entries.
ROWX_COLY	Particular rowX_columnY. If there is no entry after rowX and columnY, this field remains blank. See example: In the "More Data" window, the first row is blank.
Item	The following entries are possible: • Parameter identifier • Text See example: The header "Corner radius" appears in the second row of the window. The text "Radius" and an input field for the value appear in the third row of the window. The fourth row is blank.
Span	By default, the width of two columns is reserved for the output of a parameter header. If a header is longer, this entry can be used to increase the reserved width in the "Columns" unit (see Example 3).

Entries	Meaning
TOOLTYPE_ALL_XXX	The setting applies to all tool types of a group. XXX stands for the group number of the tool types. The hundreds position is used as the group number, for example, TOOLTYPE_ALL_100 or TOOLTYPE_ALL_200 etc. (see example).
WITHOUT_MORE_DATA	Tool types of a tool group for which no "further data" are to be displayed can be excluded with this identifier (see example).

Example 1

```
<MOREDATACONFIGURATION>
     <TOOLTYPE_111>
        <ROW1_COL1>
        </ROW1_COL1>
        <ROW2_COL1>
           <Item value="TM_DGL_CORNER_RADIUS" type="QString" />
        </ROW2_COL1>
        <ROW3_COL1>
           <Item value="GeoCornerRadius" type="QString" />
        </ROW3_COL1>
        <ROW4_COL2>
        </ROW4_COL2>
     </TOOLTYPE_111>
</MOREDATACONFIGURATION>
```

Example 2

```
<MOREDATACONFIGURATION>
     <TOOLTYPE_ALL_500>
        <ROW1_COL1>
           <Item value="TM_DGL_LENGTH" type="QString" />
        </ROW1_COL1>
        <ROW2_COL1>
           <Item value="GeoLengthGeoAx3" type="QString" />
        </ROW2_COL1>
     ...
     </TOOLTYPE_ALL_500>
     <WITHOUT_MORE_DATA>
        <TOOLTYPE_580 />
     </WITHOUT_MORE_DATA>
</MOREDATACONFIGURATION>
```

Example 3

```
<MOREDATACONFIGURATION>
     <TOOLTYPE_ALL_505>
     ...
         <ROW1_COL1>
            <Item value="TM_DGL_EDGE_ORIENTATION" type="QString" />
            <Span value ="6" type="int" />
         </ROW1_COL1>
     ...
     </TOOLTYPE_ALL_505>
</MOREDATACONFIGURATION>
```

Example 4 - Specifically hiding the clamping angle in the Additional data TO area window for the Y roughing tool

```
<MOREDATACONFIGURATION>
     <TOOLTYPE_505>
          <TOANO_1>
             <ROW9_COL1>
                   <Itemvalue=""type="QString" />
             </ROW9_COL1>
             <ROW10_COL1>
                   <Itemvalue=""type="QString" />
             </ROW10_COL1>
          </TOANO_1>
     </TOOLTYPE_505>
</MOREDATACONFIGURATION>
```

12.1.7 Configure the "New tool - favorites" window

<NEWTOOLFAVORITECONFIGURATION> tag

In the tag <NEWTOOLFAVORITECONFIGURATION> you define the tool types that are most frequently used. The specified tool types are displayed in the "New Tool - Favorites" window.

Entries	Meaning
StaticTooltypes	Numbers of tool types Enter the individual numbers, separated by blanks. The following section describes the assignment of the tool type and number: List of tool types (Page 324)
AdditionalStaticTooltypes	Numbers of the tool types that are to be displayed in addition below the "StaticTooltypes".

Example

```
<NEWTOOLFAVORITECONFIGURATION>
    <StaticTooltypes value="120 140 200 220 240 710 711" type="QString"/>
    <AdditionalStaticTooltypes value="110 111 121 155 156 157 9994 9997"
    type="QString"/>
</NEWTOOLFAVORITECONFIGURATION>
```

12.1.8 Configuring the "New Tool" window

<NEWTOOLCONFIGURATION> tag

You use the "New Tool" dialog to supply a new tool with specific tool data before creating the tool or before loading the tool to a magazine location. These are typically the tool size and/or the tool location type because these parameters can no longer be changed when the tool is loaded. It is also useful to display the tool type and the tool names in this window.

Specify the parameters in the <NEWTOOLCONFIGURATION> tag that are to be displayed in the "New Tool" dialog.

Note

The number of parameters is limited by the size of the window.

Entries	Meaning
ItemX	Entry of the identifier of the tool parameter. "X" specifies the order in which the parameters are displayed in the window. The identifier of the tool parameter can be found in Chapter: Tool parameter identifiers (Page 304)

Example

```
<NEWTOOLCONFIGURATION>
    <!-- The standard field "oversized tool" is removed and replaced by the fields -->
    <!-- "tool location type", "tool size left" and "tool size right" -->
    <!-- -->
    <Item2 value="ToolPlaceSpec" type="QString"/>
    <Item3 value="ToolSizeLeft" type="QString"/>
    <Item4 value="ToolSizeRight" type="QString"/>
</NEWTOOLCONFIGURATION>
```

12.1.9 Configuring the "New Multitool" window

<NEWMULTITOOLCONFIGURATION> tag

You use the "New Multitool" dialog to supply a new multitool with specific multitool data before creating the multitool or loading it into to a magazine location. Multitool data typically includes the number of multitool locations and spacing between locations, as these parameters cannot be changed once the multitool has been created. The same applies to the multitool size and/or multitool location type as to the tool size and tool location type. These parameters, too, cannot be changed once they have been loaded into the tool. It is also a good idea to display the multitool name in this window.

In the <NEWMULTITOOLCONFIGURATION>tag, specify the parameters that you wish to have displayed in the "New Multitool" dialog.

Entries	Meaning
ItemX	Entry of the identifier of the multitool parameter. "X" specifies the order in which the parameters will be displayed in the window. The identifier of the multitool parameter can be found in Chapter:Multitool parameter identifiers (Page 316)

12.1.10 Configuring the "New angle head" window

<NEWCROSSHEADCONFIGURATION> tag

You use the "New angle head" dialog to supply a new angle head with specific data before creating the multitool or loading it into to a magazine location. These are typically the angle head size and/or the angle head location type because these parameters can no longer be changed at the loaded angle head. It is also a good idea to display the angle head name in this window.

In the <NEWCROSSHEADCONFIGURATION> tag, specify the parameters that you wish to have displayed in the "New angle head" dialog.

Entries	Meaning
ItemX	Entry of the identifier of the angle head parameter. "X" specifies the order in which the parameters will be displayed in the window. The identifier of the angle head parameter can be found in Chapter: Identifiers of the angle head parameters (Page 318)

12.1.11 Configuring the "New Tool from Code Carrier" window

<NEWTOOLIDENTCONNECTIONCONFIGURATION> tag

You use the "New Tool from Code Carrier" dialog to identify a new tool before creating it. It is therefore a good idea to display the tool type and the tool name in this window. In addition, the same rules apply as for configuration of the window "New Tool," Section "Configuring the "New Tool" window (Page 334)."

In the <NEWTOOLIDENTCONNECTIONCONFIGURATION>tag, define the parameters that you wish to have displayed in the "New Tool of Code Carrier" dialog.

Entries	Meaning
ItemX	Entry of the identifier of the tool parameter. "X" specifies the order in which the parameters are displayed in the window. The identifier of the tool parameter can be found in Chapter: Tool parameter identifiers (Page 304)

12.1.12 Configuring the "New Tool from File" window

<NEWTOOLSAVETOOLDATACONFIGURATION> tag

You use the "New Tool from File" dialog to identify a new tool before creating it. It is therefore a good idea to display the tool type and the tool name in this window. In addition, the same rules apply as for configuration of the window "New Tool," Section "Configuring the "New Tool" window (Page 334)."

Specify the parameters in the <NEWTOOLSAVETOOLDATACONFIGURATION>tag that are to be displayed in the "New Tool from Tile" dialog.

Entries	Meaning
ItemX	Entry of the identifier of the tool parameter. "X" specifies the order in which the parameters are displayed in the window. The identifier of the tool parameter can be found in Chapter: Tool parameter identifiers (Page 304)

12.1.13 Configuring the "New tool from TTD file" window

<NEWTOOLTTDCREATETOOLCONFIGURATION> tag

You use the "New tool from TTD file" dialog to supply a new tool with specific tool data before creating the tool or before loading the tool to a magazine location. These are typically the tool size and/or the tool location type because these parameters can no longer be changed when the tool is loaded. It is also useful to display the tool type and the tool names in this window.

Specify the parameters in the <NEWTOOLTTDCREATETOOLCONFIGURATION> tag that are to be displayed in the "New tool from TTD file" dialog.

Entries	Meaning
ItemX	Entry of the identifier of the tool parameter. "X" specifies the order in which the parameters will be displayed in the window. The identifier of the tool parameter can be found in Chapter: Tool parameter identifiers (Page 304)

12.1.14 Configuring default values for new tools

<NEWTOOLDEFAULTVALUECONFIGURATION> tag

In the <NEWTOOLDEFAULTVALUECONFIGURATION> tag, you can specify default values for tools that you create via the user interface. These values can be configured depending on the tool type and, if required, also depending on the cutting edge position:

Entries	Meaning
TOOLTYPE_ALL	The parameter setting applies to all tool types.
TOOLTYPE_XXX	The parameter setting applies to a specific tool type. XXX stands for the number of the tool type.
TOANO_X	X stands for the number of the TO area. This entry can be used to make TO area-specific entries.
CUTTEDGEPOS_XXX	The parameter setting applies to a specific cutting edge position. XXX stands for the number of the cutting edge position.
Default	Default value of the parameter.

Example

```
<NEWTOOLDEFAULTVALUECONFIGURATION>
    <!-- Initialize geometry length 1 for all tool types to 70.0 -->
    <TOOLTYPE_ALL>
        <GeoLengthGeoAx1>
            <Default value="70.0" type="double" />
        </GeoLengthGeoAx1>
    </TOOLTYPE_ALL>
    <!-- Initialize tip angle of the twist drill to 120.0 -->
    <TOOLTYPE_200>
        <NoseAngle>
            <Default value="120.0" type="double" />
        </NoseAngle>
    </TOOLTYPE_200>
    <TOOLTYPE_500>
        <!-- Initialize plate angle of the roughing tool to 82.0 -->
            <PlateAngle>
```

```
            <Default value="82.0" type="double" />
        </PlateAngle>
        <!-- Initialize holder angle of the roughing tool depending on the cutting edge position -->
        <HolderAngle>
            <CUTTEDGEPOS_1>
                <Default value="93.0" type="double" />
            </CUTTEDGEPOS_1>
            <CUTTEDGEPOS_2>
                <Default value="93.0" type="double" />
            </CUTTEDGEPOS_2>
            <CUTTEDGEPOS_3>
                <Default value="93.0" type="double" />
            </CUTTEDGEPOS_3>
            <CUTTEDGEPOS_4>
                <Default value="93.0" type="double" />
            </CUTTEDGEPOS_4>
            <CUTTEDGEPOS_5>
                <Default value="49.0" type="double" />
            </CUTTEDGEPOS_5>
            <CUTTEDGEPOS_6>
                <Default value="49.0" type="double" />
            </CUTTEDGEPOS_6>
            <CUTTEDGEPOS_7>
                <Default value="49.0" type="double" />
            </CUTTEDGEPOS_7>
            <CUTTEDGEPOS_8>
                <Default value="49.0" type="double" />
            </CUTTEDGEPOS_8>
        </HolderAngle>
    </TOOLTYPE_500>
    <!-- Initialize coolant 2 of the plunge cutter to switched on -->
    <TOOLTYPE_520>
        <Coolant2>
            <Default value="true" type="bool" />
        </Coolant2>
    </TOOLTYPE_520>
</NEWTOOLDEFAULTVALUECONFIGURATION>
```

Example - initialize spindle rotation direction, cutting edge orientation vector and cutting edge normal vector for Y turning tools

Program code

```
<NEWTOOLDEFAULTVALUECONFIGURATION>
    <TOOLTYPE_505>
        <SpindleDirection>
            <Default value="512" type="int" />
        </SpindleDirection>
        <OrientVGeoAx1>
            <Default value="-1.0" type="double" />
        </OrientVGeoAx1>
        <OrientVGeoAx2>
            <Default value="0.0" type="double" />
        </OrientVGeoAx2>
        <OrientVGeoAx3>
            <Default value="0.0" type="double" />
        </OrientVGeoAx3>
        <OrientNormalVGeoAx1>
            <Default value="0.0" type="double" />
        </OrientNormalVGeoAx1>
        <OrientNormalVGeoAx2>
            <Default value="1.0" type="double" />
        </OrientNormalVGeoAx2>
        <OrientNormalVGeoAx3>
            <Default value="0.0" type="double" />
        </OrientNormalVGeoAx3>
    </TOOLTYPE_505>
    <TOOLTYPE_515>
        <SpindleDirection>
            <Default value="512" type="int" />
        </SpindleDirection>
        <OrientVGeoAx1>
            <Default value="-1.0" type="double" />
        </OrientVGeoAx1>
        <OrientVGeoAx2>
            <Default value="0.0" type="double" />
        </OrientVGeoAx2>
        <OrientVGeoAx3>
            <Default value="0.0" type="double" />
        </OrientVGeoAx3>
        <OrientNormalVGeoAx1>
            <Default value="0.0" type="double" />
        </OrientNormalVGeoAx1>
        <OrientNormalVGeoAx2>
            <Default value="1.0" type="double" />
        </OrientNormalVGeoAx2>
        <OrientNormalVGeoAx3>
            <Default value="0.0" type="double" />
        </OrientNormalVGeoAx3>
    </TOOLTYPE_515>
    <TOOLTYPE_525>
        <SpindleDirection>
            <Default value="512" type="int" />
        </SpindleDirection>
        <OrientVGeoAx1>
```

Program code

```
                    <Default value="-1.0" type="double" />
                </OrientVGeoAx1>
                <OrientVGeoAx2>
                    <Default value="0.0" type="double" />
                </OrientVGeoAx2>
                <OrientVGeoAx3>
                    <Default value="0.0" type="double" />
                </OrientVGeoAx3>
                <OrientNormalVGeoAx1>
                    <Default value="0.0" type="double" />
                </OrientNormalVGeoAx1>
                <OrientNormalVGeoAx2>
                    <Default value="1.0" type="double" />
                </OrientNormalVGeoAx2>
                <OrientNormalVGeoAx3>
                    <Default value="0.0" type="double" />
                </OrientNormalVGeoAx3>
            </TOOLTYPE_525>
            <TOOLTYPE_535>
                <SpindleDirection>
                    <Default value="512" type="int" />
                </SpindleDirection>
                <OrientVGeoAx1>
                    <Default value="-1.0" type="double" />
                </OrientVGeoAx1>
                <OrientVGeoAx2>
                    <Default value="0.0" type="double" />
                </OrientVGeoAx2>
                <OrientVGeoAx3>
                    <Default value="0.0" type="double" />
                </OrientVGeoAx3>
                <OrientNormalVGeoAx1>
                    <Default value="0.0" type="double" />
                </OrientNormalVGeoAx1>
                <OrientNormalVGeoAx2>
                    <Default value="1.0" type="double" />
                </OrientNormalVGeoAx2>
                <OrientNormalVGeoAx3>
                    <Default value="0.0" type="double" />
                </OrientNormalVGeoAx3>
            </TOOLTYPE_535>
        </NEWTOOLDEFAULTVALUECONFIGURATION>
```

Specifically initializing the direction of spindle rotation for the Y roughing tool TO area

```
<NEWTOOLDEFAULTVALUECONFIGURATION>
     <TOOLTYPE_505>
          <TOANO_1>
              <SpindleDirection>
                     <Defaultvalue="256"type="int" />
              </SpindleDirection>
          </TOANO_1>
```

```
            <TOANO_2>
                <SpindleDirection>
                        <Default value="512"type="int" />
                </SpindleDirection>
            </TOANO_2>
        </TOOLTYPE_505>
</NEWTOOLDEFAULTVALUECONFIGURATION>
```

Default values when creating new tools

System variable		Default value
Tool size		
Size on left	$TC_TP3	1
Size on right	$TC_TP4	1
Size above	$TC_TP5	1
Size below	$TC_TP6	1
Magazine location type of the tool		
Created outside of the magazine	$TC_TP7	1
Created in a magazine location	$TC_TP7	Location type of the magazine location
Tool status		
released	$TC_TP8	2

The following combinations between the cutting-edge position and the cutting direction result dependent on the tool type. When creating the new tool, the operator selects a combination in the "New Tool - Favorites" window.

Tool type $TC_DP1	Default values	
	Cutting edge $TC_DP2	Cutting direction $TC_DP11
Type 500 roughing tool	1	4
Type 505 Y roughing tool	2	3
Type 510 finishing tool	3	3
Type 515 Y finishing tool	4	4
Type 560 rotary drill	5	1
Type 570 Prime A	6	3
Type 571 Prime B	7	1
	8	3
Type 520 plunge cutter	1	2
Type 521 inclined plunge cutter	1	4
Type 525 Y plunge cutter	4	4
Type 530 cut-off tool	3	1
Type 531 inclined cutting tool	4	1
Type 535 Y cutting tool	2	3
Grinding tool types	3	3
	2	2

Tool type $TC_DP1	Default values	
	Cutting edge $TC_DP2	Cutting direction $TC_DP11
Type 540 threading tool	5	2
Type 541 inclined threading tool	5	1
	8	3
	8	4
	6	3
	6	4
	7	2
	7	1

If the ShopMill/ShopTurn option is set, the **direction of spindle rotation** is preassigned:

Tool type $TC_DP1	Default value $TC_DP25	Direction of spindle rotation
Measuring tools		
Type 580/585/710/712/713/725	Bit 8 = 0 Bit 9 = 0	Spindle stop
Turning tools dependent on SD54215 TM_FUNCTION_MASK_SET bit 1		
SD54215 bit 1 = 0	Bit 8 = 1 Bit 9 = 0	Spindle right
SD54215 bit 1 = 1	Bit 8 = 0 Bit 9 = 1	Spindle left
Special tools		
Type 730/731/732	Bit 8 = 0 Bit 9 = 0	Spindle stop
Type 505/515/525/535	Bit 8 = 0 and bit 9 = 1	Spindle left
All other tool types		
-	Bit 8 = 1 Bit 9 = 0	Spindle right

Tool type $TC_DP1	Default value $TC_DP25 for switching direction
Type 712 mono probe	Bit 17 = 1 Bit 20 = 1
Type 713 L probe	Bit 17 = 1 Bit 21 = 1
Type 714 star probe	Bit 16 = 1 Bit 17 = 1 Bit 18 = 1 Bit 19 = 1

Tool type $TC_DP1	Default value $TC_DP24 for tip angle
Type 200 twist drill	118
Type 220 centering tool	90

The plate angle rather than the clearance angle is displayed on the screen:
plate angle = 180 - holder angle - clearance angle

Tool type $TC_DP1	Default values		
	Cutting edge position $TC_DP2	Holder angle $TC_DP10	Clearance angle $TC_DP24
Type 500 roughing tool	1 ... 4	95	5
Type 505 Y roughing tool	5 ... 8	50	50
Type 510 finishing tool	1 ... 4	93	32
Type 515 Y finishing tool	5 ... 8	62.5	62.5
Type 560 rotary drill	-	90	2
Type 570 Prime A	1 ... 4	115	30
Type 570 Prime A	5 ... 8	72.5	72.5
Type 571 Prime B	1 ... 4	95	5
Type 571 Prime B	5 ... 8	50	50

Tool type $TC_DP1	Measurement unit	Default value $TC_DP8 for plate length
Type 500 roughing tool	metric	11
Type 505 Y roughing tool Type 510 finishing tool Type 515 Y finishing tool	inch	0.4

For all grinding tools, the parameter for **calculating the radius**:
$TC_TPG9 = 3 (corresponds to length 1) applies

Tool type $TC_DP1	Measurement unit	Default value $TC_DP6 for radius
Type 570 Prime A	metric	0.4
	inch	0.4/25.4
Type 571 Prime B	metric	0.8
	inch	0.4/25.4

Tool type $TC_DP1	Measurement unit	Default value $TC_DP8 for inside circle
Type 570 Prime A	metric	11
	inch	11/25.4
Type 571 Prime B	metric	12
	inch	12/25.4

For all inclined turning tools (type 521, 531, 541), the following applies to the inclined position parameter:

$TC_DP19 = 45

For all measuring tools (type 580, 710, 712, 713, 714), the following applies to the calibration data parameter:

$TC_DPNT = 1 (if MD54740 Bit5 is set)

Basic orientation for Y-turning tools

The basic orientation for Y turning tools is preset with the following default values.

System variable		Value
Tool type	$TC_DP1	Type 505/515/525/535 - Y-turning tools
Cutting edge orientation vector X	$TC_DPV3	-1.0
Cutting edge orientation vector Y	$TC_DPV4	0.0
Cutting edge orientation vector Z	$TC_DPV5	0.0
Cutting edge normal vector X	$TC_DPVN3	0.0
Cutting edge normal vector Y	$TC_DPVN4	1.0
Cutting edge normal vector Z	$TC_DPVN5	0.0

12.1.15 Configuring the "Details" window

<DETAILSCONFIGURATION> tag

In the tag <DETAILSCONFIGURATION>, you can make the settings for the "Details" window:

Entries	Meaning
ShowToolNumber	The tool number is displayed in the upper right field of the "Details" window.
UseAxisNameInLength	The tool length is displayed with axis identifiers, for example, Length X instead of Length 1. If no data is specified, then the setting is taken from the tool lists.
ShowBaseLength	The basic length is displayed in the "Details" window (AdaptLength1, AdaptLength2 and AdaptLength3). If no data is specified, then the setting is taken from the tool lists.
ShowYAxis	The Y axis is displayed in the "Details" window. If no data is specified, then the setting is taken from the tool lists.

Entries	Meaning	
AutoDisplayAxisEnabled	`true`	Default setting The selection of the displayed tool lengths is based on the display of the tool lengths in the tool list.
	`false`	Several components of the tool length are displayed, even if only one tool length is displayed in the tool list.
OrderOfAxis	The order of display of the axes for all tool types in the window is "Details" > "Cutting edge data". The possible values are XYZ, XZY, YXZ, YZX, ZXY and ZYX.	

Example 1

```
<DETAILSCONFIGURATION>
    <ShowToolNumber value="true" type="bool" />
</DETAILSCONFIGURATION>
```

Example 2

```
<DETAILSCONFIGURATION>
    <OrderOfAxis value="ZYX" type="QString" />
</DETAILSCONFIGURATION>
```

For all of the tools, the order of display in the details dialog is ZYX.

Example 3

If the entry `OrderOfAxis` is used as a tag, the order of display of the axes can be defined to be tool type-specific.

```
<DETAILSCONFIGURATION>
    <OrderOfAxis>
      <TOOLTYPE_130>
        <Item value="ZYX" type="QString" />
      </TOOLTYPE_130>
      <TOOLTYPE_131>
        <Item value="ZYX" type="QString" />
      </TOOLTYPE_131>
    </OrderOfAxis>
</DETAILSCONFIGURATION>
```

For angle heads (type 130 and 131), the order of display of the axes in the details dialog is ZYX.

Example 4

```
<DETAILSCONFIGURATION>
    <OrderOfAxis>
      <TOOLTYPE_130>
```

```
        <Item value="ZYX" type="QString" />
      </TOOLTYPE_130>
      <TOOLTYPE_131>
        <Item value="ZYX" type="QString" />
      </TOOLTYPE_131>
      <TOOLTYPE_ALL_500>
        <Item value="ZXY" type="QString" />
      </TOOLTYPE_ALL_500>
      <TOOLTYPE_ALL>
        <Item value="XYZ" type="QString" />
      </TOOLTYPE_ALL>
    </OrderOfAxis>
</DETAILSCONFIGURATION>
```

For angle heads (type 130 and 131), the order of display of the axes in the details dialog is ZYX.

For all rotary tools (type 500 and ff.), the order of display of the axes in the details dialog is ZXY.

For all remaining tools, the order of display of the axes in the details dialog is XYZ.

12.1.16 Assigning a name for magazines and magazine locations

<MAGAZINEPLACENAMECONFIGURATION> tag

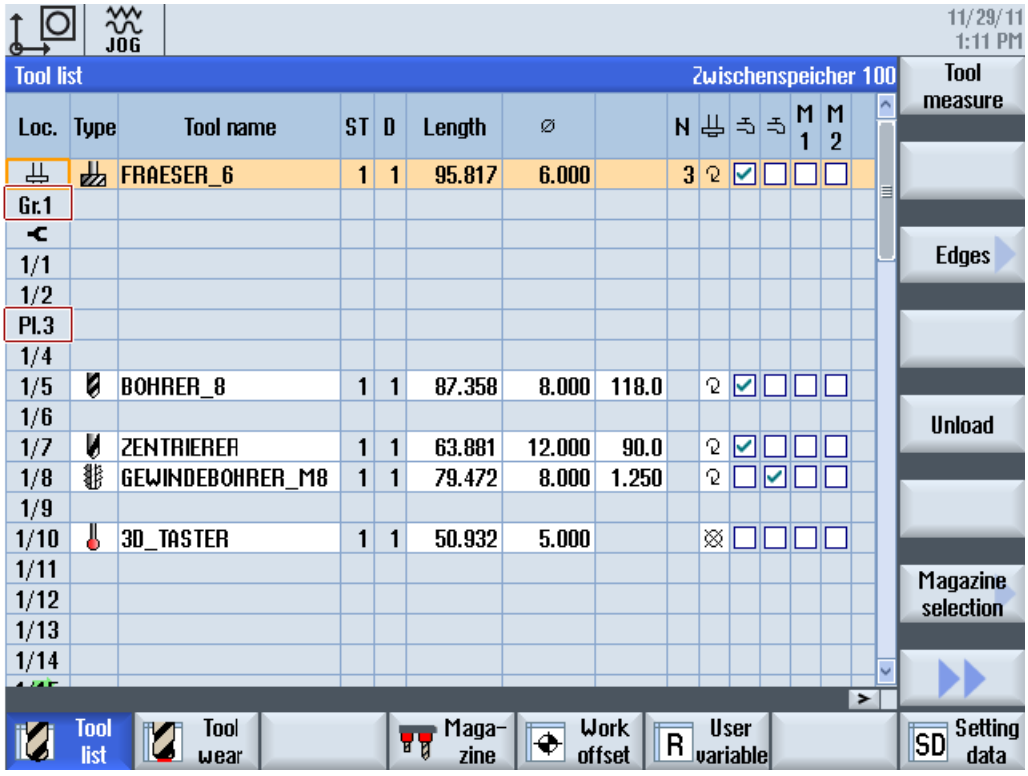
The entries to assign dedicated magazine location names are located in the <MAGAZINEPLACENAMECONFIGURATION> tag. Magazine locations can be displayed with texts. Instead of the identifier, e.g. 1/5 for the first magazine, location five, the configured text is then output in the tool lists.

Entries	Meaning
MAGAZINE_XXX	XXX stands for the magazine number.
PLACE_XXX	XXX stands for the location number.
Name	Text identifier of the magazine location.

Example

```
<MAGAZINEPLACENAMECONFIGURATION>
    <MAGAZINE_9998>
       <PLACE_2>
          <Name value="TM_SPECIAL_MAG_PLACE_1" type="QString" />
       </PLACE_2>
    </MAGAZINE_9998>
    <MAGAZINE_1>
       <PLACE_3>
          <Name value="TM_SPECIAL_MAG_PLACE_2" type="QString" />
```

```
        </PLACE_3>
    </MAGAZINE_1>
</MAGAZINEPLACENAMECONFIGURATION>
```



Name for a magazine

Dedicated texts can also be allocated for magazines. The text or the text identifier is entered into the magazine configuration file (ini file) of the NC. The system variable for the magazine name is $TC_MAP2[magNo]. The magazine name is displayed at the top right in the tool lists.

Example

Magazine 1 should have the name "main magazine".

```
$TC_MAP2[1]="main magazine"
```

If the "main magazine" text is to be output language-dependent, then the language-dependent texts for the text identifier "main magazine" must be created.

See also

Please refer to the following section for the procedure to create language-dependent text:

Creating OEM texts (Page 367)

12.1.17 Configuring the "Assign Tooltips for Multitool Locations" window

<MULTITOOLPLACETOOLTIPCONFIGURATION> tag

You will find the entries for assigning tooltips in the tag <MULTITOOLPLACETOOLTIPCONFIGURATION>. This is particularly useful if all the multitools used on a turning machine have two locations and a fixed assignment between multitool locations and the main spindle or the counter spindle exists, for example, location 1 - main spindle location 2 - counter spindle. In this case, the tooltip for location 1 could be "Tool Holder for Main Spindle."

Entries	Meaning
PLACE_XXX	XXX stands for the multitool location number.
Tooltip	Text identifier of the magazine location.

Example

```
<MULTITOOLPLACETOOLTIPCONFIGURATION>
        <PLACE_1>
            <ToolTip value="TM_HOLDER_OF_MAIN_SPINDLE" type="QString" />
        </PLACE_1>
        <PLACE_2>
            <ToolTip value="TM_HOLDER_OF_COUNTER_SPINDLE" type="QString" />
        </PLACE_2>
</MULTITOOLPLACETOOLTIPCONFIGURATION>
```

See also

Please refer to the following section for the procedure to create language-dependent text:

Creating OEM texts (Page 367)

12.1.18 Assigning a name for magazine location types

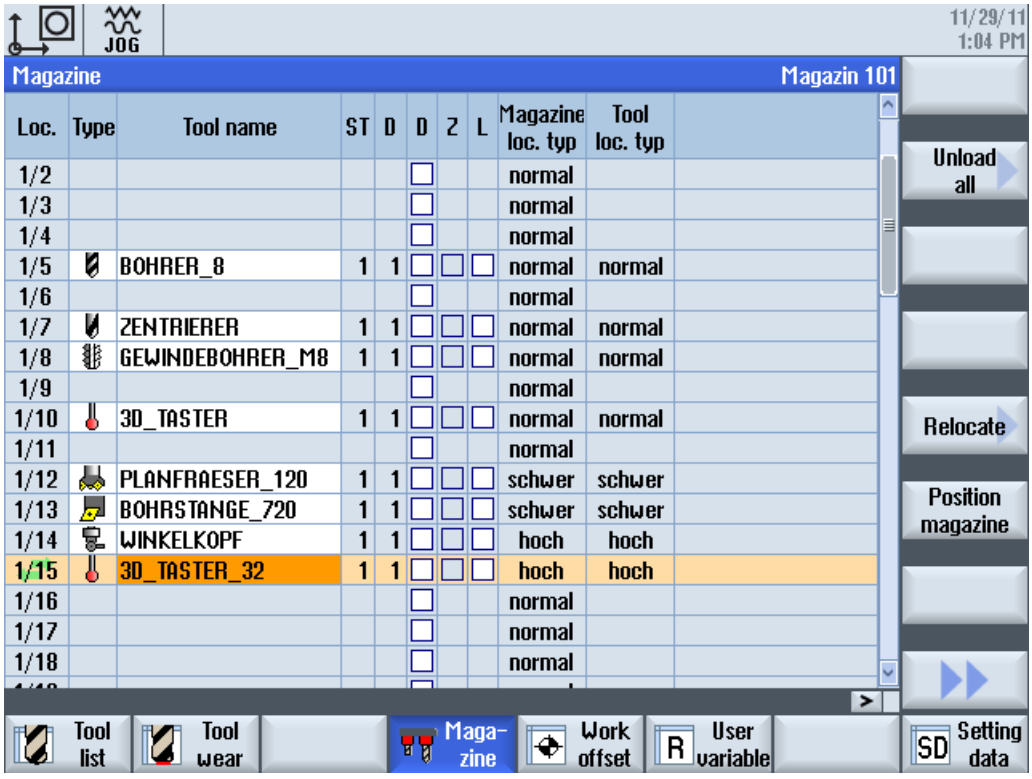
<PLACETYPECONFIGURATION> tag

You can find entries to allocate your own location type identifiers in the tag <PLACETYPECONFIGURATION>. Magazines and magazine location types can be displayed with texts in the tool lists. Instead of identifier 2, e.g. for location type 2, the configured text is then output in the tool lists.

Entries	Meaning
PLACETYPEXXX	XXX stands for the number of the location type.
Text	Text identifier of the location type.
Tooltip	Tooltip text identifier of the location type.

Example

```
<PLACETYPECONFIGURATION>
     <PLACETYPE1>
        <Text value="TM_PLACETYPE_1" type="QString" />
        <Tooltip value="TM_TT_PLACETYPE_1" type="QString" />
     </PLACETYPE1>
     <PLACETYPE2>
        <Text value="TM_PLACETYPE_2" type="QString" />
        <Tooltip value="TM_TT_PLACETYPE_2" type="QString" />
     </PLACETYPE2>
     <PLACETYPE3>
        <Text value="TM_PLACETYPE_3" type="QString" />
        <Tooltip value="TM_TT_PLACETYPE_3" type="QString" />
     </PLACETYPE3>
</PLACETYPECONFIGURATION>
```



See also

Please refer to the following section for the procedure to create language-dependent text:

Creating OEM texts (Page 367)

12.1.19 Assignment of magazines and magazine locations to channels

Requirement

Turning technology or milling technology and technology extension turning have been set:
Name of the associated configuration file sltmturninglistconfig.xml or sltmmillturnlistconfig.xml.

Assignment between magazine and magazine locations and channel

With this setting, magazines and magazine locations can be assigned to channels. This makes sense if the machine is configured so that several channels are assigned to one tool area (TOA area). If each magazine is only used in one channel, it is possible to define this assignment between the magazine or magazine locations and the channel here.

As a consequence of this configuration, the tool icons are output with reference to their position depending on the channel. Every channel can be assigned a position using the channel machine data MD52000 MCS_DISP_COORDINATE_SYSTEM. If a tool is at a magazine location which is assigned a channel via this configuration, then the tool icon is output depending on this position that has been set.

This setting option is only available in turning technology and in milling technology with the technology extension milling. This can be especially helpful if, on a multi-channel machine, a magazine is used before the turning center and another magazine after the turning center. This means that the tools can be entered into the tool list in a position visible for the operator.

The assignment between magazine locations and channels is useful when instead of two magazines, two tool holders/tool spindles exist that fetch their tools from a magazine. One tool holder is used in front of the turning center and one is used behind the turning center, so that different views onto the tool result.

<CHANNELMAGAZINEASSIGNMENT> tag

In tag <CHANNELMAGAZINEASSIGNMENT> , you will find the entries for the assignment between the magazines or magazine locations and the channels.

Entries	Meaning
CHANNELNO_XXX	XXX stands for the channel number.
Magazines	Number of magazines that are assigned to the channel.
MAGAZINE_XXX	Number of the magazine whose locations will be assigned to the channel.
Places	Number of magazine locations that will be assigned to the channel.

Example 1

The following assignment between magazine and channel should apply:

Magazine 1-3	Channel 1
Magazine 4	Channel 2
Magazine 5	Channel 3
Magazine 6	Channel 4

```
<CHANNELMAGAZINEASSIGNMENT>
   <CHANNELNO_1>
      <Magazines value="1 2 3" type="QString" />
   </CHANNELNO_1>
   <CHANNELNO_2>
      <Magazines value="4" type="QString" />
   </CHANNELNO_2>
   <CHANNELNO_3>
      <Magazines value="5" type="QString" />
   </CHANNELNO_3>
   <CHANNELNO_4>
      <Magazines value="6" type="QString" />
   </CHANNELNO_4>
</CHANNELMAGAZINEASSIGNMENT>
```

Example 2

Magazine location 1 of the buffer will be assigned to channel 1.

Magazine location 2 of the buffer will be assigned to channel 2.

```
<CHANNELMAGAZINEASSIGNMENT>
   <CHANNELNO_1>
      <MAGAZINE_9998>
         <Places value="1" type="QString" />
      </MAGAZINE_9998>
   </CHANNELNO_1>
   <CHANNELNO_2>
      <MAGAZINE_9998>
         <Places value="2" type="QString" />
      </MAGAZINE_9998>
   </CHANNELNO_2>
</CHANNELMAGAZINEASSIGNMENT>
```

12.1.20 Assignment between magazines and magazine locations and coordinate systems

Requirement

Turning technology or milling technology and technology extension turning have been set: Name of the associated configuration file sltmturninglistconfig.xml or sltmmillturnlistconfig.xml.

Assignment between the magazine or magazine locations and the coordinate system

With this setting, magazines and magazine locations can be assigned to coordinate systems. This is useful if the machine is configured in such a way that a magazine or magazine location does not match the position of the coordinate system set in the channel. The assignment between magazine or magazine location and coordinate system can be defined here.

As a consequence of this configuration, the tool icons are output with reference to their position depending on the coordinate system.

This setting option is only available in turning technology and in milling technology with the technology extension milling.

<COORDSYSMAGAZINEASSIGNMENT> tag

In tag <COORDSYSMAGAZINEASSIGNMENT>, you will find the entries for the assignment between magazines or magazine locations and coordinate systems.

Entries	Meaning
COORDSYSNO_XXX	XXX stands for the number of the coordinate system.
Magazines	Number of the magazines that are assigned to the channel.
MAGAZINE_XXX	Number of the magazine whose locations will be assigned to the coordinate system.
Places	Number of magazine locations that will be assigned to the coordinate system.

Example

The following assignment between magazines or magazine locations and coordinate systems is to apply:

Magazine 1,3, and 4	Coordinate system 34
Magazine 2	Coordinate system 33
Magazine 9998 location 2	Coordinate system 33

```
<COORDSYSMAGAZINEASSIGNMENT>
   <COORDSYSNO_34>
      <Magazines value="1 3 4" type="QString" />
   </COORDSYSNO_34>
   <COORDSYSNO_33>
      <Magazines value="2" type="QString" />
      <MAGAZINE_9998>
         <Places value="2" type="QString" />
      </MAGAZINE_9998>
   </COORDSYSNO_33>
</COORDSYSMAGAZINEASSIGNMENT>
```

12.1.21 Disabling the softkey depending on filters

Tag <FILTERCONFIGURATION> in combination with Tag <SOFTKEYLOCK>

Some softkeys can be hidden depending on filter settings. For example, the "New Tool" softkey is hidden if the filter "Only disabled tools" is set. This is not a good idea because the new tool is not created as a disabled tool and is thus directly filtered out of the display. This means that the new tool would not be displayed after being created.

Softkey identifiers of the softkey, which can be hidden:

Entries	Meaning
NewTool	New tool
Edges	Cutting edges

Identifiers for filter settings that can be reacted to:

Entries	Meaning
FirstEdge	Filter "Only first cutting edge," default setting "Edges"
StandBy	Filter "Only tools ready for use", default setting "NewTool"
Active	Filter "Only tools labeled as active", default setting "NewTool"
Locked	Filter "Only locked tools", default setting "NewTool"
Prewarning	Filter "Only tools with prewarning limit reached", default setting "NewTool"
RemainingPieces	Filter "Only tools with remaining pieces", default setting "NewTool Edges"
RemainingTime	Filter "Only tools with remaining time", default setting "NewTool Edges"
MarkedToUnload	Filter "Only tools with marked to unload", default setting "NewTool Edges"
MarkedToLoad	Filter "Only tools with marked to load", default setting "NewTool Edges"

Example

The softkeys "New tool" and "Edges" should be hidden if the "Only locked tools" filter is activated.

```
<FILTERCONFIGURATION>
   <SOFTKEYLOCK>
      <Locked value="NewTool Edges" type="QString" />
   </SOFTKEYLOCK>
</FILTERCONFIGURATION>
```

12.1.22 Configuring the "Load Station Selection" window

<LOADSTATIONSHANDLEALLTOOLS> tag

For functions "Delete all," "Unload all," and "Load all," you must specify a load station if multiple load stations are configured in the magazine configuration of the NC. The load station is defined in a dialog called "Select Load Station" or via the load station configuration with tag <LOADSTATIONSHANDLEALLTOOLS>.

Entries	Meaning
DialogEnabled	Offer window "Load Station Selection," default setting "true"
MAGAZINE_XXX	Number of the magazine to which a load station will be assigned.
LoadStation	Load station to be assigned to the magazine.

Example

```
<LOADSTATIONSHANDLEALLTOOLS>
   <DialogEnabled value="true" type="bool" />
   <MAGAZINE_1>
      <LoadStation value="2" type="uint" />
   </MAGAZINE_1>
   <MAGAZINE_2>
      <LoadStation value="1" type="uint" />
   </MAGAZINE_2>
</LOADSTATIONSHANDLEALLTOOLS>
```

12.1.23 Coolant and tool-specific functions

Assigning coolant (ShopMill/ShopTurn)

In the tool management of ShopMill/ShopTurn, you can assign coolant and tool-specific functions to each tool. These are activated when the tool is changed. You assign the coolant to the corresponding machine functions (M functions) using the following machine data:

MD52230 $MCS_M_CODE_ALL_COOLANTS_OFF	M code for all coolants OFF
MD52231 $MCS_M_CODE_COOLANT_1_ON	M code for coolant 1 ON
MD52232 $MCS_M_CODE_COOLANT_2_ON	M code for coolant 2 ON
MD52233 $MCS_M_CODE_COOLANT_1_AND_2_ON	M code for both coolants ON

Tool-specific functions (ShopMill)

Tool-specific functions 1 ... 4 are intended for other M functions that can be activated for a tool. For example, third coolant, speed monitoring, tool breakage etc.

Up to 8 M functions can be programmed for the activation or deactivation of the tool-specific functions. Definition is realized using the following machine data:

MD52281 $MCS_TOOL_MCODE_FUNC_ON[0..3]	M code for tool-specific function ON
= - 1	

MD52282 $MCS_TOOL_MCODE_FUNC_OFF[0..3]	M code for tool-specific function OFF
= - 1	

The M functions for the tool-specific functions are executed by the ShopMill cycles at the tool change. If the checkbox is activated, the corresponding M code is generated from MD52281. If the checkbox is not activated, the corresponding M code is generated from MD52282.

If you do not want to automatically execute additional M commands on a tool change, set the associated machine data to -1 for the M commands of this tool-specific function. The associated fields are also hidden in the tool list.

The default value of the machine data is -1 so that no M values are output for the corresponding functions.

Technical procedure

The M functions for the tool-specific functions are generated by the ShopMill cycles in a set (up to 4 M commands are possible).

The number and also the sequence of the M commands are dependent on the settings of the machine data MD52281 to MD52282 and the programming.

If M functions less than 100 are used, the number and sequence of the M commands has no significance. They are available in the user PLC in the decoded area (DB21.DBB194..206), independently of the quantity and the sequence when programming.

However, if M functions from 100 and higher are used, then you must decode these yourself in the user PLC. In this case, you must take into consideration the number and sequence (M function 1 to M function 4).

JOG mode

In the JOG mode, no coolant and tool-specific functions are output to the interface. For a tool change in the JOG mode, these functions can be initiated by the operator using the keys on the machine control panel (implementation via the PLC user program).

Example: Machine data settings

MD52281 $MCS_TOOL_MCODE_FUNC_ON[0]	= 90
MD52281 $MCS_TOOL_MCODE_FUNC_ON[1]	= 92
MD52281 $MCS_TOOL_MCODE_FUNC_ON[2]	= 94
MD52281 $MCS_TOOL_MCODE_FUNC_ON[3]	= 96

MD52282 $MCS_TOOL_MCODE_FUNC_OFF[0]	= 91
MD52282 $MCS_TOOL_MCODE_FUNC_OFF[1]	= - 1
MD52282 $MCS_TOOL_MCODE_FUNC_OFF[2]	= 95
MD52282 $MCS_TOOL_MCODE_FUNC_OFF[3]	= 97

As a result, fields 1 to 4 of the tool-specific functions are displayed in the tool list.

The following fields are activated:

Tool-specific function 1:	On
Tool-specific function 2:	Off
Tool-specific function 3:	Off
Tool-specific function 4:	Off

The M functions M90, M95 and M97 are generated by the ShopMill cycles at the tool change.

You can change the texts for "Coolant 1/2 on/off" and "Tool-spec. fct. 1, ..." (see section Identifiers of the standard texts (Page 368)). So that the texts between the tool list and program area machine functions match, please note the following section.

Creating user-specific texts for the "Machine Functions" window.

If you require an additional possibility of switching on or switching off coolant and tool-specific functions, then you can program this in the "Program" operating area using the input window "Machine Functions". You can create the texts and the tooltips for the tool-specific functions which are used in this window.

Define the texts for the M functions in the file "slstepforms_xxx.ts".

If you require the texts in additional languages, then you must create a separate file for each language. The only difference in the files is that the file name has the language code "xxx". Use only the specified language codes.

More information
Commissioning Manual SINUMERIK Operate; Supported languages

File "slstepforms_xxx.ts "

Tag	Meaning
source	Text identifier for label: SlStepLabels A maximum of up to 4 M functions can be defined: T_LAB_USER_MACHINE_FUNC_1 T_LAB_USER_MACHINE_FUNC_2 T_LAB_USER_MACHINE_FUNC_3 T_LAB_USER_MACHINE_FUNC_4
	Text identifier for tooltip: SlStepToolTip There are 3 tooltips for every machine function x (1 ... 4): T_TT_USER_MACHINE_FUNC_x: standard tooltip if the machine function was not selected. T_TT_USER_MACHINE_FUNC_x _ON: Tooltip for activated machine function T_TT_USER_MACHINE_FUNC_x_OFF: Tooltip for deactivated machine function
translation	Text, which is displayed at the user interface. Label texts: You can only enter texts on one line. The "lines" entry is not applicable. Tooltip: You can enter multi-line texts. A line break is set using the "%n" character.
chars	Text length The maximum text length is 30 characters.
lines	Number of lines The maximum number of lines is three.
remark	Own comment - entry is not displayed at the user interface.

Procedure

1. You can copy the sample file "oem_slstepforms_deu.ts" from the following directory: /**siemens**/sinumerik/hmi/template/lng.
2. Save or create the file in the /**oem**/sinumerik/hmi/lng or /**user**/sinumerik/hmi/lng directory.
3. Change the file name to "slstepforms_deu.ts".
   If you require texts for additional languages, then a separate file must be created for each language. Save the file with the appropriate language code in the file name.
4. Open the file and define the texts in the <message> and </message> areas.
5. Restart the system.
   The file must still be converted into a binary format in order that the texts are displayed during the program runtime. This data is only converted during power up.

Example of a file "slstepforms_deu.ts"

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE TS><TS>
<context>
   <name>SlStepLabels</name>
   <message>
      <source>T_LAB_USER_MACHINE_FUNC_1</source>
```

```
      <translation>tool-specific function 1</translation>
      <chars>20</chars>
   </message>
<context>
   <name>SlStepToolTip</name>
   <!-- tooltips of user machine function 1-->
   <message>
      <source>T_TT_USER_MACHINE_FUNC_1</source>
      <translation>User%nMachine functions 1</translation>
      <chars>25</chars>
      <lines>3</lines>
   </message>
</context
</TS>
```

12.1.24 Reason for the tool change when reactivating

Function

When reactivating a tool, you have the option of specifying the reason for the tool change, e.g. to use this information for statistical purposes. The "Reason for Tool Change" selection window is opened after pressing the "Reactivate" softkey. In a checkbox, you can select the reason for the tool change. The tool change is entered in a tool parameter to be configured or in a local variable of the SlCap service.

Tool change reasons

You can adapt or expand the tool change reasons as required. The operating software provides the following pre-defined texts for the configuration:

Text identifier	Text
TM_DGL_DESIRED_PIECES_ELAPSED	Unit quantity setpoint (desired pieces) elapsed
TM_DGL_TOOL_BREAKAGE	Tool breakage
TM_DGL_CUTTING_EDGE_DAMAGE	Cutting edge damage
TM_DGL_PREMATURE_WEAR	Premature wear
TM_DGL_TYPE_REEQUIPPING	Type reequipping
TM_DGL_TOOL_TRIAL	Tool trial
TM_DGL_PREVENTIVE_CHANGE	Preventive change

<TOOLCHANGEREASON> tag

You configure the function in the <TOOLCHANGEREASON> tag:

Parameter	Designation
Enabled	True - "Reason for tool change" selection window is displayed when reactivating.
	False - default value, selection window is not displayed.
OnlyOneReasonAllowed	It is only permissible to select one tool change reason.
UseCapLocalParameter	The reason for the tool change is entered into a local variable of the SlCap service.
REASONXXX	XXX stands for an internal number. The numbers for the tool change reason must be clear and unique.
Text	Text identifier of the tool change reason.
Parameter	Parameter identifier of the parameter in which it is entered whether the tool change reason was selected.
BitMask	Bit mask to enter the reason for the change in the local variable of the SlCap service. The bit mask is entered as integer value. Bit 0 → 1, bit 1 → 2, bit 2 → 4

Local variable of the SICap service

The identifier of the local variables is called "/Hmi/ToolEdgeReplacementReason". The variable is the string type.

The table below lists the most important arguments and their significance:

Argument	Meaning
datetime	Date and time of the event
containerNo	Number of the management area
toaNo	Number of the TO area
toolNo	Tool number
edgeNo	Cutting edge number
toolIdent	Tool name
toolDuplo	Replacement tool number
ncuName	Name of the NCU
reactivationReason	Reason for tool change
monitoringMode	Monitoring type
remainingBefore	Remaining unit quantity/tool lifetime before reactivation

Example 1

The following example uses the texts for the tool change reason, already prepared in the operating software. In this example, the tool change reason is stored in parameters that still have to be configured, e.g. the OEM tool data 1, bit 1 and following.

```
<TOOLCHANGEREASON>
   <Enabled value="true" type="bool" />
```

```
    <REASON1>
      <Text value="TM_DGL_DESIRED_PIECES_ELAPSED" type="QString" />
      <Parameter value="ToolUser_1_Bit1" type="QString" />
    </REASON1>
    <REASON2>
      <Text value="TM_DGL_TOOL_BREAKAGE" type="QString" />
      <Parameter value="ToolUser_1_Bit2" type="QString" />
    </REASON2>
    <REASON3>
      <Text value="TM_DGL_CUTTING_EDGE_DAMAGE" type="QString" />
      <Parameter value="ToolUser_1_Bit3" type="QString" />
    </REASON3>
    <REASON4>
      <Text value="TM_DGL_PREMATURE_WEAR" type="QString" />
      <Parameter value="ToolUser_1_Bit4" type="QString" />
    </REASON4>
    <REASON5>
      <Text value="TM_DGL_TYPE_REEQUIPPING" type="QString" />
      <Parameter value="ToolUser_1_Bit5" type="QString" />
    </REASON5>
    <REASON6>
      <Text value="TM_DGL_TOOL_TRIAL" type="QString" />
      <Parameter value="ToolUser_1_Bit6" type="QString" />
    </REASON6>
    <REASON7>
      <Text value="TM_DGL_PREVENTIVE_CHANGE" type="QString" />
      <Parameter value="ToolUser_1_Bit7" type="QString" />
    </REASON7>
</TOOLCHANGEREASON>
```

Example 2

Also in this example, the prepared texts for the reason for the change are used. It is only permitted to enter one reason for the change as a valid entry. The reason for the tool change is saved in the local variable of the SICap service.

```
<TOOLCHANGEREASON>
   <Enabled value="true" type="bool" />
    <OnlyOneReasonAllowed value="true" type="bool" />
    <UseCapLocalParameter value="true" type="bool" />
    <REASON1>
      <Text value="TM_DGL_DESIRED_PIECES_ELAPSED" type="QString" />
      <BitMask value="1" type="int" />
    </REASON1>
```

```
    <REASON2>
      <Text value="TM_DGL_TOOL_BREAKAGE" type="QString" />
      <BitMask value="2" type="int" />
    </REASON2>
    <REASON3>
      <Text value="TM_DGL_CUTTING_EDGE_DAMAGE" type="QString" />
      <BitMask value="4" type="int" />
    </REASON3>
    <REASON4>
      <Text value="TM_DGL_PREMATURE_WEAR" type="QString" />
      <BitMask value="8" type="int" />
    </REASON4>
    <REASON5>
      <Text value="TM_DGL_TYPE_REEQUIPPING" type="QString" />
      <BitMask value="16" type="int" />
    </REASON5>
    <REASON6>
      <Text value="TM_DGL_TOOL_TRIAL" type="QString" />
      <BitMask value="32" type="int" />
    </REASON6>
    <REASON7>
      <Text value="TM_DGL_PREVENTIVE_CHANGE" type="QString" />
      <BitMask value="64" type="int" />
    </REASON7>
</TOOLCHANGEREASON>
```

See also

For configuring new parameters, see Configuring list parameters (Page 320).

12.1.25 Configure the code carrier connection

Precondition



Software option
You require the "Tool Ident Connection" option for this function.

Tag <IDENTCONNECTIONCONFIGURATION>

The entries used for configuring the code carrier connection in the tool list of the user interface are located in the <IDENTCONNECTIONCONFIGURATION> tag. The code carrier connection is implemented via TDI Ident Connection.

The following functions in the tool list are available:

1. Creating tools for the code carrier:

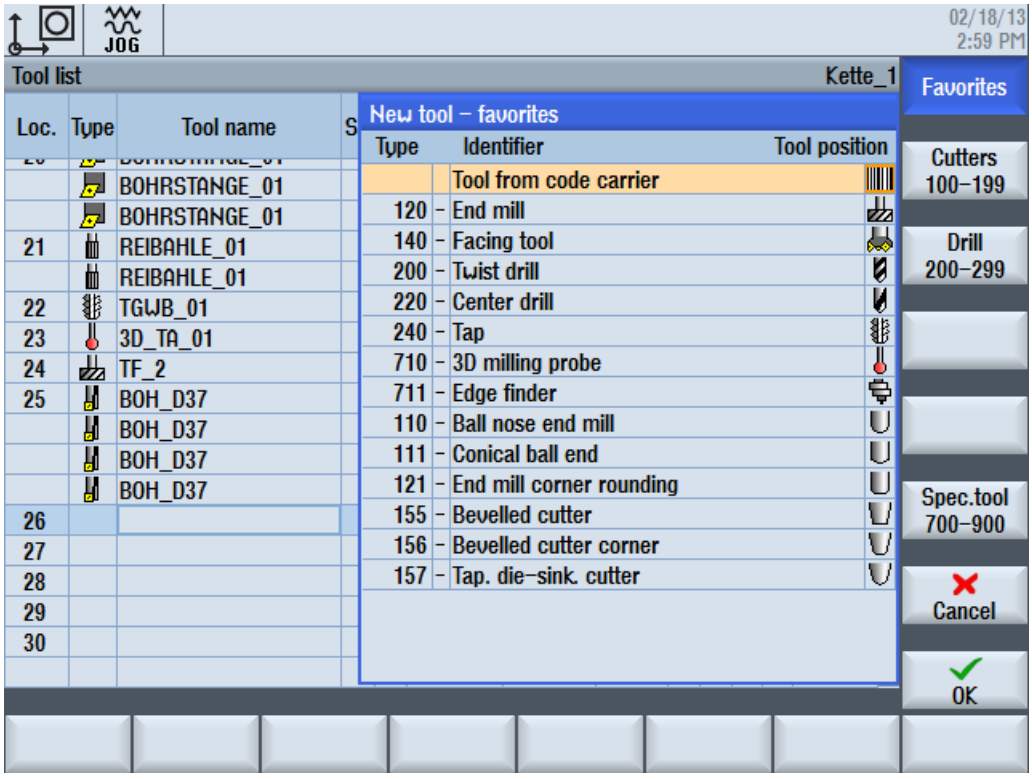


Figure 12-1 New tool from code carrier

   The data of the tool are read from the code carrier tool and displayed in the "New Tool" window. Click OK to create the tool in the NC and to initialize it with the code carrier data.

2. Unloading tools on the code carrier:
   The tool is unloaded and the data of the tool is written to the code carrier. The tool can also be deleted using a setting (see below).

3. Deleting the tool on the code carrier:
   The tool is unloaded and the data of the tool is written to the code carrier. The tool is then deleted from the NC. Using a setting (see below), the "On code carrier" softkey can also be hidden for deleting a tool.

The following settings can be made:

Entries	Meaning
Enabled	true - Code carrier connection is offered. false - Code carrier connection is not offered.
ShowInternalMessage	true - Internal messages for error analysis are also displayed. false - Default setting. No internal messages are displayed.

Entries	Meaning
UnloadWithoutIdentconnection	true - The "On code carrier" function is not offered during unloading. It is only displayed during "Delete tool." false - Default setting. The "On code carrier" function is offered during unloading.
UnloadWithDelete	true - The tool is deleted when unloaded to the code carrier. The "on code carrier" function is not offered for "Delete tool". false - Default setting. The tool is only unloaded for "Unload to code carrier" and remains in the NC memory.
OnlyInLoadStation	true - Default setting. Only loaded tools are stored "on code carrier"; new tools "from code carrier" can only be created directly in the magazine. false - Tools can be created independently of the "from code carrier" magazine slot or deleted "on code carrier".
CreateNewToolWithDialog	true - Default setting. Before creating the tool from code carrier, the tool data is displayed in the "New tool" window. false - The tool is created directly in the tool list.
ForceSisterToolNumber	true - The sister tool number specified by the code carrier data is retained when the tool is created. false - Default setting. The sister tool number is freely assigned.
UseDefaultData	true - The new tool is initialized with the configured default data before the code carrier data are applied (see tag <NEWTOOLDEFAULTVALUECONFIGURATION> Section "Configuring default values for new tools (Page 337)"). false - Default setting. The new tool is not initialized with the configured default data.
OrderFirstWriteThanUnload	true - During unloading/deletion, the data are first written to the code carrier and then the tool is unloaded or deleted. false - Default setting. During unloading/deletion the tool is first unloaded, then the data are written to the code carrier and finally, if requested, the tool is deleted.

Example

```
<IDENTCONNECTIONCONFIGURATION>
      <Enabled value="true" type="bool" />
      <ShowInternalMessage value="false" type="bool" />
      <UnloadWithDelete value="false" type="bool" />
      <OnlyInLoadStation value="true" type="bool" />
      <CreateNewToolWithDialog value="true" type="bool" />
</IDENTCONNECTIONCONFIGURATION>
```

More information

More information about tool management with code carriers can be found at Code carrier connection - Tool Ident Connection (option) (Page 385).

12.1.26 Configuring Save to file

<SAVETOOLDATACONFIGURATION> tag

The entries used for configuring the functions "Save tools to file" or "Create tools from file" are located in the <SAVETOOLDATACONFIGURATION> tag:

The following functions in the tool list are available:

1. Creating tools from file:

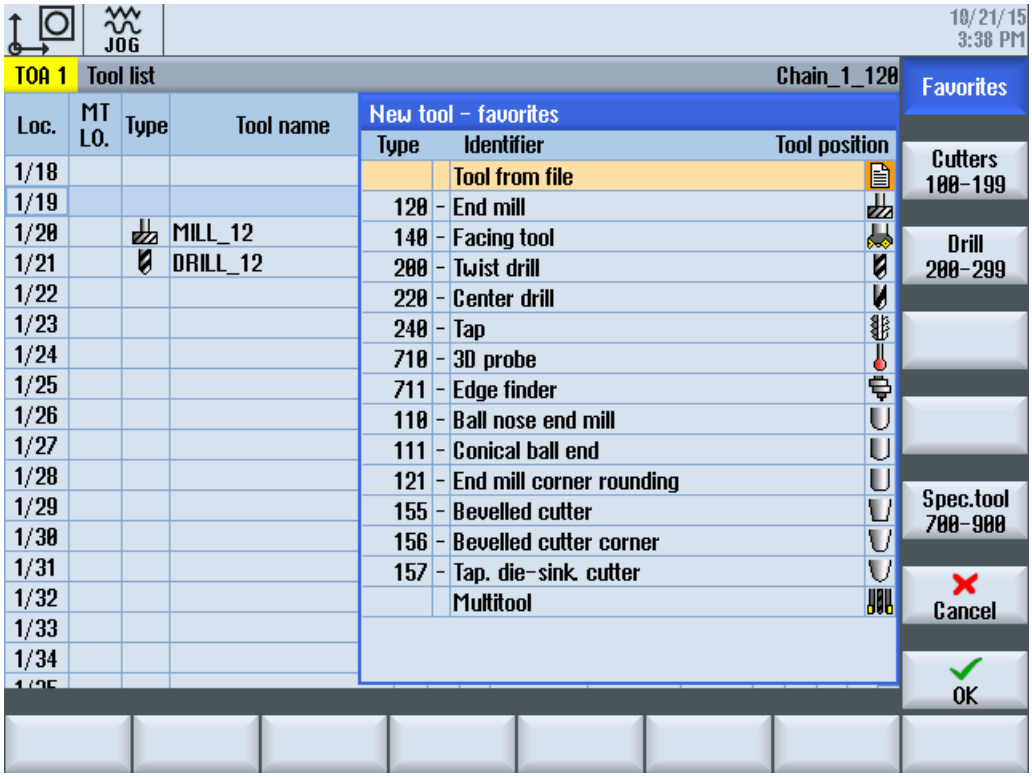


Figure 12-2 Tool from file

The data of the tool are read from a file and displayed in the "New Tool" window. Click OK to create the tool in the NC and to initialize it with the data from the file.

2. Unloading tools into a file:
   The tool is unloaded and the data of the tool are written to a file. The tool can also be deleted using a setting (see below).
3. Deleting the tool in a file:
   The tool is unloaded and the data of the tool are written to a file. The tool is then deleted from the NC. The softkey "In File" can be hidden while the tool is deleted by means of a setting (see below).

The following settings can be made:

Entries	Meaning
Enabled	true - Tool from/in file is offered. false - Tool from/in file is not offered.
UnloadWithDelete	true - The tool is deleted when unloaded to a file. The "In File" function is not offered for "Delete tool." false - Default setting, on Unload to file the tool is only unloaded and is kept in the NC memory.
CreateNewToolWithDialog	true - Default setting, before the tool is created from a file, the tool data is displayed in the "New Tool" window. false - The tool is created directly in the tool list.
ForceSisterToolNumber	true - The sister tool number specified by the code carrier data is kept when the tool is created. false - Default setting, the sister tool number is freely assigned.
UseDefaultData	true - The new tool is initialized with the configured default data before the data are taken over from the file (see tag <NEWTOOLDEFAULTVALUECONFIGURATION> Section "Configuring default values for new tools (Page 337)"). false - Default setting, the new tool is not initialized with the configured default data.

12.1.27 Configure creation from a TTD file

<TTDCREATETOOLCONFIGURATION> tag

The entries used for configuring the "Create tools from TTD file" are located in the <TTDCREATETOOLCONFIGURATION> tag.

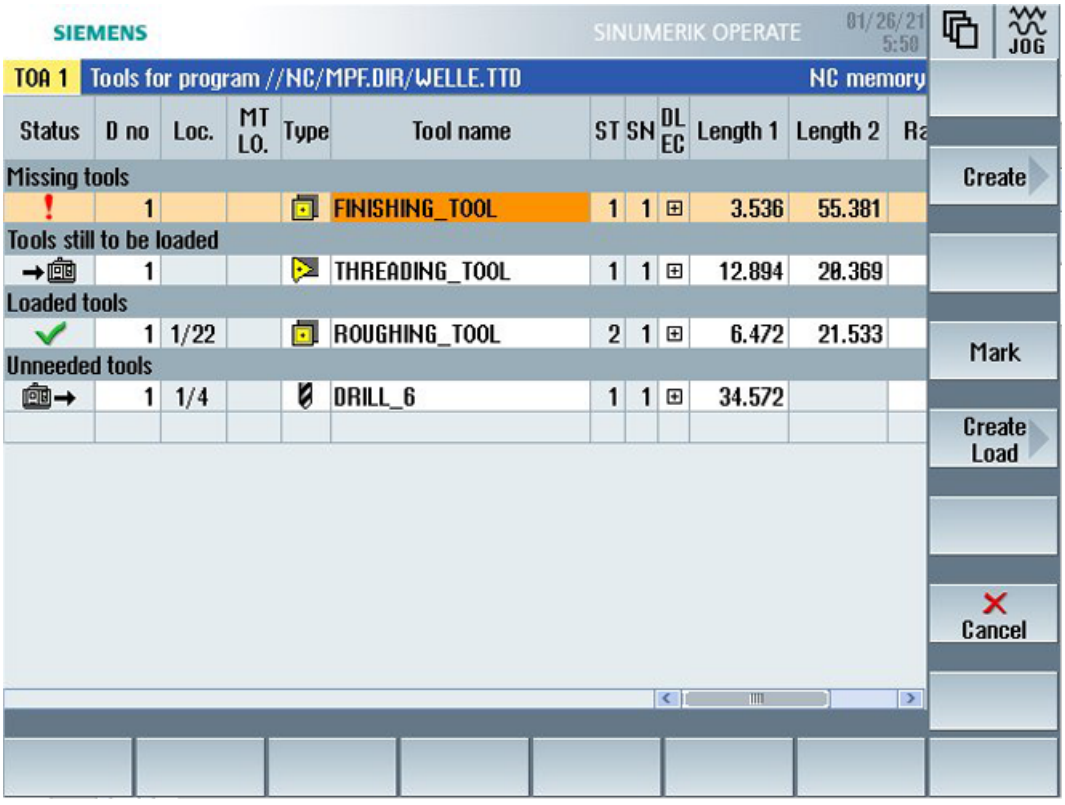


Figure 12-3 Tool from TTD file

The following settings can be made:

Entries	Meaning
CreateNewToolWithDialog	true - Before creating the tool from the TTD file, the tool data is displayed in the "New tool from TTD file" window false - Default setting, the tool is created directly in the tool list.
ForceSisterToolNumber	true - The sister tool number specified by the code carrier data is kept when the tool is created. false - Default setting, the sister tool number is freely assigned.
UseDefaultData	true - The new tool is initialized with the configured default data before the data are taken over from the file (see tag <NEWTOOLDEFAULTVALUECONFIGURATION> "Configuring default values for new tools (Page 337)") false - Default setting, the new tool is not initialized with the configured default data.

12.2 Creating OEM texts

Overview

This section describes how you create OEM texts in a particular language.

The texts are created in the "sltmlistdialog_xxx.ts" file.

A dedicated file must be created for each language. The file names differ by the language code. For "xxx", set the appropriate language code.

Further information
Commissioning Manual SINUMERIK Operate; Supported languages

Creating texts

All OEM texts from the previous sections, such as the tooltip, can be made language-dependent by using the OEM text as text identifier under the "source" tag. The translation of the text in the particular language is entered under the "translation" tag.

Entries	Meaning
source	Text identifier See 1. Example: Single-line text with new text identifier "MY_NEW_TEXT". See 2. Example: Double-line text with new text identifier "MY_NEW_TEXT_2_LINES".
translation	Text, which is displayed on the user interface. You can enter single and multi-line texts. "%n" is used to indicate a line break. See 1. Example: Single-line text "My new text" is displayed. See 2. Example: Double-line text "My new - text" is displayed.
chars	Text length See 1. Example: Text length of 30 characters is specified. See 2. Example: Text length of 10 characters per line is specified.
lines	Number of lines See 2. Example: The number of rows is 2.
remark	Own comments - are not displayed.

Procedure

1. You can copy the sample file "oem_sltmlistdialog_eng.ts" from the following directory: /**siemens**/sinumerik/hmi/template/lng.
2. Save the file in the /**oem**/sinumerik/hmi/lng or /**user**/sinumerik/hmi/lng directory.
3. Change the file name to "sltmlistdialog_eng.ts".
   If you wish to create texts for additional languages, then a separate file must be created for each language. Save the file with the appropriate language code in the file name.
4. Open the file and define the texts in the <message> and </message> areas.
5. Restart the system.
   The file must still be converted into a binary format in order that the texts are displayed during the program runtime. This data is only converted during power up.

12.2.1 Identifiers of the standard texts

Identifiers of the standard texts

With the help of the identifiers, you can change the standard texts of OEM parameters, coolants and tool-specific functions. This applies to the column header in the tool lists, the tooltip and, if available, also the designation in the detail displays.

A list of the text identifiers is provided below:

OEM tool parameters

"ToolUser_1..64"

"ToolAppl_1..10"

Text identifier of the column heading	Text identifier of the tooltip
TM_HL_TOOL_USER_1..64	TM_TT_TOOL_USER_1..64
TM_HL_TOOL_APPL_1..10	TM_TT_TOOL_APPL_1..10

Text identifiers in tool details	Text identifiers in tool details - all parameters
TM_HL_TOOL_USER_1..64_DETAILS	TM_HL_TOOL_USER_1..64_LONG
TM_HL_TOOL_APPL_1..10_DETAILS	TM_HL_TOOL_APPL_1..10_LONG

OEM cutting edge parameters

"EdgeUser_1..64"

"EdgeAppl_1..10"

Text identifier of the column heading	Text identifier of the tooltip
TM_HL_EDGE_USER_1..64	TM_TT_EDGE_USER_1..64
TM_HL_EDGE_APPL_1..10	TM_TT_EDGE_APPL_1..10

Text identifiers in tool details	Text identifiers in tool details - all parameters
TM_HL_EDGE_USER_1..64_DETAILS	TM_HL_EDGE_USER_1..64_LONG
TM_HL_EDGE_APPL_1..10_DETAILS	TM_HL_EDGE_APPL_1..10_LONG

OEM monitoring parameters

"EdgeSupUser_1..64"

"EdgeSupAppl_1..10"

Text identifier of the column heading	Text identifier of the tooltip
TM_HL_EDGE_SUPUSER_1..64	TM_TT_EDGE_SUPUSER_1..64
TM_HL_EDGE_SUPAPPL_1..10	TM_TT_EDGE_SUPAPPL_1..10

Text identifiers in tool details	Text identifiers in tool details - all parameters
TM_HL_EDGE_SUPUSER_1..64_DETAILS	TM_HL_EDGE_SUPUSER_1..64_LONG
TM_HL_EDGE_SUPAPPL_1..10_DETAILS	TM_HL_EDGE_SUPAPPL_1..10_LONG

OEM magazine location parameters

"MagPlaceUser_1..64"

"MagPlaceAppl_1..10"

Text identifier of the column heading	Text identifier of the tooltip
TM_HL_MAGPLACE_USER_1..64	TM_TT_MAGPLACE_USER_1..64
TM_HL_MAGPLACE_APPL_1..10	TM_TT_MAGPLACE_APPL_1..10

OEM multitool parameters

"MultiToolUser_1..64"

"MultiToolAppl_1..10"

Text identifier of the column heading	Text identifier of the tooltip
TM_HL_MULTITOOL_USER_1..64	TM_TT_MULTITOOL_USER_1..64
TM_HL_MULTITOOL_APPL_1..10	TM_TT_MULTITOOL_APPL_1..10

Text identifiers in tool details	Text identifiers in tool details - all parameters
TM_HL_MULTITOOL_USER_1..64_DETAILS	TM_HL_MULTITOOL_USER_1..64_LONG
TM_HL_MULTITOOL_APPL_1..10_DETAILS	TM_HL_MULTITOOL_APPL_1..10_LONG

OEM multitool location parameters

"MultiToolPlaceUser_1..64"

"MultiToolPlaceAppl_1..10"

Text identifier of the column heading	Text identifier of the tooltip
TM_HL_MTPLACE_USER_1..64	TM_TT_MTPLACE_USER_1..64
TM_HL_MTPLACE_APPL_1..10	TM_TT_MTPLACE_APPL_1..10

OEM crosshead parameters

"CrossHeadUser_1..64"

"CrossHeadAppl_1..10"

Text identifier of the column heading	Text identifier of the tooltip
TM_HL_CROSSHEAD_USER_1..64	TM_TT_CROSSHEAD_USER_1..64
TM_HL_CROSSHEAD_APPL_1..10	TM_TT_CROSSHEAD_APPL_1..10

Text identifiers in tool details	Text identifiers in tool details - all parameters
TM_HL_CROSSHEAD_USER_1..64_DETAILS	TM_HL_CROSSHEAD_USER_1..64_LONG
TM_HL_CROSSHEAD_APPL_1..10_DETAILS	TM_HL_CROSSHEAD_APPL_1..10_LONG

OEM crosshead location parameters

"CrossHeadPlaceUser_1..64"

"CrossHeadPlaceAppl_1..10"

Text identifier of the column heading	Text identifier of the tooltip
TM_HL_CHPLACE_USER_1..64	TM_TT_CHPLACE_USER_1..64
TM_HL_CHPLACE_APPL_1..10	TM_TT_CHPLACE_APPL_1..10

Coolant and tool-specific functions

"Coolant1..2"

"MFunction1..4"

Text identifier of the column heading	Text identifier of the tooltip
No column	TM_TT_STATE_COOL_1..2
TM_HL_MFCT1..4	TM_TT_MFCT1..4

12.2.2 Examples of OEM texts

Example 1 with a single-line and double-line text

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE TS><TS>
<!--      ************************************************************ -->
<!--      Definition of OEM texts
<!--      ************************************************************ -->
<context>
   <name>SlTmListForm</name>
<!--      ************************************************************ -->
<!--      enter your text behind this comment -->
<!--      ************************************************************ -->
<!--      ************************************************************ -->
<!--      1st example of a single-line text -->
<!--      ************************************************************ -->
<!--  <message>
        <source>MY_NEW_TEXT</source>
        <translation>My new text</translation>
        <chars>30</chars>
```

```
      </message>-->
<!--  ************************************************************ -->
<!--  2nd example of a double-spaced text -->
<!--  ************************************************************ -->
<!--  <message>
        <source>MY_NEW_TEXT_2_LINES</source>
        <translation>My new%n text</translation>
        <remark>My comment to the text</remark>
        <chars>10</chars>
        <lines>2</lines>
      </message>-->
<!--  ************************************************************ -->
<!--  enter your text ahead of this comment -->
<!--  ************************************************************ -->
</context>
</TS>
```

Example 2

A new parameter is configured in the following example from section Configuring list parameters (Page 320):

```
<PARAMETERCONFIGURATION>
    <EdgeUser_1_Bit0>
        <Base value="EdgeUser_1" type="QString" />
        <ItemType value="CheckBox" type="QString" />
        <DisplayMode value="AnyMode" type="QString" />
        <BitMask value="1" type="int" />
        <Width value="17" type="int" />
        <HeadLine value="TM_HL_EDGE_USER_1_Bit0" type="QString" />
        <ToolTip value="TM_TT_EDGE_USER_1_Bit0" type="QString" />
    </EdgeUser_1_Bit0>
</PARAMETERCONFIGURATION>
```

The two texts under HeadLine and ToolTip should be output depending on the language. To do this, the two text identifiers must be included in the sltmlistdialog_xxx.ts file:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE TS><TS>
<!--  ************************************************************ -->
<!--  Definition of OEM texts
<!--  ************************************************************ -->
<context>
  <name>SlTmListForm</name>
<!--  ************************************************************ -->
<!--  enter your text behind this comment -->
<!--  ************************************************************ -->
```

```
<!--    ************************************************************ -->
<!--    Translate the text for the headline -->
<!--    ************************************************************ -->
      <message>
        <source>TM_HL_EDGE_USER_1_Bit0</source>
        <translation>heavy %n tool</translation>
        <chars>10</chars>
        <lines>2</lines>
      </message>
<!--    ************************************************************ -->
<!--    Translate the text for the tooltip -->
<!--    ************************************************************ -->
      <message>
        <source>TM_TT_EDGE_USER_1_Bit0</source>
        <translation>heavy tool</translation>
        <chars>20</chars>
      </message>
<!--    ************************************************************ -->
<!--    enter your text ahead of this comment -->
<!--    ************************************************************ -->
</context>
</TS>
```

Operate tool list window adapting "Additional data"

The description texts for the "Additional data" window must be defined in a separate context.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE TS><TS>
<!--    ************************************************************ -->
<!--    Definition of OEM texts
<!--    ************************************************************ -->
<context>
   <name>SlTmMoreDataForm</name>
<!--    ************************************************************ -->
<!--    enter your text behind this comment -->
<!--    ************************************************************ -->
<!--    ************************************************************ -->
<!--    Translate the text for the headline -->
<!--    ************************************************************ -->
      <message>
        <source>PUNCH_TAP_TWIST_TEXT</source>
        <translation>Twist pitch mm/rev</translation>
        <chars>15</chars>
        <lines>1</lines>
```

```
      </message>
</context>
</TS>
```

12.3 Examples

12.3.1 Example: Configuring an OEM tool list

12.3.1.1 Adapting the configuration file

The following excerpt shows the contents of the configuration file:

- Milling technology: sltmlistconfig.xml
- Turning technology: sltmturninglistconfig.xml

The file in this example is stored in the "user/sinumerik/hmi/cfg" directory.

This example describes a configuration of the OEM tool list.

- OEM tool parameter 1 is displayed in column 6.
- A new "Coolant 3" parameter is inserted in column 7 that displays bit 0 from OEM tool parameter 2.
- OEM cutting edge parameter 1 is displayed in column 8.

Example

"sltmlistconfig.xml" or "sltmturninglistconfig.xml" file:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CONFIGURATION>
   <!-- **** Identifier for the configuration of the lists *** -->
   <LISTCONFIGURATION>
     <!-- **** OEM tool list *** -->
     <SlTmTooloemForm>
       <!-- **** Enabling the OEM tool list *** -->
       <Enabled value="true" type="bool" />
       <!-- **** Columns 1 to 5 have already been assigned default values. *** -->
       <!-- **** Column 6 *** -->
       <COLUMN6>
         <!-- **** For all tool types *** -->
         <TOOLTYPE_ALL>
           <!-- **** Display OEM tool parameter 1 *** -->
           <Item value="ToolUser_1" type="QString" />
         </TOOLTYPE_ALL>
```

```
    </COLUMN6>
    <!-- **** Column 7 *** -->
    <COLUMN7>
      <!-- **** For all tool types *** -->
      <TOOLTYPE_ALL>
        <!-- **** Priority tool, own parameter *** -->
        <!-- **** Display OEM tool parameter 2 bit 0 *** -->
        <Item value="PriorTool" type="QString" />
      </TOOLTYPE_ALL>
    </COLUMN7>
    <!-- **** Column 8 *** -->
    <COLUMN8>
      <!-- **** For all tool types *** -->
      <TOOLTYPE_ALL>
        <!-- **** Display OEM cutting edge parameter 1 *** -->
        <Item value="EdgeUser_1" type="QString" />
      </TOOLTYPE_ALL>
    </COLUMN8>
  </SlTmTooloemForm>
</LISTCONFIGURATION>
<!-- **** Identifier for the configuration of own parameter *** -->
<PARAMETERCONFIGURATION>
  <!-- **** Change OEM tool parameter 1 *** -->
  <ToolUser_1>
    <!-- **** Column width *** -->
    <Width value="36" type="int" />
    <!-- **** Type change to unsigned integer *** -->
    <DisplayMode value="UnsignedIntegerMode" type="QString" />
  </ToolUser_1>
  <!-- **** Identifier of own parameter *** -->
  <!-- **** Priority tool *** -->
  <PriorTool>
    <!-- ** New parameter is based on OEM tool parameter 2**-->
    <Base value="ToolUser_2" type="QString" />
    <!-- **** Bit 0 is evaluated *** -->
    <BitMask value="1" type="int" />
    <!-- **** Text identifier of the headline *** -->
    <HeadLine value="TMO_HL_PRIOR_TOOL" type="QString" />
    <!-- **** Text identifier of the tooltip *** -->
    <ToolTip value="TMO_TT_PRIOR_TOOL" type="QString" />
    <!-- **** Column width *** -->
    <!-- **** Recommended widths 640x480 - 19 pixels *** -->
    <!-- **** 800x600 - 21, 1024x768 - 17 *** -->
    <Width value="19" type="int" />
```

```
      <!-- **** Display item of the parameter *** -->
      <ItemType value="CheckBox" type="QString" />
    </PriorTool>
  </PARAMETERCONFIGURATION>
</CONFIGURATION>
```

12.3.1.2 Adaptation of the customer text file

The following excerpt shows the contents of the "sltmlistdialog_eng.ts" customer text file.

The file in this example is stored in the "user/sinumerik/hmi/lng" directory.

Example

"sltmlistdialog_eng.ts" file:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE TS><TS>

<!-- ************************************************************ -->
<!-- Definition of OEM texts                                      -->
<!-- ************************************************************ -->
<context>
    <name>SlTmListForm</name>

<!-- ************************************************************ -->
<!-- Enter your text behind this comment                          -->
<!-- ************************************************************ -->

      <!-- **** OEM tool parameter 1 headline *** -->
      <!-- **** Weight *** -->
      <message>
        <!-- **** Text identifier *** -->
        <source>TM_HL_TOOL_USER_1</source>
        <!-- **** Text that is output *** -->
        <translation>%nweight</translation>
        <!-- **** Comment only for the translator *** -->
        <remark>priority tool</remark>
        <!-- **** Maximum line length in characters *** -->
        <!-- **** only for the translator *** -->
        <chars>7</chars>
        <!-- **** Number of possible lines *** -->
        <!-- **** only for the translator *** -->
        <lines>2</lines>
      </message>
```

```
<!-- **** OEM tool parameter 1 tooltip *** -->
<!--  **** weight  *** -->
<message>
  <source>TM_TT_TOOL_USER_1</source>
  <translation>weight of the tool</translation>
  <chars>35</chars>
  <lines>2</lines>
</message>
<!--  **** title for user's parameters *** -->
<!--  **** priority tool  *** -->
<message>
  <source>TMO_HL_PRIOR_TOOL</source>
  <translation>V</translation>
  <remark>priority tool</remark>
  <chars>1</chars>
  <lines>2</lines>
</message>
<!--  **** to for user's parameters *** -->
<!-- **** Priority tool *** -->
<message>
  <source>TMO_TT_PRIOR_TOOL</source>
  <translation>priority tool, %nuse tool preferably< /translation>
  <chars>35</chars>
  <lines>3</lines>
</message>
<!--  **** OEM cutting edge parameter 1 headline *** -->
<!--  **** Maximum tool width  *** -->
<message>
  <source>TM_HL_EDGE_USER_1</source>
  <translation>maximum%nwidth</translation>
  <chars>8</chars>
  <lines>2</lines>
</message>
<!--  **** OEM cutting edge parameter 1 tooltip *** -->
<!--  **** Maximum tool width  *** -->
<message>
  <source>TM_TT_EDGE_USER_1</source>
  <translation>maximum width%nof the tool%noverall</translation>
  <chars>35</chars>
  <lines>2</lines>
</message>

<!--  ************************************************************ -->
<!--  Enter your text ahead of this comment                        -->
```

```
<!-- ************************************************************ -->

</context>
</TS>
```

12.3.2 Example: Configuring magazine location types with names

12.3.2.1 Adapting the configuration file

The following excerpt shows the contents of the configuration file:

- Milling technology: sltmlistconfig.xml
- Turning technology: sltmturninglistconfig.xml

The file in this example is stored in the "user/sinumerik/hmi/cfg" directory.

This example describes a configuration of the location types in the magazine list.

In this example, the magazine location type and the tool location type are to be displayed as a toggle field with plain texts.

The assignment of the parameters to the columns of the list is made in the "LISTCONFIGURATION" area. The magazine location type is displayed in column 10. The tool location type is displayed in column 11.

The texts to the location types are assigned in the "PLACETYPECONFIGURATION" area. The number of the location type is the number at the end of the "PLACETYPE" identifier.
A text identifier for a toggle box entry and a text identifier for a tooltip can be stored for each location type.

Example

"sltmlistconfig.xml" or "sltmturninglistconfig.xml" file:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CONFIGURATION>
   <!-- **** Configuration of the lists *** -->
   <LISTCONFIGURATION>
     <!-- **** Magazine list *** -->
     <SlTmToolmagazinForm>
       <!-- **** Column 10 *** -->
       <COLUMN10>
         <!-- **** For all tool types *** -->
         <TOOLTYPE_ALL>
           <!-- **** Display magazine location type *** -->
           <Item value="MagPlaceTypeIdent" type="QString" />
         </TOOLTYPE_ALL>
       </COLUMN10>
```

```
      <!-- **** Column 11 *** -->
      <COLUMN11>
        <!-- **** For all tool types *** -->
        <TOOLTYPE_ALL>
          <!-- **** Display tool location type *** -->
          <Item value="ToolPlaceSpecIdent" type="QString" />
        </TOOLTYPE_ALL>
        <!-- **** For all multitools *** -->
        <TOOLTYPE_9997>
          <!-- **** Display for all multitool location types *** -->
          <Item value="MultiToolPlaceSpecIdent" type="QString" />
        </TOOLTYPE_9997>
      </COLUMN11>
    </SlTmToolmagazinForm>
  </LISTCONFIGURATION>
  <!-- **** Configuration of the magazine location types *** -->
  <PLACETYPECONFIGURATION>
    <!-- **** Magazine location type 1 *** -->
    <PLACETYPE1>
      <!-- **** Text identifier of the toggle box entry *** -->
      <Text value="TMMP_ITEM_NORMAL_TOOL" type="QString" />
      <!-- **** Text identifier of the tooltip *** -->
      <Tooltip value="TMMP_TT_NORMAL_TOOL" type="QString" />
    </PLACETYPE1>
    <!-- **** Magazine location type 2 *** -->
    <PLACETYPE2>
      <Text value="TMMP_ITEM_HEAVY_TOOL" type="QString" />
      <Tooltip value="TMMP_TT_HEAVY_TOOL" type="QString" />
    </PLACETYPE2>
    <!-- **** Magazine location type 3 *** -->
    <PLACETYPE3>
      <Text value="TMMP_ITEM_BIG_TOOL" type="QString" />
      <Tooltip value="TMMP_TT_BIG_TOOL" type="QString" />
    </PLACETYPE3>
    <!-- **** Magazine location type 4 *** -->
    <PLACETYPE4>
      <Text value="TMMP_ITEM_SENSITIVE_TOOL" type="QString" />
      <Tooltip value="TMMP_TT_SENSITIVE_TOOL" type="QString" />
    </PLACETYPE4>
  </PLACETYPECONFIGURATION>
</CONFIGURATION>
```

12.3.2.2 Adaptation of the customer text file

The following excerpt shows the contents of the "sltmlistdialog_eng.ts" customer text file.

The file in this example is stored in the "user/sinumerik/hmi/lng" directory.

Example

"sltmlistdialog_eng.ts" file:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE TS><TS>

<!-- ************************************************************ -->
<!-- Definition of OEM texts                                      -->
<!-- ************************************************************ -->
<context>
     <name>SlTmListForm</name>

<!-- ************************************************************ -->
<!-- Enter your text behind this comment                          -->
<!-- ************************************************************ -->
     <!-- **** Magazine location type 1 item *** -->
     <!--  **** Normal tool *** -->
     <message>
       <!-- **** Text identifier *** -->
       <source>TMMP_ITEM_NORMAL_TOOL</source>
       <!-- **** Text that is output *** -->
       <translation>normal</translation>
       <!-- **** Maximum line length in characters *** -->
       <!-- **** only for the translator *** -->
       <chars>9</chars>
     </message>
     <!-- **** Magazine location type 1 tooltip *** -->
     <!-- **** Normal tool *** -->
     <message>
       <source>TMMP_TT_NORMAL_TOOL</source>
       <translation>normal tool,%nno restrictions</translation>
       <chars>35</chars>
       <lines>2</lines>
     </message>
     <!-- **** Magazine location type 2 item *** -->
     <!-- **** Heavy tool *** -->
     <message>
       <source>TMMP_ITEM_HEAVY_TOOL</source>
       <translation>heavy</translation>
```

```
    <chars>9</chars>
  </message>
  <!-- **** Magazine location type 2 tooltip *** -->
  <!-- **** Heavy tool *** -->
  <message>
    <source>TMMP_TT_HEAVY_TOOL</source>
    <translation>heavy tool</translation>
    <chars>35</chars>
    <lines>2</lines>
  </message>
  <!-- **** Magazine location type 3 item *** -->
  <!-- **** Big tool *** -->
  <message>
    <source>TMMP_TT_BIG_TOOL</source>
    <translation>big</translation>
    <chars>9</chars>
  </message>
  <!-- **** Magazine location type 3 tooltip *** -->
  <!-- **** Big tool *** -->
  <message>
    <source>TMMP_TT_BIG_TOOL</source>
    <translation>big tool</translation>
    <chars>35</chars>
    <lines>2</lines>
  </message>
  <!-- **** Magazine location type 4 item *** -->
  <!-- **** Sensitive tool *** -->
  <message>
    <source>TMMP_ITEM_SENSITIVE_TOOL</source>
    <translation>sensitive</translation>
    <chars>9</chars>
  </message>
  <!-- **** Magazine location type 4 tooltip *** -->
  <!-- **** Sensitive tool *** -->
  <message>
    <source>TMMP_TT_SENSITIVE_TOOL</source>
    <translation>Notice: Tool can%nbe destroyed easily</translation>
    <chars>35</chars>
    <lines>2</lines>
  </message>

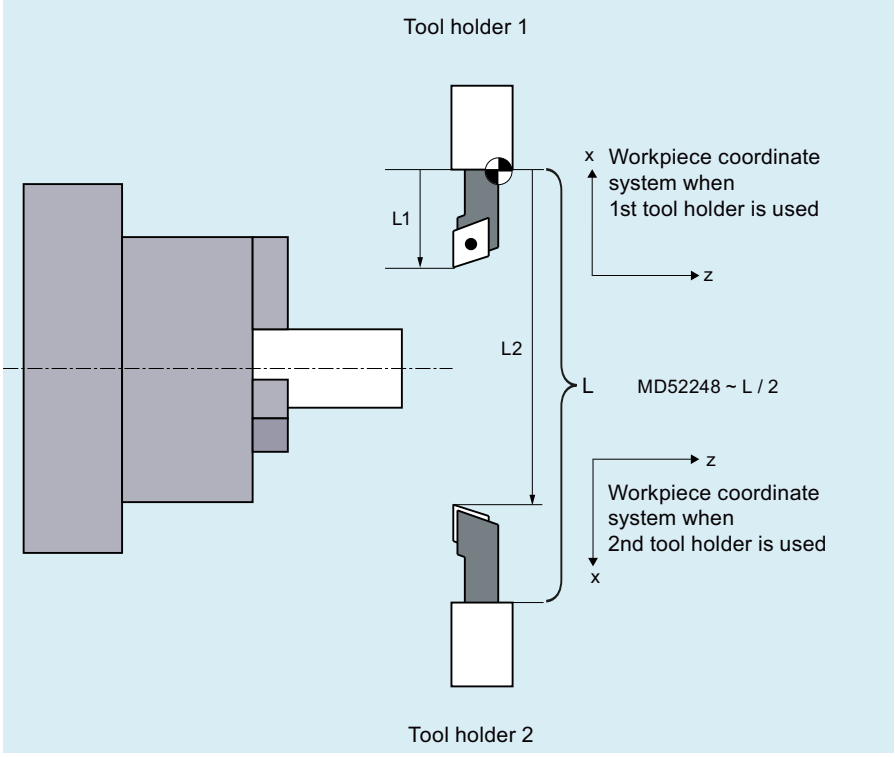
<!-- ************************************************************ -->
<!--  Enter your text ahead of this comment                       -->
<!-- ************************************************************ -->
```

```
</context>
</TS>
```

12.4 Working with two tool holders

It is possible to work on a single turning machine with two tool holders, both of which are mounted on an X axis. The tool holders may be revolvers, multifix, or a combination of both. As both tool adapters are mounted on the same axis it is only possible to machine with one tool. The workpiece is located between the two tool adapters.

All tools have the same reference point (on tool holder 1) irrespective of which tool holder they are on. Consequently, the tool lengths of the second tool holder are always longer than the tool lengths of the first tool holder. The main machining is performed in the negative X axis direction.



Tool holder definition

The MD52248 REV_2_BORDER_TOOL_LENGTH defines a tool length limit X so that the tools can be assigned to the two tool holders automatically:

Tool length X < MD52248: Tool belongs to the 1st tool holder
Tool length X ≥ MD52248: Tool belongs to the 2nd tool holder

Additional parameter assignments

For the used mirrorings of the X and Y axes and the offsets of the C axis, the following machine data has to be configured:

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	Clear system frames after RESET
Bit 4 = 0	Do not clear system frame workpiece reference after RESET

MD28082 $MC_MM_SYSTEM_FRAME_MASK	Setting-up system frames (SRAM)
Bit 4 = 1	Workpiece reference ($P_WPFRAME)

MD28083 $MC_MM_SYSTEM_DATAFRAME_MASK	Setting-up data management, system frames (SRAM)
Bit 4 = 1	Workpiece reference ($P_WPFRAME)

Due to the coordinate system being mirrored, the following machine/setting data must be set:

MD21202 $MC_LIFTFAST_WITH_MIRROR	Lift fast with mirror
= 1	For the determination of the lifting direction, it is also mirrored for active mirroring. The mirroring of the lifting direction only refers to the direction components perpendicular to the tool direction.

SD42900 $SC_MIRROR_TOOL_LENGTH	Sign change, tool length when mirroring
= 0	The signs of the tool length components are independent of system frames with mirroring.

SD42910 $SC_MIRROR_TOOL_WEAR	Sign change, tool wear when mirroring
= 1	Inverting the sign mirrors the wear values of the tool length components whose associated axes are mirrored.

Settings for ShopTurn programs

In ShopTurn, MD21202, SD42900, and SD42910 are automatically set for the necessary mirroring.

Settings for G code programs

Additional settings must be made if the functions of the second tool holder with mirroring of the X axis are also to be active for G code programs and tool calls from the T,S,M screenform:

MD10717 $MN_T_NO_FCT_CYCLE_NAME="F_T_REV2"

MD15710 $MN_TCA_CYCLE_NAME="F_T_REV2"

Note

The settings of MD10717 and MD15710 cannot be used if old G code programs are available and will be used without modification for the second tool holder without mirroring of the X axis.

If transformation TRANSMIT is set up, the necessary mirroring can also be activated automatically if programmed. For this, the TRANSMIT command must be replaced by a subprogram (for example, in the manufacturer directory):

```
MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[0]="TRANSMIT"
MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[1]="TRANSM"
```

Subprogram TRANSMIT.SPF:

```
PROC TRANSMIT(INT_NR)
F_T_REV2
TRANSM(_NR)
F_T_REV2
RET
```

Note

In the status display, TRANSM is now displayed instead of TRANSMIT.

General conditions

- The taper turning is not enabled for "working with two tool holders."
- The B-axis function (kinematics) is also not enabled for "working with two tool holders."

Code carrier connection - Tool Ident Connection (option) 13

13.1 Installation

13.1.1 System requirements

Tool Ident Connection is intended for the SINUMERIK 828D target system with SINUMERIK Operate.

Note

Tool Ident Connection is only available in conjunction with the SINUMERIK Operate operating software as of SW 4.9 SP4.

PLC interface

PLC interface for the Parameter operating area (Page 423)

Command codes (Page 424)

Tool management

Active magazine management is required to use the Tool Ident Connection option.

DB1900

The PLC data block DB1900 is available as of V4.9 SP4 via the PLC interface for "Tool Ident Connection", see PLC interface in DB1900 (Page 421).

Hardware configuration for read/write devices

The read/write devices for the tool data are connected to the SINUMERIK 828D as PROFINET IO devices. The hardware configuration is stored in SDBs. For the SINUMERIK 828D, the bus is predefined for different standard configurations and entered into preconfigured SDBs.

Using a Balluff read/write head

When using a read/write head from the Balluff company, the special SDB must be installed like a software update. The special SDB can be found in Pridanet under "SINUMERIK 828D Software".

Information on how you handle Balluff components is provided in the following FAQ (SIOS):

FAQ number: 109797865

Title: SINUMERIK 828: Which Balluff components have been released for the Ident Tool?

13.1.2 Installation of Tool Ident Connection

Note

We recommend that you generate a system backup before setup.

According to the system requirement, any additional components required must be installed first.

The storage medium for software applications is a CompactFlash card with fixed directory structure. The "Tool Ident Connection" application is stored within this structure in the "OEM" directory.

Data conversion from V5.25

With version V6.25, it is no longer necessary to convert the data with the WKonvert Wizard. Instead, the data is copied to a fixed directory on the NCU. From there, the data is automatically converted into the correct format: Converting data (Page 392)

Hardware configuration for read/write devices

The read/write devices for the tool data are connected to the SINUMERIK 828D as PROFINET IO devices. The hardware configuration is stored in SDBs. For the SINUMERIK 828D, the bus is predefined for different standard configurations and entered into preconfigured SDBs. These SDBs are supplied along with the control.

The SDB must be installed for Balluff devices.

13.1.2.1 Activation of Tool Ident Connection

Configuration files

The following files are available to configure Tool Ident Connection:

Standard configuration files	Directory: SD card
tdiidentcfg.xml	/siemens/sinumerik/hmi/template\toolidentconnection
toolSpec.xml	
wkonvert.mcx	
DefToolDat.txt	
wkonvert.mcc	
systemconfiguration.ini	
oem_sltmlistconfig.xml	

Setup of the Tool Ident Connection option

The following entry must be made in the "systemconfiguration.ini" in the section [services] to set up the software:

```
SVCxxx=name:=SlMcisTdiIdentConnectionService,
implementation:=slmcistdiidentconnectionservice.SlMcisTdiIdentConnectionService,
process:=SlHmiHost1,
```

Note

The `xxx` in `SVCxxx` must be replaced by the free number of a service. Only numbers as of 200 can be used. The service number 200 is already provided in the template.

To adapt the Tool Ident Connection configuration function, the systemconfiguration.ini and, if adapted, the configuration files (from the directory mentioned above) must be copied to the following directory.

- SD Card: /OEM/

The oem_sltmlistconfig.xml file must be renamed, depending on the technology. This is explained in Chapter "Configuring the user interface".

An exhaustive description of the configuration options of the oem_sltmlistconfig.xml can be found in Configuring the user interface (Page 291).

The files `wkonvert.txt`, `DefToolDat.txt` and `wkonvert.mcc` are configuration files. They must be converted into a `wkonvert.mcx` file.

13.2 Configuration

13.2.1 Configuration schematic

The following figure displays the configuration scheme of Tool Ident Connection:

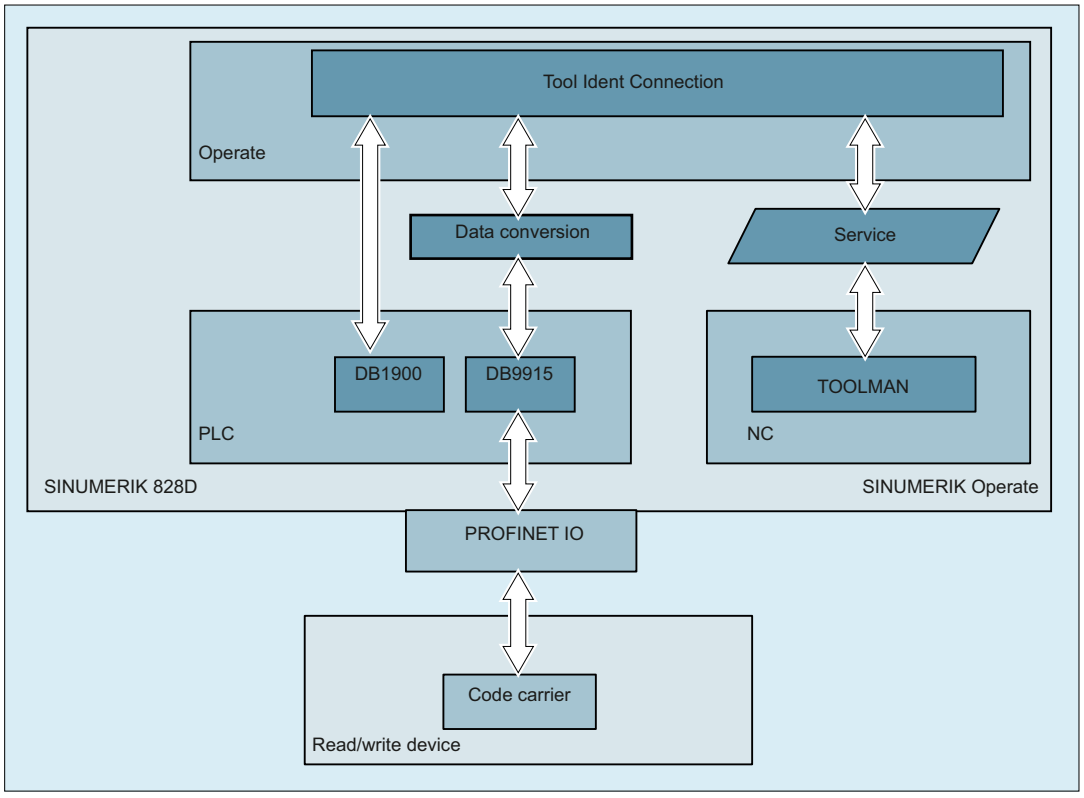


Figure 13-1 Configuration scheme of Tool Ident Connection

13.2.2 Configuration of the tool lists

<IDENTCONNECTIONCONFIGURATION> tag

The entries used for configuring the code carrier connection in the tool list of the user interface are located in the <IDENTCONNECTIONCONFIGURATION> tag. The code carrier connection is implemented via Tool Ident Connection.

For additional information on this topic, please refer to Configuring the code carrier connection (Page 361).

13.2.3 TDIIdentCfg.xml

The following entry must be made in the "systemconfiguration.ini", in Section `[services]` to activate the software:

```
[services]
SVCxxx=name:=SIMcisTdiIdentConnectionService,
implementation:=slmcistdiidentconnectionservice.SIMcisTdiIdentConnectionService,
process:=SIHmiHost1, timeout:=5000, shutdowntime:=5000
```

Note

xxx in SVCxxx must be replaced by the free number of a service.

Tool Ident Connection is parameterized using a configuration file based on xml. This configuration file allows the integration and parameterization of various read/write devices via different communication technologies.

Structure

The configuration file is structured as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- (C) Copyright Siemens AG DI MC 2024. All rights reserved. -->
<!-- Configuration-File for MCIS TDI Ident Connection sl -->
<!-- Do not manipulate this file -->
<Configuration>
  <Logging>
    <ErrorFile size="3" enable="1">/user/sinumerik/hmi/log/
IDIIdentLog.txt</ErrorFile>
    <TraceFile size="3" enable="1">/user/sinumerik/hmi/log/
TDIIdentTrace.Log</TraceFile>
  </Logging>
  <!-- parametrisation of KCode-Interface -->
  <GlobalParam>
    <InterfaceDB>1900</InterfaceDB>
    <InterfaceOffset>3110</InterfaceOffset>
    <DisableInterface>false</DisableInterface>
    <TimeOut>1000</TimeOut>
    <InputDB>1900</InputDB>
    <InputOffset>3000</InputOffset>
    <OutputDB>1900</OutputDB>
    <OutputOffset>3058</OutputOffset>
  </GlobalParam>
  <!-- parametrisation of ParamTM-Interface -->
  <ParamTMInterface enable="1">
    <InterfaceDB>1900</InterfaceDB>
    <InterfaceOffset>3116</InterfaceOffset>
  </ParamTMInterface>
  <MergeMode>
    <LoadPlaceNo>1</LoadPlaceNo>
    <UnloadPlaceNo>1</UnloadPlaceNo>
 </MergeMode>
  <!-- parametrisation of read and write unit -->
  <!-- Type = 1 -> PLC -->
  <!-- Type = 2 -> USB -->
  <!-- Type = 3 -> RJ45 -->
  <Device Type="1">
    <Units>1</Units>
    <Data Unit="1">
      <DataDB>9915</DataDB>
      <Offset>0</Offset>
     <MaxData>1000</MaxData>    <!-- ! a value more than 1000 doesn't work!
-->
    </Data>
  </Device>
</Configuration>
```

Description

Note

Only the entries described in the following table can be changed/used to parameterize and configure Tool Ident Connection.

Always generate a backup before making any change.

The configuration file can be changed using a text editor, e.g. notepad. It is subdivided into various nodes which contain different entries. The sections and their entries are described in the following table.

Section	Parameters	Meaning
GlobalParam	InterfaceDB	Data block information for the PLC interface
	InfaceOffset	Offset for the PLC interface
	TimeOut	Execution monitoring time
	DisableInterface	De/activation of the PLC interface
	InputDB	Data block for transfer parameters
	InputOffset	Offset for transfer parameters
	OutputDB	Data block for return parameters
	OutputOffset	Offset for return parameters
Logic	mcxPath	File + path data to the configured mcx file
	specPath	File + path data to ToolSpec.xml
MergeMode	LoadPlaceNo	Default loading position
	UnloadPlaceNo	Default unloading position
Device Type="1" -> PLC	Units	Number of read heads used
	DataBlock	Data block of the read head
	Offset	Offset of the read head data
	MaxDataBytes	Maximum data of the read head in bytes
	EOT	End of the read head character string

If the execution of the PLC services for Tool Ident Connection is to be canceled because of a timeout, a longer time can be defined using the "TimeOut" entry. A timeout can e.g. occur, if the operator does not acknowledge a tool being loaded within the parameterized time window.

You can use the "MergeMode" parameter and the entries "LoadPlaceNo" and "UnloadPlaceNo" to change the default loading and unloading location number of the loading magazine.

The "Device" node has the "Type" attribute; the type of read device being used is saved there. The attribute defines the entries below this range. The parameters of the range contain the configuration data of the read/write device.

Overview of the "Type" attribute of the "Device" parameter:

Type	Designation
1	PLC read device → connection via PROFINET IO

13.2.4 Read/write device at the PLC

Configuration of the PLC connection

In the configuration for Tool Ident Connection, the "Type" attribute must be set to pulse value "1" for a PLC read/write device. The following parameters must be supplied with valid values below the node:

Parameter	Data type	Value range
Units	int	1
DataBlock	int	0 ... 32767
Offset	int	0 ... 32767
MaxDataBytes	int	0 ... 32767
EOT	string	10 characters

"Device" extract from a configuration file (TDIIdentCfg.xml) with PLC connection:

```
<Device Type="1">
        <Units>1</Units>
        <Data Unit="1">
              <DataDB>9915</DataDB>
              <Offset>0</Offset>
              <MaxData>1000</MaxData>
              <EOT>0*2F2F</EOT>
        </Data>
</Device>
```

13.2.5 Data conversion

13.2.5.1 Converting data

The tool data is saved in the code carrier as a sequence of bytes. The PLC user program writes the code carrier data to the previously defined data block (TDIIdentcfg.xml). The data conversion accesses the data block and generates the tool data record and vice versa.

Generating conversion rules

As part of the commissioning of Tool Ident Connection, a conversion file must be created. Binary file `wkonvert.mcx` is created. It contains the conversion rule and default tool data.

The following conversion rules can be processed:

- Conversion rule from Tool Ident Connection
  Files: wkonvert.txt and DefToolDat.txt
  Benefits: Guarantee of compatibility with earlier versions of Tool Ident Connection.

  Note

  The structure of the wkonvert.txt file is described in Chapter Structure of the description file wkonvert.txt (Page 394).

  The structure of the DefToolDat.txt file is described in Chapter Tool data initialization file DefToolDat.txt (Page 402).

- Conversion rule in the form of source text in the MCIS-C script language.
  Benefits: Extended conversion functionality.

  Note

  If you want Siemens to create a customer-specific conversion rule for Tool Ident Connection, please contact your Siemens sales partner.

Converting data

During commissioning, the input files are now converted during the startup of the `slmcistdiidentconnectionservice` during the startup of Operate.

For this purpose, a check is made to see whether one of the following input files is located in the following directory:

- `...oem/sinumerik/hmi/cfg`

When running up, a search is made for the file formats in this sequence and then converted:

1. `...oem/sinumerik/hmi/cfg`
   A search is made for files of the following type:
   `wkonvert.mcx`, `wkonvert.mcc`, `wkonvert.txt` or `deftooldat.txt`
   The file with the most recent change timestamp is converted

Errors are output in a log file: Messages during conversion (Page 394)

Note

Observe the path

All configuration data for ToolIdentConnection must be located in the path `...oem/sinumerik/hmi/cfg`. If data is located in other directories, e.g. in the addon directory, it will no longer be taken into account.

Note

Access level "Manufact"

The "Manufact" access level is required to convert the data.

13.2.5.2 Messages during conversion

Messages that can occur while interpreting the wkonvert.txt and DefToolDat.txt or MCIS-C scripts are described in this section.

All messages are written to the ToolIdentConnection log file:

- oem/sinumerik/hmi/log/toolidentconnection/TDIIdentLog.txt
- oem/sinumerik/hmi/log/toolidentconnection/TDIIdentTrace.Log

To resolve the error, the error that has occurred must be removed in the source file.

The following table lists possible error messages:

ErrorCode	ErrorMessage	Meaning
200	File <abcd> could not be read!	File: DefToolDat.txt; file cannot be read. Additional information is specified.
201	Unknown data type.	File: DefToolDat.txt; for one item, no value has been assigned.
202	Line has no value.	File: DefToolDat.txt; "=" missing.
203	Invalid line format.	File: DefToolDat.txt; "[" or "]" missing.
100	File <abcd> could not be read!	File: wKonvert.txt; file cannot be read. Additional information is specified
101	Error in value ‚DataLen'	File: wKonvert.txt; error when specifying the value 'DataLen'.
102	New block before BlockEnd	File: wKonvert.txt; BlockEnd identification missing.
103	Missing block numer <xx>	File: wKonvert.txt; Block <xx> missing.
104	Line is empty	File: wKonvert.txt; the specified line is empty.
105	No item number defined	File: wKonvert.txt; the item has no number.
110	Missing item: act=<x>, last=<y>.	File: wKonvert.txt; item numbering is incorrect.
111	Finishing block number <xx>	File: wKonvert.txt; block number is incorrect.

13.2.6 Conversion files wkonvert.txt and DefToolDat.txt

For Tool Ident Connection, the conversion rules are used for the files Wkonvert.txt and DefToolDat.txt. There are also the code carrier data formats: REAL and DOUBLE.

13.2.6.1 Structure of the description file wkonvert.txt

The data on the code carrier is stored in a particular order. A conversion rule in the form of a description file is provided so that the tool management can read or write this data flow.

This description file consists of correctly defined tool and cutting edge dialog data. Only this dialog data can actually be processed by the tool management. All the other data on the code carrier must not be assigned to any dialog variables; they will then not be processed by the standard version.

Tool Ident Connection enables the connection of optional software components. These components are used to adapt this special code carrier data to the SINUMERIK standard. This means that customer-specific code carrier formats can be read and written by the tool manager; also, special logic can be implemented in the transport operations of the tool data (e.g. replacement tools can be taken into account in the NC, additional cutting edges can be created, etc.).

The description file can be generated as an ASCII file using a standard text editor. This description file is called wkonvert.txt. It represents the precise image of the data which is on the code carrier.

Tool dialog data

The tool dialog data is defined as follows:

Table 13-1 Tool dialog data

Dialog variable	Data type	Designation	Assignment $TC...
T1	String	Tool name, max. 32 characters	$TC_TP2
T2	Integer	Duplo number	$TC_TP1
T3	Integer	Number of cutting edges	$P_TOOLND[tno] tno=tool number
T4	Integer	Tool size left in half locations	$TC_TP3
T5	Integer	Tool size right in half locations	$TC_TP4
T6	Integer	Tool size top in half locations	$TC_TP5
T7	Integer	Tool size bottom in half locations	$TC_TP6
T8	String	Magazine location type	$TC_TP7 [1)]
T9	Integer	Tool status	$TC_TP8
T10	Integer	Type of tool monitoring	$TC_TP9
T11	Integer	Type of tool search	$TC_TP11
T12 [2)]	Integer	Magazine location type	$TC_TP7

[1)] The character string which is stored there, is an internal control location type that is assigned to the value in $TC_TP7. This text is defined via the tool management start-up and stored in the database.

[2)] This number corresponds to the string from T8 and should be used as an alternative to T8.

Cutting edge dialog data

Table 13-2 Cutting edge dialog data

Dialog variable	Data type	Designation	Assignment $TC...
C1	Integer	Subtype	$TC_DP1
C4	Integer	Cutting edge position	$TC_DP2
		Geometry tool length compensation	
C5	Double	Length 1	$TC_DP3
C6	Double	Length 2	$TC_DP4
C7	Double	Length 3	$TC_DP5
		Geometry tool radius compensation	
C8	Double	Length 1	$TC_DP8
C9	Double	Length 2	$TC_DP9
C10	Double	Radius 1	$TC_DP6
C11	Double	Radius 2	$TC_DP7
C12	Double	Angle 1	$TC_DP10

Dialog variable	Data type	Designation	Assignment $TC...
C13	Double	Angle 2	$TC_DP11
		Wear tool length compensation	
C14	Double	Length 1	$TC_DP12
C15	Double	Length 2	$TC_DP13
C16	Double	Length 3	$TC_DP14
		Wear tool radius compensation	
C17	Double	Length 1	$TC_DP17
C18	Double	Length 2	$TC_DP18
C19	Double	Radius 1	$TC_DP15
C20	Double	Radius 2	$TC_DP16
C21	Double	Angle 1	$TC_DP19
C22	Double	Angle 2	$TC_DP20
		Base dimension/adapter dimension tool length compensation	
C23	Double	Basic length 1	$TC_DP21
C24	Double	Basic length 2	$TC_DP22
C25	Double	Basic length 3	$TC_DP23
C26	Double	Undercut angle	$TC_DP24
C27	Integer	Reverse insert	$TC_DP25
C28	Integer	Cutting edge number for addressing variables	
C29 *)	Integer	Tool life in minutes	$TC_MOP2
C30 *)	Integer	Prewarning limit for tool life in minutes	$TC_MOP1
C31 *)	Integer	Unit quantity	$TC_MOP4
C32 *)	Integer	Prewarning limit for workpiece count	$TC_MOP3
C33	Double	Setpoint tool life in minutes	$TC_MOP11
C34 *)	Integer	Setpoint unit quantity	$TC_MOP13
C35	Double	Prewarning limit for wear	$TC_MOP5
C36	Double	Wear	$TC_MOP6
C37	Double	Setpoint wear	$TC_MOP15
C38 *)	Double	Tool life in minutes	$TC_MOP2
C39 *)	Double	Prewarning limit for tool life in minutes	$TC_MOP1
C40 *)	Double	Unit quantity	$TC_MOP4
C41 *)	Double	Prewarning limit for workpiece count	$TC_MOP3
C42 *)	Double	Setpoint unit quantity	$TC_MOP13
A1 ... A10	Double/String	User tool data	$TC_TPCx
U1 ... U10	Double/String	User cutting edge data	$TC_DPCx
S1 ... S10	Double/String	User monitoring data	$TC_MOPCx

*) C38 and C39 can only be used as an alternative to C29 and C30; C40, C41 and C42 can only be used as an alternative to C31, C32 and C34.

That data on the code carrier, which is not relevant for tool management, but which may not be changed when writing to the code carrier, is shown in the description file with a hyphen (-) instead of the dialog variable (data gaps).

Data types

The following data types are defined for dialog variables:

- Integer: Value range -32768 ... +32767
- Double: Floating point - double the accuracy
- String: String made up of ASCII characters

Keywords

The code carrier description file is structured line-by-line, whereby each line starts with one of the following keywords:

- Inverted comma
  The apostrophe (') marks the beginning of a comment. The characters that follow up to the end of the line are skipped. Example:
  `' This is a comment`
- Datalen
  `DATALEN=CONST | VARIABLE 0x<delimiter>`
  The following data has a constant (CONST) or a variable (VARIABLE) data length. Data with variable length is terminated with 0x<delimiter>. Example:
  `DATALEN=VARIABLE 0x0A ' variable data length, delimiter LF`
- Item
  `Item<n>=<line>`
  `<n>` := Consecutive number of the code carrier data, ascending from 1 without gaps
  `<line>` := <(max.) Length in bytes> `<code carrier data format> <dialog variable>`
  `<code carrier data format>` : See table, Code carrier data formats
  `<dialog variable>` : Assignment to the dialog variable, see table, Cutting edge dialog data
  Example:
  ```
  Item1 32 ASCII T3 ' tool identifier
  Item2 3 BCD T2 ' duplo number
  ```
- BItem
  `BItem<n>=<line>`
  `<n>` := Consecutive number of the code carrier data, ascending from 1 without gaps
  `<line>` := analog `Item<n>`
  Conversion rule for code carrier data <n> within a block. If the code carrier data is assigned a tool dialog data T<n>, the dialog data is assigned the first value of the code carrier data in the block.
  Example:
  ```
  BItem1 1 BCD C1 ' subtype
  BItem2 1 BCD C4 ' cutting edge position
  ```

- Block
  `Block<n> <repetition rule>`
  `<n>` := Consecutive number of block, ascending from 1 without gaps
  `<repetition rule>` := * `Item<n>`
  A block of data `BItem<n>` follows (up to keyword `End_Block<n>`) which is stored on the code carrier according to the `<repetition rule>`.
  In the case of `Block <n> * Item <n>` Item `<n>` must be defined before Block `<n>` .
  Example:
  `Block1 * Item6 ' Block1 corresponds to repeating the value from Item6`
  `End_Block<n>`
- End_Block
  `End_Block<n>`
  `<n>` := consecutive block number
  End identifier for a data block defined with "Block".

Code carrier data formats - <code carrier data format>

The following code-carrier data formats (see `Item` / `BItem`) are supported:

Table 13-3 Code carrier data formats

Data format	Explanation	Range of values
ASCII	ASCII character set	-
INT	16-bit integer (Intel format)	-32768 ... +32767
BCD	Binary-coded decimal number (if necessary, with sign and decimal point): Non-relevant decades are preassigned the value 0, left-justified.	-
REAL	16-bit floating-point number (Intel format): Accuracy, 7 digits	$\pm 1.5\times10^{-45}$ ... $\pm 3.4\times10^{38}$
DOUBLE	32-bit floating-point number (Intel format): Accuracy 15 ... 16 digits	$\pm 5.0\times10^{-324}$ ... $\pm 1.7\times10^{308}$
FPX2	Bit integer (SINUMERIK): Least significant byte at the higher order address (inverse to the Intel format)	-32768 ... +32767

Assignment between code carrier data and dialog data - <dialog variable>

The conversion rule for `Item<n>` or `BItem<n>` also contains the assignment to none/one/several dialog variables, if necessary with a conversion that is explained in detail in this section.

`<dialog variable> :=` `<dvar1>[=(<uv>)] [, <dvar2>[=(<uv>)]`
`[,<dvar3>[&<dvar4>]=(<uv>)]`
`[,dvarN>[=(<uv>)]`

`<dvar> :=` `T<index> | C<index> | _`
T = tool data
C = cutting edge data
index = index within tool / cutting edge dialog data
– = no assignment to a dialog variable

`<dvar1>&<dvar2>=` `<uv>`
Conversion rule applies to <dvar1> and <dvar2>

`uv :=` `<Tetn>`
Tetn := nth tetrad in byte sequence
Byte1, = Tet1 and Tet2
Byte2, = Tet3 and Tet4

Example of the tetrad allocation of the code carrier variables (in BCD format) to dialog variables:

```
T4=(Tet1), T5=(Tet2), T6=(Tet3), T7=(Tet4)
```

If the code carrier variable has the value 0x1234, for example, dialog variable T4 is assigned the value 1, dialog variable T7 the value 4.

13.2.6.2 Examples for the description files wkonvert.txt

After installation, the name of the file is "wkonvert.txt".

Table 13-4 Wkonvert.txt Items

Code carrier variable	Length (bytes)	Data format	Dialog variable	Comment
Item1	32	ASCII	T1	' Identifier
Item2	3	BCD	T2	' Duplo
Item3	2	BCD	T4=(Tet1), T5=(Tet2), T6=(Tet3), T7=(Tet4)	' Tool size: Left, right, top, bottom
Item4	32	ASCII	T8	' Location type
Item5	2	BCD	T9	' Status
Item6	1	BCD	T3	' Number of cutting edges
Item7	1	BCD	T10	' Type of tool monitoring
Item8	1	BCD	T11	' Type of tool monitoring
' Cutting edge data Block1 * Item6				
BItem1	2	BCD	C1	' Subtype, type
BItem2	3	BCD	C4	' Cutting edge position
' Tool length compensation				
BItem3	4	BCD	C5	' Length 1
BItem4	4	BCD	C5	' Length 2
BItem5	4	BCD	C7	' Length 3
' Tool radius compensation				
BItem6	4	BCD	C8	' Length 1
BItem7	4	BCD	C9	' Length 2
BItem8	4	BCD	C10	' Radius 1
BItem9	4	BCD	C11	' Radius 2
BItem10	4	BCD	C12	' Angle 1
BItem11	4	BCD	C13	' Angle 2
' Wear length compensation				

Code carrier variable	Length (bytes)	Data format	Dialog variable	Comment
BItem12	4	BCD	C14	' Length 1
BItem13	4	BCD	C15	' Length 2
BItem14	4	BCD	C16	' Length 3
' Wear radius compensation				
BItem15	4	BCD	C17	' Length 1
BItem16	4	BCD	C18	' Length 2
BItem17	4	BCD	C19	' Radius 1
BItem18	4	BCD	C20	' Radius 2
BItem19	4	BCD	C21	' Angle 1
BItem20	4	BCD	C22	' Angle 2
' Basic dimension length compensation				
BItem21	4	BCD	C23	' Basic length 1
BItem22	4	BCD	C24	' Basic length 2
BItem23	4	BCD	C25	' Basic length 3
BItem24	4	BCD	C26	' Undercut angle
BItem25	1	BCD	C27	' Reverse insert
BItem26	2	BCD	C29	' Tool life in minutes
BItem27	2	BCD	C30	' Pre-warning limit tool life
BItem28	2	BCD	C33	' Setpoint tool life in minutes
BItem29	2	BCD	C31	' Unit quantity
BItem30	2	BCD	C32	' Pre-warning limit unit quantity
BItem31	2	BCD	C32	' Setpoint unit quantity
End_Block1				

Example with tool and cutting edge OEM data as well as data gaps:

Table 13-5 OEM data for Wkonvert.txt

Code carrier variable	Length (bytes)	Data format	Dialog variable	Comment
Item1	32	ASCII	T1	' Identifier
Item2	3	BCD	T2	' Duplo
Item3	2	BCD	T4=(Tet1), T5=(Tet2), T6=(Tet3), T7=(Tet4)	' Tool size: Left, right, top, bottom
Item4	32	ASCII	T8	' Location type
Item5	2	BCD	T9	' Status
Item6	1	BCD	T3	' Number of cutting edges
Item7	1	BCD	T10	' Type of tool monitoring
Item8	1	BCD	T11	' Type of tool monitoring
Item9	5	ASCII	-	' Tool data gap
Item10	4	BCD	A1	' Tool OEM data 1

Code carrier variable	Length (bytes)	Data format	Dialog variable	Comment
' Cutting edge data Block1 * Item6				
BItem1	2	BCD	C1	' Subtype, type
BItem2	1	BCD	C4	' Cutting edge position
' Tool length compensation				
BItem3	4	BCD	C5	' Length 1
BItem4	4	BCD	C5	' Length 2
BItem5	4	BCD	C7	' Length 3
' Tool radius compensation				
BItem6	4	BCD	C8	' Length 1
BItem7	4	BCD	C9	' Length 2
BItem8	4	BCD	C10	' Radius 1
BItem9	4	BCD	C11	' Radius 2
BItem10	4	BCD	C12	' Angle 1
BItem11	4	BCD	C13	' Angle 2
' Wear length compensation				
BItem12	4	BCD	C14	' Length 1
BItem13	4	BCD	C15	' Length 2
BItem14	4	BCD	C16	' Length 3
' Wear radius compensation				
BItem15	4	BCD	C17	' Length 1
BItem16	4	BCD	C18	' Length 2
BItem17	4	BCD	C19	' Radius 1
BItem18	4	BCD	C20	' Radius 2
BItem19	4	BCD	C21	' Angle 1
BItem20	4	BCD	C22	' Angle 2
' Basic dimension length compensation				
BItem21	4	BCD	C23	' Basic length 1
BItem22	4	BCD	C24	' Basic length 2
BItem23	4	BCD	C25	' Basic length 3
BItem24	4	BCD	C26	' Undercut angle
BItem25	1	BCD	C27	' Reverse insert
BItem26	2	BCD	C29	' Tool life in minutes
BItem27	2	BCD	C30	' Pre-warning limit tool life
BItem28	2	BCD	C33	' Setpoint tool life in minutes
BItem29	2	BCD	C31	' Unit quantity
BItem30	2	BCD	C32	' Pre-warning limit unit quantity
BItem31	2	BCD	C32	' Setpoint unit quantity
BItem32	4	BCD	U1	' Cutting edge OEM data 1
BItem33	3	BCD	-	' Cutting edge data gap
End_Block1				

13.2.6.3 Tool data initialization file DefToolDat.txt

The default data contained in the "DefToolDat.txt" file initializes the tool data that is not available in the data source for the services of Tool Ident Connection. This situation can occur when reading data from the code carrier from an import file or from the production host computer. The data of a default tool with a cutting edge in the known NC-$ syntax is located in this file.

The contents of "DefToolDat.txt" must be adapted by the user when configuring Tool Ident Connection. By removing the comment character in front of the appropriate optional tool data, its quantity should be adapted to the existing NC configuration. Also these values assigned to the tool data in the file can be modified according to the users requirements.

Example of a description file DefToolDat.txt

```
; Example of a tool default data file.
; NOTICE: Certain data values may not be practical
in this example!
;
;Tool Data
$TC_TP1[1]=1                    ; Duplo Number
$TC_TP2[1]=1                    ; Tool Name
$TC_TP3[1]=1                    ; Tool size left
$TC_TP4[1]=1                    ; Tool size right
$TC_TP5[1]=1                    ; Tool size up
$TC_TP6[1]=1                    ; Tool size low
$TC_TP7[1]=1                    ; Mag. location type
$TC_TP8[1]=2                    ; Tool Status
$TC_TP9[1]=0                    ; Watchdog type
$TC_TP10[1=0                    ;
;
;OEM Tool Data
;$TC_TPC1[1]=1.0
;$TC_TPC2[1]=2.0
;$TC_TPC3[1]=3.0
;$TC_TPC4[1]=4.0
;$TC_TPC5[1]=5.0
;$TC_TPC6[1]=6.0
;$TC_TPC7[1]=7.0
;$TC_TPC8[1]=8.0
;$TC_TPC9[1]=9.0
;$TC_TPC10[1]=10.0
;
;Cutting Edge Data
```

```
$TC_DP1[1,1]=120              ; Tool Sub-Type
$TC_DP2[1,1]=0                ; Cutting Edge Orientation
$TC_DP3[1,1]=0.0              ; Geom Length 1
$TC_DP4[1,1]=0.0              ; Geom Length 2
$TC_DP5[1,1]=0.0              ; Geom Length 3
$TC_DP6[1,1]=0.0              ; Geom Radius 1
$TC_DP7[1,1]=0.0              ; Geom Radius 2
$TC_DP8[1,1]=0.0              ; Geom-Radius Length 1
$TC_DP9[1,1]=0.0              ; Geom-Radius Length 2
$TC_DP10[1,1]=0.0             ; Geom-Radius Angle 1
$TC_DP11[1,1]=0.0             ; Geom-Radius Angle 2
$TC_DP12[1,1]=0.0             ; Wear Length 1
$TC_DP13[1,1]=0.0             ; Wear Length 2
$TC_DP14[1.1] =0.0            ; Wear Length 3
$TC_DP15[1,1]=0.0             ; Wear Radius 1
$TC_DP16[1,1]=0.0             ; Wear Radius 2
$TC_DP17[1,1]=0.0             ; Wear-Radius Length 1
$TC_DP18[1,1]=0.0             ; Wear-Radius Length 2
$TC_DP19[1,1]=0.0             ; Wear-Radius Angle 1
$TC_DP20[1,1]=0.0             ; Wear-Radius Angle 2
$TC_DP21[1,1]=0.0             ; Base Length 1
$TC_DP22[1,1]=0.0             ; Base Length 2
$TC_DP23[1,1]=0.0             ; Base Length 3
$TC_DP24[1,1]=0.0             ; Tool clearance angle
$TC_DP25[1,1]=0               ; Use of tool reversed
;
;OEM Cutting Edge Data
;$TC_DPC1[1,1]=0.0
;$TC_DPC2[1,1]=20.0
;$TC_DPC3[1,1]=30.0
;$TC_DPC4[1,1]=40.0
;$TC_DPC5[1,1]=50.0
;$TC_DPC6[1,1]=60.0
;$TC_DPC7[1,1]=70.0
;$TC_DPC8[1,1]=80.0
;$TC_DPC9[1,1]=90.0
;$TC_DPC10[1,1]=100.0
;
;Edge monitoring Data
$TC_MOP1[1,1]=0.0             ; Prewarning limit [min]
$TC_MOP2[1,1]=0.0             ; DownTime [min]
$TC_MOP3[1,1]=0               ; Prewarning limit [part]
$TC_MOP4[1,1]=0               ; Part countdown
$TC_MOP5[1,1]=0               ; Wear prewarning limit
$TC_MOP6[1,1]=0               ; Wear
$TC_MOP11[1,1]=0              ; Target service life [min]
$TC_MOP13[1,1]=0              ; Target number of parts
$TC_MOP15[1,1]=0              ; Target wear
```

```
;OEM edge monitoring data
;$TC_MOPC1[1,1]=0
;$TC_MOPC2[1,1]=0
;$TC_MOPC3[1,1]=0
;$TC_MOPC4[1,1]=0
;$TC_MOPC5[1,1]=0
;$TC_MOPC6[1,1]=0
;$TC_MOPC7[1,1]=0
;$TC_MOPC8[1,1]=0
;$TC_MOPC9[1,1]=0
;$TC_MOPC10[1,1]=0
;
;Location-dependent additive cutting edge offsets
;$TC_SCP13[1,1]=0.0
;$TC_SCP14[1,1]=0.0
;$TC_SCP15[1,1]=0.0
;$TC_SCP16[1,1]=0.0
;$TC_SCP17[1,1]=0.0
;$TC_SCP18[1,1]=0.0
;$TC_SCP19[1,1]=0.0
;$TC_SCP20[1,1]=0.0
;$TC_SCP21[1,1]=0.0
M17
```

13.2.7 Conversion rule based on the MCIS-C script language

Tool Ident Connection enables conversion rules to be created for the code carrier based on the MCIS-C script language (Motion Control Information System (MCIS)). Using the script language, the user obtains additional functions to interpret the data on the code carrier.

Complex code carrier formats can also be processed by using control structures (e.g. if) in conjunction with relational operators (e.g. >, <).

Note

Optimized blocks

The C functions described in this chapter cannot be used for optimized data blocks.

13.2.7.1 Structure of the conversion rule

The conversion rule is generated as ASCII file with the MCC extension (MCIS-C). The file has the following structure:

```
Line          Program code
1             _VersionInfo := "Sample for WKONVERT-Logic";
2
3             TmTool Tool = new TmTool(); // Tool-Object
4             unsigned char ToolArray[0]; // Array-Object
5
```

```
6           void DecodeToolData()
7           {
8                       Tool.TC_TP2 = readASCII( ToolArray, 2, 10 );
9                       ...
10          }
11
12          void EncodeToolData()
13          {
14                      writeASCII( ToolArray, 2, 10, Tool.TC_TP2 );
15                      ...
16          }
```

Line 1: Using the construct _VersionInfo, users can save their own version information within the conversion rule.

Line 3: A TmTool object is defined with the name Tool in this line. Using this object, it is possible to access individual tool / cutting edge data of a tool.

Line 4: An array object with the ToolArray name is defined in this line. Using this object, data on the code carrier can be accessed (reading or writing). Initially, the data carrier size is pre-assigned 0. The actual size of the code carrier is determined during the run time and can be read with the construct: ToolArray.ArraySize(). Individual bytes of the code carrier are accessed using the construct: ToolArray[offset], where the value of the offset may assume values from 0 to ToolArray.ArraySize (-1).

In addition to directly accessing the individual bytes of the code carrier, additional help functions can be used. The auxiliary functions are always required if data is to be read or written in a specific coding (e.g. readBCD()/writeBCD()).

Line 6: The function with the name DecodeToolData() is always called from Tool Ident Connection if data has been read from the code carrier. When this function is called, the array object contains: ToolArray where all of the code carrier data is saved. The tool object: Tool initially does not contain any data and is "filled" with data within these functions.

Line 8: This line represents as example the procedure within the DecodeToolData() function. Here, the tool identifier – using the help function readASCII() – is read-out of the code carrier data and assigned to the data element TC_TP2 of the tool data set.

Line 12: Tool Ident Connection always calls the function with the name EncodeToolData() if data is to be written to the code carrier. When this function is called, the array object contains: ToolArray the actual data from the code carrier (this is initially read each time before writing to the code carrier). The tool object contains data of the tool, which should be written to the code carrier.

Line 14 This line represents as example the procedure within the EncodeToolData() function. Here, the tool identifier TC_TP2 is written to the code carrier using the help function writeASCII(). This line represents as example the procedure within the EncodeToolData() function. Here, the tool identifier TC_TP2 is written to the code carrier using the help function writeASCII().

The identifiers used: Tool, ToolArray, DecodeToolData, EncodeToolData must **not** be changed.

13.2.7.2 Access to tool data

Within the conversion rule, individual tool data can be accessed reading and writing via the tool object. All of the data elements that are available are listed in this section. The identifiers used are oriented to the tool manager (e.g. Tool.TC_TP2 - tool identifier). The meaning of the individual tool data should be taken from the description of the tool management in Section "NC programming (Page 115)". Cutting edge-related data is accessed by specifying the cutting edge number (e.g. Tool.TC_DP2[2] – cutting edge position of the cutting edge with cutting edge number 2).

Identifier	Designation	MCIC-C data type
Tool-related data		
TC_TP1	Duplo number	long
TC_TP2	Tool identifier	string
TC_TP3	Size on left	long
TC_TP4	Size on right	long
TC_TP5	Size above	long
TC_TP6	Size below	long
TC_TP7	Magazine location type	long
TC_TP8	Status	long
TC_TP9	Tool monitoring method	long
TC_TP10	Replacement change strategy	long
TC_TP11	Tool information	long
TC_TP_PROTA	Name for the protection area	string
TC_TP_MAX_VELO	Maximum tool speed	double
TC_TP_MAX_ACC	Maximum tool acceleration	double
Tool-related user data		
TC_TPC1	OEM user data 1	double
...	...	...
TC_TPC64	OEM user data 64 (as a double)	double
TC_TPC1Str	OEM user data 1 (as a string)	string
...	...	...
TC_TPC64Str	OEM user data 64 (as a string)	double
Cut parameters		
TC_DP1 [EdgeNo]	Tool type	long
TC_DP2 [EdgeNo]	Cutting edge position	double
TC_DP3 [EdgeNo]	Geometry length 1	double
TC_DP4 [EdgeNo]	Geometry length 2	double
TC_DP5 [EdgeNo]	Geometry length 3	double
TC_DP6 [EdgeNo]	Geometry radius	double
TC_DP7 [EdgeNo]	Geometry - corner radius	double
TC_DP8 [EdgeNo]	Geometry length 4	double
TC_DP9 [EdgeNo]	Geometry length 5	double
TC_DP10 [EdgeNo]	Geometry - angle 1	double
TC_DP11 [EdgeNo]	Geometry - angle 2	double

Identifier	Designation	MCIC-C data type
TC_DP12 [EdgeNo]	Wear - length 1	double
TC_DP13 [EdgeNo]	Wear - length 2	double
TC_DP14 [EdgeNo]	Wear - length 3	double
TC_DP15 [EdgeNo]	Wear - radius	double
TC_DP16 [EdgeNo]	Wear - rounding radius	double
TC_DP17 [EdgeNo]	Wear - projection	double
TC_DP18 [EdgeNo]	Wear - length 5	double
TC_DP19 [EdgeNo]	Wear - angle 1	double
TC_DP20 [EdgeNo]	Wear - angle 2	double
TC_DP22 [EdgeNo]	Adapter - length 1	double
TC_DP23 [EdgeNo]	Adapter - length 2	double
TC_DP24 [EdgeNo]	Adapter - length 3	double
TC_DP25 [EdgeNo]	Clearance angle	double
TC_DPCE [EdgeNo]	Cutting edge number CE	long
TC_DPH [EdgeNo]	H parameter	double
TC_DPV [EdgeNo]	Tool cutting edge orientation	double
TC_DPV3 [EdgeNo]	L1 component of the tool cutting edge orientation	double
TC_DPV4 [EdgeNo]	L2 component of the tool cutting edge orientation	double
TC_DPV5 [EdgeNo]	L3 component of the tool cutting edge orientation	double
TC_DPVN3	L1 component of the orientation normals	double
TC_DPVN4	L2 component of the orientation normals	double
TC_DPVN5	L3 component of the orientation normals	double
TC_DPNT	Number of teeth of this cutting edge (cutting edge parameter 34)	long
TC_DPROT	Basic angle of rotation of cutting edge Note: The TDI Wkonvert-Wizard version 02.03.00.03 and higher is used to access variable TC_DPROT	double
$TC_DP_RADIUS3	Geometry - radius 3	double
$TC_DP_WEAR_RADIUS3	Wear - radius 3	double
$TC_DP_LEN6	Geometry - length 6	double
$TC_DP_WEAR_LEN6	Wear - length 6	double
Cutting edge supplementary information		
TC_DPC1 [EdgeNo]	OEM cutting edge user data 1	double
...	...	...
TC_DPC10 [EdgeNo]	OEM cutting edge user data 10	double
Cutting edge-related tool monitoring		
TC_MOP1 [EdgeNo]	Prewarning limit service life	double
TC_MOP2 [EdgeNo]	Residual tool life	double
TC_MOP3 [EdgeNo]	Prewarning limit unit quantity	long
TC_MOP4 [EdgeNo]	Residual unit quantity	long
TC_MOP11 [EdgeNo]	Reference tool life	double
TC_MOP13 [EdgeNo]	Unit quantity reference	long

Identifier	Designation	MCIC-C data type
TC_MOP5 [EdgeNo]	Wear prewarning limit	double
TC_MOP6 [EdgeNo]	Wear actual value	double
TC_MOP15 [EdgeNo]	Wear setpoint	double
User cutting edge monitoring		
TC_MOPC1	OEM user cutting edge monitoring[1]	long
...		...
TC_MOPC10	OEM user cutting edge monitoring[10]	long
Help attributes		
EDGE_NUM	Number of cutting edges (read only)	long
EDGE_NO [EdgeIdx]	Cutting edge number (read only) Supplies the cutting edge number of a cutting edge using the cutting edge index. The cutting edge index is a value between 1 and Tool.EDGE_NUM . Example: Tool with three cutting edges:1, 3, 5 Tool.EDGE_NUM supplies 3 Tool.EDGE_NO[1] supplies 1 Tool.EDGE_NO[2] supplies 3 Tool.EDGE_NO[3] supplies 5 Tool.TC_DP2[5] cutting edge position of the cutting edge with cutting edge number 5.	individual

13.2.7.3 Auxiliary functions for numerical coding

The tool data can be saved on the code carrier using various codings. The following auxiliary functions are available when reading and writing this data within MCIS-C:

Note

Syntax

All of these functions always have as the first parameter, the array object: Array and an offset within this array as second parameter. The offset can assume values between 0 and ToolArray.ArraySize()-1. For data types with a variable length, the length of the data on the code carrier is also transferred, (e.g. readASCII(array, offset, length)).

readASCII() / writeASCII()

```
string      readASCII ( array, offset, length )
            writeASCII ( array, offset, length, string [,precision] )
```

The functions allow ASCII strings to be read and written to. When writing, the string is aligned "right justified". If the string that is written is longer than the parameter length, then the excess characters are cut off. If the string is shorter, then the written range is filled with spaces. When reading, leading spaces and spaces at the end of a line are ignored. The writeASCII() function contains an optional "precision" parameter with which the number of

decimal places can be specified. If this parameter is not used, then the number of decimal places configured in SINUMERIK is used.

Examples:

```
writeASCII ( ToolArray, 0, 5, "ABC" );
readASCII ( ToolArray, 0, 5 ); => "ABC"
```

Hex value	41	42	43	20	20										
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

```
writeASCII ( ToolArray, 0, 5, "ABCDEFGH" );
readASCII ( ToolArray, 0, 5 ); => "ABCDE"
```

Hex value	41	42	43	44	45										
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

```
writeASCII ( ToolArray, 0, 5, 3.94 );
writeASCII ( ToolArray, 5, 5, 3.94, 0 );
writeASCII ( ToolArray, 10, 5, 3.94, 1 );
readASCII ( ToolArray, 0, 5 ); => "3.94"
readASCII ( ToolArray, 5, 5 ); => "00004"
readASCII ( ToolArray, 10, 5 ); => "003.9"
```

Hex value	30	33	2E	39	34	30	30	30	30	34	30	30	33	2E	39	
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

readBCD() / writeBCD()

```
double      readBCD ( array, offset, length )
            writeBCD ( array, offset, length, val [,precision] )
```

The functions allow BCD numbers to be read and written.

Examples:

```
writeBCD ( ToolArray, 0, 5, 3.11 );
readBCD ( ToolArray, 0, 5 ); => "3.11"
writeBCD ( ToolArray, 5, 5, -47.11 );
readBCD ( ToolArray, 0, 5 ); => "-47.11"
writeBCD ( ToolArray, 10, 5, 4711 );
readBCD ( ToolArray, 10, 5 ); => "4711"
```

Hex value	B0	00	00	3E	11	D0	00	04	7E	11	B0	00	00	47	11	
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

```
writeBCD ( ToolArray, 0, 5, -3.1234 );
readBCD ( ToolArray, 0, 5 ); => "-3.12"
writeBCD ( ToolArray, 5, 5, 1.2 );
readBCD ( ToolArray, 5, 5 ); => "1.2"
```

Hex value	D0	00	00	3E	32	B0	00	04	1E	20						
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

```
writeBCD ( ToolArray, 0, 4, 3.94 );
readBCD ( ToolArray, 0, 4 ); => "3.94"
writeBCD ( ToolArray, 4, 4, 3.94, 0 );
readBCD ( ToolArray, 4, 4 ); => "4"
writeBCD ( ToolArray, 8, 4, 3.94, 1 );
readBCD ( ToolArray, 8, 4 ); => "3.9"
writeBCD ( ToolArray, 12, 4, 3.94, 2 );
readBCD ( ToolArray, 12, 4 ); => "3.94"
```

Hex value	B0	00	3E	94	B0	00	00	04	B0	00	03	E9	B0	00	3E	94
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

readINT() / writeINT()

```
int         readINT ( array, offset )
            writeINT ( array, offset, val )
```

The functions allow signed 16-bit numbers to be read and written. The data is stored in the S7 format (Big Endian).

Example:

```
writeINT ( ToolArray, 0, 1 );
readINT ( ToolArray, 0, ); => "1"
writeINT ( ToolArray, 2, 4711 ); // hex: 1267
readINT ( ToolArray, 2 ); => "4711"
writeINT ( ToolArray, 4, -300 ); // hex: FED4
readINT ( ToolArray, 4 ); => "-300"
writeINT ( ToolArray, 6, -1 ); // hex: FFFF
readINT ( ToolArray, 6 ); => "-1"
```

Hex value	00	01	12	67	FE	D4	FF	FF								
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

readDINT() / writeDINT()

```
long        readDINT ( array, offset )
            writeDINT ( array, offset, val )
```

The functions allow signed 32-bit numbers to be read and written. The data is stored in the S7 format (Big Endian).

Example:

```
writeDINT ( ToolArray, 0, 1234 ); // hex: 4D2
readDINT ( ToolArray, 0, ); => "1234"
writeDINT ( ToolArray, 4, -1234 ); // hex: FFFFFB2E
readDINT ( ToolArray, 4 ); => "-1234"
```

Hex value	00	00	04	D2	FF	FF	FB	2E								
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

readWORD() / writeWORD()

```
long        readWORD ( array, offset )
            writeWORD ( array, offset, val )
```

The functions allow unsigned 16-bit numbers to be read and written. The data is stored in the S7 format (Big Endian).

Example:

```
writeWORD ( ToolArray, 0, 1 ); // hex: 1
readWORD ( ToolArray, 0, ); => "1"
writeWORD ( ToolArray, 2, 50000 ); // hex: C350
readWORD ( ToolArray, 2, ); => "50000"
```

Hex value	00	01	C3	50												
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

readDWORD() / writeDWORD()

```
unsigned long          readDWORD ( array, offset )
                       writeDWORD ( array, offset, val )
```

The functions allow unsigned 32-bit numbers to be read and written. The data is stored in the S7 format (Big Endian).

Example:

```
writeDWORD ( ToolArray, 0, 12345678 ); // hex: 075BCD15
readDWORD ( ToolArray, 0, ); => "12345678"
writeDWORD ( ToolArray, 4, 987654321 ); // hex: 3ADE68B1
readDWORD ( ToolArray, 4, ); => "987654321"
```

Hex value	07	5B	CD	15	3A	DE	68	B1								
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

readREAL() / writeREAL()

```
float       readREAL ( array, offset )
            writeREAL ( array, offset, val )
```

The functions allow 32-bit floating-point numbers to be read and written. The data is stored in the S7 format (REAL).

Example:

```
writeREAL ( ToolArray, 0, 3.11f );
readREAL ( ToolArray, 0, ); => "3.11"
writeREAL ( ToolArray, 4, 08.15f );
readREAL ( ToolArray, 4, ); => "08.15"
```

Hex value	40	47	0A	3D	41	02	66	66								
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Note

The notation 3.11f ensures that the constant 3.11 is interpreted as 32-bit floating-point number (float). 64-bit (double) is used as default.

readDOUBLE() / writeDOUBLE()

```
double      readDOUBLE ( array, offset )
            writeDOUBLE ( array, offset, val )
```

The functions allow 64-bit floating-point numbers to be read and written. The data correspond to the IEEE 754 standard.

Example:

```
writeDOUBLE ( ToolArray, 0, 3.11 );
readDOUBLE ( ToolArray, 0, ); => "3.11"
writeDOUBLE ( ToolArray, 8, 08.15 );
readDOUBLE ( ToolArray, 8, ); => "08.15"
```

Hex value	40	08	E1	47	AE	14	7A	E1	40	20	4C	CC	CC	CC	CC	CD
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

ReverseByteOrder ()

```
ReverseByteOrder ( val )
```

The functions xxxINT(), xxxDINT(), xxxWORD() and xxxDWORD() read and write values in the Big-Endian format. This corresponds to S7 formats. If values are to be read / written in the Little-Endian format, then the function ReverseByteOrder() can be used for this purpose.

Example:

```
writeINT ( ToolArray, 0, 1 );
writeINT ( ToolArray, 2, ReverseByteOrder(1) );
ReverseByteOrder( readINT( ToolArray, 2 ) ); => "1"
writeDINT ( ToolArray, 4, 4711 );
writeDINT ( ToolArray, 8, ReverseByteOrder(4711L) );
ReverseByteOrder( readDINT( ToolArray, 8 ) ); => "4711"
```

Hex value	00	01	01	00	00	00	12	67	67	12	00	00				
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Note

The notation 4711L ensures that the constant 4711 is interpreted as 32-bit constant. 16-bit (int) is used as default.

13.2.7.4 Configuration of ToolSpec.xml

PlaceTypeToString()/PlaceTypeFromString ()

The location type of a tool (TC_TP7) is always listed as a numeric value in SINUMERIK Operate. A string value is used for the location type in the HM. The assignment between the numeric value and the string value is performed in the WZACCESS.MDB database.

To guarantee the compatibility of code carrier formats between HMI-Advanced and SINUMERIK Operate, the "ToolSpec.xml" configuration file is introduced in Tool Ident Connection. This file must be saved in the control in the same directory as the mcx file.

Example 1:

```
<!--The following data structure is used to convert the location
type information ($TC_TP7) from a numeric value to a string value
and vice versa. The string comparison is case-sensitive in the
conversion from text → code-->
<ToolSpecConversion>
   <ToolSpec code="1" text="normal"/>
```

```
   <ToolSpec code="2" text="large"/>
   <ToolSpec code="3" text="small"/>
</ToolSpecConversion>
```

The ToolSpec.xml can be accessed within the conversion rule with the `PlaceTypeToString()` and `PlaceTypeFromString()` auxiliary functions.

```
PlaceTypeToString (NumType)
PlaceTypeFromString (StrType)
```

Example 2:

```
PlaceTypeToString (1); ⇒ "normal"
PlaceTypeToString (2); ⇒ "large"
PlaceTypeToString (3); ⇒ "small"

PlaceTypeFromString ("normal"); ⇒ 1
PlaceTypeFromString ("large");  ⇒ 2
PlaceTypeFromString ("small");  ⇒ 3
```

13.2.7.5 Error handling

Handling errors

The tool data can be checked within one conversion rule based on the MCIS-C script language.

If the tool data check proves negative (e.g.: tool diameter too large, wrong tool type, etc.),

the loading procedure can be aborted with an error message.

The error message appears in field "Error_Code" on the PLC interface.

(See also PLC interface in DB1900 (Page 421))

For the application case described here, the user can use Error_Code from value range 0x80 ... 0xFF.

See also Error codes (Page 434) for error group 0B: Tool conversion error group in the appendix.

To initiate the error handling described here within the conversion rule, the conversion rule must be expanded to include the declaration of the Error_Code variable.

Example

```
Variable                  Error_Code         = new variable<int>();     // error code
TmTool                    Tool               = new TmTool();            // tool object
unsigned char             ToolArray[0];                                 // data of the data
carrier

void DecodeToolData()
{
  …
  int EdgeCount = readBCD( ToolArray, 71, 1 );
  if( EdgeCount > 4 )
  {
     Error_Code = 0xFF;
     return;
   }
   …
}
```

13.2.7.6 Description of the MCIS-C script language

In this section, relevant MCIS-C specific subjects are discussed, which must be taken into account when configuring logical functions.

Keywords

Keywords are reserved for the compiler as language elements and must not be used as names of classes, variables and functions.

bool	do	if	true
break	double	int	unsigned
case	else	long	void
char	false	return	while
continue	float	string	
default	for	switch	

Byte order

Data is written and read by the SINUMERIK 828D in the "Little Endian" byte order in the "Tool Ident Connection" function (the least significant byte is stored at the start address).

Data types

The following table lists all of the data types for the MCIS-C script language, which can be used for programming:

Data type	Variable	Value range
bool	1 bit	true/false
char	8 bits	-128 ... 127
int	16 bits	-32768 ... 32767
long	32 bits	-2147483648 ... 2147483647
unsigned char	8 bits	0 ... 255
unsigned int	16 bits	0 ... 65535
unsigned long	32 bits	0 ... 4294967295
float	32 bits	$1.175494351e^{-38}$ ... $3.402823466e^{+38F}$
double	64 bits	$2.2250738585072014e^{-308}$ ... $1.7976931348623158e^{+308}$
string	-	Characters must be encoded by the appropriate number of bytes (compare programming language C).

Operators and special characters

The MCIS-C script language supports the following operators and special characters:

!	*	-=	<=	[	}
!=	*=	/	=	]	~
%	+/-	/=	==	^	
%=	++	:	>	^=	
&	+=	;	>=	{	
&&	,	<	>>	|	
&=	-	<<	>>=	|=	
(	--	<<=	?	||	
)	.				

13.2.7.7 Auxiliary functions in conversion specifications

Standard C++ type conversions

In addition to the script language commands for creating conversion specifications, there are other standard C ++ commands available. The following commands convert, for example, strings into numerical data types (Double):

- atof
- _atof_l
- _wtof
- _wtof_l

A detailed description of these commands can be found, for example, in the Microsoft Knowledge Base.

13.2.8 Examples

Overview

The following source code examples are in the format of the MCIS-C script language for creating conversion examples for Tool Ident Connection.

SampleWkonvert.mcc

```
_VersionInfo := "SampleWkonvert.mcc: Example for WKONWERT logic";

TmTool    Tool = new TmTool();      // tool object
unsigned char ToolArray[0];          // data of the data carrier

void DecodeToolData()
{

  Tool.TC_TP1   = readWORD( ToolArray, 0 );                    // Duplo number          byte 00-01
  Tool.TC_TP2   = readASCII( ToolArray, 2, 10);                // tool name         byte 02-11
  Tool.TC_TP3   = ToolArray[12];                               // size to the left        byte 12
  Tool.TC_TP4   = ToolArray[13];                               // size to the right     byte 13
  Tool.TC_TP5   = ToolArray[14];                               // size to the top        byte 14
  Tool.TC_TP6   = ToolArray[15];                               // size to the bottom       byte 15
  Tool.TC_TP7   = readINT(ToolArray, 16 );                     // location type                byte 16-17

  int EdgeCount = readINT(ToolArray, 18 );                     // number of cutting edges         byte 18-19

  int EdgeIndex;                                              // running variable for cutting edges

  for ( EdgeIndex = 1 ; EdgeIndex <= EdgeCount ; EdgeIndex++ )
  {
    int EdgeOffset    = 20 + (EdgeIndex-1) * 24;          // offset at the beginning of the particular cutting edge
                                                          // 24: Size of the cutting edge data

    int EdgeNo              = readINT ( ToolArray, EdgeOffset +  0 );     // cutting edge number  byte 20-21
    Tool.TC_DP1[EdgeNo]   = readINT ( ToolArray, EdgeOffset +  2 );    // tool type            byte 22-23
    Tool.TC_DP2[EdgeNo]   = readREAL( ToolArray, EdgeOffset +  4 );   // cutting edge position         byte 24-27
    Tool.TC_DP3[EdgeNo]   = readREAL( ToolArray, EdgeOffset +  8 );   // geometry length 1   byte 28-31
    Tool.TC_DP4[EdgeNo]   = readREAL( ToolArray, EdgeOffset + 12 );   // geometry length 2  byte 32-35
    Tool.TC_DP5[EdgeNo]   = readREAL( ToolArray, EdgeOffset + 16 );   // geometry length 3  byte 36-39
    Tool.TC_DP6[EdgeNo]   = readREAL( ToolArray, EdgeOffset + 20 );   // geometry radius    byte 40-43
  }
}

void EncodeToolData()
{
  writeWORD ( ToolArray, 0,    Tool.TC_TP1 );                  // Duplo number          byte 00-01
  writeASCII( ToolArray, 2, 10, Tool.TC_TP2 );                 // tool name         byte 02-11

  ToolArray[12]    = Tool.TC_TP3;                              // size to the left        byte 12
  ToolArray[13]    = Tool.TC_TP4;                              // size to the right     byte 13
  ToolArray[14]    = Tool.TC_TP5;                              // size to the top        byte 14
  ToolArray[15]    = Tool.TC_TP6;                              // size to the bottom       byte 15

  writeINT  ( ToolArray, 16,   Tool.TC_TP7 );                  // location type                byte 16-17

  int EdgeCount = Tool.EDGE_NUM;

  writeINT   ( ToolArray, 18, EdgeCount );                     // number of cutting edges      byte 18-19

  int EdgeIndex;

  for ( EdgeIndex = 1 ; EdgeIndex <= EdgeCount ; EdgeIndex++ )
```

```
  {
    int EdgeNo     = Tool.EDGE_NO[EdgeIndex];                    // cutting edge number
    int EdgeOffset  = 20 + (EdgeIndex-1) * 24;                   // offset to the beginning of the particular cutting edge
                                                                 // 24: Size of the cutting edge data

    writeINT     ( ToolArray, EdgeOffset +  0, EdgeNo );         // cutting edge number     byte 20-21
    writeINT      ( ToolArray, EdgeOffset +  2, Tool.TC_DP1[EdgeNo] );  // tool type             byte 22-23
    writeREAL   ( ToolArray, EdgeOffset +  4, Tool.TC_DP2[EdgeNo] );  // cutting edge position          byte 24-27
    writeREAL   ( ToolArray, EdgeOffset +  8, Tool.TC_DP3[EdgeNo] );  // geometry length 1     byte 28-31
    writeREAL   ( ToolArray, EdgeOffset + 12, Tool.TC_DP4[EdgeNo] ); // geometry length 2     byte 32-35
    writeREAL   ( ToolArray, EdgeOffset + 16, Tool.TC_DP5[EdgeNo] ); // geometry length 3     byte 36-39
    writeREAL   ( ToolArray, EdgeOffset + 20, Tool.TC_DP6[EdgeNo] ); // geometry radius       byte 40-43
  }
}
```

SampleToolSize.mcc

```
_VersionInfo := "SampleToolSize.mcc: Example for WKONWERT logic";

TmTool Tool = new TmTool();       // tool object
unsigned char ToolArray[0];        // data of the data carrier

void DecodeToolData()
{

   Tool.TC_TP1   = readWORD( ToolArray, 0 );                          // Duplo number      byte 00-01
   Tool.TC_TP2   = readASCII( ToolArray, 2, 10);                       // tool name     byte 02-11
   Tool.TC_TP7   = readINT(ToolArray, 12 );                           // location type              byte 12-13

   int EdgeCount = readINT(ToolArray, 14 );                            // number of cutting edges     byte 14-15

   int EdgeIndex;                                                     // running variable for cutting edges
   int ToolSize = 1;                                                  // tool size in half locations

   for ( EdgeIndex = 1 ; EdgeIndex <= EdgeCount ; EdgeIndex++ )
   {
      int EdgeOffset    = 16 + (EdgeIndex-1) * 24;                        // offset that the beginning of the particular cutting
edge
                                                                      // 24: Size of the cutting edge data

      int EdgeNo           = readINT ( ToolArray, EdgeOffset +  0 );          // cutting edge number byte16-17

      Tool.TC_DP1[EdgeNo]    = readINT ( ToolArray, EdgeOffset +  2 );    // tool type          byte 18-19
      Tool.TC_DP2[EdgeNo]    = readREAL( ToolArray, EdgeOffset +  4 );  // cutting edge position        byte 20-23
      Tool.TC_DP3[EdgeNo]    = readREAL( ToolArray, EdgeOffset +  8 );  // geometry length 1  byte 24-27
      Tool.TC_DP4[EdgeNo]    = readREAL( ToolArray, EdgeOffset + 12 ); // geometry length 2  byte 28-31
      Tool.TC_DP5[EdgeNo]    = readREAL( ToolArray, EdgeOffset + 16 ); // geometry length 3  byte 32-35
      Tool.TC_DP6[EdgeNo]    = readREAL( ToolArray, EdgeOffset + 20 ); // geometry radius    byte 36-39

      double to radius = Tool.TC_DP6[EdgeNo];

      if       ( to radius >  30.0 && ToolSize < 4 )       ToolSize = 4;
      else if ( tool radius >= 20.0 && ToolSize < 3 )      ToolSize = 3;
      else if ( tool radius >= 10.0 && ToolSize < 2 )      ToolSize = 2;
   }

   Tool.TC_TP3     = ToolSize;                                        // size to the left
   Tool.TC_TP4     = ToolSize;                                        // size to the right
   Tool.TC_TP5     = ToolSize;                                        // size to the top
   Tool.TC_TP6     = ToolSize;                                        // size to the bottom
}

void EncodeToolData()
{
   writeWORD   ( ToolArray, 0,     Tool.TC_TP1 );                     // Duplo number byte 00-01
   writeASCII    ( ToolArray, 2, 10, Tool.TC_TP2 );                    // tool name       byte 02-11
   writeINT      ( ToolArray, 12,    Tool.TC_TP7 );                    // location type              Byte 12-13

   int EdgeCount = Tool.EDGE_NUM;

   writeINT      ( ToolArray, 14,    EdgeCount );                      // number of cutting edges     Byte 14-15
```

```
   int EdgeIndex;

   for ( EdgeIndex = 1 ; EdgeIndex <= EdgeCount ; EdgeIndex++ )
   {
      int EdgeNo          = Tool.EDGE_NO[EdgeIndex] ;                    // cutting edge number
      int EdgeOffset    = 16 + (EdgeIndex-1) * 24;                        // offset at the beginning of the particular cutting
edge
                                                                          // 24: Size of the cutting edge data

      writeINT ( ToolArray, EdgeOffset +  0, EdgeNo );                    // cutting edge number  byte 16-17
      writeINT ( ToolArray, EdgeOffset +  2, Tool.TC_DP1[EdgeNo] );     // tool type           byte 18-19
      writeREAL( ToolArray, EdgeOffset +  4, Tool.TC_DP2[EdgeNo] );   // cutting edge position         byte 20-23
      writeREAL( ToolArray, EdgeOffset +  8, Tool.TC_DP3[EdgeNo] );   // geometry length 1   byte 24-27
      writeREAL( ToolArray, EdgeOffset + 12, Tool.TC_DP4[EdgeNo] );  // geometry length 2   byte 28-31
      writeREAL( ToolArray, EdgeOffset + 16, Tool.TC_DP5[EdgeNo] );  // geometry length 3   byte 32-35
      writeREAL( ToolArray, EdgeOffset + 20, Tool.TC_DP6[EdgeNo] );  // geometry radius     byte 36-39
   }
}
```

13.3 PLC interface in DB1900

13.3.1 Function call of the PLC interface

Assignment of the PLC interface

The interface occupies 6 bytes in the DB1900 and has the following assignment:

Address 828D	Identifier	Type	Value range	Meaning
DB1900.DBX3110.0	A_Request	BOOL	ON, OFF	Request task
DB1900.DBB3111	A_K_Code	BYTE	0...127	Command code
DB1900.DBX3112.0	E_Done	BOOL	ON, OFF	Task successfully completed
DB1900.DBX3112.1	E_Error	BOOL	ON, OFF	Task ended with error
DB1900.DBX3112.2	E_Active	BOOL	ON, OFF	Task is being processed
DB1900.DBX3112.7	E_Ident_Ready	BOOL	ON, OFF	Status for power up
DB1900.DBB3113	E_K_CodeError	BYTE	0 ... 127	K code in case of an error
DB1900.DBW3114	E_Error_Code	INT	0...0xFFFF	Error code

13.3.2 Assignment of the PLC for transfer parameters

Assignment of the PLC interface

The required transfer parameters are also stored in DB1900; this data area occupies 56 bytes. The data type "STRING" is not available; instead it is substituted by BYTE.

The data area has the following assignment:

Address 828D	Identifier	Type	Value range	Meaning
DB1900.DBW3000	A_Subtype	INT	-1 ... 32000	Subtype of the tool
DB1900.DBW3002	A_Duplo	INT	-1 ... 32000	Duplo number of the tool
DB1900.DBB3004	A_Ident_Max	BYTE	32	Max. length of the identifier
DB1900.DBB3005	A_Ident_Len	BYTE	1 ... 32	Length of the identifier
DB1900.DBB3006 ... DB1900.DBB3037	A_Ident_1 ... A_Ident_32	BYTE	ASCII (32 ... 255)	Tool identifier
DB1900.DBW3038	A_Tnumber	INT	-1 ... 32000	T number of the tool
DB1900.DBW3040	A_MagazinePlace	INT	-1 ... 32000	Magazine location number of the tool
DB1900.DBW3042	A_MagazinePlaceType	INT	-1 ... 32000	Magazine location type of the tool
DB1900.DBB3044	A_ToolSize_Upper	BYTE	1 ... 127	Tool size top in half locations
DB1900.DBB3045	A_ToolSize_Down	BYTE	1 ... 127	Tool size bottom in half locations
DB1900.DBB3046	A_ToolSize_Left	BYTE	1 ... 127	Tool size left in half locations
DB1900.DBB3047	A_ToolSize_Right	BYTE	1 ... 127	Tool size right in half locations
DB1900.DBB3048	A_Channel	BYTE	1 ... 127	Channel number of the tool
DB1900.DBB3049	A_LoadingPlace	BYTE	1 ... 127	Number of the loading/unloading location
DB1900.DBW3050	A_Magazine	INT	0 ... 32000	Magazine number
DB1900.DBB3052	A_Unit	BYTE	-1 ... 127	Code carrier number
DB1900.DBB3053	A_NCU_Index	BYTE	-1 ... 127	Name of the NCU
DB1900.DBW3054	A_FileIndex	INT	0 ... 32000	File index (import/export)
DB1900.DBX3056.0	A_ActiveEnable	BOOL	ON, OFF	Hide status bit "Tool active"
DB1900.DBX3056.1	A_Override	BOOL	ON, OFF	Overwrite file
DB1900.DBX3056.2	A_LoadTool	BOOL	ON, OFF	Load tool after creation
DB1900.DBX3056.3	A_DeleteTool	BOOL	ON, OFF	Delete tool after unloading

13.3.3 Assignment of the PLC for return parameters

Assignment of the PLC interface

The return values are stored in DB1900, this data area occupies 50 bytes. The data type "STRING" is not available; instead it is substituted by BYTE.

The data area has the following assignment:

Address 828D	Identifier	Type	Value range	Meaning
DB1900.DBW3058	E_Subtype	INT	-1 ... 32000	Subtype of the tool
DB1900.DBW3060	E_Duplo	INT	-1 ... 32000	Duplo number of the tool
DB1900.DBB3062	E_Ident_Max	BYTE	32	Max. length of the identifier
DB1900.DBB3063	E_Ident_Len	BYTE	1 ... 32	Length of the identifier
DB1900.DBB3064 ... DB1900.DBB3095	E_Ident_1 ... E_Ident_32	BYTE	ASCII (32 ... 255)	Tool identifier
DB1900.DBW3096	E_Tnumber	INT	-1 ... 32000	T number of the tool
DB1900.DBW3098	E_ToolState	INT	-1 ... 32000	Tool status
DB1900.DBW3100	E_MagazinePla-ceType	INT	-1 ... 32000	Magazine location type of the tool
DB1900.DBB3102	E_ToolSize_Up-per	BYTE	1 ... 127	Tool size top in half locations
DB1900.DBB3103	E_Tool-Size_Down	BYTE	1 ... 127	Tool size bottom in half locations
DB1900.DBB3104	E_ToolSize_Left	BYTE	1 ... 127	Tool size left in half locations
DB1900.DBB3105	E_Tool-Size_Right	BYTE	1 ... 127	Tool size right in half locations
DB1900.DBW3106	E_Magazine	INT	0 ... 32000	Magazine number of the tool
DB1900.DBW3108	E_Magazine-Place	INT	0 ... 32000	Magazine location number of the tool

13.3.4 PLC interface for the Parameter operating area

Assignment of the PLC interface

The PLC interface is used in conjunction with the following functions:

- New tool from code carrier
- Unload tool/delete from code carrier

These are also stored in DB1900, the structure has the following assignment

Address 828D	Identifier	Type	Value range	Meaning
DB1900.DBX3116.0	E_Req	BOOL	ON, OFF	Requests the task
DB1900.DBX3118.0	A_Done	BOOL	ON, OFF	Task successfully completed
DB1900.DBX3118.1	A_Error	BOOL	ON, OFF	Task ended with error
DB1900.DBX3118.2	A_Active	BOOL	ON, OFF	Task is being processed
DB1900.DBX3119.0	E_TDC_Read	BOOL	ON, OFF	Read tool data from CT
DB1900.DBX3119.1	E_TDC_Write	BOOL	ON, OFF	Write tool data to CT
DB1900.DBX3119.2	E_TDC_Cancel	BOOL	ON, OFF	Operator has canceled read/write operation

Address 828D	Identifier	Type	Value range	Meaning
DB1900.DBX3119.3	E_TDC_Read_before_Write	BOOL	ON, OFF	Selective writing of CT data
DB1900.DBW3120	A_Error_Code	INT	0 ... 0xFFFF	PLC error code
Transfer parameters				
DB1900.DBW3122	E_PMagazine	INT	0 ... 0xFFFF	Magazine number
DB1900.DBW3124	E_PMagazinePlace	INT	0 ... 0xFFFF	Magazine location number
DB1900.DBW3126	E_PTnumber	INT	0 ... 0xFFFF	T number

13.4 Command codes

13.4.1 Functional scope of the PLC interface (K codes)

Functional scope of the PLC interface

The PLC interface has a permanently defined range of functions. Every defined function/service is assigned a unique command code (K code). Depending on the command to be executed, before setting the request bit, various transfer parameters must be written to the PLC.

The following functions can be executed via Tool Ident Connection using the PLC interface:

K code	Description
0	Loading from the code carrier (K code = 0) (Page 425)
1	Unload to code carrier (K code = 1) (Page 426)
7	Loading a tool from the tool list (K code = 7) (Page 428)
8	Unloading a tool in the tool list (K code = 8) (Page 429)
9	Empty location search (K code = 9) (Page 430)
10	Delete tool in NC (K code = 10) (Page 431)
14	Updating the code carrier (K code = 14) (Page 432)
15	Reading data from the code carrier (K code = 15) (Page 433)

The individual functions of Tool Ident Connection are described in detail in the following sections. The optional and the mandatory transfer parameters are listed for each service.

There is also an overview of the return parameters supplied by Tool Ident Connection after the successful execution of a command code.

13.4.2 Parameter description

Transfer parameters for the services

The terms of the transfer parameters are described in the following table:

Parameter	Description
ActiveEnable	Transfer of the tool status bit from the database in the NC (TRUE) / hiding the status bit during the transfer (FALSE)
Channel	Channel number
DeleteTool	Delete tool after unloading (TRUE) / do not delete (FALSE)
Duplo	Duplo number
FileIndex	Index of the file from which data is to be imported or to which data is to be exported. The file name which is extended by the index can be specified in the INI file.
Ident	Tool name
LoadingPlace	Number of the loading/unloading location in the loading magazine
LoadTool	Load tool after reading (TRUE) / do not load (FALSE)
Magazines	Magazine number
MagazinePlace	Magazine location number of the tool
MagazinePlaceType	Magazine location type
NCU_Index	NCU index (parameter is presently not used)
Override	Overwrite export file (TRUE) / extend (FALSE)
Subtype	Subtype or type
TNumber	T number
ToolSize_Down	Tool size downwards in half locations
ToolSize_Left	Tool size to the left in half locations
ToolSize_Right	Tool size to the right in half locations
ToolSize_Upper	Tool size upwards in half locations
ToolState	Tool status
Unit	Code carrier unit (read/write head) from which data is read or to which data should be written. Depending on the unit: 1...4.

13.4.3 Loading from the code carrier (K code = 0)

Reads tool data from the code carrier; whose unit is specified with "Unit". The tool is then created if it is not already available in the NC. It is also loaded if "LoadTool" has been set. The number of the loading location in the loading magazine from "LoadingPlace" is used during this process if "LoadingPlace" is > 0. However, if "LoadingPlace" is set to 0, the value from "LoadPlaceNo" node is used in the configuration file as loading location (place) number.

The channel number (channel) and the number of the read head (unit) must be specified.

If the parameter "Magazine" is assigned with ">0" and "MagazinePlace" with "0", then an attempt is made to load the tool into an appropriate empty location in the specified magazine.

If the values for "Magazine" and "MagazinePlace" are > 0, then an attempt is made to load the tool to this magazine location.

In all other cases, the tool is loaded into the magazine in which an empty location is found.

Table 13-6 PLC parameters for "loading from code carrier"

PLC data	Identifier	Type	Value range	Meaning
Transfer parameters				
DBW3040	Magazine location	INT	-1 ... 32000	Magazine location number of the tool
DBB3048	Channel	BYTE	1 ... 127	TO area
DBB3049	Loading Place	BYTE	1 ... 127	Number of the loading/unloading location
DBW3050	Magazines	INT	0 ... 32000	Magazine number
DBB3052	Unit	BYTE	1 ... 127	Code carrier number
DBX3056.2	LoadTool	BOOL	True, False	Load tool after creation
Return parameters				
DBW3058	Subtype	INT	-1 ... 32000	Subtype of the tool
DBW3060	Duplo	INT	-1 ... 32000	Duplo number of the tool
DBW3096	T number	INT	-1 ... 32000	T number of the tool
DBW3098	ToolState	INT	-1 ... 32000	Tool status
DBW3100	Magazine location type	INT	-1 ... 32000	Magazine location type of the tool
DBB3102	ToolSize_Upper	BYTE	1 ... 127	Tool size top in half locations
DBB3103	ToolSize_Down	BYTE	1 ... 127	Tool size bottom in half locations
DBB3104	ToolSize_Left	BYTE	1 ... 127	Tool size left in half locations
DBB3105	ToolSize_Right	BYTE	1 ... 127	Tool size right in half locations
DBW3106	Magazines	INT	0 ... 32000	Magazine number of tool
DBW3108	Magazine place	INT	0 ... 32000	Magazine location number of tool

13.4.4 Unload to code carrier (K code = 1)

Unloads the tool that was parameterized using the transfer parameters. The channel number and the number of the write head must be specified. If the "Magazine" and "MagazinePlace" parameters are > 0, then an attempt is made to unload the tool, which is at this location. Otherwise the tool is selected with the parameters "Duplo" and "Ident" or via the parameter "TNumber".

In this case, the number of the unloading location in the loading magazine from "LoadingPlace" is used, if "LoadingPlace" is > 0. If "LoadingPlace" is set to 0, then the value from the "LoadPlaceNo" node is used in the configuration file as unload location number (loading point).

If "DeleteTool" is set, then the tool is also deleted from the NC after the unload operation.

Table 13-7 PLC parameters for "unload to the code carrier"

PLC data	Identifier	Type	Value range	Meaning
Transfer parameters				
DBW3002	Duplo	INT	0 ... 32000	Duplo number
DBW3038	TNumber	INT	0 ... 32000	T number
DBW3040	MagazinePlace	INT	0 ... 32000	Magazine location
DBB3048	Channel	BYTE	1 ... 127	TO area
DBB3049	Loading Place	BYTE	1 ... 127	Number of the loading/unloading location
DBW3050	Magazines	INT	0 ... 32000	Magazine number
DBB3052	Unit	BYTE	1 ... 127	Code carrier number
DBX3056.3	DeleteTool	BOOL	TRUE, FALSE	Delete tool after unloading
Return parameters				
None				

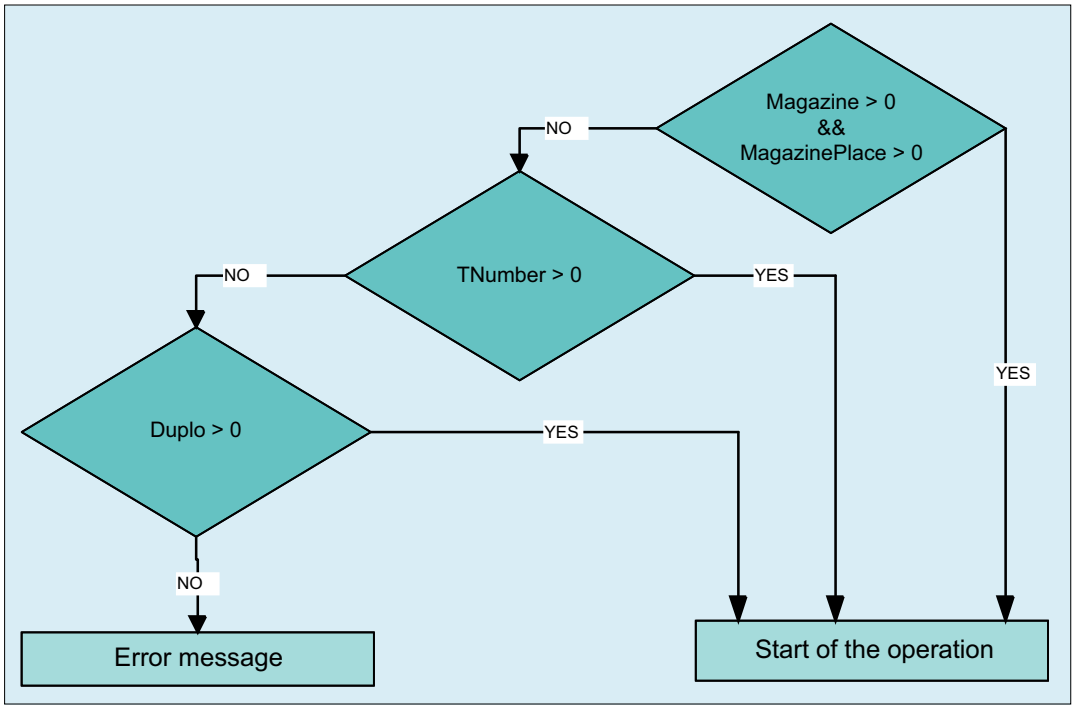


Figure 13-2 Flowchart to identify tools

13.4.5 Loading a tool from the tool list (K code = 7)

Loading a tool from the tool list which was parameterized using the transfer parameters. The tool can be selected with the parameters "Duplo" and "Ident" or via the parameter "TNumber". To load, the number of the loading location (place) in the loading magazine from "LoadingPlace" is used, if "LoadingPlace" is > 0. However, if "LoadingPlace" is set to 0, the value from "LoadPlaceNo" node is used in the configuration file as loading location (place) number. The channel number must be specified for the operation.

If the parameter "Magazine" is assigned with ">0" and "MagazinePlace" with "0", then an attempt is made to load the tool into an appropriate empty location in the specified magazine.

If the values for "Magazine" and "MagazinePlace" are > 0, then an attempt is made to load the tool to this magazine location. In all other cases, the tool is loaded into the magazine in which an empty location is found.

Table 13-8 PLC parameters for "Load tool - tool list"

PLC data	Identifier	Type	Value range	Meaning
Transfer parameters				
DBW3002	Duplo	INT	1 ... 32000	Duplo number
DBW3038	TNumber	INT	0 ... 32000	T number
DBW3040	MagazinePlace	INT	0 ... 32000	Magazine location number
DBB3048	Channel	BYTE	1 ... 127	TO area
DBB3049	Loading Place	BYTE	1 ... 127	Number of the loading/unloading location
DBW3050	Magazines	INT	0 ... 32000	Magazine number
Return parameters				
None				

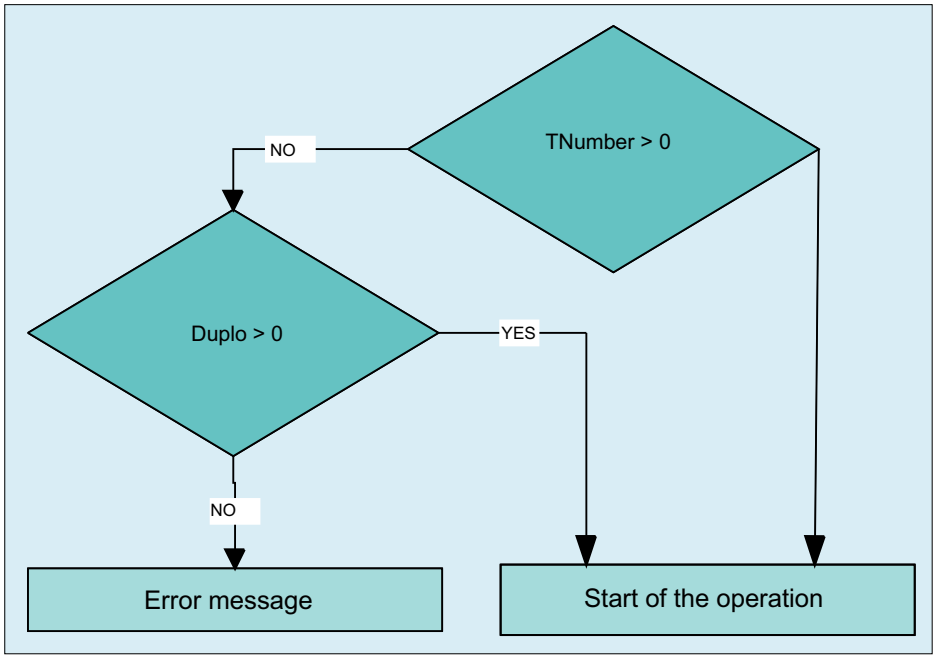


Figure 13-3 Flowchart to identify tools

13.4.6 Unloading a tool in the tool list (K code = 8)

Unloads the tool in the tool list that was parameterized using the transfer parameters. The channel number must be specified for this operation.

If the "Magazine" and "MagazinePlace" parameters are > 0, then an attempt is made to unload the tool, which is at this location. Otherwise the tool is selected with the parameters "Duplo" and "Ident" or via the parameter "TNumber".

In this case, the number of the unloading location in the loading magazine from "LoadingPlace" is used, if "LoadingPlace" is > 0. If "LoadingPlace" is set to 0, then the value from the "LoadPlaceNo" node is used in the configuration file as unload location number (loading point).

If "DeleteTool" is set, then the tool is also deleted from the NC after the unload operation.

Table 13-9 PLC parameters for "Unload tool - tool list"

PLC data	Identifier	Type	Value range	Meaning
Transfer parameters				
DBW3002	Duplo	INT	0 ... 32000	Duplo number
DBW3038	TNumber	INT	0 ... 32000	T number
DBB3048	Channel	BYTE	1 ... 127	TO area
DBB3049	LoadingPlace	BYTE	0 ... 127	Number of loading point
DBW3050	Magazines	INT	0 ... 32000	Magazine number
DBW3040	MagazinePlace	INT	0 ... 32000	Magazine location
DBX3056.3	DeleteTool	BOOL	True, False	Delete tool after unloading
Return parameters				
None				

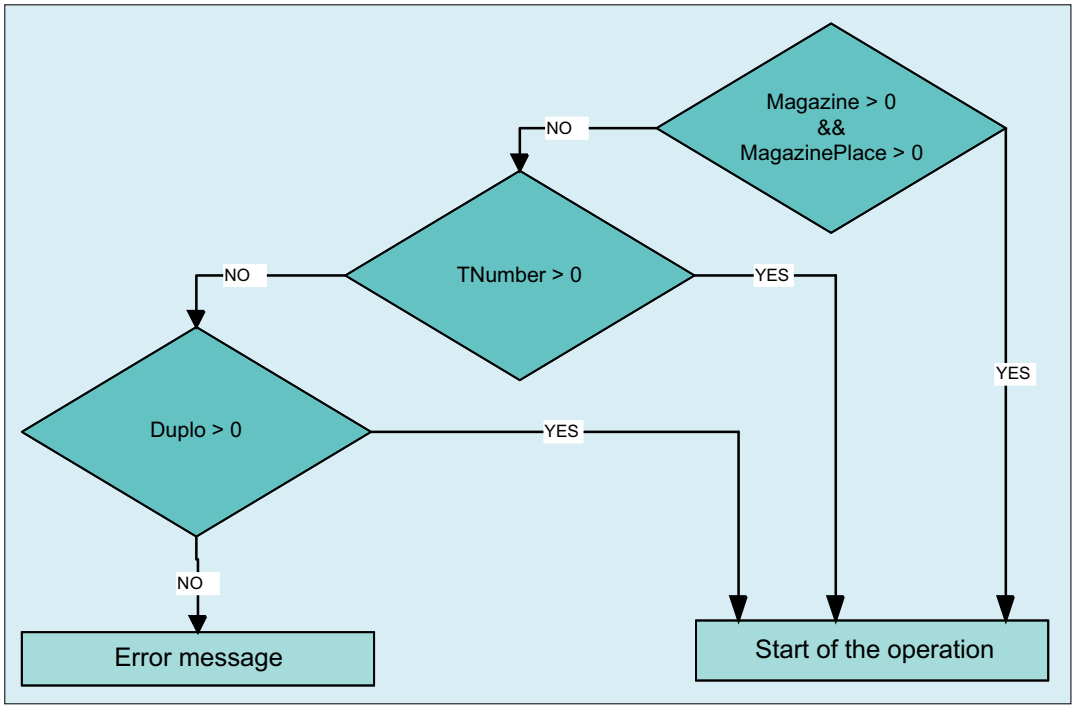


Figure 13-4 Flowchart to identify tools

13.4.7 Empty location search (K code = 9)

For the tool specified with "TNumber" or with "Ident" and "Duplo", an empty location search or target location check is performed. For "TNumber" = 0 an empty location is searched for the tool specified using "Ident" and "Duplo"; for "TNumber" > 0 using the T number.

A valid value must be set for the "Channel" parameter in order to execute the function. Using the specified loading point, an attempt is made to find an empty location in the magazines with which the loading point is connected. To do this, the number of the loading point in the loading magazine from "LoadingPlace" is used. However, if "LoadingPlace" is set to 0, the value from "LoadPlaceNo" node is used in the configuration file as loading location (place) number.

If "Magazine" =0, a search is made for an empty location in any magazine; if a magazine number is specified, then a search is made in the corresponding magazine. If, in addition, a magazine location >0 is specified, then the function performs a target location check. This is required for tools that are coded for specific locations.

Table 13-10 PLC parameters for "empty location search"

PLC data	Identifier	Type	Value range	Meaning
Transfer parameters				
DBW3002	Duplo	INT	0 ... 32000	Duplo number
DBW3038	TNumber	INT	0 ... 32000	T number
DBB3048	Channel	BYTE	1 ... 127	TO area
DBB3049	LoadingPlace	BYTE	0 ... 127	Number of loading point

PLC data	Identifier	Type	Value range	Meaning
DBW3050	Magazines	INT	0 ... 32000	Magazine number
DBW3040	MagazinePlace	INT	0 ... 32000	Magazine location
Return parameters				
DBW3058	Subtype	INT	-1 ... 32000	Subtype of the tool
DBW3060	Duplo	INT	-1 ... 32000	Duplo number of the tool
DBW3096	T number	INT	-1 ... 32000	T number of the tool
DBW3098	ToolState	INT	-1 ... 32000	Tool status
DBW3100	Magazine location type	INT	-1 ... 32000	Magazine location type of the tool
DBB3102	ToolSize_Upper	BYTE	1 ... 127	Tool size top in half locations
DBB3103	ToolSize_Down	BYTE	1 ... 127	Tool size bottom in half locations
DBB3104	ToolSize_Left	BYTE	1 ... 127	Tool size left in half locations
DBB3105	ToolSize_Right	BYTE	1 ... 127	Tool size right in half locations
DBW3106	Magazines	INT	0 ... 32000	Magazine number of the tool
DBW3108	MagazinePlace	INT	0 ... 32000	Magazine location number of the tool

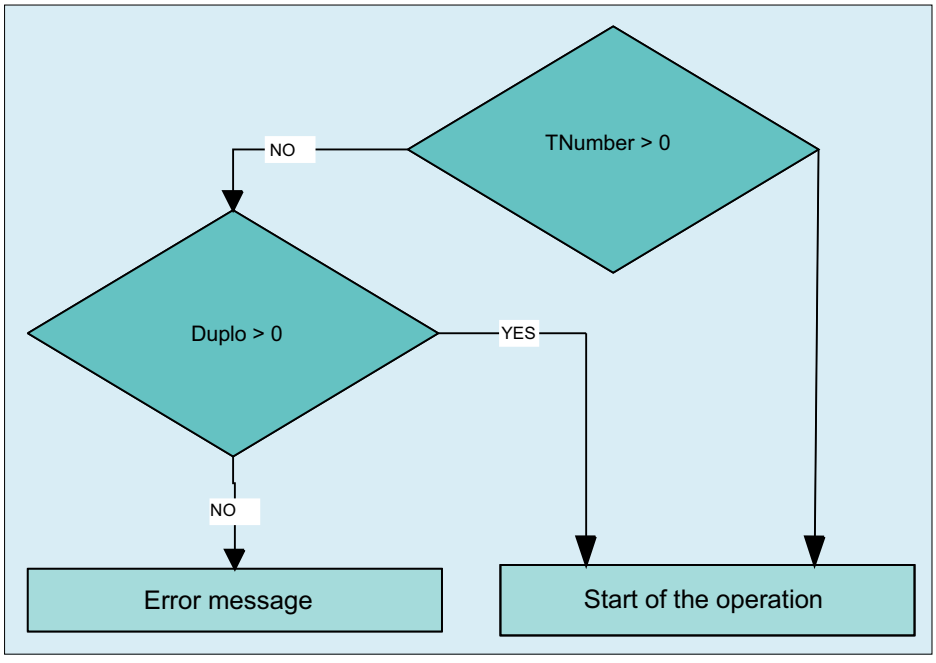


Figure 13-5 Flowchart to identify tools

13.4.8 Delete tool in NC (K code = 10)

Deletes the tool specified with "TNumber" or with "Ident" and "Duplo" in the NC if this is not loaded.

If a T number > "0" is specified, an attempt is made to delete the tool with this number in the NC.

If "TNumber = 0" is set, an attempt is made to delete the tool with the specified "Ident" and "Duplo" in the NC.

Table 13-11 PLC parameters for "Delete tool in the NC"

PLC data	Identifier	Type	Value range	Meaning
Transfer parameters				
DBW3002	Duplo	INT	0 ... 32000	Duplo number
DBW3038	TNumber	INT	0 ... 32000	T number
DBB3048	Channel	BYTE	1 ... 127	TO area
Return parameters				
None				

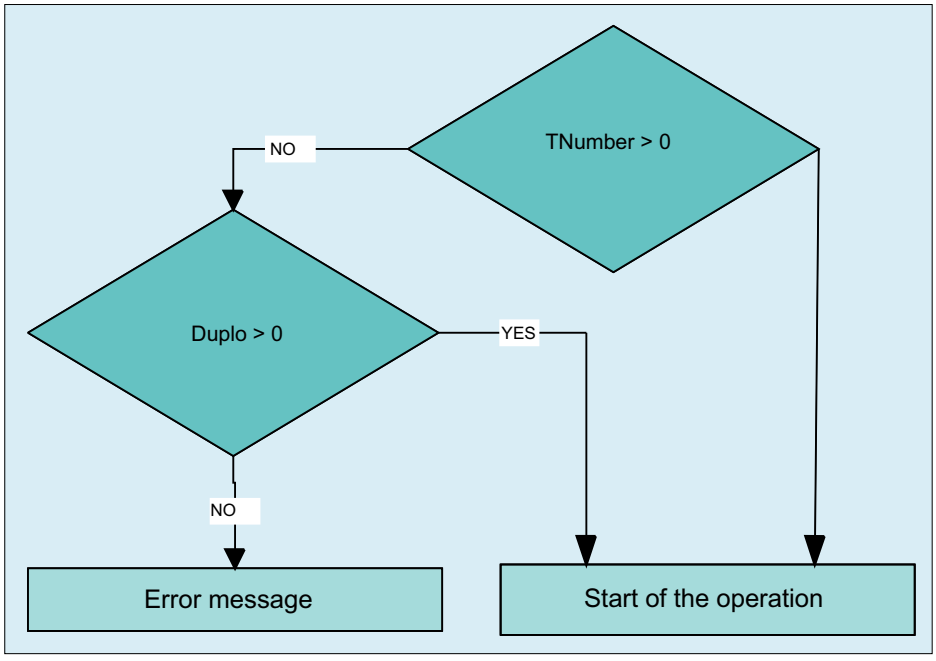


Figure 13-6 Flowchart to identify tools

13.4.9 Updating the code carrier (K code = 14)

Updates the code carrier with the data of the tool, which was parameterized using the transfer parameters.

The channel number and the number of the read head must be specified.

If the "Magazine" and "MagazinePlace" parameters are > 0, then an attempt is made to write the data of the tool, which is at this location, to the code carrier. Otherwise the tool is selected with the parameters "Duplo" and "Ident" or "TNumber".

Table 13-12 PLC parameters for "Update code carrier"

PLC data	Identifier	Type	Value range	Meaning
Transfer parameters				
DBW3002	Duplo	INT	0 ... 32000	Duplo number
DBW3038	TNumber	INT	0 ... 32000	T number
DBB3048	Channel	BYTE	1 ... 127	TO area
DBW3050	Magazines	INT	0 ... 32000	Magazine number
DBW3040	MagazinePlace	INT	0 ... 32000	Magazine location
DBB3052	Unit	BYTE	1 ... 127	Number of the code carrier
Return parameters				
None				

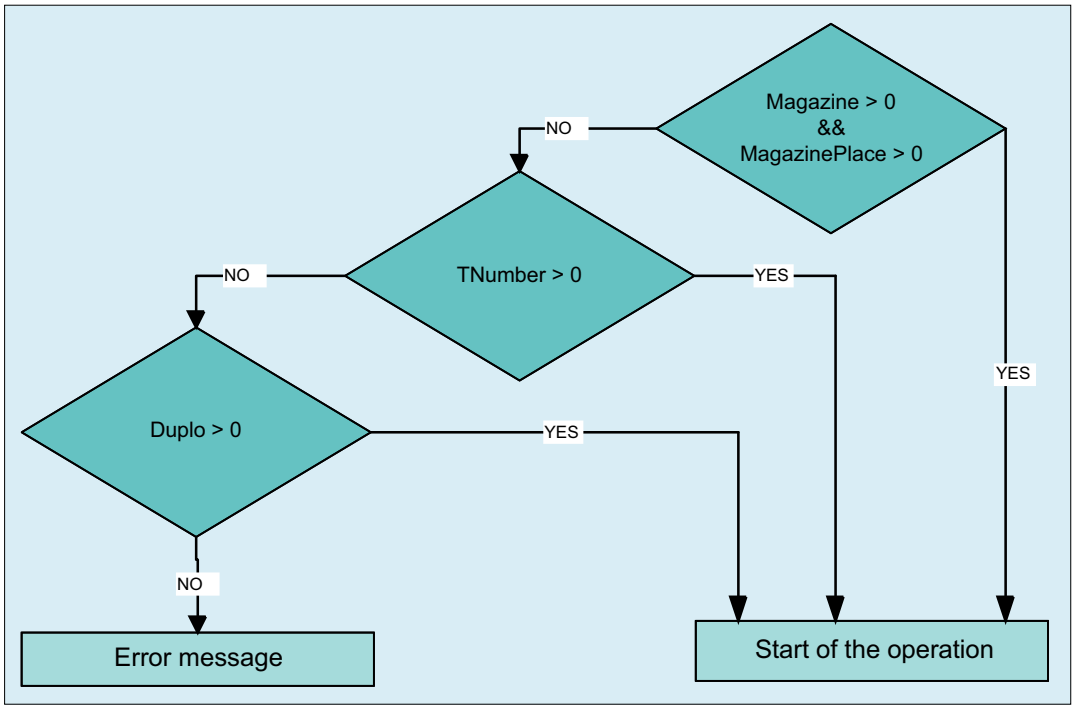


Figure 13-7 Flowchart to identify tools

13.4.10 Reading data from the code carrier (K code = 15)

Reading tool data from the code carrier and writing return parameters into the PLC. The code carrier is identified using "Unit".

It is then possible, for example, to read these two data from a code on which only the "Duplo" number and the tool identifier are saved. The "Loading/unloading from the tool list" service can then be called, for example.

This means that also favorably-priced "read only" code chips can be used.

Table 13-13 PLC parameters for "Reading data from the code carrier"

PLC data	Identifier	Type	Value range	Meaning
Transfer parameters				
DBB3052	Unit	BYTE	1 ... 127	Number of the read head
Return parameters				
DBW3058	Subtype	INT	-1 ... 32000	Subtype of the tool
DBW3060	Duplo	INT	-1 ... 32000	Duplo number of tool
DBW3098	ToolState	INT	-1 ... 32000	Tool status
DBW3100	Magazine location type	INT	-1 ... 32000	Magazine location type of the tool
DBB3102	ToolSize_Upper	BYTE	1 ... 127	Tool size top in half locations
DBB3103	ToolSize_Down	BYTE	1 ... 127	Tool size bottom in half locations
DBB3104	ToolSize_Left	BYTE	1 ... 127	Tool size left in half locations
DBB3105	ToolSize_Right	BYTE	1 ... 127	Tool size right in half locations

13.4.11 Error codes

The error codes that are saved in the PLC byte "K_CodeError" or PLC word "Error_Code" of the PLC interface are in the hexadecimal format and refer to the location where the task execution was canceled with an error.

If the error code is to be shown in the decimal notation, then this must be converted into the hexadecimal format in order to determine the error cause from tables A-1 to A-8.

The error code in "K_CodeError" identifies the command code, where the error occurred.

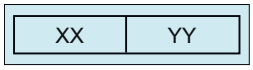
Table 13-14 Command codes of the task in the case of an error

K_code error	Service
00	No error
01	Load from the code carrier
02	Unload to the code carrier
03	
04	
05	Load from production host computer
06	Unload to the production host computer
07	
08	Load tool
09	Unload tool
0A	Empty location search
0B	Delete tool in NC
0C	
0D	
0E	

K_code error	Service
0F	Update code carrier
10	Read data from code carrier
11	
...	Reserved

The error codes "Error_Code" comprise a high byte and low byte, which have a different significance. The high byte (XX) identifies the error type group, in which the error occurred. The low byte (YY) specifies the cause of the error, which is responsible for canceling the task.

Format of the Error_Code



The contents of XX = 00 and YY = 00 mean that no error has occurred.

XX = 01: General group error

Table 13-15
Low byte (YY) of the function error code

	Cause of the error
01	Error when reading parameter from DB1900
02	Error when calling the internal service
03	Timeout when executing the service
04	No tool data determined yet
05	Error when executing the internal service
...	Reserved
	Transfer parameter error
10	Subtype
11	Duplo
12	Ident
13	TNumber
14	MagazinePlace
15	MagazinePlaceType
16	ToolSize_Upper
17	ToolSize_Down
18	ToolSize_Left
19	ToolSize_Right
1A	Channel
1B	Magazines
1C	Unit
1D	NCU_Index
1E	FileIndex

	Cause of the error
1F	ActiveEnable
20	Override
21	LoadTool
22	DeleteTool
23	Length of Ident
...	Reserved

XX = 02: Code carrier error group

Table 13-16
Low byte (YY) of the function error code

	Cause of the error
01	No code carrier available
02	Error when reading code carrier
03	Reading interrupted as code carrier was removed
04	Error when writing to code carrier
05	Writing interrupted as the code carrier was removed
0C	Command string unknown
0D	Number of bytes illegal
0E	Non-BCD characters found in the received write data
17	Invalid data
18	Device not available
19	Timeout
1A	Hardware faults
1B	Invalid head number
1C	Read head not connected
1E	Device is read only
1F	General read error
20	General write error
28	Unknown error
68	Data storage medium not recognized (reading/writing, first page)
69	Data storage medium not recognized (reading/writing, following page)
6D	Data storage medium not recognized (writing, first pages)
6E	Data storage medium not recognized (reading before writing)
6F	Data storage medium not recognized (writing the last page)
73	Non-BCD characters found in the received write data
74	Read BCD format: Non-BCD characters found on the data storage medium
8D	Data storage medium access error
8E	Data in the data storage medium incomplete

XX = 05: NC error group

Table 13-17
Low byte (YY) of NC error code

	Cause of the error
00	Connection not established to the requested NCU
05	Reading the NC configuration unsuccessful
0A	An invalid channel number was specified in the task
3C	An invalid magazine number was specified in the task
3E	An invalid magazine location number was specified in the task
3F	An invalid cutting edge OEM monitoring parameter number was specified in the task
41	The MMC semaphore for the search/create tool is already assigned
46	The tool specified with ident and duplo was not found in the NC
47	The tool specified with the T number was not found in the NC
48	No tool was found in the specified magazine
49	No tool is loaded at the specified magazine location
4A	The tool specified with ident and duplo was not able to be uniquely determined in the NC
4B	Number of found tools was not able to be read in the NC
50	The tool specified with ident and duplo was not created in the NC
55	The specified tool is already loaded into the magazine
5A	The tool cutting edge with the specified number has not been created in the NC
5B	The tool cutting edge with the specified number has not been deleted in the NC
5C	The specified cutting edge does not exist in the NC
5D	The specified tool cutting edge number is not valid
5E	The specified cutting edge is already in the NC
5F	Tool state was not able to be read in the NC
64	The MMC semaphore for the empty location search is already assigned
69	An empty magazine location was not able to be found when loading into the NC
6E	Checking the specified magazine location to load negatively acknowledged
6F	Tool deletion was exited with error
70	The empty location search was exited with error
71	Creating a tool was exited with error
72	Operation to relocate tools unsuccessful
73	The MMC semaphore for tool unloading/unloading already assigned
74	Alarm extension was exited with error
78	The magazine is disabled or has not been enabled for loading
7D	Tool loading was exited with error
82	The magazine number of the tool was not able to be read in the NC
87	The specified tool is not loaded into the NC
8C	The specified tool is loaded in a different magazine
8E	The specified tool is loaded at a different magazine location
91	Magazine state not able to be read in the NC
96	Magazine not in the loading position or not enabled for unloading
9B	The tool unload operation was exited with error

	Cause of the error
A0	The tool was not able to be deleted in the NC
A5	The tool is active in the NC! The tool was not able to be deleted in the NC
D2	The specified tool is loaded into the NC! The tool was not deleted in the DB

XX = 0B: Tool conversion error group

Table 13-18
Low byte (YY) of the function error code

	Cause of the error
01	1401 Error when opening the MCX file
02	1402 Error when reading the MCX file
03	1403 Error when reading the MCX file
04	1404 Error when reading the MCX file
05	1405 Error when reading the MCX file
06	1406 Error when reading the MCX file
07	1410 Error when reading the MCX file
08	1628 System error
09	1801 System error
0A	1803 System error
0B	1804 System error
0C	1805 Incorrect version of the MCX file
0D	1806 MCX file contains functions that are not supported
0E	1807 System error
0F	1808 Error when reading the MCX file
10	1809 System error
11	1810 System error
12	2401 System error
13	2402 System error
14	2501 System error
15	2502 System error
16	2503 System error
17	2701 System error
18	2702 System error
19	2703 Index error when accessing the array
1A	2801 System error ConvertFunctions::CheckAccess()
1B	2802 System error
1C	2803 Error in conjunction with the conversion functions readXXX()/writeXXX(). Basis type of the array parameter must be 'unsigned char'.
1D	2804 Error in the function writeBCD(). The data length (val) exceeds the size of the target range.
1E	2805 Error in the function PlaceTypeFromString(). ToolTypeText used is not defined in the ToolSpec.xml.

	Cause of the error
1F	2806 Error in the function PlaceTypeToString(). ToolTypeCode used is not defined in the ToolSpec.xml.
20	2807 Error in the function writeREAL(). Data loss due to rounding.
21	2901 System error
22	2903 System error
23	2904 System error
24	2905 System error
25	2908 System error
26	2909 System error
27	2910 System error
28	2912 System error
29	2913 Error when accessing SlTmService. setVal()
2A	2914 Error when accessing SlTmService. setVal()
2B	2915 Error when accessing SlTmService. setVal()
2C	2917 Error when accessing SlTmService. rData().val()
2D	2918 Error when accessing SlTmService. rData().val()
2E	2919 Error when accessing SlTmService. setVal()
2F	2921 Error when accessing SlTmService. val()
30	2922 System error
31	2923 Error when accessing SlTmService. val()
32	2924 System error
33	7001 System error
34	7002 The conversion rule must contain an object with the 'Tool' name.
35	7003 The 'Tool' object must be the 'TmTool' type.
36	7004 The conversion rule must contain an object with the 'ToolArray' name.
37	7005 The 'ToolArray' object must be an array.
38	7006 The 'ToolArray' object must be an array, basic type 'unsigned char'.
39	7007 The 'DecodeToolData' function was not found in the conversion rule.
3A	7008 The object 'DecodeToolData' must be a function.
3B	7009 The 'EncodeToolData' function was not found in the conversion rule.
3C	7010 The object 'EncodeToolData' must be a function.
3D	7011 System error
3E	7012 System error
3F	7013 System error
40	7014 System error
41	7015 Error when reading ToolSpec.xml'
42	7016 Error when interpreting ToolSpec.xml
43	7017 The ToolSpec.xml. The document element should have the name: "ToolSpecConversion".
44	7018 Format error in the ToolSpec.xml. Attribute code="" or text="" are not included.
45	System error when converting data
80 ... FF	User-defined error from conversion rule (see Error handling (Page 414))

13.5 Examples

Note

Project example

A project example for loading and unloading from the code carrier via the PLC interface is provided with the toolbox.

13.5.1 Loading the code carrier via PLC interface

In this case, the data of the imaginary tool "WZ_Test13" is read from a code carrier, the tool is then created in the NC and loaded. The tool should be loaded into Magazine "1" of the standard NCU in channel "1". The operation should be administered via loading location "1" of the loading magazine.

The parameters required for this service are listed in the following table.

Before executing the service, the PLC writes a "0" into the command code byte (K_code) of the PLC interface in data block DB1900. This service is then started by setting the "Request" bit. If the "Active" bit is reset and the "Done" bit was set, then the PLC can evaluate the "Error" byte. The "Request" bit must then be reset again.

Table 13-19 Transfer parameters for "Loading from code carrier"

PLC data		Identifier	Value
DBW	3040	MagazinePlace	0
DBB	3048	Channel	1
DBB	3049	LoadingPlace	1
DBW	3050	Magazines	1
DBB	3052	Unit	1
DBB	3053	NCU_Index	0
DBX	3056.2	LoadTool	TRUE

Table 13-20 Return parameters from Tool Ident Connection for "Loading from the code carrier"

PLC data		Identifier	Value
DBW	3058	Subtype	120
DBW	3060	Duplo	1
DBW	3096	TNumber	289
DBW	3098	ToolState	14
DBW	3100	MagazinePlaceType	1
DBB	3102	ToolSize_Upper	1
DBB	3103	ToolSize_Down	1
DBB	3104	ToolSize_Left	1

DBB	3105	ToolSize_Right	1
DBW	3106	Magazines	1
DBW	3108	MagazinePlace	1

The sequence of the individual operations can be clearly seen in the following flowchart:

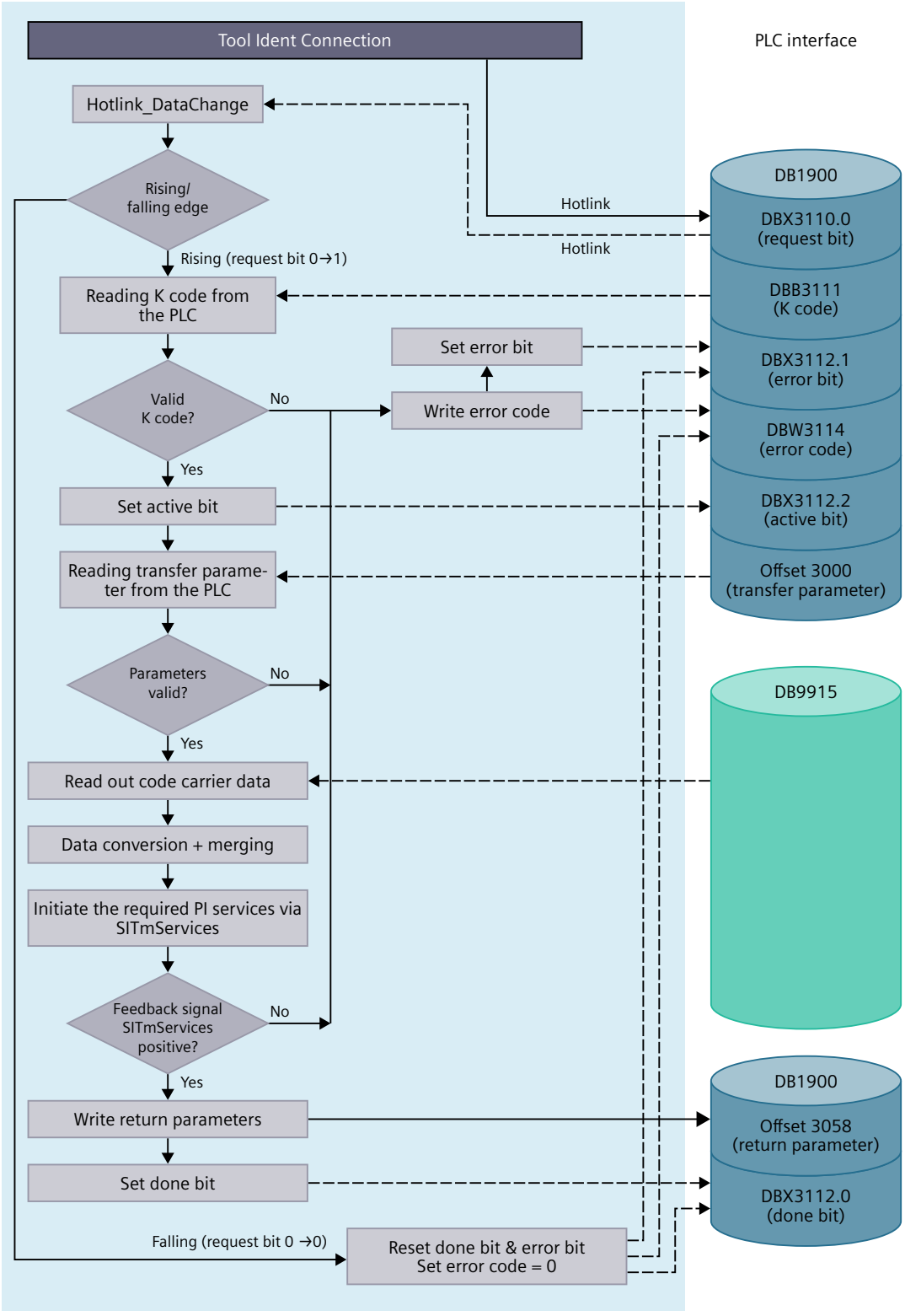
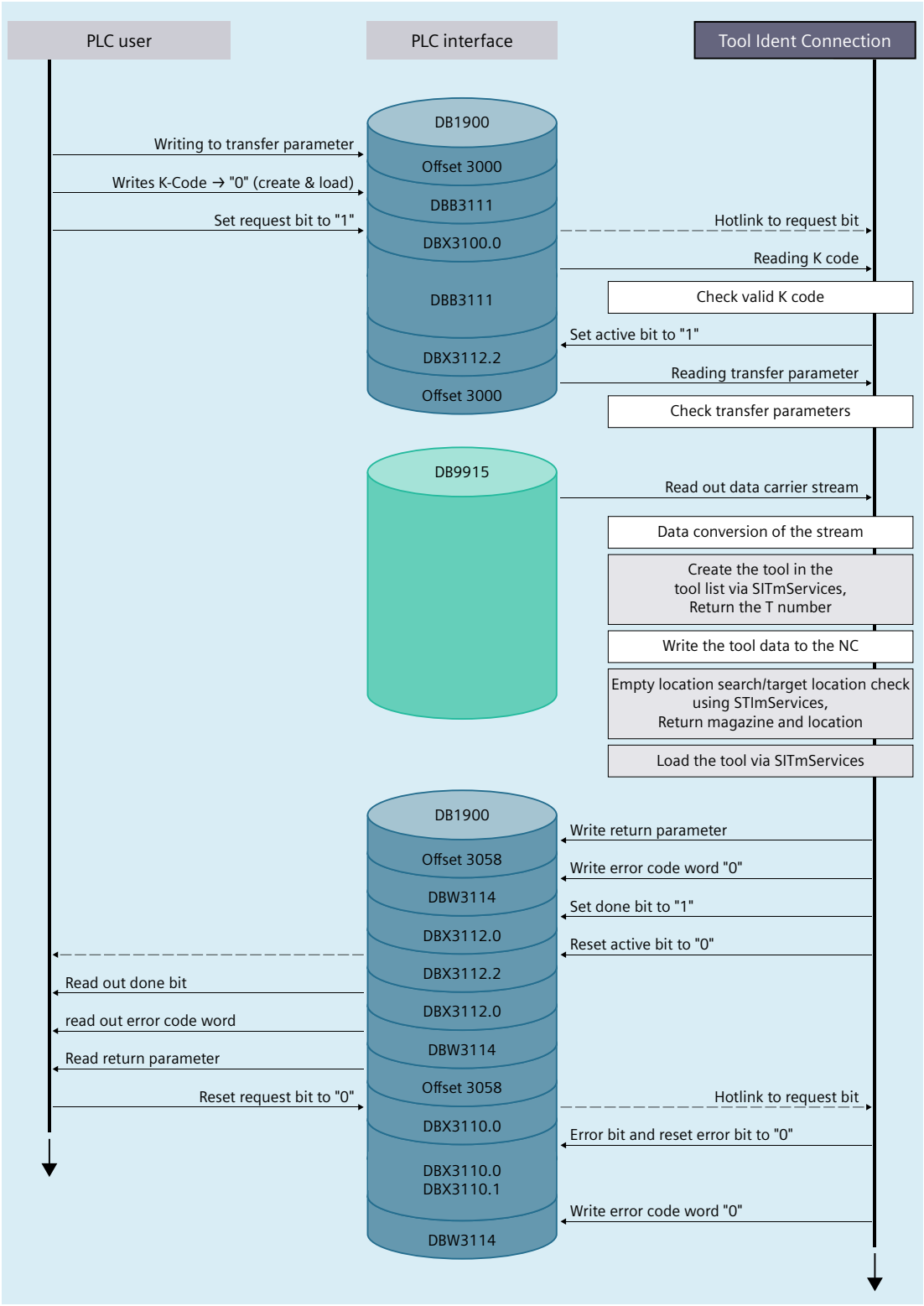


Figure 13-8 Flowchart: Loading from the code carrier via the PLC interface


PLC user
PLC interface
Tool Ident Connection
DB1900
Offset 3000
DBB3111
DBX3100.0
DBB3111
DBX3112.2
Offset 3000
DB9915
DB1900
Offset 3058
DBW3114
DBX3112.0
DBX3112.2
DBX3112.0
DBW3114
Offset 3058
DBX3110.0
DBX3110.0
DBX3110.1
DBW3114
Writing to transfer parameter
Writes K-Code → "0" (create & load)
Set request bit to "1"
Hotlink to request bit
Reading K code
Check valid K code
Set active bit to "1"
Reading transfer parameter
Check transfer parameters
Read out data carrier stream
Data conversion of the stream
Create the tool in the tool list via SITmServices, Return the T number
Write the tool data to the NC
Empty location search/target location check using STImServices, Return magazine and location
Load the tool via SITmServices
Write return parameter
Write error code word "0"
Set done bit to "1"
Reset active bit to "0"
Read out done bit
read out error code word
Read return parameter
Reset request bit to "0"
Hotlink to request bit
Error bit and reset error bit to "0"
Write error code word "0"

13.5.2 Unloading to the code carrier via PLC interface

The tool loaded in the previous example should be unloaded and the actual data written to the code carrier. After unloading, the tool is deleted in NC. The procedure should be managed via unload location "2" of the loading magazine.

The transfer parameters required for this are listed in the following table. The service does not return any parameters, and as a result, there are no new values in the return parameters of the PLC interface.

Before executing the service, command code "1" must be written to the corresponding PLC byte (K code). The start of the service and the further procedure are equivalent to the procedure in Loading the code carrier via PLC interface (Page 440).

Table 13-21 Transfer parameters for "Unloading from code carrier"

PLC data		Identifier	Value
DBW	3002	Duplo	1
DBW	3040	MagazinePlace	1
DBB	3048	Channel	1
DBB	3049	LoadingPlace	2
DBW	3050	Magazines	1
DBB	3052	Unit	1
DBB	3053	NCU_Index	0
DBX	3056.3	DeleteTool	TRUE

The sequence of the individual operations can be clearly seen in the following flowchart:

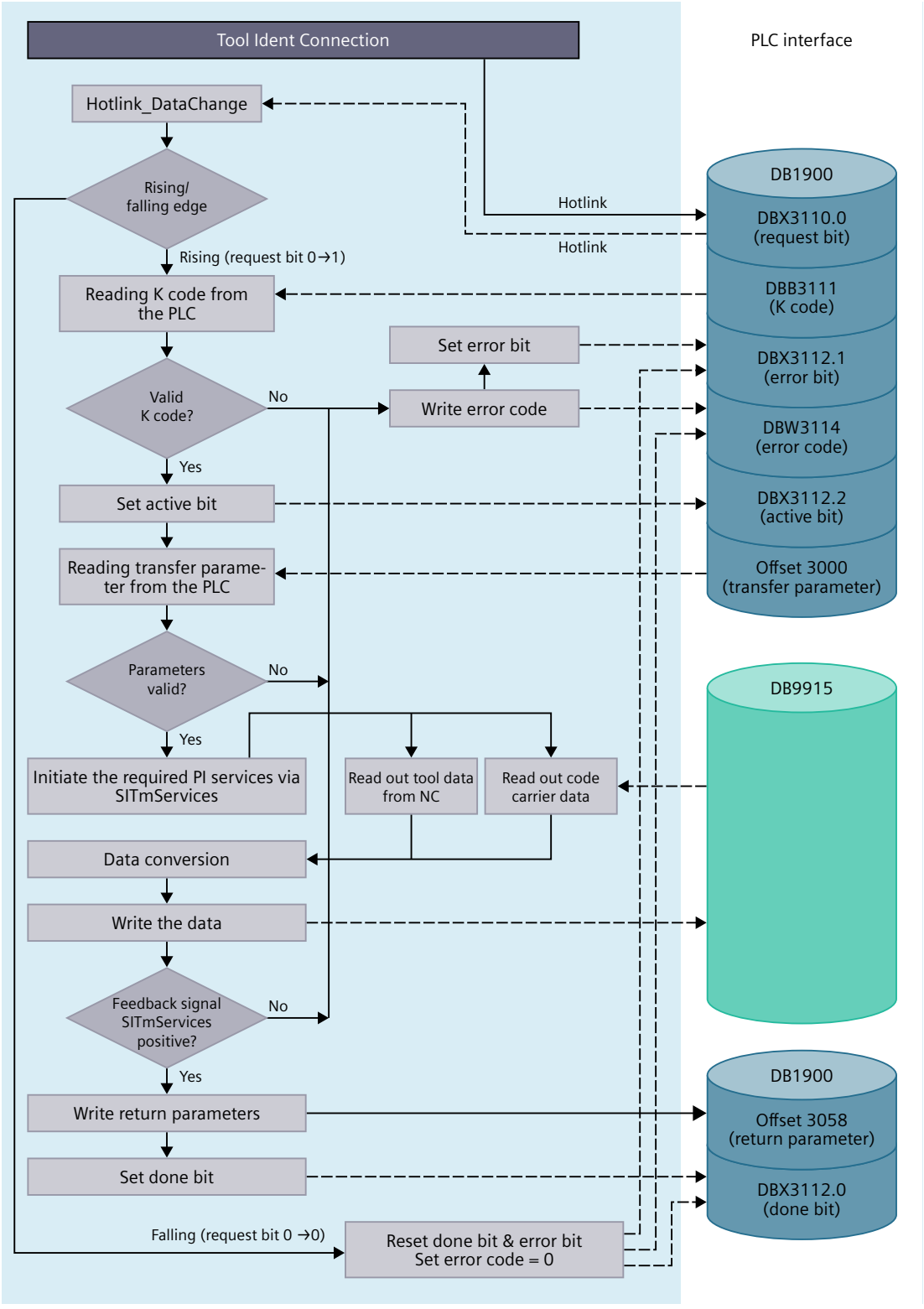


Figure 13-9 Flowchart: Unloading from the code carrier via the PLC interface

Appendix

A

A.1 List of abbreviations

ASUB	Asynchronous subroutine
OPI	Operator Panel Interface
CC	Compile cycle or OEM or user area
CUTOM	CUTter radius cOMpensation: Tool Radius Compensation
DB	Data Block in the PLC
DBB	Data Block Byte in the PLC
DBW	Data Block Word in the PLC
DBX	Data Block Bit in the PLC
DDE	Dynamic Data Exchange: Dynamic Data Exchange
DW	Data Word
ENC	Encoder: Actual value encoder
EPROM	Erasable Programmable Read Only Memory: erasable, electrically programmable read-only memory
FB	Function Block
FC	Function Call: Function block in the PLC
GUD	Global User Data: Global user data
HEX	Abbreviation for hexadecimal number
HMI	Human Machine Interface
MSD	Main Spindle Drive
IBN	Commissioning
INC	Increment: Increment
INI	Initializing Data: Initializing data
IPC	Industrial PC
ISO code	Special punched tape code, number of holes per character always even
K1 ... K4	Channel 1 to channel 4
C Bus	Communication bus
MD	Machine Data
MDI	Manual Data Input: Manual input
Machine	Machine coordinate system
NCK	Numerical Control Kernel
OA	Open Architecture
OB	Organization Block in the PLC
OEM	Original Equipment Manufacturer: Manufacturer whose products are marketed under a different name
OP	Operator Panel: Operating equipment
PI	Program Invocation: Programming Instance
PLC	Programmable Logic Controller: Interface control

PLC	Programmable logic controller
TCA	ToolChangeAbsolute
TCI	ToolChangeIntermediateLocation
TO	Tool Offset: Tool offset
TOA	Tool Offset Active: Identifier (file type) for tool offsets
TOOLGNT	ToolGroupNumber OfTools
TOOLGT	TOOLGroupToolNumber
USEKT	UserKindOfTools
VDI	Virtual Device Interface: Virtual interface
V.24	Serial interface (definition of the exchange lines between data terminal equipment and data communication equipment)
Work	Workpiece coordinate system
T	Tool
TLC	Tool Length Compensation
TRC	Tool Radius Compensation
TO	Tool offset
TMBF	Tool Management Basic tool function (basic function)
TMMO	Tool monitoring function or monitor
TMMG	Tool magazine management
TM	Tool management

Index

$

$A_MONIFACT, 241
$A_MYMLN, 247
$A_MYMN, 247
$A_TOOLMLN, 240
$A_TOOLMN, 237
$A_USEDD, 253
$A_USEDND, 251
$A_USEDT, 252
$AC_MONMIN, 243
$AC_MSNUM, 271, 272, 288
$AC_MTHNUM, 272, 288
$MC_TOOL_CHANGE_ERROR_MODE (MD22562), 27, 75
$MC_TOOL_CHANGE_MODE (MD22550), 26
$P_AD[n], 268
$P_ADT[n], 269
$P_ADTD[n], 270, 271
$P_D_BEFORE_SEARCH, 275
$P_DL_BEFORE_SEARCH, 275
$P_DLNO, 266
$P_ISTEST, 280
$P_MAGA, 263
$P_MAGDISL, 255
$P_MAGDISS, 255
$P_MAGHLT, 260
$P_MAGNA, 263
$P_MAGNA1, 264
$P_MAGNA2, 264
$P_MAGNDIS, 255
$P_MAGNH, 260
$P_MAGNHLT, 260
$P_MAGNREL, 257
$P_MAGNS, 256
$P_MAGREL, 257
$P_MAGS, 256
$P_MSNUM, 272, 288
$P_MTHNUM, 272, 288
$P_MTHNUM_BEFORE_SEARCH, 274
$P_MTHSDC, 273, 274
$P_TC, 266, 267
$P_TCANG[n], 267
$P_TCDIFF[n], 267
$P_TH_OF_D, 274
$P_TOOL, 265
$P_TOOLD, 250
$P_TOOLL[n], 266
$P_TOOLND, 241
$P_TOOLNDL, 250
$P_TOOLNG, 246
$P_TOOLNO, 265
$P_TOOLNT, 249
$P_TOOLP, 265
$P_TOOLR, 266
$P_TOOLT, 249
$P_USEKT, $TC_TP11, 231
$P_VDITCP[x], 172
$TC_DPCx[t,D], 124
$TC_DPx[t,D], 122
$TC_MAMPx [n], 149
$TC_MAP10, 140
$TC_MAP3, 139
$TC_MAP8, 139
$TC_MAPCx[n], 140
$TC_MDPx[n,m], 146
$TC_MLSR[x,y], 157
$TC_MOPCx[t,D], 126
$TC_MOPx[t,D], 125
$TC_MPP1, 142
$TC_MPP5, 143
$TC_MPP6, 143
$TC_MPPCx[n,m], 144
$TC_MPPx[n,m], 140
$TC_MPTH[n,m], 145
$TC_MTPCx[t], 170
$TC_MTPPCx[t], 170
$TC_SCPx[t,D], 127
$TC_TP1 ... $TC_TP2, 130
$TC_TP3 ... TP 6, 130
$TC_TP7, 131
$TC_TP8, 131
$TC_TPCx[t], 136
$TC_TPGx[t], 135
$TC_TPx[t], 128

A

Acknowledgment
  Process, 57
  Rules, 68
  Status, 59
  synchronous, 58
  TOOLMAN, 40
Acknowledgment step, 66
Acknowledgment-step table, 55
Adjacent location, 26

Angle head location parameters
Identifiers, 319
Angle head parameters
Identifiers, 318

B

Buffer, 25
Byte order, 415

C

Chain magazine, 25
CHKDM, 174
Circular magazine, 26
Code carrier data
Data conversion, 392
Code carrier data formats, 398
Code carrier data, dialog data
Assignment, 398
Code carrier description file, 397
Configuring the tool change reason, 358
Coolant
Assigning, 354
Identifier, 370
CRCEDN, 286
CREACE, 286
CREATO, 286
CRTOCE, 286
Cutting edge parameters
Identifiers, 306
OEM identifier, 368
Cutting edge selection after tool change, 279

D

D numbers of replacement tools, 174
DefToolDat.txt, 386, 402
DELDL, 178
DELECE, 286
DELETO, 286
DELMLOWNER, 229
DELMLRES, 228
DELMT, 181
DELT, 181
DELTC, 216
DZERO, 177

E

End acknowledgment, 58, 59
Example
Change manual tool (1), 75
Change manual tool (2), 76
Milling machine, 91
Milling part program, 92
Turning machine, 79
Turning part program, 80

F

Feedback, 41

G

G-code program, 383
GETACTT, 195
GETACTTD, 175
GETEXET, 189
GETFREELOC, 222
GETSELT, 186
GETT, 183
Grinding parameters
Identifiers, 311

I

IDENTCONNECTIONCONFIGURATION, 388
Identifier
Angle head location parameters, 319
Angle head parameters, 318
Coolant, 370
Cutting edge parameters, 306
Grinding parameters, 311
Magazine location adapter parameters, 313
Magazine location parameters, 312
Monitoring parameters, 311
Multitool location parameter, 317
Multitool parameters, 316
OEM crosshead parameters, 369, 370
OEM cutting edge parameters, 368
OEM magazine location parameters, 369
OEM monitoring parameters, 368
OEM multitool parameters, 369
OEM tool parameters, 368
Tool parameters, 304
Tool-specific functions, 370

Intermediate acknowledgment, 58, 59

J

Job status, 53

L

List parameters
    Changing, 320
    Creating new, 322
Loading magazine, 25
Location-dependent offsets, 127
Location-dependent offsets, coarse, 127

M

Machine function, defining texts, 356
Magazine, 25, 64
    Configuration, 88
Magazine list, 25
Magazine location adapter parameters
    Identifiers, 313
Magazine location parameters
    Identifiers, 312
    OEM identifier, 369
Main entry, 341
Manual tools, 27
MCIS-C script language, 404
MD10717
    $MN_T_NO_FCT_CYCLE_NAME, 34
MD22550
    TOOL_CHANGE_MODE, 26
MD22562
    TOOL_CHANGE_ERROR_MODE, 27, 75
MD52270
    TM_FUNCTION_MASK, 33
Measuring in JOG, 86
Message, asynchronous, 58
MMCSEM, 286
Monitoring parameters
    Identifiers, 311
    OEM identifier, 368
Multitool, 27
Multitool location parameter
    Identifiers, 317
Multitool parameters
    Identifiers, 316
    OEM identifier, 369
MVTOOL, 205

N

NC variables, 68
NEWMT, 179
NEWT, 178

O

OEM tool types, 326
oem_sltmlistconfig.xml, 386
OPI, 115
OPI block AD, 171
OPI block C/S, 187, 190
OPI block MTUD, 170
OPI block MTUP, 171
OPI block TD, 129
OPI block TG, 136
OPI block TM, 137
OPI block TMC, 149, 157
OPI block TMV, 283, 284
OPI block TO, 122, 124
OPI block TOE, TOET, 128
OPI block TOS, 127
OPI block TP, 141
OPI block TPM, 147
OPI block TS, 125
OPI block TT, 145
OPI block TU, 136
OPI block TUM, 140
OPI block TUP, 144
OPI block TUS, 126

P

PI service, 71
PI services, 286
PLC firmware, 23
PLC program
    Acknowledging jobs, 67
    TM, 23
POSM, 200
POSM (Multitool), 202
POSMT, 203
Programming data
    Tool and cutting edge data, 119
Programming the tool selection, 278

R

Replacement tools, 21
RESETMON, 212
RESETMON (multitool), 214

S

SETDNO, 177
SETMS, 196
SETMTH, 197
SETPIECE, 184
SETTA (Multitool), 210
SETTIA, 210
SETTIA (multitool), 211
Setting-up offsets, 127
ShopMill
  Tool-specific functions, 354
Subprogram replacement technique, 275
System variables, 115
systemconfiguration.ini, 386, 387, 389

T

T=location number, 280
T=location, automatic tool selection, 234
TCA, 217
TCI, 220
TDIdentCfg.xml, 390
tdiidentcfg.xml, 386
TMCRTC, 286
TMCRTO, 286
TMFDPL, 286
TMFPBP, 286
TMGETT, 284, 286
TMMVTL, 286
TMMVTL (PI service), 71
TMPCIT, 286
TMPOSM, 286
TMRASS, 286
Tool call, 86
Tool change, 44, 96, 97
  End acknowledgment, 45
  Programming, 278
Tool holder, 381
Tool Ident Connection
  Configuration, 389
  PLC connection, 392
Tool list, 25
  List views, 300
Tool management, 21
  Block splitting, 97
  Grinding technology, 292
  Milling technology, 292
  PLC,TRANSLINE 2000, 292
  Turning technology, 292
  Turning technology on milling machine, 292
Tool parameters
  Identifiers, 304
  OEM identifier, 368
Tool types
  Drill, 325
  Grinding tools, 325
  Milling tool, 324
  Special tools, 326
  Turning tools, 325
TOOLGNT, 234
TOOLGT, 234
Tool-related data, 128
ToolSize.mcc, 420
toolSpec.xml, 386
ToolSpec.xml
  Configuration, 413
Tool-specific functions
  Identifier, 370
Total acknowledgment, 59
Transfer step, 64
Transfer-step table
  Constant, 54
  variable, 54
TRANSMIT command, 383
TSEARC, 284, 286

U

User interface, 35

V

Variable
  Location state, 69
  Location type, 69
  T number, 70

W

wkonvert.mcc, 386, 418
wkonvert.mcx, 386
wkonvert.txt
  Description file, 395