SIEMENS

SINUMERIK

SINUMERIK 840D sl Tool Management

Function Manual

Preface	
Fundamental safety instructions	1
Fundamentals	2
Description of functions	3
Commissioning	4
Programming	5
Code carrier connection - Tool Ident Connection (option)	6
Machine data	7
Signal description of the PLC interface	8
Alarms	9
Appendix	A

Valid for:
Control system
SINUMERIK 840D sl / 840DE sl

Software	version
CNC software	4.7 SP2

10/2015
6FC5397-6BP40-5BA3

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 personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

Proper use of Siemens products

Note the following:

WARNING

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

Trademarks

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

Disclaimer of Liability

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

Siemens AG
Division Digital Factory
Postfach 48 48
90026 NÜRNBERG
GERMANY

0-5BA3

Preface

SINUMERIK documentation

The SINUMERIK documentation is organized in the following categories:

- General documentation
- User documentation
- Manufacturer/service documentation

Additional information

You can find information on the following topics at www.siemens.com/motioncontrol/docu:

- Ordering documentation/overview of documentation
- Additional links to download documents
- Using documentation online (find and search in manuals/information)

Please send any questions about the technical documentation (e.g. suggestions for improvement, corrections) to the following address:

docu.motioncontrol@siemens.com

My Documentation Manager (MDM)

Under the following link you will find information to individually compile OEM-specific machine documentation based on the Siemens content:

www.siemens.com/mdm

Training

For information about the range of training courses, refer under:

- www.siemens.com/sitrain
  SITRAIN - Siemens training for products, systems and solutions in automation technology
- www.siemens.com/sinutrain
  SinuTrain - training software for SINUMERIK

FAQs

You can find Frequently Asked Questions in the Service&Support pages under Product Support. http://support.automation.siemens.com

SINUMERIK

You can find information on SINUMERIK under the following link:

www.siemens.com/sinumerik

Target group

The Function Manual is intended for:

- Project engineers
- Technologists (from machine manufacturers)
- System and machine commissioning engineers
- Programmers

Benefits

The Function Manual describes the functions so that the target group knows them and can select them. It provides the target group with the information required to implement the functions.

Standard scope

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

Other functions not described in this documentation might be executable in the control. This does not, however, represent an obligation to supply such functions with a new control or when servicing.

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

Technical Support

You will find telephone numbers for other countries for technical support in the Internet under http://www.siemens.com/automation/service&support

Table of contents

Preface....3

1 Fundamental safety instructions....13

1.1 General safety instructions....13

1.2 Industrial security....14

2 Fundamentals....15

2.1 Explanation of terms....15

2.2 Function structure of tool management....21

2.3 HMI/PLC - NCK data structure (OPI)....22

2.4 PLC – NCK interfaces....24

2.5 Magazine configuration....26

2.6 Access protection, protection levels....27

3 Description of functions....29

3.1 Magazines....29
3.1.1 Buffer....29
3.1.2 Loading magazine....29
3.1.3 Box-type and chain magazines....30
3.1.4 Circular magazine....32
3.1.5 Other magazine types....32
3.1.6 Wear group....33
3.1.7 Background magazine....34
3.1.8 Consider adjacent location....35

3.2 Several tools at a magazine location (multitool)....37
3.2.1 Brief description....37
3.2.2 Programming....40
3.2.3 $TC_MTP - multitool data....41
3.2.4 $TC_MTPP - Multitool location data....51
3.2.5 Tool change with a tool from a multitool....55

3.3 Tool change box-type, chain, circular magazines....72
3.3.1 Prepare a tool change....72
3.3.2 General tool change sequence....74
3.3.3 Monitoring the maximum speed of a tool....78
3.3.4 Selecting a tool and a cutting edge....83
3.3.5 Predecoding (preprocessing) and block execution (main run)....85
3.3.6 Traverse axes while tool is being changed....87
3.3.7 Tool change in the spindle for chain and box-type magazines....87
3.3.8 Special cases "T0", empty spindle, multiple T selection....91
3.3.9 Tool change with turret....92
3.3.10 Number of replacement tools....92
3.3.11 Tool changing errors....93

3.3.12 Manual tools (adding tools during machining)....94
3.3.13 Tool changes in the NCK using synchronized actions....100
3.3.14 Function replacement....103
3.3.15 Block search....107
3.3.16 Block search (SSL) in conjunction with active tool management....109
3.3.17 System variables for state before block search....112
3.3.17.1 Description of the variables....112
3.3.17.2 Example....113
3.3.18 Program test....116
3.3.19 Several spindles in one channel or TO unit....118
3.3.20 Decoupling the tool management from the spindle number....119
3.3.21 Several spindles/toolholders....125
3.3.22 Several magazines in one channel or one TO unit....126
3.3.23 Reset and start mode....127
3.3.24 Repeating a tool change with the same tool identifier....134

3.4 Search for tool....137
3.4.1 Strategies for tool searches....137

3.5 Empty location search....139
3.5.1 Empty location search for a tool – from spindle to magazine....139
3.5.2 Search strategy for empty locations....140
3.5.3 Assigning a hierarchy to the location types....141
3.5.4 New type of location type hierarchy....142
3.5.5 Search procedure for empty locations....143
3.5.6 Search strategy, 1:1 exchange (old for new)....144
3.5.7 Tool search in wear group....146

3.6 Loading....150
3.6.1 Loading sequence....150
3.6.2 Function of the PLC when loading?....150
3.6.3 Loading by directly assigning the T number....152
3.6.4 Load tools via a part program....152
3.6.5 Retroload tool data....153

3.7 Unloading....156
3.7.1 Overview....156
3.7.2 PLC function when unloading....156
3.7.3 Unloading by deleting the T number from the magazine location....158

3.8 Relocating tools and positioning the magazine....159
3.8.1 Relocate (task from TM system)....159
3.8.2 Relocation by the PLC....159
3.8.3 Positioning the magazine....161

3.9 Tool monitoring (workpiece count, tool life, wear)....163
3.9.1 Monitoring types....163
3.9.2 Tool life monitoring....165
3.9.3 Workpiece count monitoring....166
3.9.4 Wear monitoring....167
3.9.5 Signals to and from the PLC....170

3.10 Tool monitoring without active tool management....173
3.10.1 Tool monitoring overview....173
3.10.2 Tool life monitoring....174
3.10.3 Workpiece count monitoring....175

3.10.4 Wear monitoring....176
3.10.5 Monitoring the additive offset....176
3.10.6 Working with replacement tools....176
3.10.7 Examples....176
3.10.8 NC language commands....177

3.11 Variants of D-number assignments....178
3.11.1 Relative D No. for each T - standard....178
3.11.2 Absolute D No. without reference to the T number (flat D No.)....178
3.11.3 Free selection of D numbers for every T....179
3.11.4 Location-dependent offsets (additive offsets)....181

3.12 Adapter data....185
3.12.1 Overview....185
3.12.2 Description of function....185
3.12.3 Activation....186
3.12.4 Transformed data of the active tool $P_ADT[n]....195

3.13 Power failure while tool command is in progress....196

3.14 PLC description....197
3.14.1 Interfaces....197
3.14.2 Definitions of acknowledgement status....202
3.14.3 Simplified acknowledgement of tool management commands....206
3.14.4 Diagnostics for communication between NC and PLC....207
3.14.5 Function blocks....210

4 Commissioning....211

4.1 Input of the machine data....211

4.2 Load the machine manufacturer PLC blocks....215
4.2.1 Overview....215
4.2.2 Create PLC data....216
4.2.3 Description of the test blocks....218
4.2.4 Delete pending tasks....222

5 Programming....223

5.1 Overview of OPI and system variables....223

5.2 Cutting edge data....226
5.2.1 Cutting edge data....226
5.2.2 Cutting edge parameters....226
5.2.3 User cutting edge data....228
5.2.4 Cutting edge-related tool monitoring....229
5.2.5 User cutting-edge monitoring....230
5.2.6 Location-dependent offsets, fine (additive offsets)....231
5.2.7 Location-dependent offsets, coarse (setting-up offsets)....231

5.3 Tool data....233
5.3.1 Overview....233
5.3.2 Tool-related data....233
5.3.3 Tool-related grinding data....239
5.3.4 Tool-related user data....239

5.4 Magazine data....241
5.4.1 Overview of magazine data....241

5.4.2 Magazine description data....241
5.4.3 Magazine user data....243
5.4.4 Magazine location data....244
5.4.5 Magazine location user data....248
5.4.6 Magazine location type hierarchy....249
5.4.7 Distance to change position....250
5.4.8 Magazine blocks....253
5.4.9 Assigning buffers to spindles....260

5.5 Adapter data....262

5.6 Toolholder data....263

5.7 User-definable variables....266

5.8 NC language commands....268
5.8.1 CHKDNO - Uniqueness check for D number....268
5.8.2 CHKDM - Uniqueness check within a magazine....269
5.8.3 GETACTTD - Determining the T No. for a unique D No....269
5.8.4 GETDNO - Rename D numbers....270
5.8.5 SETDNO - Rename D numbers....270
5.8.6 DZERO - Invalidate D numbers....271
5.8.7 DELDL - Delete additive offsets....271
5.8.8 NEWT - Create a new tool....271
5.8.9 NEWMT - create new multitool....272
5.8.10 DELT - Delete tool....273
5.8.11 DELMT - deleting a multitool....273
5.8.12 $TC_MTPN - Deletion of one or all multitools....274
5.8.13 GETT - Read T No....274
5.8.14 SETPIECE - decrementing the workpiece counter....275
5.8.15 GETSELT - Read the selected T No....277
5.8.16 GETEXET - Reading of the loaded T number....279
5.8.17 $P_MTHSDC - master tool holder with regard to the D offset selection....283
5.8.18 $P_TH_OF_D - master tool holder with regard to the current D offset....284
5.8.19 GETACTT - Read the active internal T No....285
5.8.20 SETMS - Spindle can be declared a master spindle....287
5.8.21 SETMTH - Set master toolholder number....287
5.8.22 POSM - Positioning the magazine....289
5.8.23 POSMT - positioning a multitool on a toolholder to the location number....292
5.8.24 MVTOOL - Language command to move tool....295
5.8.25 SETTIA - Deactivate tool from wear group....298
5.8.26 SETTA - Activate tool from wear group....299
5.8.27 RESETMON - Language command for setpoint activation....300
5.8.28 DELTC - Delete toolholder data record....304
5.8.29 TCA - Tool selection/tool change irrespective of tool status....305
5.8.30 Replacement of the TCA command....308
5.8.31 TCI - Change tool from buffer into magazine....310
5.8.32 GETFREELOC - Search for empty location....311
5.8.33 DELMLRES - Delete location status "reserved for tool in the buffer"....317
5.8.34 DELMLOWNER - Delete owner magazine location of the tool....318
5.8.35 $P_USEKT - Tool change only with tools of subgroup....319
5.8.36 TOOLGNT/TOOLGT - Tool groups....323
5.8.37 $P_TMNOIS - is a number, T number, magazine number or MT number....323
5.8.38 $P_TOOLEXIST - define the existence of a tool....324

5.8.39 $A_TOOLMN - Read magazine No. of tool....325
5.8.40 $P_MTOOLN / $P_MTOOLMT - determine the number of multitools /MT number....325
5.8.41 $P_MTOOLNT / $P_MTOOLT - number of tools in the multitool....326
5.8.42 $A_TOOLMLN - Read magazine location No. of tool....327
5.8.43 $P_TOOLND - Read number of cutting edges for tool....327
5.8.44 $A_MONIFACT - Factor for reading tool life monitoring....328
5.8.45 $AC_MONMIN - Factor for tool search....329
5.8.46 $P_TOOLNG - Number of tool groups....333
5.8.47 $A_MYMN / $A_MYMLN - Owner magazine/location of the tool....333
5.8.48 $A_MYMTN / $A_MYMTLN - $A_TOOLMTN / $A_TOOLMTLN - tools in the multitool....334
5.8.49 $P_TOOLNT / $P_TOOLT - T numbers....336
5.8.50 $P_TOOLD - D numbers....336
5.8.51 $P_TOOLNDL - Number of defined DL offsets....337
5.8.52 $A_USEDND - Workpiece counting....338
5.8.53 $A_USEDT - Workpiece counting....339
5.8.54 $A_USEDD - Workpiece count....340
5.8.55 $P_MAGN / $P_MAG - Magazine....341
5.8.56 $P_MAGNDIS / $P_MAGDISS / $P_MAGDISL - magazine distance tables....342
5.8.57 $P_MAGNS / $P_MAGS - Toolholder....343
5.8.58 $P_MAGNREL / $P_MAGREL - Assigned buffer....344
5.8.59 Example of magazine configuration system variables....345
5.8.60 $P_MAGNH / $P_MAGNHLT / $P_MAGHLT - Location type hierarchies....348
5.8.61 $P_MAGNA / $P_MAGA - Tool adapter....349
5.8.62 Additional language commands....350
5.8.63 Variables for subroutine replacement technique....358
5.8.64 Variables for tool change in synchronized action....359

5.9 Definitions when programming data....361
5.9.1 Tool and cutting edge data....361
5.9.2 Magazine data....362
5.9.3 Tool change....364
5.9.4 Cutting edge selection....365
5.9.5 Tool transfer from program test....366

5.10 Programming T=location number....368

5.11 Call multiple turrets with "T=location number"....370

5.12 Programming examples....371

5.13 Overview of the remaining OPI blocks of tool management....372
5.13.1 Magazine data, directory....372
5.13.2 Tool data, directory....372
5.13.3 Parameterization, return parameters _N_TMGETT, _N_TSEARC....372
5.13.4 Working offsets....373
5.13.5 PI services and language commands for tool management....374

6 Code carrier connection - Tool Ident Connection (option)....381

6.1 Installation....381
6.1.1 System requirements....381
6.1.2 Installation of Tool Ident Connection....382
6.1.2.1 Activation of Tool Ident Connection....382
6.1.2.2 Installation of Wkonvert wizard....383

6.2 Configuration....389

6.2.1 Configuration schematic....389
6.2.2 Configuration of the tool lists....389
6.2.3 TDIIdentCfg.xml....390
6.2.4 Read/write device at the PLC....393
6.2.5 Data conversion....394
6.2.5.1 Using the Wkonvert wizard....396
6.2.5.2 Messages of the Wkonvert wizard....397
6.2.6 Conversion files wkonvert.txt and DefToolDat.txt....398
6.2.6.1 Structure of the description file wkonvert.txt....398
6.2.6.2 Examples for the description files wkonvert.txt....403
6.2.6.3 Tool data initialization file DefToolDat.txt....406
6.2.7 Conversion rule based on the MCIS-C script language....408
6.2.7.1 Structure of the conversion rule....408
6.2.7.2 Access to tool data....410
6.2.7.3 Auxiliary functions for numerical coding....412
6.2.7.4 Configuration of ToolSpec.xml....418
6.2.7.5 Description of the MCIS-C script language....419
6.2.8 Examples....420

6.3 PLC interface in the DB19....425
6.3.1 Structure....425
6.3.2 Function call of the PLC interface....425
6.3.3 Assignment of the PLC for transfer parameters....426
6.3.4 Assignment of the PLC for return values....427
6.3.5 Executing functions....427
6.3.5.1 General sequence when executing functions....427
6.3.5.2 Pulse diagram for positive execution of a function....429
6.3.5.3 Pulse diagram when negative execution of a function....430
6.3.5.4 Pulse diagram "Ident_Ready" signal....430
6.3.6 PLC interface for the Parameter operating area....431
6.3.6.1 Assignment of the PLC interface for the Parameter area....431
6.3.6.2 Load the tool....432
6.3.6.3 Sequence 433
6.3.6.4 Sequence when canceled by the PLC....434
6.3.6.5 Unloading a tool....435
6.3.6.6 Diagnostics....435
6.3.6.7 Restrictions....437
6.3.6.8 Error messages....437
6.3.7 PLC data types....437

6.4 Command codes....439
6.4.1 Functional scope of the PLC interface (K codes)....439
6.4.2 Parameter description....439
6.4.3 Loading from the code carrier (K code = 0)....440
6.4.4 Unload to code carrier (K code = 1)....441
6.4.5 Loading a tool from the tool list (K code = 7)....442
6.4.6 Unloading a tool in the tool list (K code = 8)....443
6.4.7 Empty location search (K code = 9)....445
6.4.8 Deleting a tool in the NCK (K code = 10)....446
6.4.9 Updating the code carrier (K code = 14)....447
6.4.10 Reading data from the code carrier (K code = 15)....448
6.4.11 Error codes....449

6.5 Examples....455

6.5.1 Loading the code carrier via PLC interface....455
6.5.2 Unloading to the code carrier via PLC interface....458

7 Machine data....461

7.1 NC-specific machine data....461

7.2 Channel-specific machine data....485

7.3 Machine data for function replacement....503

7.4 Machine data for the Siemens user data....507

8 Signal description of the PLC interface....509

8.1 Overview of data blocks....509

8.2 Interface for loading/unloading magazine....510

8.3 Interface for spindle as change position....515

8.4 Interface for turrets as change position....522

8.5 Interface for loading/unloading magazine (multitool) (DB 1071)....527

8.6 Interface for spindle (multitool) (DB 1072)....529

8.7 Interface for revolver (multitool) (DB 1073)....534

8.8 Interface NC channels....539

8.9 Interface magazine configuration....541

9 Alarms....543

9.1 Overview....543

9.2 Alarm description....546

A Appendix....577

A.1 List of abbreviations....577

A.2 Documentation overview....579

Glossary....581

Index....591

1 Fundamental safety instructions

1.1 General safety instructions

> ⚠ WARNING
>
> **Risk of death if the safety instructions and remaining risks are not carefully observed**
>
> If the safety instructions and residual risks are not observed in the associated hardware documentation, 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
>
> **Danger to life or malfunctions of the machine as a result of incorrect or changed parameterization**
>
> As a result of incorrect or changed parameterization, machines can malfunction, which in turn can lead to injuries or death.
>
> - Protect the parameterization (parameter assignments) against unauthorized access.
> - Respond to possible malfunctions by applying suitable measures (e.g. EMERGENCY STOP or EMERGENCY OFF).

1.2 Industrial security

Note

Industrial security

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, solutions, machines, equipment and/or networks. They are important components in a holistic industrial security concept. With this in mind, Siemens' products and solutions undergo continuous development. Siemens recommends strongly that you regularly check for product updates.

For the secure operation of Siemens products and solutions, it is necessary to take suitable preventive action (e.g. cell protection concept) and integrate each component into a holistic, state-of-the-art industrial security concept. Third-party products that may be in use should also be considered. For more information about industrial security, visit this address (http://www.siemens.com/industrialsecurity).

To stay informed about product updates as they occur, sign up for a product-specific newsletter. For more information, visit this address (http://support.automation.siemens.com).

WARNING

Danger as a result of unsafe operating states resulting from software manipulation

Software manipulation (e.g. by viruses, Trojan horses, malware, worms) can cause unsafe operating states to develop in your installation which can result in death, severe injuries and/or material damage.

- Keep the software up to date.
  You will find relevant information and newsletters at this address (http://support.automation.siemens.com).
- Incorporate the automation and drive components into a holistic, state-of-the-art industrial security concept for the installation or machine.
  You will find further information at this address (http://www.siemens.com/industrialsecurity).
- Make sure that you include all installed products into the holistic industrial security concept.

Fundamentals 2

2.1 Explanation of terms

The "tool management" function (TOOLMAN) ensures that at any given time, the correct tool is in the correct location and that the data assigned to the tool is up to date. The function is used on machine tools with circular magazines, chain magazines or box magazines. It also allows fast tool changes and avoids scrap by monitoring the tool service life and machine downtimes by using spare tools.

Quantity structure	NCU 730	NCU 720	NCU 710
Number of magazines	64	32	32
Number of cutting edges	3000	1500	1500
Number of tools per multitool	64	32	32

Functions of the tool management

There are four different function versions when handling tools:

- **TMBF**: **T**ool **M**anagement (**B**ase **F**unctions) (also available without active tool management)
  Default setting in NC
  (TMBF = Tool Management Base Functions)
- **TMFD: T**ool **M**anagement **F**lat **D** Numbers (only without active tool management)
  (TMFD = Tool Management Flat D Numbers)
- **TMMO: T**ool **M**anagement Tool (**Mo**nitoring)
  (TMMO = Tool Management Tool Monitoring)
- **TMMG: T**ool **M**anagement **M**a**g**azines (only available with active tool management)
  (TMMG = Tool Management Magazines)

Included in the basic version of SINUMERIK 840D sl are:

- TMBF or
- TMBF + TMFD + TMMG

The function is capable of managing up to 64 real magazines with a total of 1500 magazine locations and 1500 tools, and up to 12 cutting edges per tool (max. 3000 cutting edges). The maximum number of edges per tool depends on the software version and machine data settings.

Basic functions

The basic functions are generally available, even in systems without active tool management. Basic functions include, for example, creating and deleting tools, entering offsets and tool changes. On the basic function level, a specific number (max. 12) of cutting edges (D numbers) is assigned to each T number (tool identification).

Alternatively, the TMFD or "Flat D numbers" (freely selectable D number independently of the T number) function can be activated in systems without active tool management. You can select any number of tool edges per tool; the number of tool edges per tool is not limited to 12. With "Flat D numbers", the user is responsible for the management and assignment of T numbers to D numbers.

Note

SINUMERIK Operate does not support the "Flat D numbers" function.

Basic functions (standard)

Basic functions of the tool management (standard)	SINUMERIK Operate
System screens in the standard software	X
Options for configuring screen forms and softkeys	X
User-friendly commissioning via system screens	X
Editing tool data	X
Magazine and tool list	X
Empty location search and location positioning	X
Loading and unloading of tools	X
Easy search for empty locations using softkeys	X
Several real magazines are possible	X
Several loading and unloading points per magazine	X
"Relative" D numbers that can be freely numbered	-
Adapter data	-
Location-dependent offsets	-
Data backup via USB	X
Data backup on hard disk	-
Data backup on CF Card	X

Special functions

The special functions of tool management are magazine management, tool and empty location search and monitoring the tool life, workpiece numbers or wear values. These additional functions are only available when tool management is active.

Without active tool management, the magazine management must be implemented by the machine manufacturer in the PLC user program.

Magazine management

Magazine management administers magazine locations. These locations might be empty, loaded with tools or assigned to oversized tools in adjacent locations.

Empty locations can be "loaded" with other tools. The tool management provides the machine manufacturer with optimized management of tools and magazine locations.

Magazine management provides extended functions such as load, unload or position tools. It also includes searches for tools, magazine locations and search strategies for replacement tools. For the tool monitoring functions, while the activated monitoring is running, tools are disabled and no longer used. For further machining, if a similar tool (duplo tool) is present which is not disabled, then this tool is used automatically.

In the simplest case, all that needs to be configured when tool management is active are magazines, loading magazines, spindles, grippers, etc. Furthermore, the interfaces (DB 71 to DB 73) must be processed in the PLC.

Task-related tool motions (e.g. position chain, swivel gripper) are derived from the interface processing. After tool movements, the positions and task status should be acknowledged via basic program blocks (FC6, FC7 and FC8). A detailed description of the acknowledgements is provided under the index entries "Acknowledgement: simplified", "Acknowledgement status" and "Changing acknowledgement data".

If necessary, create another cycle (or ASUP) for the NC program where tool changing is programmed with the required travel motions. An identifier is programmed for the tool change or tool preselection when the TOOLMAN system is active. A duplo number is also available to support unique identification of replacement tools. Tool identifier and duplo numbers are always mapped on an internally assigned T number. This internally assigned T number is used for addressing the variables described in the following.

OPI variables

Additional functions are available using OPI variables on the PLC or HMI side. The NC program (e.g. cycle, ASUP) provides corresponding language commands to achieve optimum adaptation of the tool management to the machine environment. You can obtain a clear overview from the data structures that form the basis of the tool management. They are represented in the form of NC data blocks.

Operator panels

The following operator panels can be used for tool management:

- OP177, e.g. for loading magazines
- OP8T, e.g. for box-type magazines

Data

Data storage and management is carried out in the NC and SINUMERIK Operate. All data can be read and written manually via the NC program or by data transfer.

Operation via system screens

Operation is mainly performed via system screens. There are system screens for commissioning and for tool management operation (magazine lists, tool lists, loading/ unloading).

Configuration of user-specific screens

Tool-specific data that is not displayed via SINUMERIK Operate system screens can be displayed and changed in configurable user-specific screens.

The configuration of user-specific screens is described in:

References

- SINUMERIK 840D sl: Commissioning Manual, Base Software and Operating Software; Section "SINUMERIK Integrate Run MyScreens (BE2)"
- SINUMERIK 828D: Commissioning Manual; Section "SINUMERIK Integrate Run MyScreens (BE2)"

Programming in the NC part program

The tool management function makes it possible to call a tool in the part program using a name (identifier), e.g. T = "end mill 120 mm".

Tool call is still possible via the T no. (tool number). The T No. is then the name of the tool (e.g. T=12345678).

A tool is uniquely defined by its name and duplo number. Furthermore, each tool can be uniquely identified by its "internal" T number. The internal T number is generally assigned by the NC and is not used for programming a tool change in the part program.

Using a machine data, it can be set as to whether the change is realized using the T command (typical for lathes) or whether the tool is prepared using T programming and then it is first changed with M06. This machine data setting is independent of the magazine type.

The following characters are permitted for the identifier:

[_] [a...z] [A...Z] [0...9] ; [+ - . ,]

Identifiers are case-sensitive, i.e. differentiate between upper-case and lower-case characters.

PLC

There are data blocks (DB71-73) as well as DB1071-1073 for receiving tool management commands and function blocks (FC6, FC7, FC8) for acknowledging the tool management commands. A "fast acknowledgement" is implemented for simple applications in data blocks DB71...73.

Another block, FC22, is used as a direction selection for magazines.

Tool management data can also be read and written via FB2 and FB3. Complex tool management services can be initiated via FB4.

Magazine types

Turret, chain magazines and box magazines can be managed. Other magazine types, e.g. pick-up magazine, are mapped onto these.

Real magazines can be defined as a turret, chain or box-type. 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, ...).

Location coding

Fixed location coding as well as variable location coding are supported for the tools.

Location type

The location type defines the type and shape of the location. By assigning location types to magazine locations it is possible to subdivide a magazine into areas. This means that different types of special tools, e.g. "especially_large, "especially_heavy" can be assigned to specific locations.

The location types can be placed in ascending order - a so called hierarchy. This order determines that a tool that is supposed to be inserted in a "small" location type can also be placed in a "larger" location type if no "smaller" location type is vacant.

Monitoring functions

In tool management, it is possible to select either workpiece counts or tool life monitoring (with reference to the cutting edges). Further, a wear monitoring function is available. Replacement tools (duplo tools) are differentiated by means of a duplo number.

Search strategy

Various strategies are possible for tool search and to search for the empty location of the "old tool".

The empty location search strategy is used for loading tools.

Excerpt from TOOLMAN basic data

Term	Data/Range
Magazine configuration per channel	1
Total number of magazines	Max. 64
Total number of magazine locations	Max. 1500
Total number of tools	Max. 1500 per TO unit / max. 1500 across all TO units
Programming tools (names) in the NC program using 32 alphanumeric characters	e.g. T="angled miller_32"
Duplo number [1)]	1 – 32000
Total number of cutting edges	Max. 3000
Location type definition	Yes
Viewing adjacent locations in half locations	2dimensional
Location coding	Fixed or variable
Strategy for tool search	can be set (programmable) via system variables
Strategy for empty location search	can be set (programmable) via system variables

Term	Data/Range
M06 command for tool change	M code, settable via MD, channel-specific
Tool change with M06 or T command	settable via MD, channel-specific
Wear monitoring	For every cutting edge
Wear monitoring according to tool life	Resolution ms
Wear monitoring according to number of workpieces	Counters
Access to tool management data via NC program	System variables
Automatic decoding stop until tool is selected	Yes
T=Location No.	settable via MD

1) The duplo number or the duplo tool is called "sistertool" in SINUMERIK Operate.

Option

Tool management with more than two real magazines is an OPTION.

2.2 Function structure of tool management

HMI

- Tool list
- Tool wear list
- OEM tool list
- Magazine list

NCK

- Manage tool data
  State
  Monitoring
  Offsets
- Manage magazine data
  Magazine
  Magazine locations
- Tool management
  Search for tool
  Search for empty location
  Change tool
  Load, unload

PLC

- Magazine control
- Gripper control
- Spindle control
- Safety interlocks
- Execute tool change
- Calculation of position, if necessary
- Own change strategy, if required

2.3 HMI/PLC - NCK data structure (OPI)

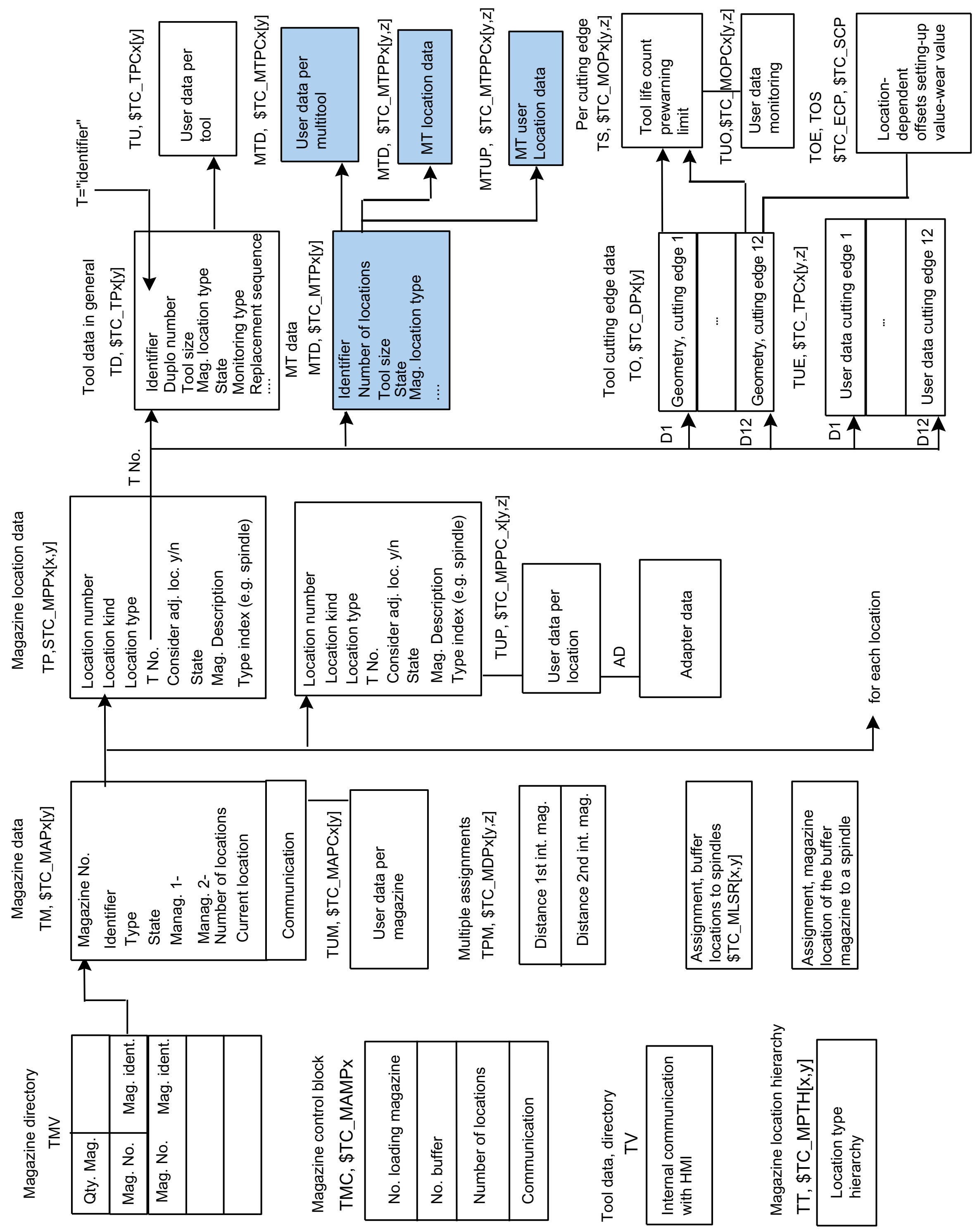


Image 2-1 Structure of magazine data and tool data

Unchecked boxes mark the previous data of the tool management. Checked boxes show the user data.

New data blocks are displayed as checked and grayed boxes.

TOA area

A TOA area constitutes and independent group in tool management. There is no link existing to other TOA areas.

Up to 10 independent TOA areas may be created depending on the number of channels available. Several channels can be assigned to one TOA area but one channel cannot be assigned to more than one TOA area. A subset of magazines, buffer locations and loading magazines can be assigned to one TOA area.

2.4 PLC – NCK interfaces

Overview

The heart of the SINUMERIK 840D sl tool management system is located on the NCK. The PLC merely contains the interfaces for the machine-specific part.

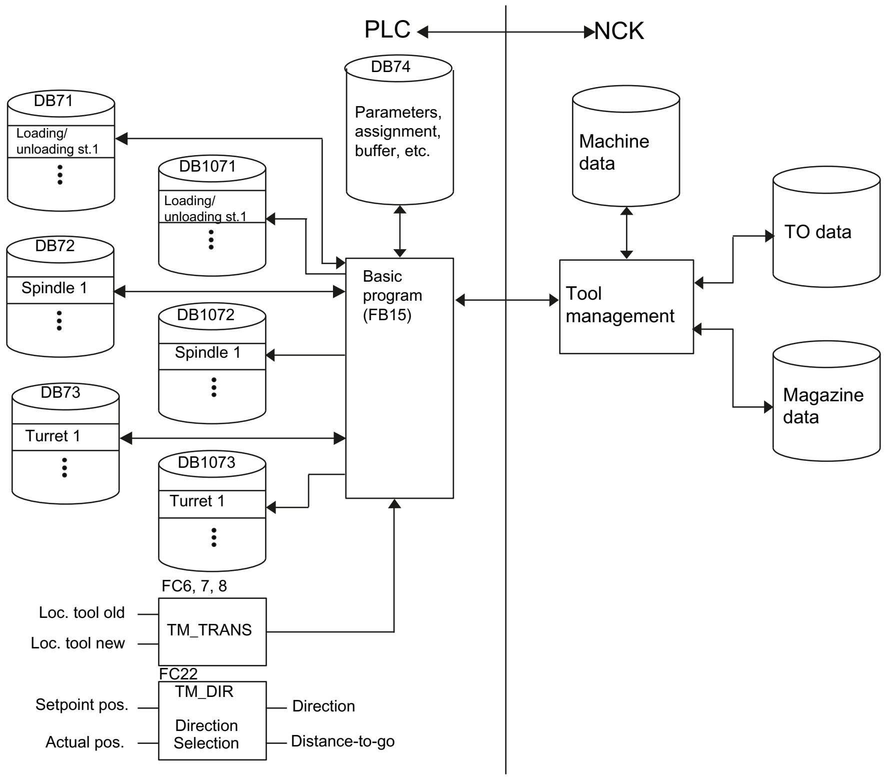


Image 2-2 Data structure and PLC - NCK interface

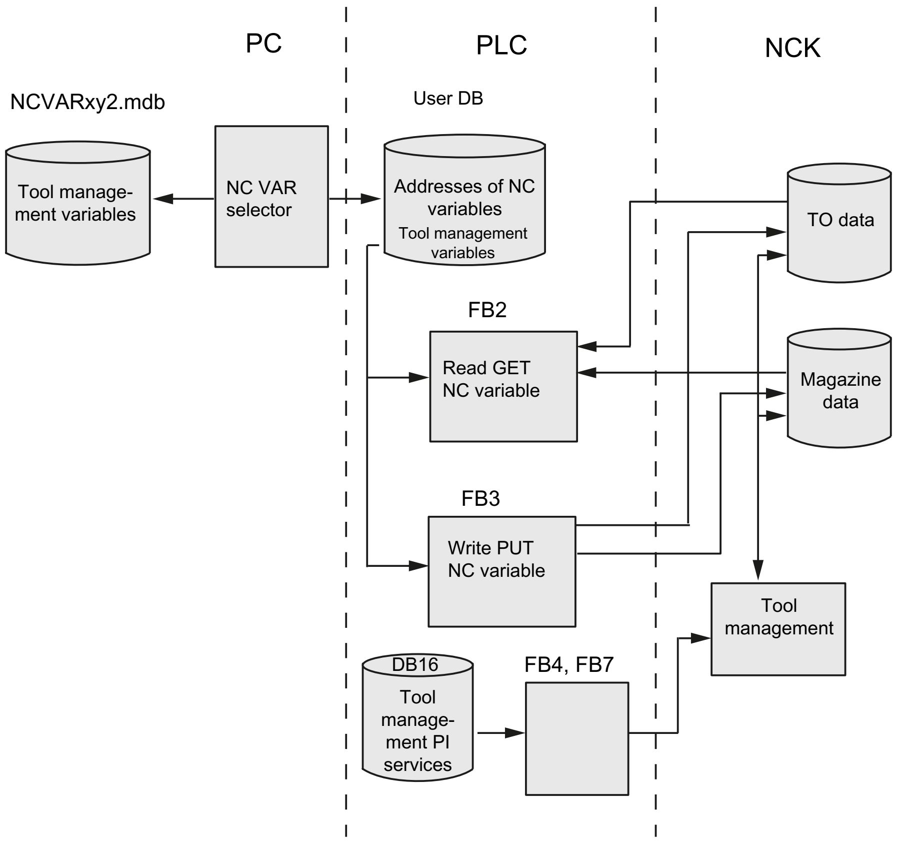


Image 2-3 Extended interface for tool management between PLC and NCK

2.5 Magazine configuration

Magazine configuration

In a configuring process, (HMI or NC program) one or several real (actual magazine to store tools, NCK can manage several magazines), magazines can be combined on the HMI to form a magazine configuration. All the magazines of one configuration can be operated simultaneously in one channel. Several magazine configurations can be defined but only one configuration can be active in one channel at one time.

Magazine and tool data are stored in the NC in the so-called TO area. The TO area can in turn be sub-divided by machine data into individual TO units. It must further be defined by machine data, which channel works or which channels work on which TO units. Only one magazine configuration can be active at any one time per TO unit. If several channels are assigned to a TO unit, then the magazine configuration applies for all assigned channels.

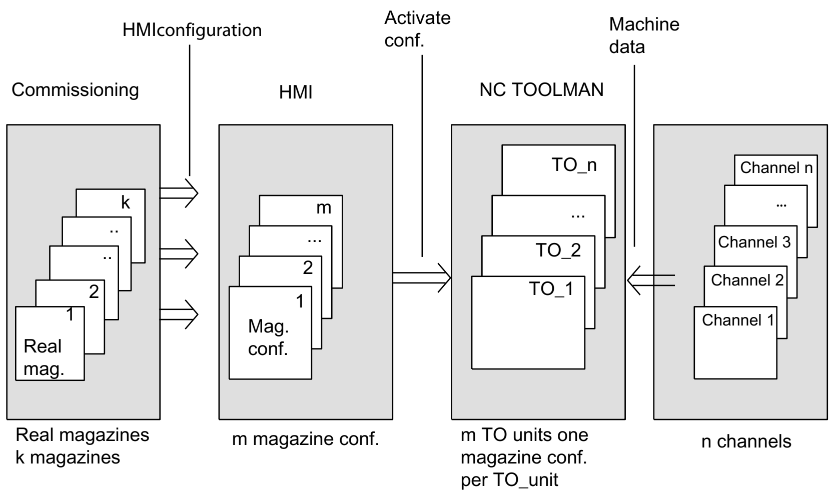


Image 2-4 Assignment of magazines to channels

2.6 Access protection, protection levels

The access to programs, data and functions is protected via eight hierarchical levels according to customer requirements. These are divided into

- Four password levels for Siemens, machine manufacturers and end users
- Four keyswitch positions for end users

Protection level	Locked by:	User
0	Password	Siemens
1	Password	Machine manufacturer: Development
2	Password	Machine manufacturer: Commissioning engineer
3	Password	End user: Service
4	Keyswitch, position 3	End user: Programmer, machine setter
5	Keyswitch, position 2	End user: Qualified operator, who does not program
6	Keyswitch, position 1	End user: Selected operator, who does not program
7	Keyswitch, position 0	End user: Semi-skilled operator

Examples of functions that can be disabled with SINUMERIK Operate:

- Loading
- Unloading
- Magazine list, tool list display
- Tool cabinet, tool catalog
- Loading the magazine configuration

Description of functions 3

3.1 Magazines

3.1.1 Buffer

A buffer involves the 2nd internal magazine. The magazine number is always 9998. The buffer includes the spindle, tool holder, gripper, loader and transfer location. The locations are numbered, just like a real magazine, ascending from 1-n. Every location has a location index. Here, it is assigned as to which is the 1st spindle or the 1st gripper. Any numbering can be used. However, transparency is improved if the locations of a location type (all spindles, all grippers,...) are numbered in an ascending sequence. See also Section "Overview of magazine data (Page 241)".

The NCK only makes a differentiation between spindles or toolholders and grippers. Other "location types" such as loaders or the transfer location are mapped on the gripper.

Example: Assigning the locations in the buffer magazine

No.	Name	Type	Index	Assignment to spindles	Distances to magazine
1	Spindle_1	Spindle	1		0
2	Gripper_1	Gripper	1		0
3	Gripper_2	Gripper	2		0
4	Loader_1	Loader	3		0
5	Loader_2	Loader	4		0
6	Transfer_1	Transfer location	5		0

3.1.2 Loading magazine

The loading magazine is the 1st internal tool magazine and always has magazine number 9999. The loading magazine contains loading locations.
These are sub-divided into

- Loading points and
- Loading stations

Loading points and stations are provided for loading and unloading tools. The allocation of locations 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 in all spindles / tool holders, as well as to load and replace manual tools (see Manual tools (adding tools during machining) (Page 94)). A loading point is an open entry to the magazine where tools can be **manually** put into and taken directly from the magazine.

A loading station is viewed as an "external magazine location" which a gripper, for example, can access to transport a tool to the magazine during loading.

The difference between a loading point and a loading station is as follows:

When loading a tool at a loading/unloading point, the tool is automatically removed from this location, wheres when unloading to an unload station, it remains at the loading or unloading location - it must then be removed from there by the application (HMI, NC cycle or PLC).

3.1.3 Box-type and chain magazines

Fundamentals

Chain and box-type magazines do not as a rule have any additional buffer available for transportation between magazine and spindle. These additional buffers can temporarily contain tools.

The command is distributed in the PLC using the basic program. In this case, DB72 is used as the user interface. There is a separate interface area for each spindle in the interface. A new command from the NC is only then entered in the interface once the previous command has been acknowledged with status values 1 to 7 via FC8/FC6.

1. The programming function T = identifier is implemented in the PLC in DB72. Bit "Prepare tool" is set in the associated interface.
2. Programming function M06 is also implemented in DB72. In this instance, bit "Change tool" is set in the activated interface. The bit "Prepare tool" from an earlier T command is not reset here. If the bit "Prepare tool" shall no longer be set for the M06, then it is the task of the user program to reset this bit as part of acknowledging the last T command.
3. Programming functions T and M06 in the same block set the "Prepare tool" and "Change tool" bits simultaneously in the activated DB72 interface.

Exceptional cases which are mapped in the PLC identically to case 3 above are as follows:

- Initiation of a tool change after block search (last accumulated tool change for the active tool)
- Initiation of a tool change for initialization blocks (reset and start mode), if this was set via the machine data.

Note

Special cases, box-type and chain magazines

In these exceptional cases, the subroutine (macro, cycle) in which M06 is normally programmed is not executed.

Example for machine tools with chain and box-type magazines

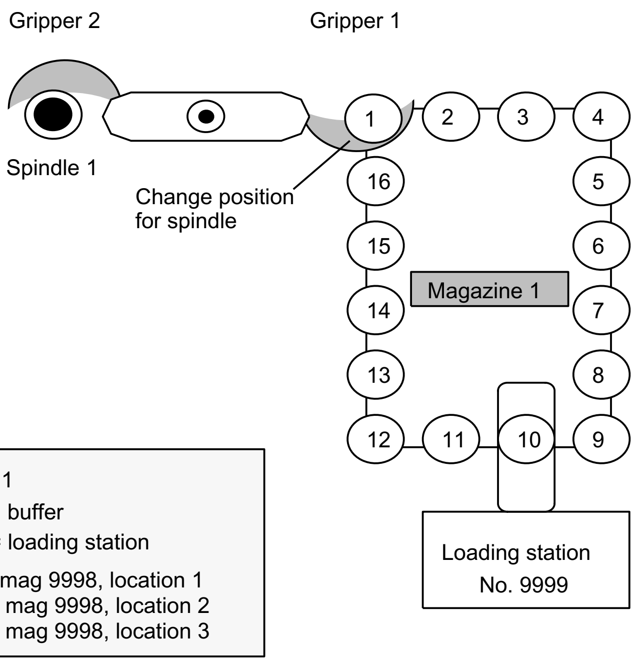

Gripper 2
Gripper 1
Spindle 1
Change position for spindle
Magazine 1
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Loading station No. 9999
Mag 1= no. 1
Mag 9998 = buffer
Mag 9999 = loading station
Spindle 1 = mag 9998, location 1
Gripper 1 = mag 9998, location 2
Gripper 2 = mag 9998, location 3

Image 3-1 Machine tool with chain magazine

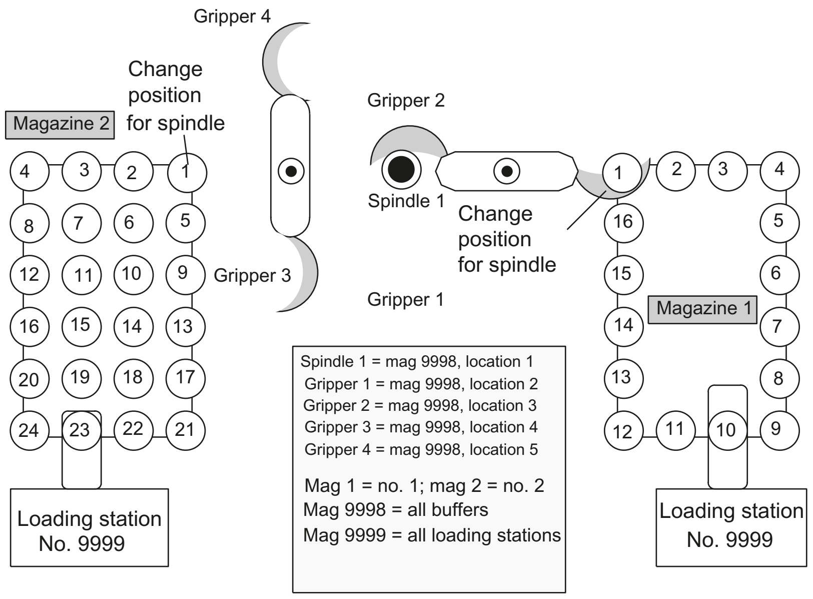

Gripper 4
Change position for spindle
Magazine 2
Gripper 3
Loading station No. 9999
Gripper 2
Spindle 1
Change position for spindle
Gripper 1
Magazine 1
Spindle 1 = mag 9998, location 1
Gripper 1 = mag 9998, location 2
Gripper 2 = mag 9998, location 3
Gripper 3 = mag 9998, location 4
Gripper 4 = mag 9998, location 5
Mag 1 = no. 1; mag 2 = no. 2
Mag 9998 = all buffers
Mag 9999 = all loading stations
Loading station No. 9999

Image 3-2 Machine tool with chain and box-type magazine

The magazine zero point is defined by $TC_MDP2 (with value assignment of 0). The change position (spindle) is normally assigned on this basis.

3.1.4 Circular magazine

Generally, turret magazines do not have any additional buffer with which tools can be transported from the magazine to the spindle. The tools on turret 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 holder in the software.

If the value 1 is also set for the turret in MD22550: $MC_TOOL_CHANGE_MODE, the statement made for the chain and box-type magazines also applies here.

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.

The command is distributed in the PLC using the basic program. In this case, DB73 is shown as the interface for the user (if TOOL_CHANGE_MODE is set to 0). There is a separate interface area for each turret. The turret numbers are assigned successively in ascending sequence according to magazine numbers during commissioning. The permissible magazine range is 1 ... maximum number of real magazines. A new command from the NCK is only entered in the interface when the previous command has been acknowledged via FC7 (alternatively also via FC8/FC6).

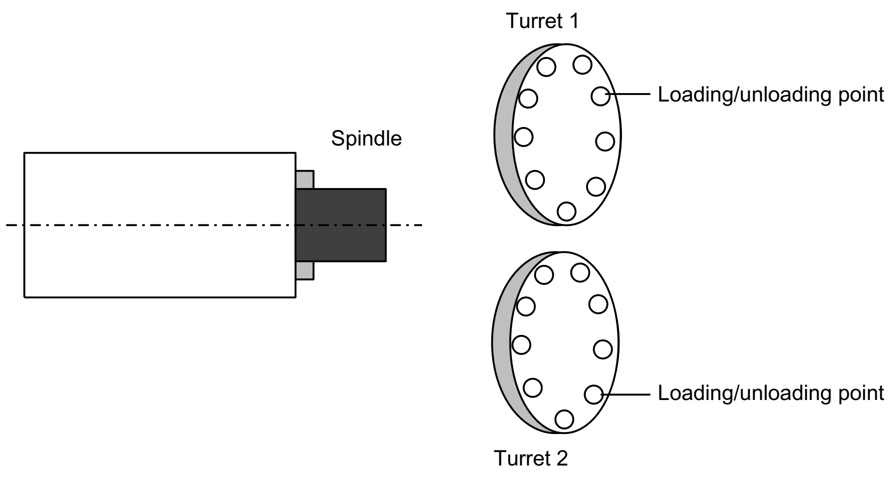


Image 3-3 Double-slide turning machine with direct loading/unloading point in the turret

3.1.5 Other magazine types

In practice, there are other types of magazines in addition to the ones listed above. These are e.g. disk-type, washer, pick-up, rack, cage magazines (and many more). Such types must be mapped to the three types of magazines supported by the tool management.

3.1.6 Wear group

Fundamentals

Locations in a magazine are linked to form an area referred to as the wear group. In this way, locations of a magazine can be reserved for particular machining operations.

A wear group number is assigned to each of these locations and the magazine is thereby divided into several different areas. Only tools from one of the areas are then used for a specific machining operation.

The wear group number for each magazine location is defined via system variable **$TC_MPP5[m,p]** (m: Magazine number, p: Location number).

Values from -32000 to +32000 are possible.

Value > 0:

The specified number is assigned to the location (e.g. **$TC_MPP5[1,3] = 2** assigns the third location of magazine 1 to wear group number 2).

Value = 0:

The location is not assigned to a wear group, threfore the magazine locations are generally included in tool searches.

Values < 0:

The wear group whose number corresponds to the absolute value of this number is disabled (e.g. **$TC_MPP5[1,3] = - -2** disables wear group number 2 of the magazine with number 1).

This applies even if there is only one disabled location in the wear group.

Note

Wear groups are only available for real magazines. The definitions for $TC_MPP5 do not affect the status of tools.

Up to eight wear groups per magazine are possible.

Activate wear group

System variable $TC_MAP9 defines which wear group (magazine area) is active. To change the active wear group, the corresponding number is set in this system variable, thereby defining which wear group will be used to start the machining operation.

The default setting is 0.

The wear group can also be set active internally by a tool change or by the user via NC commands/OPI.

Disable wear group

If there is no longer any tool that is ready for use at the location of an active wear group, then the system switches to the next wear group and the old wear group is disabled.

Machining is continued by activating the next group and searching for a suitable replacement tool.

The wear group is also disabled if one of the locations has been disabled via system variable $TC_MPP5 (negative value).

Activate (internally)

Bit 0 of system variable **$TC_MAMP3** can be set to determine how internal activation of a wear group will affect the status of the associated tools.

Value 0:
The tool status is not changed (preset).

Value 1:
When activated, one tool from each included tool group is set to "active". Tools already set earlier as active are not reset.

Disable (internally)

Bit 1 of system variable $TC_MAMP3 can be set to determine how internal deactivation of a wear group will affect the status of the associated tools.

Value 0:
The tool status is not changed (preset).

Value 1:
When a wear group is disabled all active tools are reset.

Note

For a tool search in the wear group, see Section "Tool search in wear group (Page 146)".

3.1.7 Background magazine

Background magazines are not directly supported by the tool management. However, functions for background magazines can be activated by setting the system variable selectively. System variable $TC_7 MAMP2, bit 7 can be used to set whether the tool search begins in the magazine last used for tool replacement (bit 7 = 0) or whether the search is carried out in the order defined by "Spindle to magazine" (bit 7 = 1).

Generally, this system variable is set when configuring the magazine. A change can be made at any time via the part program or OPI.

The assignment of "spindle to magazine" is set via system variable $TC_MDP2[n,m]; the order corresponds to the order in which this variable is written. This is also pre-assigned by the magazine configuration.

Example for four magazines and one spindle:

$TC_MAMP2=385 (bits 0, 7 and 8 set).

$TC_MDP2[1,1]=0

$TC_MDP2[2,1]=0

$TC_MDP2[3,1]=0

$TC_MDP2[4,1]=0

... this assigns the first buffer (spindle) to magazines 1 to 4; a tool search would therefore start in magazine 1, followed by magazine 2, etc. up to magazine 4.

You can modify this search order by setting this system variable as follows:

1. Deleting the assignment
   $TC_MDP2[1,0]=9999
   $TC_MDP2[2,0]=9999
   $TC_MDP2[3,0]=9999
   $TC_MDP2[4,0]=9999
2. Re-assigning in a different order
   $TC_MDP2[2,1]=0
   $TC_MDP2[3,1]=0
   $TC_MDP2[4,1]=0
   $TC_MDP2[1,1]=0
   ... resulting in the search order of magazine 2, 3, 4, 1

The trigger criterion for changing the order of assignment can be the information in the change cycle that the new tool was found in another magazine. This can be read in the program via $A_TOOLMN[t], whereby "t" is the internal T number of the tool. The new tool is obtained via GETSELT. You must remember the previous foreground magazine.

3.1.8 Consider adjacent location

Fundamentals

Consider adjacent location is used for oversized tools. When searching for empty locations (loading, relocating, changing) bits 4 to 11 are evaluated in the magazine location parameter $TC_MPP4 (half location occupied/reserved). As this function requires additional memory space, the default setting is 0.

To activate the function, set the following:

$MN_TOOL_MANAGEMENT_MASK, bit 3 = 1

$MC_TOOL_MANAGEMENT_MASK, bit 3 = 1

In addition, for every magazine location that is to be considered, parameter $TC_MPP3 = 1 must be set.

Two functions are available when the consider adjacent location is active.

Overlap disabled magazine locations

The function is activated by setting the magazine location parameter $TC_MPP4 bit 13 = 1. If a location is disabled, it can now be "overlapped" by an oversized tool. This means the consider adjacent location ignores the disabled state of a magazine location.

Example:

Chain magazine, location 12 is disabled (e.g. the tool acceptance is defective). An oversized tool (size 2/2/1/1) is loaded or is positioned in the spindle. The tool can now be stored at location 11 or 13.

The following can be selected as default setting:

As soon as a location is disabled, "Overlapping active" is automatically set or reset as soon as the location is enabled.

This setting is made in machine data $MN_TOOL_DEFAULT_DATA_MASK, bit 4 = 1.

Overlapping magazine edge locations

This function is activated by setting the magazine description parameter $TC_MAP3, bits 8 to 11.

The following definition applies:

(Definition: smallest magazine location number is at the top left, the largest magazine location number is at the bottom right).

Bit 8 **Left** edge location must not be covered

Bit 9 **Right edge** location must not be covered

Bit 10 **Top** edge location must not be covered

Bit 11 **Bottom** edge location must not be covered

The default setting for these bits is 0.

Example:

Box-type magazine

Due to the mechanical conditions, oversized tools can cover the edge at the top and on the right.

The following must be set:

$TC_MAP3[magazine no.], bit 8 = 1

$TC_MAP3[magazine no.], bit 11 = 1

3.2 Several tools at a magazine location (multitool)

3.2.1 Brief description

Overview

With the TMMG function (magazine management) and when the "multitools" function (MT) is activated, so-called multitools (mini turrets) with a number of tools can be loaded into a magazine and unloaded from a magazine like a tool.

Note

This function is supported in SINUMERIK Operate only.

Note

The following functions are not implemented in the current version:

- Manual tool (a multitool cannot be a manual tool)
- Magazine location adapter (a multitool can be located on an adapter, but adapters are not permitted on multitool locations)

The T selection in the part program also detects and checks the tools of a multitool when searching for a tool.

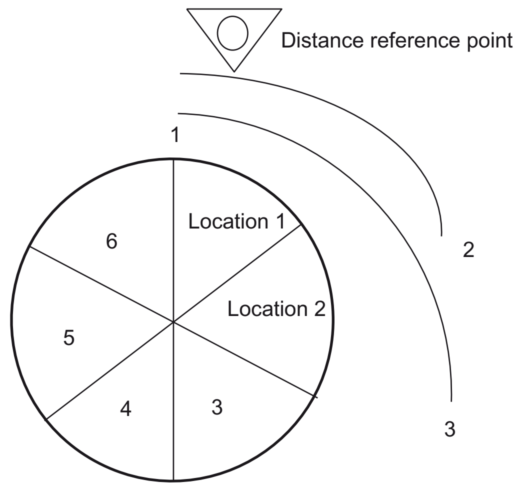
A multitool has a definable number of MT locations, where tools can be located. The geometrical arrangement of the MT locations can either be defined using the MT location number, an angle or via a distance.

The commands to select a tool and to change a tool sent to the PLC have additional information regarding the distance reference point or the machining position:

- MT location number of the tool in the multitool, or
- Angle, or
- Distance of the tool within the multitool

This type of distance coding can be defined for each MT. The PLC then initiates the corresponding machine handling, e.g. moving a positioning axis.

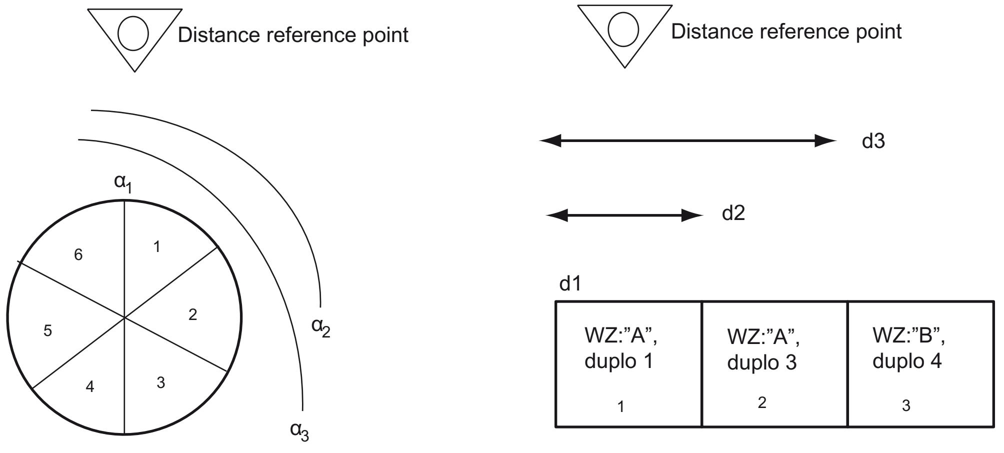
A new acknowledgement block is available in the basic PLC program, the FC6. The block corresponds to the already known FC8. It only has one additional parameter "MultitoolPosition". The FC6 includes the complete FC8 functionality, so that the FC6 can completely replace the FC8. Analogous to DB71, DB72 and DB73, data blocks DB1071, DB1072 and DB1073 are available as user interface. If, for example, a multitool is changed, then the complete information of the "carrier tool" is contained in DB72, and all the information about the individual tool in DB1072.



Location number definition for the MT location distance:
MT location 1 = 1, MT location 2 = 2, ...MT location 6 = 6

Image 3-4 Location number

The figure shows the distance coding of the locations in the multitool, i.e. the MT location number of the particular MT location itself. This distance coding is especially suitable for machines that operate with similar multitool geometries, e.g. only mini turrets.



Angle definition
for the MT location distance:
α_1 =0, ... α_3 = angle 1 to location 3

Length definition
for the MT location distance:
d1 = 0, ...d3 = angle to location 3

Image 3-5 Angle and length

Typical geometries can be seen in this figure. Angle α and the lengths are output on the PLC. This means, for example, that multitools can be used on a machine and handled by the PLC, which have different geometries (MT locations are not symmetrically arranged: Although multitools are inherently symmetrical, different multitools have different distances between the MT locations).

The distance reference point or the machining position is defined machine-specifically. The assignment of the MT location numbers to the multitool locations must correspond to the machine construction.

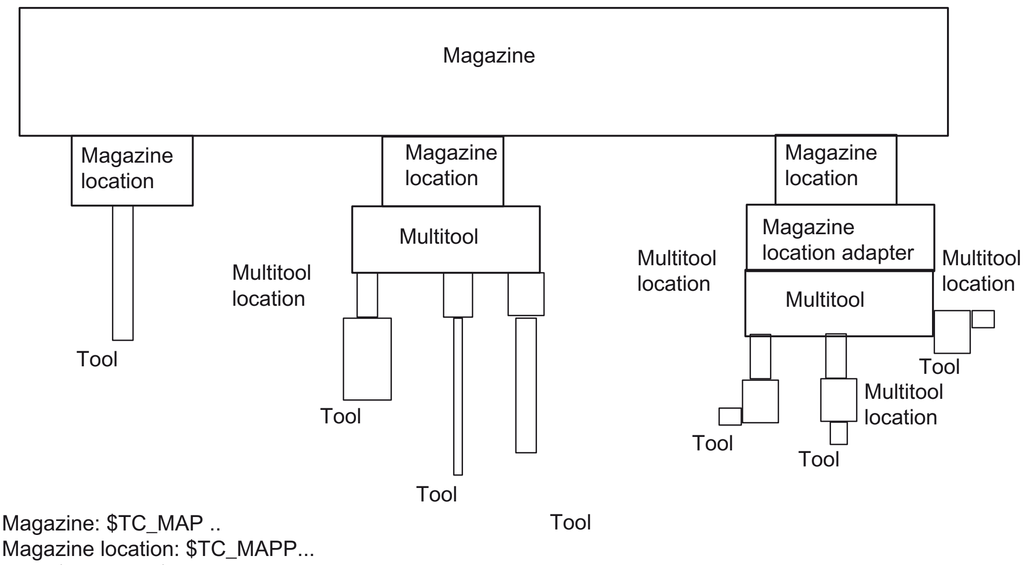
A multitool can contain tools from different tool groups, i.e. the names can be different.

Nothing changes when programming a tool change in the part program. Formally, the commands are identical to existing commands; i.e. the tool change is programmed using T and/or M06.

For the function T = turret location number, it is also necessary to define which tool of the turret magazine is meant when the location contains more than one tool.

Note

Presently, SINUMERIK Operate supports the distance coding "angle" and "location number".



Magazine: $TC_MAP ..
Magazine location: $TC_MAPP...
Tool: $TC_TP.../$TC_DP...
Multitool: $TC_MTP...
Multitool location: $TC_MTPP...

Image 3-6 Multitool shown in the magazine

The following machine data and language commands are available for the "multitool" function:

Machine data

- $MN_MM_TOOL_MANAGEMENT_MASK, bit 10
  Activation of the MT function
- $MN_MM_NUM_MULTITOOL
  Number of defined MT (limit values: 0-1500; max. 600 per channel)
- $MN_MAX_TOOLS_PER_MULTITOOL
  Max. number of tool locations per MT (limit values: 2-64)

Language commands / system parameters

- $TC_MTP, $TC_MTPP
  Data definition for MT and MT location
- $TC_MTPC, $TC_MTPPC
  MT OEM data for tool management
- $TC_MTPCS, $TC_MTPPCS
  MT OEM data for Siemens
- NEWMT, DELMT
  Creating, deleting
- POSMT
  Positioning MT on tool holder MT location number
- $P_TMNOIS
  What type does the number refer to: Magazine, tool or MT
- $P_MTOOLN, $P_MTOOLMT
  Number of multitools, MT number of the i-th MT
- $P_MTOOLNT, $P_MTOOLT
  Number of tools in the multitool, T number of the i-th tool
- $A_TOOLMTN, $A_TOOLMTLN
  Location of the tool in the multitool, MT number / MT location number
- Synchronized action access in the tool management PLC command
  $AC_TC_MTTN, $AC_TC_MTLTN, $AC_TC_MTNLOC, $AC_TC_TOOLIS,
  $AC_TC_MTDIST

NC commands

- RESETMON
  Can reset the tool monitoring data of the tools in an MT to their setpoints
- GETT
  Determines T no., MT no. for the name
- DELMLOWNER
  Deletes the magazine owner location of the tool or MT
- MVTOOL
  Moves tool or MT
- GETFREELOC
  Searches for an empty location in the magazine for tool or MT, searches for an empty location in the MT for tool

3.2.2 Programming

MT numbers

The multitool numbers originate from the same number space as the T numbers and the magazine numbers.

Numbers 1-32000 are permitted for the T and magazine numbers. This allows multitools to be handled just like simple tools (T number at the magazine location, T number as parameter in the OPI PI service to load and unload tools, etc.). In addition, in some instances MT numbers can be handled just like magazine numbers.

Users can explicitly enter T or MT numbers in the NC programming, if they want to create a tool or multitool by programming $TC_TPx or $TC_MTPx. Explicitly programmed MT numbers must not collide with already defined T numbers or magazine numbers. The programming is then aborted with an alarm. However, if users create a new tool with the language command NEWT – or for multitools with NEWMT – then the NCK automatically allocates the T or MT numbers. When automatically allocating an MT number, it is ensured that this is neither an already defined tool nor an already defined magazine.

NC language commands or system parameters, which have a tool T number or magazine number as parameter or index, and which are called using an MT number, generate an alarm if the associated functionality for multitool is not explicitly defined.

The same number can be used for a tool and a magazine.

Example 1:

The magazine numbers 1, 2, 3, -, 5, -, -, - and the T numbers 1, -, 3, -, 5, 6, -, -, have already been defined.
Then -, -, -, 4, -, -, 7, 8 are available for the MT numbers.

Example 2:

The magazine numbers 1, 2, 3, -, 5, -, -, - and the MT numbers -, -, -, 4, -, 6, 7, -, have already been defined.
Then 1, 2, 3, -, 5, -, -, 8 are available for the T numbers.

MT names

Multitool names come from the same namespace as the tool names and magazine names. Different multitools must have different names, i.e. a duplo number is not defined.

An MT name must not be the same as a tool name and not the same as a magazine name. Hoever, a magazine name can be the same as a tool name.

3.2.3 $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 systemology of the tool magazine function.

All of the parameters in the following table have no physical units. The default access right is "key-operated switch 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: Parameters 1 …8, …PROTA

y: = Multitool number MT = 1 …32000

NCK identifier	Description	Format	Pre-assigned limit values
$TC_MTPN	The number of locations is defined as delete command	INT	2 2-$MN_MAX_TOOLS_PER_MULTITOOL
$TC_MTP2	Identifiers	STRING	MT number
$TC_MTP3	Size to the left in half locations (= towards a lower location number)	INT	1 1 - 7
$TC_MTP4	Size to the right in half locations (= towards a higher location number)	INT	1 1 - 7
$TC_MTP5	Size upwards in half locations (= towards a lower location number)	INT	1 1 - 7
$TC_MTP6	Size downwards in half locations (= towards a higher location number)	INT	1 1 - 7
$TC_MTP7	Magazine location type	INT	9999 (not defined) 0 - 32000
$TC_MTP8	State; bit-coded	INT	0 (not enabled) 0 - "4000"
$TC_MTP_POS	MT position. The following applies: Position 1 is defined as "location 1 in the machining position"	INT	0 (not defined) 0-$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[mtNr] = 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. for tool search strategies, empty location search strategies. Since the results depend on the type of magazine, the displayed property is 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 that apply when changing the multitool size correspond to the rules that apply when changing the tool size.

The MT size of a multitool can be changed, as 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, which 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	**enabled**	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.

Bit	Meaning	Comment
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.
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 disabled**	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	**Master tool**	Multitool can be selected as master tool
13	Reserved	Reserved
14	**Selected for 1:1 exchange**	Multitool is selected for the pending 1:1 exchange. The rules as for a 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 state synchronously. The tools not involved with the tool selection do not experience a state change.
16	**MT is disabled, if a tool in the MT is disabled**	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 state 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	enabled	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.
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, then 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 disabled	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.

Bit	Meaning	Comment
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 disabled, if a tool in the MT is disabled**	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, then 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 results in the states of the tools loaded to the MT being checked after the operation and the MT state "disa-bled" 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 is set to 1.

If the "disabled" state of any tool in this MT is set (either manually or automatically), then in addition, the "disabled" state is set in the MT, although there are still two tools where 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 into the "disabled" state when this tool – as the last tool – also assumes 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 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:

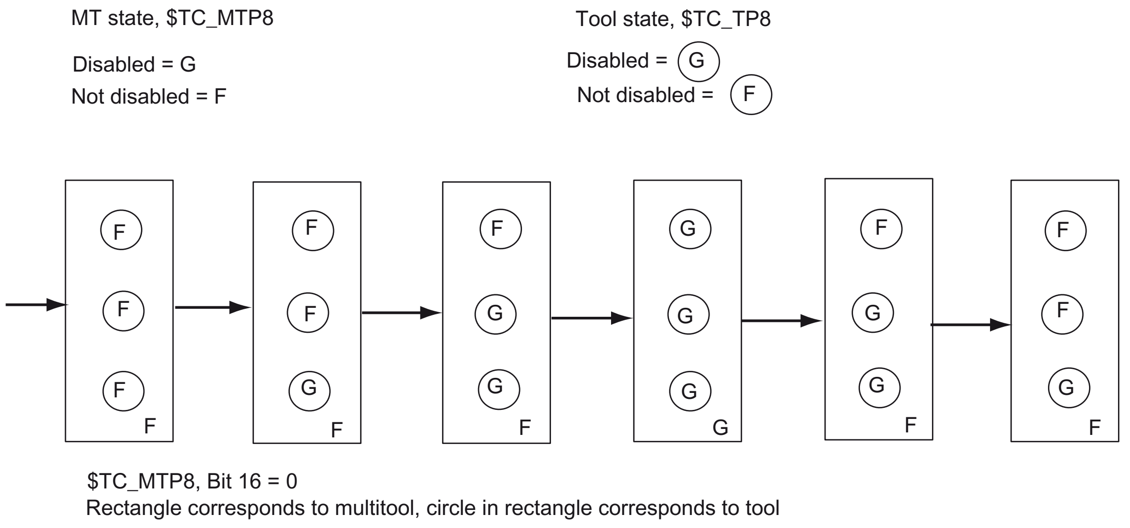


Image 3-7 Automatic state change of the multitool depending on the state of the loaded tools and bit 16 of $TC_MTP8

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.

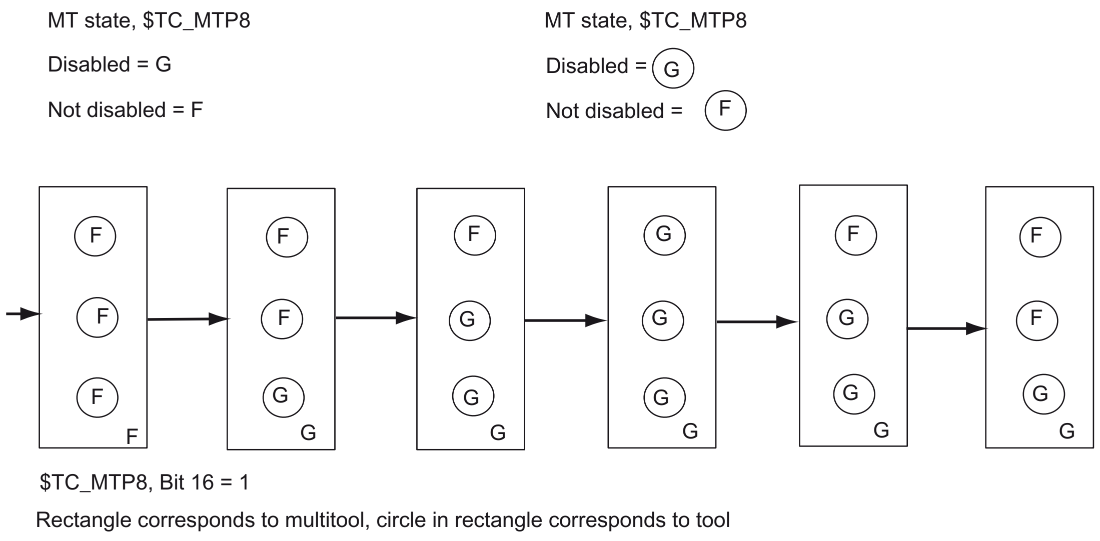


Image 3-8 Automatic state change of the multitool depending on the state of the loaded tools and bit 16 of $TC_MTP8

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 NCK 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 revolvers.

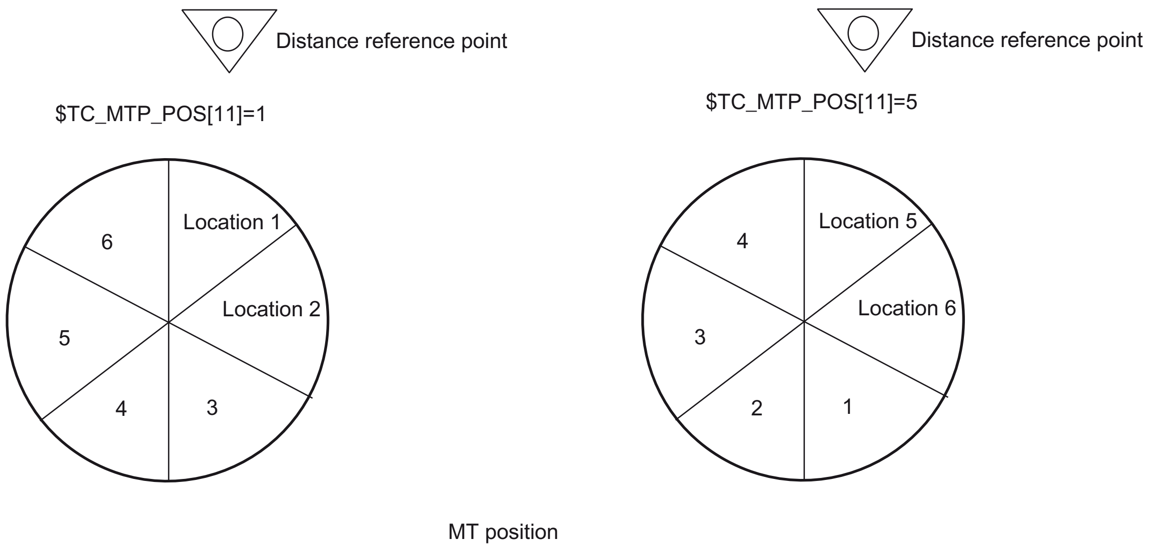


Image 3-9 The location on the machining position determines the MT position

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 final acknowledgement of the MT motion command, the PLC can inform the NCK whether and how the position of the multitool is to be updated.

- The FC6 is parameterized with MultitoolLoc=0. NCK 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 NCK 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 final 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 final acknowledgement is sent to the NCK. The PLC must not change the MT position for the tool selection, tool change commands specified by the NCK 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 master 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 NCK as the tool at the MT location displayed at the MT position output to the PLC. This means that in this example, it depends on the PLC logic for the tool motion on the master master tool holder, which tool is activated at RESET. 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 NCK 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 multitool
                       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"
```

3.2.4 $TC_MTPP - Multitool location data

Overview

The default access right of the following parameters is "key switch 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 numer 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.

NCK identifier	Description	Format	Physical unit	Pre-assigned 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

Example:Transporting a multitool (loading into the magazine or onto the spindle)

"Multitool_2" has been created (T-no. 25), with six locations and distance coding "location".

The multitool is directly loaded onto the spindle, e.g. via the user interface. As a consequence positioning is not involved.

The NCK generates a transport task (CMD1)

```
ID:00000/00000 ------ CMD:00001
NewTool: from M: 09999 P: 00001 to M: 09998 P: 00001 TNo: 00025 TMMVTL-chan: 00000
                   isMultitool: 00001 MTPN: 00006 MT-Dist: 00000/0.00
OldTool: from M: 00000 P: 00000 to M: 00000 P: 00000
```

The multitool is loaded just like an individual tool.

Additional information is contained in the command: Multitool, distance coding "location"

The user interface looks like this:

DB71	DBB0	00000001	Interface 1 active
	DBBn	00000001	Loading
	DBB(n+1)	10000000	Data available in the extended range, i.e. it is a multitool
	DBW(n+20)	0	Magazine no. (source)
	DBW(n+22)	0	Location no. (source)
	DBW(n+24)	9998	Magazine no. (target)
	DBW(n+26)	1	Location no. (target)
DB1071	DBW(n+0)	1	Multitool, distance coding is "location"
	....		

All other data is "0" – there is no additional information from the NCK

There are now two options for acknowledging:

- The multitool position remains undetermined, i.e. the init blocks do not output a command and no offset is selected. This option makes sense if the tool is loaded into the magazine (the multitool position is not evaluated) or if the tool is loaded via the spindle. The multitool must be positioned if a tool in the multitool is now to be brought into the machining position (from a data perspective).
- The PLC specifies a position. This means that the NCK synchronizes to the acknowledged position. An init block can activate the tool that is at this location.

FC6 can be parameterized as follows:

FC6 parameters	Values	Comment
Start		Starts task
TaskIdent	1	DB71 interface
TaskIdentNo	1	No. of the active interface
NewToolMag	9998	Target for the new tool: Magazine DB71.DBW(n+24) –
NewToolLoc	1	Target for the new tool: Location DB71.DBW(n+26)
OldToolMag	0	Source for the old tool: Magazine= "0"" there is no old tool
OldToolLoc	0	Source for the old tool: Location= "0" there is no old tool
Status	1	Operation completed
MultitoolLoc	0/x	In case_1, "0" should be entered here. The multitool position remains undetermined, alternatively "-1". In case_2, the multitool location, which is in the machining position, is entered here.
Ready		Feedback from FC6
Error		Feedback from FC6

Positioning a multitool

It is only possible to position a multitool, if

a) It is located on a tool holder

b) No offset is active. This means that none of the tools in this multitool may be active.

Positioning can be performed via the language command POSMT or the similar PI service (POSMT). Positioning from the PLC via an asynchronous transfer is also possible.

The multitool loaded onto the spindle in the previous example is now positioned to location_5 using a part program.

N10 POSMT(state 5,1); position the multitool to tool holder_1 at position 5

NCK generates a command 1

```
ID:00001/00000 ------ CMD:00001
NewTool: from M: 09998 P: 00001 to M: 09998 P: 00001 TNo: 00025 Spindle: 00001
                    isMultitool: 00001 MTPN: 00006 MT-Dist: 00005/5.00
OldTool: from M: 00000 P: 00000 to M: 00000 P: 00000
```

New tool from spindle to spindle, i.e. it is already located on the spindle. T-No=25, spindle=1
Multitool with distance coding "location", number of locations of the multitool=6, target position=5.
There is no old tool

The user interface looks like this:

DB71	DBB0	00000001	Interface 1 active
	DBBn	00110000	Positioning the multitool / command comes from the part program
	DBB(n+1)	10000000	Data available in the extended range, i.e. it is a multitool
	DBW(n+20)	9998	Magazine no. (source)
	DBW(n+22)	1	Location no. (source)
	DBW(n+24)	9998	Magazine no. (target)
	DBW(n+26)	1	Location no. (target)
DB1071	DBW(n+0)	1	Multitool, distance coding is "location"
	DBW(n+2)	6	Number of locations of the multitool
	DBD(n+4)	5.0	Location distance
	DBW(n+8)	25	T no. of the multitool
	DBW(n+10)	5	Location no. in the multitool
	....		

All other data is "0"

FC6 can be parameterized as follows:

FC6 parameters	Values	Comment
Start		Starts task
TaskIdent	1	DB71 interface
TaskIdentNo	1	No. of the active interface
NewToolMag	9998	Target for the new tool: Magazine DB71.DBW(n+24) –

NewToolLoc	1	Target for the new tool: Location DB71.DBW(n+26)
OldToolMag	0	Source for the old tool: Magazine = "0" there is no old tool
OldToolLoc	0	Source for the old tool: Location= "0" there is no old tool
Status	1	Operation completed
MultitoolLoc	5	Multitool position - DB1071.DBW(n+10)
Ready		Feedback from FC6
Error		Feedback from FC6

$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. After removing a tool from the MT, the states of the remaining tools in the MT determine whether and how the state of the multitool 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, ...

3.2.5 Tool change with a tool from a multitool

Overview

Programming tool change commands remains unchanged. The programming

T= "Miller_150"

M06

D1

selects an appropriate tool from the tool group "Miller_150" and loads this. If, corresponding to the tool search strategy, the tool found is located in a multitool, then this does not play a role regarding activating the offset. The multitool is only involved when the data for the tool change command has been compiled. The NCK determines the distance value (–>MT location number or $TC_MTPPL/$TC_MTPPA) and adds this to the PLC command. For the tool change itself, the multitool (with its individual tools) is moved from the magazine to the tool holder. Using the data "Multitool location distance" that the NCK sends with the tool change data to the PLC, the PLC program must appropriately initiate the machine so that the active tool within the multitool can machine the workpiece.

The state of tools within the multitool in question, which are not involved in an MT motion operation, tool selection, tool change, does not change.

If programmed

T="name"' M06 D1

and the name is that of a multitool, then Alarm 6404 "Tool change not possible" is generated.

Tool change from a multitool to a tool holder - PLC request

Example 1:

A multitool with four locations is in magazine 1, location 10, and has the TNo = 23. Angle distance coding has been selected for the multitool; i.e. $TC_MTP_KD[23] = 3

The setting is $MC_TOOL_CHANGE_MODE=1, i.e. change with M06.

The data of the defined and loaded multitool looks like this:

$TC_MTPN[23]=4	Number of locations
$TC_MTP2[23]="Multitool_0"	Identifiers
$TC_MTP3[23]=2	Number of half locations (right), i.e. the multitool is oversized
$TC_MTP4[23]=2	Number of half locations (left)
$TC_MTP5[23]=2	Number of half locations (top)
$TC_MTP6[23]=2	Number of half locations (bottom)
$TC_MTP7[23]=1	Location type
$TC_MTP8[23]=130	Tool status (enabled, was in use)
$TC_MTP_POS[23]=2	MT position (is not evaluated, if the tool is located in the magazine)
$TC_MTP_KD[23]=3	Distance coding (3 = angle)
$TC_MTP_PROTA[23]=""	Data for protection zone
$TC_MTPP2[23,1]=0	Location_1 - location type
$TC_MTPP4[23,1]=0	Location_1 - location state (occupied)
$TC_MTPP6[23,1]=1	Location_1 - equipped with T-No.1 ("Tool1")
$TC_MTPP7[23,1]=0	Location_1 - adapter No.
$TC_MTPPA[23,1]=0	Distance-angular value
$TC_MTPP2[23,2]=0	Location_2 ...
$TC_MTPP4[23,2]=0	
$TC_MTPP6[23,2]=2	equipped with T-No.2 ("Tool2")
$TC_MTPP7[23,2]=0	
$TC_MTPPA[23,2]=90	
$TC_MTPP2[23,3]=0	Location_3 ...
$TC_MTPP4[23,3]=0	
$TC_MTPP6[23,3]=3	equipped with T-No.3 ("Tool3")
$TC_MTPP7[23,3]=0	
$TC_MTPPA[23,3]=180	
$TC_MTPP2[23,4]=0	Location_4
$TC_MTPP4[23,4]=0	
$TC_MTPP6[23,4]=4	equipped with T-No.4 ("Tool4")
$TC_MTPP7[23,4]=0	
$TC_MTPPA[23,4]=270	

The following is programmed:

T="Tool3" ; generates the PLC command = 2 with data

```
ID:00002/00000 --------- CMD:00002
NewTool: from M: 00001 P: 00010 to M: 09998 P: 00001 TNo: 00023/00003 spindle: 00001
                              isMultitool: 00003 MTPN: 00004 MT–Dist: 00003/180.00
OldTool: from M: 00000 P: 00000 to M: 00000 P: 00000
```

change tool or multitool from location 1/10 to tool holder 9998/1
the T number 23 of the multitool is output,
as well as the T number 3 of the tool selected in the multitool
isMultitool = 3 tool is contained in the multitool; distance has the unit angle

MT dist = 3/180 - with this data the PLC must bring the tool with T-No. = 3 in multitool T-No. = 23 to location=3 and distance180 degrees into the machining position

The PLC user interface looks like this:

DB72	DBB0	00000001	Interface 1 active
	DBB(n+4)	00000100	Prepare change
	DBB(n+5)	10000000	Data available in the extended range (DB 1072)
	...		
	DBW(n+20)	1	Source new tool (magazine)
	DBW(n+22)	10	Source new tool (location)
	DBW(n+24)	0	Target old tool (magazine)
	DBW(n+26)	0	Target old tool (location)
	DBW(n+28)	1	New tool: Location type
	DBW(n+30)	2	New tool: Size left
	...		
	DBW(n+36)	2	New tool: Size bottom
	DBW(n+38)	162	New tool: Status (enabled, was in use, being changed)
	DBW(n+40)	23	New tool: T number

Up until now, data in the usual form. "Carrier tool" is transported in which the preselected tool is loaded. That it now involves a multitool is indicated by setting DB72.DBB(n+1), bit7 "extended data available (DB1072)".

The data of the specifically preselected tool ("Tool3") is now in DB1072.

DB1072	DBW(n+0)	3	Distance coding (angle)
	DBW(n+2)	4	Number of locations in the multitool
	DBD(n+4)	180	Location distance (3rd location, corresponds to 180 degrees)
	DBW(n+8)	23	New tool: Multitool number
	DBW(n+10)	3	New tool: Location number in the multitool → it is mandatory that the PLC acknowledges this position
	DBW(n+12)	0	Old tool: Multitool number
	DBW(n+14)	0	Old tool: Location number in the multitool

DBW(n+16)	1	New tool: Location type
DBW(n+18)	1	New tool: Size left
...		
DBW(n+24)	1	New tool: Size bottom
DBW(n+26)	42	New tool: Status
DBW(n+28)	3	New tool: T number
DBW(n+30)	1	New tool: Tool holder or spindle number
DBW(n+32)	1	New tool: Original magazine (is identical with DB72.DBW(n+44)
DBW(n+34)	10	New tool: Original location (is identical with DB72.DBW(n+46)

The FC6 can be parameterized as follows.

Assumption: the prepared tool or multitool is not moved and remains in the magazine

FC6 parameters	Values	Comment
Start		Starts task
TaskIdent	2	DB72 interface
TaskIdentNo	1	No. of active interface
NewToolMag	1	Source for new tool: Magazine DB72.DBW(n+20) –
NewToolLoc	10	Source for new tool: Location DB72.DBW(n+22)
OldToolMag	0	Target for old tool: Magazine = "0" there is no old tool
OldToolLoc	0	Target for old tool: Location= "0" there is no old tool
Status	1	Exit operation
MultitoolLoc	xx	Multitool position - DB1072.DBW(n+10) The position, in this case (tool remains in the magazine), has no significance. It can be acknowledged with the position from DBW(n+10), but also with "0" or "-1". This also applies to intermediate acknowledgements. The NCK only evaluates the multitool position with the final acknowledgement of the change.
Ready		Feedback from FC6
Error		Feedback from FC6

M06 ; generates the PLC command = 3 with the data

```
ID:00003/00002 ------ CMD:00003
NewTool: from M: 00001 P: 00010 to M: 09998 P: 00001 TNo: 00023/00003 spindle: 00001
                isMultitool: 00003 MTPN: 00004 MT-dist: 00003/180.00
OldTool: from M: 00000 P: 00000 to M: 00000 P: 00000
```

Just as in all previous versions, this means that the user interface is no longer updated. Only DB72.DBB(n+0)bit1 "perform change" is set

The FC6 can be parameterized as follows.
Assumption: The prepared tool or multitool was already transported and here, in the example, only the final acknowledgement is programmed.

FC 6 parameters	Values	Comment
Start		Starts task
TaskIdent	2	DB72 interface

TaskIdentNo	1	No. of active interface
NewToolMag	9998	Target for new tool: Magazine = buffer
NewToolLoc	10	Target for new tool: Location = spindle
OldToolMag	0	Target for old tool: Magazine = "0" there is no old tool
OldToolLoc	0	Target for old tool: Location= "0" there is no old tool
Status	1	Exit operation
MultitoolLoc	3	Multitool position - DB1072.DBW(n+10) must now be forcibly acknowledged with the correct position from DBW(n+10).
Ready		Feedback from FC6
Error		Feedback from FC6

D2 ; activate the offset of the tool with T–No. = 3

A new tool is now programmed, which is located in the same multitool.

T="Tool1" ; generates the PLC command = 2 with data
ID:00004/00004 ------ CMD:00002
NewTool: from M: 09998 P: 00001 to M: 09998 P: 00001 TNo: 00023/00001 spindle: 00001
myM: 00001 myP: 00010 isMultitool: 00003 MTPN: 00004 MT-dist: 00001/0.00
OldTool: from M: 00000 P: 00000 to M: 00000 P: 00000

The tool/multitool is already on the tool holder. NCK detects the situation and sets 0/0—>0/0 to remove the old tool - there is no old tool to transport. The old tool is removed implicitly by positioning the new tool (rigid coupling of the new and old tool in the multitool)

For the user interface this means

- No transport task for the "carrier tool".

DB72	DBB0	00000001	Interface 1 active
	DBB(n+4)	00000100	Prepare change
	DBB(n+5)	10000000	Data in the extended range (DB1072)
	...		
	DBW(n+20)	9998	Source new tool (magazine)
	DBW(n+22)	1	Source new tool (location)
	DBW(n+24)	0	Target old tool (magazine)
	DBW(n+26)	0	Target old tool (location)
	DBW(n+28)	1	New tool: Location type
	DBW(n+30)	2	New tool: Size left
	...		
	DBW(n+36)	2	New tool: Size bottom
	DBW(n+38)	162	New tool: Status (enabled, was in use, being changed
	DBW(n+40)	23	New tool: T number

- The relevant data for the new tool is located in DB1072.

DB1072	DBW(n+0)	3	Distance coding (angle)
	DBW(n+2)	4	Number of locations in the multitool

DBD(n+4)	0	Location distance (1st location, corresponds to 0 degrees)
DBW(n+8)	23	New tool: Multitool number
DBW(n+10)	1	New tool: Location number in the multitool → it is mandatory that the PLC acknowledges this position
DBW(n+12)	23	Old tool: Multitool number
DBW(n+14)	3	Old tool: Location number in the multitool
DBW(n+16)	1	New tool: Location type
DBW(n+18)	1	New tool: Size left
...		
DBW(n+24)	1	New tool: Size bottom
DBW(n+26)	42	New tool: Status
DBW(n+28)	1	New tool: T number
DBW(n+30)	1	New tool: Tool holder or spindle number
DBW(n+32)	1	New tool: Original magazine (is identical with DB72.DBW(n+44)
DBW(n+34)	10	New tool: Original location (is identical with DB72.DBW(n+46)

M06 ; generates the PLC command = 3 with the data

```
ID:00005/00005 ------ CMD:00003
NewTool: from M: 09998 P: 00001 to M: 09998 P: 00001 TNo: 00023/00001 spindle: 00001
myM: 00001 myP: 00010 isMultitool: 00003 MTPN: 00004 MT-dist: 00001/0.00
OldTool: from M: 00000 P: 00000 to M: 00000 P: 00000
```

The FC6 can be parameterized as follows.

FC6 parameters	Values	Comment
Start		Starts task
TaskIdent	2	DB72 interface
TaskIdentNo	1	No. of active interface
NewToolMag	9998	Target for new tool: Magazine = buffer
NewToolLoc	1	Target for new tool: Location = spindle
OldToolMag	0	Target for old tool: Magazine = "0" there is no old tool
OldToolLoc	0	Target for old tool: Location= "0" there is no old tool
Status	1	Operation completed
MultitoolLoc	1	Multitool position - DB1072.DBW(n+10) must now be forcibly acknowledged with the correct position from DBW(n+10)
Ready		Feedback from FC6
Error		Feedback from FC6

D1 ; activate the offset of the tool with T-No. = 17

Tx - Tx programmed: Configuration of the tool selection

$MC_TOOL_MANAGEMENT_MASK, bit 11 makes it possible to adjust the behavior when repeatedly programming Tx – Tx. The setting is valid, just as before for the tool to be selected, however also for the MC itself.

Example:

The following has been set: Tool selection with T, tool change with M code. The following is programmed:

T="Tool5"
T="Tool5"

The second tool preparation command is a "superfluous" command, which as default setting does not result in a command being output to the PLC. If "Tool5" is a tool belonging to a multitool, then with the command, generally

a) a transport task of the MT "from the magazine to the tool holder" is generated, and
b) a positioning command of the MT location with the programmed tool.

If "Tool5" is in an MT, then two basic situations can exist for the MT:

1 the MT is already located on the tool holder

1.1 the programmed tool has already been positioned

1.2 the programmed tool is a different tool than the tool that was positioned

2 the MT is still not located on the tool holder

For case 1.1, the data for a) "MT from tool holder to tool holder"
b) "Position to the already positioned location of the tool "Tool5""

For case 1.2, the data for a) "MT from tool holder to tool holder"

b) "Position to the location of the tool that has still not been positioned "Tool5""

For case 2, the data for a) "MT from magazine to tool holder"

b) "Position to the location of the tool "Tool5""

The general criterion as to whether the second program command is a "superfluous" command is whether the generated command data for the PLC is identical to that of the first command. The second command is then not output to the PLC.

This command output can be forced using the bit value 1 of bit 11 from $MC_TOOL_MANAGEMENT_MASK.

Tool change command canceled before the final acknowledgement

If, within the scope of a tool change, the MT has already been mechanically loaded, but the MT has not been positioned, and the operation is interrupted in this state (fault, RESET, EMERGENCY STOP, power off, power failure...), then before continuing, the PLC must ensure that the previously pending MT positioning is carried out. The PLC must then make sure that the NCK is informed about the MT change (e.g. using an asynchronous transfer with acknowledgement status 4 or 5. In so doing, the NCK synchronizes the MT position with the command data from the PLC). With an asynchronous transfer, the PLC can also communicate the MT position to the NCK.

Note

Init blocks for START and/or RESET

Especially if init blocks for START and/or RESET are generated via MD settings, with the property "activate the tool on the tool holder", and the tool on the tool holder is an MT, then the PLC must ensure that the MT is correctly positioned on the tool holder before the NCK makes the correction selection (i.e. before the RESET command is sent to the NCK, the MT must have been positioned)

T0 - tool deselection and return transport of the multitool into the magazine

T0 deselects the tool and loads the associated MT back into the magazine.

Continuation of the example of the previous section:

T0 ; generates the PLC command = 2 with the data

```
ID:00005/00004 ——— CMD:00002
NewTool: from M: 00000 P: 00000 to M: 00000 P: 00000 TNo: 00000/0000 spindle: 00001
                         isMultitool: 0 MTPN: 0 MT-dist: 0/0.0
OldTool: from M: 09998 P: 00001 to M: 00001 P: 00010
```

This means, as usual, a T0 is output at the PLC user interface. The "tool carrier" must be removed from the tool holder. It does not make any difference as to whether it involves an individual tool or a multitool. DB72.DBB(n+1), bit 7 is not set, as there is no relevant data.

M06 ; generates the PLC command = 3 with the data

```
ID:00006/00005 ------ CMD:00003
NewTool: from M: 00000 P: 00000 to M: 00000 P: 00000 TNo: 00000 spindle: 00001
OldTool: from M: 09998 P: 00001 to M: 00001 P: 00010
```

If a tool from this multitool that has just been exchanged is again selected/loaded, then the multitool is again loaded onto the tool holder.

It is not possible to program a T0 with the meaning to deselect the active tool, and at the same time to keep the MT on the tool holder. T0 M06 D0 means, that after successful execution

- the correction is deselected

- there is no active tool

- there is no tool or MT on the tool holder.

PLC acknowledgement status 2 and 7

The PLC acknowledgement status numbers 2 and 7 refer to the tool within the multitool. This means that the tool is rejected, not the multitool. If the newly selected tool is in another multitool, then the transport task data for the multitool as well as for the newly selected tool contained in it, are newly determined and output to the PLC. The following apply:

Status=2:
The tool cannot be made available The NCK locks this tool and tries to repeat the tool preparation with another tool. This status is permissible only in conjunction with a tool change.

Status=7:
Similar to status 2 – with the property that a tool is not disabled (i.e. the same tool can be selected again).

Tool change with "T=magazine location number" programming ("T=location")

With a multitool, there are three variants of T=location programming:

1. Specification of multitool location and magazine location, e.g. N100 MTL=2 T4
2. With MD20274 $MC_MULTITOOLLOC_DEFAULT
   The magazine location is programmed, e.g. T4
   The tool at the multitool location specified in the machine data is selected.
3. Via the current multitool position
   The magazine location is programmed, e.g. T6
   The tool at the multitool location specified in the multitool parameter $TC_MTP_POS is selected.

In these three options, variant 1 has the highest priority and variant 3 the lowest.

The following examples illustrate this.

Example 1:

$MC_TOOL_CHANGE_MODE=0 applies (tool change only with the T command, i.e. the T programming results in the output of command number 4 to the PLC).

Multitool with distance coding, location number, $TC_MTP_KD[500]=1, is in turret 1, location 4 and has the MT no. = 500; `$TC_MPP6[1, 4]` = 500. The MT position is `$TC_MTP_POS[500]=`2.

The tool at magazine location 1/1 is loaded onto tool holder 9998/1. The MT on magazine location 4 has 3 locations.

The following tools are loaded in the multitool:

"tool_1": `$TC_MTPP6[500, 1]` =11 MT distance `$TC_MTPPA[500, 1]` = 1

"tool_2": `$TC_MTPP6[500, 1]` =22 MT distance `$TC_MTPPA[500, 2]` = 2

– `$TC_MTPP6[500, 1]` =0 MT distance `$TC_MTPPA[500, 3]` = 3

There is no tool at location 3 of the multitool. $MC_MULTITOOLLOC_DEFAULT=0 applies.

Tool selection and tool change are programmed with T. The "T=magazine location number" function is used.

The following command is then programmed:

;MT-Position $TC_MTP_POS[500]=2

T4 ;MT position `$TC_MTP_POS[500]` = 2
;at magazine location 4, in the multitool loaded there, select the tool,
;which is loaded at MT location 2 of the multitool and load the MT - if it is not disabled - and
;activate the tool.
;If the tool at MT location 2 of the multitool is disabled, then a search is made for
;another matching tool with the name "tool_2", which under certain circumstances –
;depending on the tool –;search strategy, tool equipping –**is at some location** ;of another multitool ;or which is contained as individual tool on another
;turret location.

The resulting command to the PLC is as follows:

T4 programming with the result "Multitool 500 is at the programmed magazine location and at the MT location selected (via the MT position), there is tool "tool_2" with the T no.=22" - supplies the following (essential) data to the PLC:

ID:00000/00000 ——— CMD:00004
NewTool: from M: 00001 P: 00004 to M: 09998 P: 00001 TNo: 00500/00022 spindle: 00001 isMultitool: 1 MTPN: 3 MT dist: 2/2.0
OldTool: from M: 09998 P: 00001 to M: 00001 P: 00001

No tool on MT location

For "T=magazine location" and "No tool on the magazine location", it is defined that a regular tool change command to the PLC is output with the special property "T number of the new tool = 0".

Similarly for "T=magazine location" and

"An MT is on the magazine location" and

"No tool is on the selected MT location",

it is defined that a regular tool change command to the PLC is output with the special property

a) MT is in the magazine

T number of the tool that is to be moved on the tool holder is the MT number of the MT that is on the magazine location and the T number of the new tool in the MT = 0.

With the information of the PLC command, the PLC can decide what is to be done in this situation: Either position the MT on the empty MT location or issue an alarm "There is no tool on the MT location".

Example 2:

If **$MC_MULTITOOLLOC_DEFAULT** =**3** applies, this results in example 1 – the multitool position is 2 (`$TC_MTP_POS[500]`=2), **there is no tool on MT location 3** – that the programming of

T4

has the same effect as if T0 were programmed (a new tool is not available, but the request to dispose of the old tool is still issued):

ID:00000/00000 ——— CMD:00004
NewTool: from M: 00001 P: 00004 to M: 09998 P: 00001 TNo: 00500/00000 spindle: 00001

isMultitool: 1 MTPN: 3 MT dist: 3/3.0

OldTool: from M: 09998 P: 00001 to M: 00001 P: 00001

b) MT is on the tool holder

T number of the tool that is to be moved on the tool holder is the MT number of the MT that is already on the tool holder (the transport request "From → to" for the MT is "From tool holder → tool holder") and the
T number of the new tool in the MT = 0.

With the information of the PLC command, the PLC can decide what is to be done in this situation: Either position the MT on the empty MT location or issue an alarm "There is no tool on the MT location".

Example 3:

The preconditions are the same as in example 2 with the condition that the MT from location 4 is already on the tool holder. The programming of

T4

has the same effect as if T0 were programmed (a new tool is not available, but the request for the offset deselection is still issued; the old tool transport data is 0/0 → 0/0) the MT remains on the tool holder:

ID:00000/00000 ——— CMD:00004
NewTool: from M: 09998 P: 00001 to M: 09998 P: 00001 TNo: 00500/00000 spindle: 00001

isMultitool: 1 MTPN: 3 MT dist: 3/3.0

OldTool: from M: 00000 P: 00000 to M: 00000 P: 00000

Tool change with "T=magazine location number MTL=MT location" programming

As an alternative to the programming described in the section above, the MT location can be programmed with MTL=p in the multitool (p=location number).

The following applies for the programming:

- MTL is **non-modal**.
- MTL must be programmed in the NC block **before the T command**.
- The programmed value for MTL must correspond to an existing location number in the multitool to be selected. Otherwise, an alarm is output.

MD $MC_MULTITOOLLOC_DEFAULT can be used to define the MT location number which is used when MTL is not programmed. The MTL programming replaces the default defined with $MC_MULTITOOLLOC_DEFAULT.

Example 4:

If **$MC_MULTITOOLLOC_DEFAULT =3** applies, this results in example 2 of the section above – the multitool position is 2 (**$TC_MTP_POS[500]=2**), there is no tool on MT location 3 – that the programming of

```
MTL=1 T4        ;affects precisely T4 in this block,
                ;which is programmed subsequently in the block.
```

The value of $MC_MULTITOOLLOC_DEFAULT=3 and the current MT position 2 are ignored as MTL is explicitly programmed. This selects "tool_1" with the T number 11 and the command to the PLC is:

ID:00000/00000 ——— CMD:00004

NewTool: from M: 00001 P: 00004 to M: 09998 P: 00001 TNo: 00500/00011 spindle: 00001

isMultitool: 1 MTPN: 3 MT-dist: 1/1.0

OldTool: from M: 09998 P: 00001 to M: 00001 P: 00001

Example 5:

If **$MC_MULTITOOLLOC_DEFAULT =3** applies, this results in example 2 of the section above – the multitool position is 2 ($TC_MTP_POS[500]=2), there is no tool on MT location 3 – that the programming of

```
MTL=1           ;does not affect following block T4 because this is non-modal
...
T4              ;the setting of the MD takes effect
```

The programmed value MTL=1 remains ineffective and the value of $MC_MULTITOOLLOC_DEFAULT=3 takes effect. There is no tool on MT location 3 and therefore the command to the PLC is:

ID:00000/00000 ——— CMD:00004

NewTool: from M: 00001 P: 00004 to M: 09998 P: 00001 TNo: 00500/00000 spindle: 00001
isMultitool: 1 MTPN: 3 MT-dist: 3/3.0

OldTool: from M: 09998 P: 00001 to M: 00001 P: 00001

Example 6:

If **$MC_MULTITOOLLOC_DEFAULT =0** applies, this results in example 2 of the section above – the multitool position is 2 ($TC_MTP_POS[500]=2), there is no tool on MT location 3 – that the programming of

```
MTL=1           ;does not affect following block T4 because this is non-modal
...
T4              ;the setting of the MD takes effect - that is now MT position=2
```

The programmed value MTL=1 remains ineffective and the value of $MC_MULTITOOLLOC_DEFAULT=0 takes effect, i.e. the value of the MT position=2. The tool with Tno.=22 is on MT location 2 and therefore the command to the PLC is:

ID:00000/00000 ——— CMD:00004

NewTool: from M: 00001 P: 00004 to M: 09998 P: 00001 TNo: 00500/00022 spindle: 00001 isMultitool: 1 MTPN: 3 MT-dist: 2/2.0

OldTool: from M: 09998 P: 00001 to M: 00001 P: 00001

$C_MTL - transfer of the MTL_value to the replacement cycle

The programmed value for MTL can be read in an existing replacement cycle for T with `$C_MTL`.

$C_MTL_PROG - if MTL has been programmed

If MTL has been programmed in the NC block, then this can be determined in an existing replacement cycle for T with `$C_MTL_PROG`. `$C_MTL_PROG==1` means that MTL has been programmed. The programmed value can be queried with `$C_MTL`.

Example:

The multitool function and the function T=location have been configured.

```
N10 MTL=3 T=8
```

Then the replacement cycle contains T code of the type for the command:

```
N140 IF $C_T_PROG == 1
N142   IF $C_MTL_PROG == 1
N144     MTL=$C_MTL T[$C_TE]=$C_T
N146   ELSE
N148     T[$C_TE]=$C_T
N150   ENDIF
N152 ENDIF
```

Load/move multitool on tool holder and init block

If a multitool is loaded onto a tool holder other than through a tool change (loading operation, movement command) and for the init block handling (if it is appropriately configured), the active tool must be subsequently determined, then the following applies:

- If necessary: After loading/moving, set the MT position so that the requested tool can be activated – or so that the MT position matches the mechanical MT position
- Always issue a tool change command to the PLC which contains the MT position of the tool to be activated.
- Take the MT which is located on the master tool holder.

- Take the tool which is at the MT location with MT location number = "MT position" (= $TC_MTP_POS), and select it.
- Activate the D offset of this tool with the lowest D number.
  Irrespective of the value of bit 12 (bit value==1 corresponds to ’H1000’) of the MD $MC_TOOL_MANAGEMENT_MASK, a command is output to the PLC if the activated tool is not on the MT location which was output last for this tool holder or was contained in the acknowledged tool change command.

Note

The MT position of the multitool on the tool holder must be synchronized to the mechanical position of the multitool on the machine. If the MT position when loading does not already match the MT loading position, then under certain circumstances the PLC can initiate this synchronization with an asynchronous transfer before the RESET command to the NCK (that generates the init block).

Example

The init block generation is activated ($MC_TOOL_MANAGEMENT_MASK, bit 14 (bit value==1 corresponds to ’H4000’)).

The part program is exited. The MT remains on the tool holder. The last tool change in the part program was performed with regard to MT position = 3. The PLC acknowledged the MT tool change command with MT position = 3.

The init block generated with the operator panel RESET determines the MT on the tool holder, the tool on the MT location with the number of the MT position. The compensation of the smallest D number is activated.

Positioning is now performed on a different location with the OPI PI _N_POSMT_. Whereby, a tool command with the number = 1 (move/position) is output to the PLC and acknowledged accordingly. The MT now has a different position than at the time of the previous end of the program. With the operator panel reset, the tool on this MT location is determined and the compensation activated accordingly. There is no tool change command output to the PLC.

Only the data of the MT position is now changed (e.g. by a write command with $TC_MTP_POS). Machine and NCK data are now asynchronous. With the operator panel reset, the tool on this MT location is determined and the compensation activated accordingly. A tool change command is output to the PLC irrespective of the setting of bit 12 of $MC_TOOL_MANAGEMENT_MASK. After correct positioning, the machine and the NCK data are synchronous again.

Not possible to move the tool from or to the MT location

A tool cannot be transported to or from a multitool location. The PI service as well as the language command or asynchronous transfer are rejected with an error.

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 (complete entity). Movement of tools in the multitool have not been realized.

Tool change - tool search

The tool search strategies are described in Section "Prepare a tool change (Page 72)". They are essentially also applied to the multitool.

NCK identifier	Description	Format	Preassignment
$TC_MAMP" $TC_MAP10	Type of search strategy Bit 0 - 7 tool search strategy	INT	0

Regarding tools in the multitool, the individual tool search strategies behave as follows:

Bit 0 = 1: Select the "active" tool.

If there is more than one suitable tool in the multitool with the "active" state, then the first checked tool of these tools is selected.

Bit 1 = 1: Select the tool that can be reached over the shortest distance.

Within the multitool there is no "shortest distance" search. With shortest distance, the "shortest distance within the magazine" is always meant. It does not make any difference whether a tool or a multitool is at the particular magazine location.

Bit 2 = 1: Select the "active" tool. If there is no active tool available, then select the replacement tool with the lowest number contained in $TC_TP10.

If there is more than one suitable tool in the multitool with the "active" state, then the first checked tool of these tools is selected.

Bit 3 = 1: Lowest actual value (residual value) of the monitoring functions.

If there is more than one suitable tool with "lowest actual value" in the multitool, then the first checked tool of these tools is selected.

Bit 4 = 1: Highest actual value (residual value) of the monitoring functions.

If there is more than one suitable tool with "highest actual value" in the multitool, then the first checked tool of these tools is selected.

Note

Further, the tool search is tool-oriented, i.e. the tool of a tool group is checked as to whether it can be used corresponding to the actual configuration and search strategy.

The MT with suitable tool is already on the tool holder.

If an MT with a tool suitable for the programmed tool change command is already on the programmed tool holder, then – irrespective of the defined tool search strategy – this tool is selected or loaded.

Example:
The MT with number = 22 is on the tool holder / spindle with number = 1. The tool with T no. = 7 is active and is disabled. This tool is loaded to MT location = 1. A tool with the same name "Tool_1" and the T no. = 8 is on MT location = 2. The tool change command with the tool name of the tool that is disabled now selects the tool with T no. = 8:

Extract from NC program:

```
...
N600 T="Tool_1" M06 ; T no. = 7 is loaded
... ; tool is disabled (e.g. because of tool monitoring)
N700 T="Tool_1" M06 ; T no. = 8 is loaded
```

PLC data output (from the diagnostics file):

```
ID:00005/00004 N:N600 CMD:00005
NewTool: from M: 09998 P: 00001 to M: 09998 P: 00001 TNo: 00022/00007 spindle: 00001
myM: 00002 myP: 00005
                        isMultitool: 00003 MTPN: 00003 MT-dist: 00001/28.10
OldTool: from M: 09998 P: 00001 to M: 09998 P: 00001

ID:00005/00005 ——— ACK:00005 St: 00001
NewTool: from M: 09998 P: 00001 to M: 09998 P: 00001 MTPOS: 00001
OldTool: from M: 09998 P: 00001 to M: 09998 P: 00001

ID:00006/00005 N:N700 CMD:00005
NewTool: from M: 09998 P: 00001 to M: 09998 P: 00001 TNo: 00022/00008 spindle: 00001
myM: 00002 myP: 00005
                        isMultitool: 00003 MTPN: 00003 MT-dist: 00002/30.10
OldTool: from M: 09998 P: 00001 to M: 09998 P: 00001

ID:00006/00006 ——— ACK:00005 St: 00001
NewTool: from M: 09998 P: 00001 to M: 09998 P: 00001 MTPOS: 00002
OldTool: from M: 09998 P: 00001 to M: 09998 P: 00001
```

The MT itself remains at its location (from–>to addresses are 9998/1 –> 9998/1. Only a new MT position must be assumed (MTPos value 1 –> 2)

Wear group and multitool

All tools within a multitool belong to the wear group of the multitool itself. The locations within a multitool have the "wear group" property.

Manual tool and multitool

The manual tool function is also available for multitools. As far as this function is concerned, the MT itself is the manual tool and not the programmed tool. If a tool is programmed which is in an MT, but this MT is not loaded, then the MT is treated just like a manual tool.

Example:

The name of the multitool is "MT1", the name of the tool - programmed for tool holder 1 - is "Tool4" with duplo number 5.

The following is programmed
T="Tool4" M06 D1.
This is then followed by the request to load with alarm 17212 "Load manual tool "Tool4", duplo no. 5 to spindle / tool holder 1".

Manual tools (adding tools during machining) (Page 94)

T0 M06 D0
then correspondingly generates alarm 17214 "Unload manual tool "Tool4" from spindle /tool holder 1"

and correspondingly requests that the manual tool (that is an MT or is located in an MT) is again removed from the tool holder.

Suppressing alarms 17212, 17214, 17215, 17216

The manual tool change request alarms can be suppressed using bit 16 in MD $MN_SUPPRESS_ALARM_MASK_2. Then, using the tool change command data, the PLC must correctly load and remove the manual tool.

Empty location search strategy - "replace old tool by new tool" and multitool

Just like tools, for an empty location search, multitools can use the search strategy - "replace old tool by new tool" (1:1 exchange). The MT has the necessary defined data required for this purpose - such as location type, tool size.

3.3 Tool change box-type, chain, circular magazines

3.3.1 Prepare a tool change

Fundamentals

The tool change type is set using machine data 22550 $MC_TOOL_CHANGE_MODE.

$MC_TOOL_CHANGE_MODE = 0

No separate T preparation, change with the T command.

$MC_TOOL_CHANGE_MODE = 1

Separate T preparation, the change is performed with M06 or the M function, which is set in machine data $MC_TOOL_CHANGE_M_CODE.

This setting setting is independent of the magazine type. This means that also for a turret, T and M06 can be programmed just the same as for a chain magazine only with a T command.

$MC_TOOL_CHANGE_MODE=0

```
T="Tool identifier"   ;Tool preparation and tool change with an NC command (= within
                      an NC block)
                      ;NCK sends a command to the PLC
```

If an error is detected during tool preparation, then machining is stopped when the block T = identifier is read-in.

After correction and NC Start, the block with T = identifier is interpreted again and program processing is continued.

$MC_TOOL_CHANGE_MODE=1

- Within an NC block

```
T="Tool identifier" M06      ;Tool preparation and tool change from this programming
                             results in a command to the PLC
```

Program tool preparation and tool change in one block (**T**= "Tool identifier" **M06**), corresponds to setting TOOL_CHANGE_MODE = 0.

- Distributed over two NC blocks

```
T="Tool identifier"   ;Tool preparation
                      ;NCK sends a command to the PLC
M06                   ;Tool change (the number of the M code can be set)
                      ;NCK sends a command to the PLC
```

Tool preparation and tool change are typically programmed in different blocks. Two commands are transferred to the PLC.

An alarm is triggered if an error occurs in T="tool identifier". If the MD22562 TOOL_CHANGE_ERROR_MODE is set accordingly, the alarm is delayed until the associated tool change command M06 is interpreted in the program sequence. Only then is the alarm output. The operator can make corrections in this block.

Note

Offset selection or deselection is automatically made with the tool change command. In so doing, machine data MD20270 CUTTING_EDGE_DEFAULT is evaluated.

Empty spindle

Program commands T0 and M06 remove the tool from the spindle and return it to the magazine. The spindle is then empty.

Possible problems when programming T/M06

$MC_TOOL_CHANGE_MODE=0; tool change with T address

The part program is executed through to the block T = "identifier". The following problems can occur and are handled as follows:

- The tool data record is in the NCK but not assigned to a magazine location. The tool must be reloaded mechanically, if necessary, e.g. directly onto the spindle. The assignment of the tool to the magazine location/the spindle takes place, e.g. with the function "Overstore"; $TC_MPP6[m,p] = T no., or by the HMI operation "Load (onto spindle)".
- The tool data record is not in the NCK: Set up data record in the NCK, e.g. by HMI operation.
- Programming error in part program: Correct NC block with error in the part program.
- Alarm 22067:
  The desired tool change is not possible. The specified tool group does not contain a "ready to use" replacement tool which could be loaded. The tool monitoring function may have set all potentially suitable tools to the "disabled" status.

The Start pushbutton is pressed once the operation has been completed. The NC block T="tool identifier" is interpreted again and program processing is continued provided operator intervention was correct. If not, the alarm will be generated again.

3.3.2 General tool change sequence

Overview

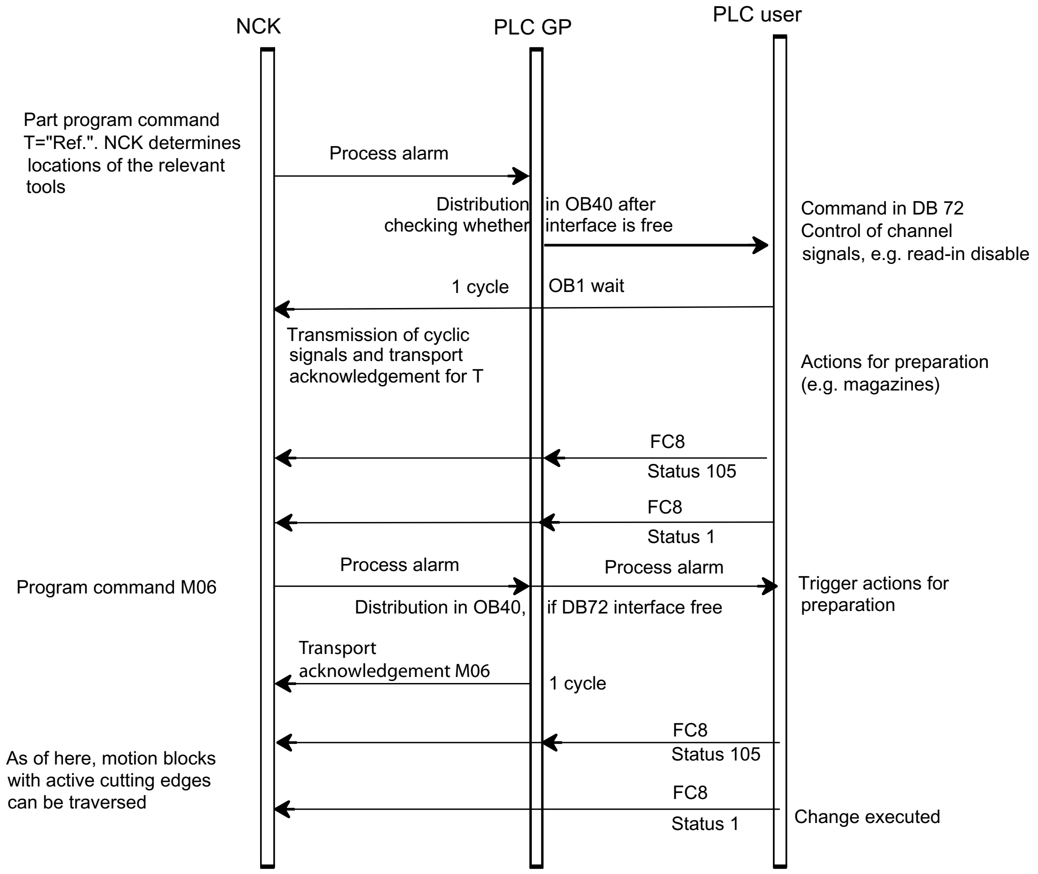

NCK
PLC GP
PLC user
Part program command T="Ref.". NCK determines locations of the relevant tools
Process alarm
Distribution in OB40 after checking whether interface is free
Command in DB 72 Control of channel signals, e.g. read-in disable
1 cycle
OB1 wait
Transmission of cyclic signals and transport acknowledgement for T
Actions for preparation (e.g. magazines)
FC8
Status 105
FC8
Status 1
Process alarm
Process alarm
Program command M06
Trigger actions for preparation
Distribution in OB40, if DB72 interface free
Transport acknowledgement M06
1 cycle
FC8
Status 105
As of here, motion blocks with active cutting edges can be traversed
FC8
Status 1
Change executed

Image 3-10 Preparing and changing a tool

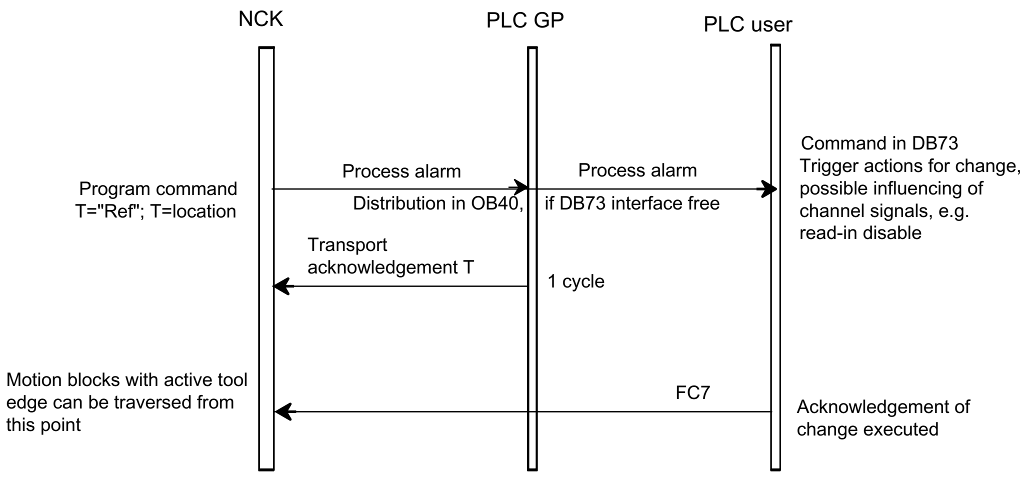

NCK
PLC GP
PLC user
Command in DB73 Trigger actions for change, possible influencing of channel signals, e.g. read-in disable
Process alarm
Process alarm
Program command T="Ref"; T=location
Distribution in OB40, if DB73 interface free
Transport acknowledgement T
1 cycle
Motion blocks with active tool edge can be traversed from this point
FC7
Acknowledgement of change executed

Image 3-11 Tool changing with T command

The prompt for changing the tool is issued by the part program via T command or M command.

1. The tool-management function in NCK searches in accordance with the tool search strategy for a tool suitable for use (preparation) and, at the same time, searches for an empty location for the tool to be exchanged.
2. The determined data is provided in DB72/73. The user program must react by making a new tool available.
3. If machine data MD22550 $MC_TOOL_CHANGE_MODE is set to 1, with the "M06 command" in the part program, the PLC executes the tool change and signals the end of the change operation.
   If the machine data is set to 0, then by programming T, the tool is loaded. In both cases, the PLC has the option of applying its own tool change strategy. It can choose its own empty location for storing the old tool.

Note

The tool change is an NCK-internal operation that is executed as an interaction with the PLC. The HMI only has the task of displaying data and facilitating data input.

Spindle and tool holder

Tool management can also be used for machines that have no spindle (e.g. punch presses or turrets). In this case the term "spindle" is replaced by "tool holder";

This is defined in MD20124 $MC_TOOL_MANAGEMENT_TOOLHOLDER. If MD > 0, then the spindle numbers $TC_MPP5 are interpreted as tool holder numbers.

Fixed location coding

If fixed location coding is selected for a tool, the tool will always be returned to the same location when it is replaced.

Variable location coding

Tools defined with variable location coding can be returned to any location for the appropriate tool size and location type in the magazine.

Automatic tool return to real magazine

1. An automatic tool return is initiated by the TM only if the tool is transported via several stations (status 105) after a T preparation command from the PLC and the T preparation command is finally acknowledged positively with status 1.
   The return of a preselected tool from the buffer can be suppressed by setting MD20310: TOOL_MANAGEMENT_MASK, bit 15 = 1.
2. If a tool change is interrupted because the control is switched off but the tool is already located in a buffer location (gripper), the next tool change must either return the tool in the buffer to the spindle or to the real magazine.
3. If several tools are located in the buffer the spindle tool is considered first. If there is no tool on the spindle, the order for return is in accordance with system variable $TC_MLSR.

Example for the time sequence of a tool change

The following example shows a typical cut-to-cut sequence of operations for a tool change with a tool changer and a fixed absolute tool change point on a milling machine.

Machining program

N970 G0 X = Y = Z	;Retraction from the contour
N980 T1	;Tool preselection
N990 T_CHANGE	;Subprogram call without parameters
N1000 G90 G0 X= Y= Z= M3 S1000	;Continue machining

Subprogram for the tool change

PROC W_WECHSEL	
N10 SPOSA= S0	;Spindle positioning
N20 G75 FP=2 X1=0 Y1=0 Z1=0;	;Approach tool change point
N30 M06	;Change tool
N40 M17	

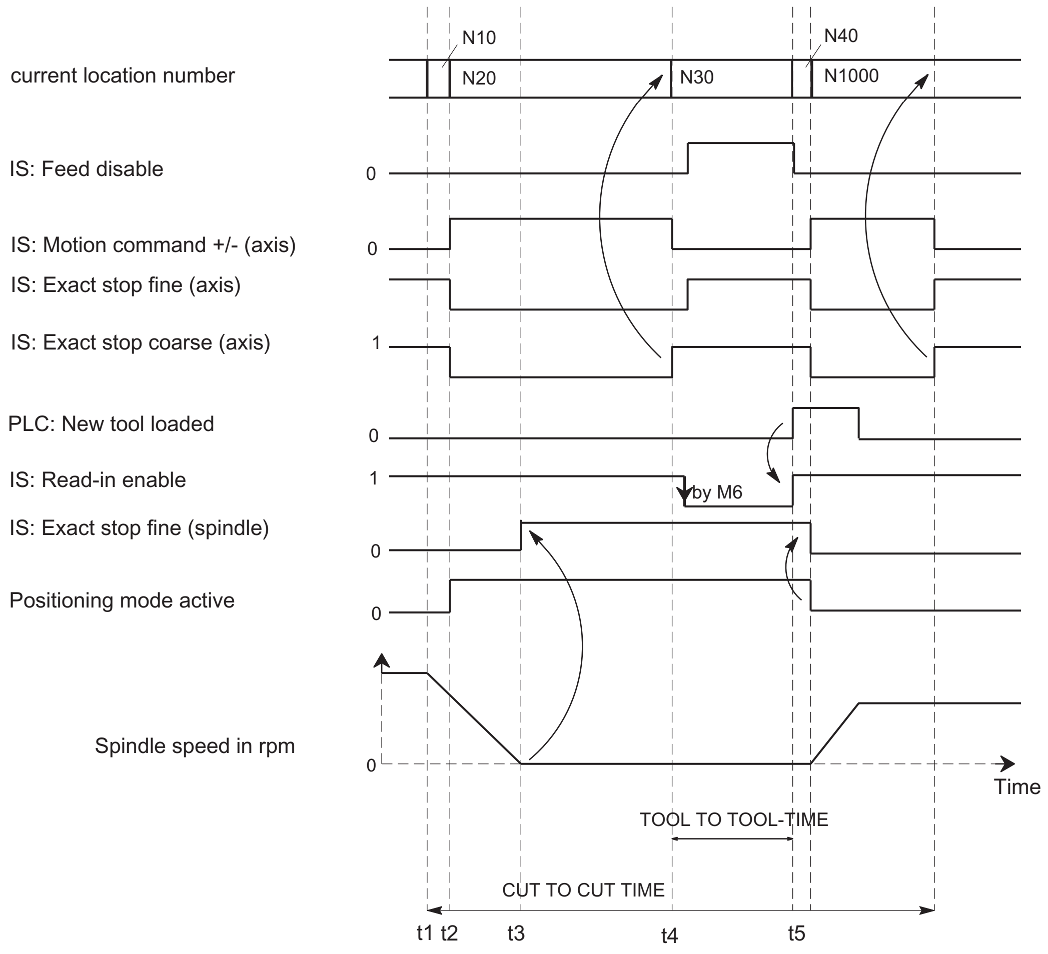


Image 3-12 Chronological sequence of tool change

t1 axes stationary, spindle rotating, start of the tool change cycle in N10

t2 Traverse axis with G75 in N20 to tool change point

t3 Spindle reaches programmed position from block N10

t4 Axes reach exact stop coarse from N20; N30 now starts:

M06 removes the previous tool from the spindle and loads and clamps the new tool.

t5 Tool changer swivels back to original position.

Then, in N1000 of the calling main program,

- the new tool offset can be selected,
- the axes can be returned to the contour, or
- the spindle can be accelerated.

3.3.3 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 rate (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.

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:

References

- SINUMERIK 840D sl: Commissioning Manual, Base Software and Operating Software; Section "SINUMERIK Operate (IM9)" → "Tool management" → "Configuring the user interface" → "Configuring tool lists" (→ "Identifiers of the tool parameters")
- SINUMERIK 828D: Commissioning Manual; Section "SINUMERIK Operate" → "Tool management" → "Configuring the user interface" → "Configuring tool lists" (→ "Identifiers of the tool parameters")

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 dynamics 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 an extension has not been programmed, then the tool change - as well as the programmed speed - refers to the actual master 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 lathes), 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.]. It is only possible to change this parameter if there is no tool at this buffer magazine location.

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

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 actual master tool holder and the programmed speed on the actual master spindle.

Activation of the limiting when the tool is changed

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 change has taken place on the spindle or the tool holder, the NC/PLC interface signal "Tool with dynamic response limit active" (DB31, ...DBX85.0) is set.

Note

The new NC/PLC interface signal "Tool with dynamic response limit active" (DB31, ...DBX85.0) displays 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, "Miller" tool group is loaded into the spindle (new tool, NWZ).

Step No.	NC program	NC → PLC	PLC → NC
1	N100 T1="Miller	---	---
2	---	CMD=2 (tool preparation) NWZ T no=8: Mag./mag.loc.1→ Sp1 AWZ T no=6: Sp1 → Mag./mag.loc. 2	---
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

Step No.	NC program	NC → PLC	PLC → NC
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

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 channel is interchanged, the spindle keeps the dynamic limits.

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_SMAXVELO_INFO[<n>] == 12, where n = spindle number

Maximum values

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

- Speed: $AC_SMAXVELO[<n>]
- Rotary acceleration: $AC_SMAXACCEL_IDX[<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:

DB31, ... DBX85.0 == 1 (tool with dynamic response limitation)

Commissioning

Preconditions

- The "Tool monitoring for maximum speed/acceleration" option must be set.
- The tool is loaded into the spindle **after** setting the speed/acceleration limitation.

Note

A change of the speed limitation in TC_TP_MAX_VELO[<T number>] is **not** effective if the tool is already in the spindle / tool holder.

- 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.

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

Program example

Program code	Comment
`N10 $TC_TP[10]="Miller"`	`;T10 = "Miller"`
`N20 $TC_TP_MAX_VELO[10]=3500`	`;Speed limitation from T10: 3500 rpm`
`...`	
`N100 SETMS(1)`	`;Spindle 1 = master spindle`
`N110 M5`	`;Spindle stop`
`N120 T="Miller" M6`	`;Tool with T10 is loaded into spindle 1 □`
	`;DB31, ... .DBX85.0 = 1`
`N125 $AC_SMAXVELO[1]=3000`	`;Spindle speed limitation: 3000 rpm`
`N130 S5000 M3`	`;M3: 5000 rpm but limited to 3000 rpm`

3.3.4 Selecting a tool and a cutting edge

Precondition

Note

The T number and the M function are no longer transferred to the PLC as an auxiliary function if tool management is activated.

Numbers are also valid tool names, e.g. "3" instead of T = "3" can be programmed more simply than T3.

There must be a tool with the T number as the identifier available when working with the T number.

Example: If you want to call a tool using T3, the tool must have the name "3". A tool CANNOT be called with the internal T number managed by NCK only.

Select/deselect tool offset for Reset

The following machine data can be used to control the behavior for RESET:

- MD20310 TOOL_MANAGEMENT_MASK bit 14
- MD20122 TOOL_RESET_NAME
- MD20110 RESET_MODE_MASK
- MD20130 CUTTING_EDGE_RESET_VALUE
- MD20132 SUMCORR_RESET_VALUE

You can determine whether to:

- deselect the active tool
- keep the active tool selected or
- set a particular tool (according to MD20122 $MC_TOOL_RESET_NAME).

If, in terms of its data, a new tool is selected that is not yet on the master spindle or the master tool holder (or main spindle, main tool holder), then a tool change is executed for a RESET or the end of the program. With this type of tool change (in a similar manner to block searches), the PLC is not capable of influencing the selection of the tool.

Select a tool at start of program

The machine data

- MD20310: TOOL_MANAGEMENT_MASK bit 14
- MD20122 TOOL_RESET_NAME
- **MD20112** START_MODE_MASK
- MD20130 CUTTING_EDGE_RESET_VALUE

can be set to define whether:

- the tool on the main spindle or the main tool holder is selected again or remains selected
- or a particular tool is selected (according to MD20122 $MC_TOOL_RESET_NAME)

If a new tool is selected which in the data is not yet specified as being on the spindle, a tool change is performed when the program is started. For this type of tool change the PLC cannot influence the selection of the tool, just as for block search.

Tool rejection by the PLC

For a block search, selection on reset or program start as well as the setting "PrepSelect", then the tool is selected during preprocessing. In this case, the PLC is no longer allowed to reject the tool.

Note

If bit 4 of the machine data MD20310 $MC_TOOL_MANAGEMENT_MASK is set, then the PLC usually has the possibility to again request preparation for a tool change, yet this time with different parameters, i.e. to reject the tool.

Communication between PLC and tool management

The communication between PLC and NCK during a tool change is via the VDI interface. Tool change is triggered by the tool management in the NCK. The tool management outputs commands to the PLC which acknowledges them either positively or negatively depending on the situation. The PLC also has the possibility of independently (non-synchronously) transferring commands to the NCK, e.g. communicating the magazine position, transport of a tool, etc.

3.3.5 Predecoding (preprocessing) and block execution (main run)

Sequence

The cutting edge geometry cannot be calculated until the tool management knows the tool that is actually to be used. Only the identifier is stated in the part program for tool change. Generally, the tool with the status "active" is then used. But if this is disabled, then one of the other twin tools, i.e. the replacement tool, is selected instead. The precoding delays selection of the new offsets until it is clear which tool is to be used. Only then can precalculation of the blocks be restarted.

Tool change must have been completed before the path can be traversed with the tool offset of the new tool.

The block is split if the preprocessing run detects that a cutting edge of a new tool has been selected for the first time and tool preparation has been initiated, but not yet completed.

The following synchronization points exist between predecoding and block execution:

Example:

Programmed NC block:
```
N1 D1 M06 Txx X100 Y100
```

Sequential blocks:
```
N1 Txx M06 end of block
N2 D1 X100 Y100
```

1.	Interpreter detects an offset selection (D number)
2.	It determines that a tool change was previously programmed which has not yet resulted in selection of a tool.
3.	Interpreter carries out "block splitting".
4.	`Output of block N1:` Block 1 receives a request from the execution blocks to output their collective blocks, and also if programmed, M06, T numbers, ...
5.	`Output of block N2:` Block 2 receives the rest, in particular all travel information and any D numbers if programmed.
6.	Tool management stops execution of the block during preprocessing until it is clear which tool is to be used.
7.	After receiving the tool preparation acknowledgement, execution of block 2 is continued, or first the new T number is entered in the block and is used to calculate the contour again.

Tool change at the main spindle or master tool holder

The main run waits in synchronism with tool change block for transport acknowledgement.

1. Main run waits in synchronism with tool change block for end of acknowledgement (if bit 5 or bit 6 of MD20310 $TC_TOOL_MANAGEMENT_MASK is set), or
2. After a tool change in the main run, the NCK automatically synchronizes with the end of the tool change in the block in which a cutting edge of the new tool was selected for the first time.

Note

The transport acknowledgement is an internal acknowledgement of an NCK command. It indicates to the NCK that the output command was accepted. When a new command is output to the PLC, the NCK waits for the acknowledgement of the previous command.

Tool already in spindle

If the programmed tool is already in the spindle, by default no task is sent to the PLC. (The response can be defined by the MD setting.)

Tool change at the secondary spindle or secondary tool holder

1. Main run does not wait. There is no synchronization.
2. Main run waits in synchronism with tool change block for transport acknowledgement.
3. Main run waits in synchronism with tool change block for end of acknowledgement.

Tool change preparation in a main spindle

1. In the main run, the tool management decides which tool is to be used. Until then, the preprocessor waits at the point in the program at which the compensation values of the new tool are to be considered for the first time.
2. The PLC can also decide which tool is to be used. In this case, the PLC can reject the recommended tool with a negative acknowledgement. If rejected by the PLC, the NCK selects a new, different tool (only if MD20300 $MC_TOOL_MANAGEMENT_MASK, bit 4 = 1, see also FC8/FC6-description).
3. Even if the function "GETSELT(...,x)" is programmed, the preprocessor again has to wait until a decision has been made as to which tool is to be used.

Prepare to change tool in a secondary spindle

1. The main run does not wait. There is no synchronization

Note

During a synchronization operation where the new offset is used or allowed for by the preprocessor, "block splitting" must be performed. This ensures that a preprogrammed tool change T or M06 is actually performed and not collected in run blocks.

Unlike the STOPRE command, the preprocessor does not necessarily wait until all blocks have been processed, but only waits if tool selection has not taken place by the appropriate time. The appropriate time is when programming new offsets after tool change or when programming GETSELT.

3.3.6 Traverse axes while tool is being changed

After the tool change command M06 the axes can continue travel without having to wait for the tool change acknowledgement and, e.g., execute traversing blocks without tool offset. Travel only stops in a block with an offset selected (D No.) until the PLC signals that a tool change has been completed.

Precondition: MD20270 $MC_CUTTING_EDGE_DEFAULT= 0 or = -2

Example: Traversing blocks between tool change and cutting edge selection

```
N10 T="Drill18"            ;Tool change preparation
N15 M06                    ;Tool change
N20 D0                     ;Offset deselection
N25 G00 X100 Z200          ;Traverse machine axes
N30 Y150 M79               ;Traverse machine axes
N35 G01 D1 X10             ;Activate the tool offset.
                           ;Check whether the tool has been changed. Preprocessing
                           stop until tool change preparations have been completed.
                           ;Main run waits until tool change has been acknowledged
                           from PLC.
```

The preprocessing stop is maintained until the tool change preparations have been completed. The main run waits at N35 (D1) until the tool change has been executed and acknowledged.

3.3.7 Tool change in the spindle for chain and box-type magazines

Spindle/buffer DB72

DB72 is responsible for changing tools in the spindle. This data block also prepares the tool change. This data block has an interface for every spindle.

User data is available at each interface (sequence in accordance with the spindle number) as for the loading and unloading points. Data are additionally available for the new tool to be loaded at change. This data includes location type, sizes, tool status and the T number internally assigned in the NC.

The buffer address of the spindle is contained in DB72.DBW(n+16) and DBW(n+18) as the target for the new tool. This position is communicated as the target position of the new tool in parameters "NewToolMag" and "NewToolLoc" when the tool change has been successfully completed. The target position of the old tool (DB72.DBW(n+24) and DBW(n+26)) is transferred to FC8/FC6 in parameters "OldToolMag, "OldToolLoc" together with parameter "Status = 1" after the change tool command has been executed.

Description of tool change operation in spindle

The tool in location 1, magazine 1 is to be loaded to the spindle (magazine no. 9998, location 1) and the tool in the spindle is to be returned to magazine 1 location 8.

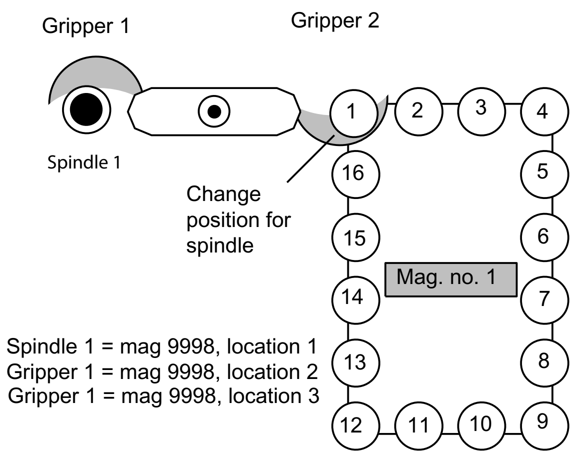

Gripper 1
Gripper 2
Spindle 1
Change position for spindle
Mag. no. 1
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Spindle 1 = mag 9998, location 1
Gripper 1 = mag 9998, location 2
Gripper 1 = mag 9998, location 3

Image 3-13 Tool change in the spindle

The tool change in the spindle is split into two steps (for TOOL_CHANGE_MODE=1):

1. Prepare change:
   Search for new tool and move to the change position
2. Execute change:
   New tool into the spindle and old tool into the magazine at the correct location

1. Prepare change

Bit 2 is set in DB72.DBB n+0. In preparing for the change, the current positions of the tools are communicated to FC8/FC6 in the associated parameters once the preparation step has been completed. In this case, "Status" = 1 is parameterized for FC8. This means that the "old tool" is still in the spindle and that the "new tool" is either still in the source magazine at the same location or has been placed in a buffer.

The following information is passed to FC8/FC6:

- The new tool is in the change position, but is still located in magazine (NewToolMag = 1 and NewToolLoc = 1).
- The old tool is still in the spindle (OldToolMag = 9998 and OldToolLoc = 1).

FC 8 parameters	Values	Comment
Start		Starts task
TaskIdent	2	DB72 interface
TaskIdentNo	1	No. of active interface
NewToolMag	1	(n+20) Mag No. new tool
NewToolLoc	1	(n+22) Loc. No. new tool
OldToolMag	9998	(n+16) Mag. no. Target new tool
OldToolLoc	1	(n+18) Loc. no. Target new tool
Status	1	Operation complete
Ready		Feedback from FC8
Error		Feedback from FC8

2. Change tool

If the preparation command has been correctly acknowledged with status = 1, the "Change" bit DB72.DBB n+0 bit 1 is set with the M06 command from the part program. The free parameters are also transferred again. All other values remain unaffected by the "Change tool" operation.

Two tools are involved in the tool change procedure. The old tool is in the spindle and the new tool is in the magazine. The tool transport is executed in this example with gripper 1 and gripper 2. Any change in the position of the tools must be communicated to the tool management with FC8/FC6. FC8/FC6 must be called twice for this purpose.

FC8/FC6 call with status 105 change in progress

The tool with the dual gripper is withdrawn from the magazine and the spindle. The old tool is now in gripper 2 at location no. 3 and the new tool in gripper 1 in location no. 2. This results in the following FC8/FC6 call:

FC 8 parameters	Values	Comment
Start		Starts task
TaskIdent	2	DB72 interface
TaskIdentNo	1	No. of active interface
NewToolMag	9998	(n+16) Mag. no. Spindle
NewToolLoc	2	(n+18) Loc. No. new tool New tool now in gripper 1
OldToolMag	9998	Mag. No. old tool
OldToolLoc	3	Loc. No. old tool Old tool now in gripper 2
Status	105	Procedure running
Ready		Feedback from FC8
Error		Feedback from FC8

Note

The operator uses FC8/FC6 to notify the tool management of the new positions of the exchanged tools.

Tool management knows which is the new (called) tool and which is the old (spindle) tool.

The current positions are also known to the tool management. If these positions change, the tool management is only informed about this through FC8/FC6.

Note

If T preparation and change signals are simultaneously present, then the tool call and the change command (T and M) are programmed in one block. When calling FC8/FC6, in such a case, only the change has to be acknowledged and not the selection.

FC8/FC6 call with status 1 tool change complete

While the gripper is moving the tools, the PLC can read the magazine location for the old tool (from the spindle) from DB72.DBW (n+24) and (n+26) and move the magazine to the change position. This position is location 8 in magazine 1 in this example. The tool change can now be mechanically ended by "inserting" the tools. Tool management shall be informed of this change in tool positions by a FC8/FC6 call with status = 1. The new tool is placed in the spindle of magazine No. 9998, location No. 1 and the old tool in magazine No. 1 at location 8.

FC 8 parameters	Values	Comment
Start		Starts task
TaskIdent	2	DB72 interface
TaskIdentNo	1	No. of active interface
NewToolMag	9998	(n+16) Mag. no. Spindle
NewToolLoc	1	(n+22) Loc. no. Spindle
OldToolMag	1	(n+24) Mag No. old tool
OldToolLoc	8	(n+26) Loc. No. old tool
Status	1	Operation complete
Ready		Feedback from FC8
Error		Feedback from FC8

If the dual gripper is to place the spindle tool in the magazine location of the new tool, the user must ensure that the magazine location is of the same size and location type as the spindle tool.

Here too, a 1:1 exchange is supported through appropriate setting of the search strategy by the tool management; in this case, the NCK checks the location type, size, adjacent location allocation,...

If this is the case, the transfer can be performed simultaneously (on the dual gripper in the spindle and in the magazine location at the change position).

FC8/FC6 must be parameterized as follows:

FC 8 parameters	Values	Comment
Start		Starts task
TaskIdent	2	DB72 interface
TaskIdentNo	1	No. of active interface
NewToolMag	9998	(n+16) Mag No. new tool
NewToolLoc	1	(n+18) Loc. No. new tool
OldToolMag	1	(n+20) Mag No. old tool
OldToolLoc	1	(n+26) Loc. No. old tool
Status	1	Operation complete
Ready		Feedback from FC8
Error		Feedback from FC8

3.3.8 Special cases "T0", empty spindle, multiple T selection

T0: Empty spindle

DB72.DBX(n+0).3 indicates that T0 has been programmed. If the spindle is to be emptied using T0, this can be identified by the fact that in DB72, DBW (n+20), DBW(n+22) data for new tool have the value "0".

Then, values NewToolMag and NewToolLoc of FC8/FC6 must be set to "0".

This applies to the preparation and to the change procedure.

Spindle is empty

A tool is to be loaded. This status is indicated by the fact that parameters OldToolMag and OldToolLoc have the value "0".

In this case, FC8/FC6 parameters OldToolMag and NewToolLoc must be set to "0" for tool preparation and change.

Multiple T selection

It can happen with multiple T selection that the program cannot be aborted by a RESET.

The abort response can be enhanced as follows:

- Cancel the read-in enable to prevent following blocks from being accepted in the main run.
- Then acknowledge with status 3 via FC8/FC6 (the tool command is rejected by the PLC).

3.3.9 Tool change with turret

Turret DB73

DB73 is the block used to "change" tools in the turret (i.e. by rotating the turret so that the required tool is in the working position). This data block has an interface for every turret. The turrets are numbered using ascending magazine numbers. User data are available at each interface as for the loading and unloading point. Data are additionally available for the new tool to be loaded at change. This data includes location type, sizes, tool status and the T number internally assigned in the NC.

Following completion of the tool-change operation, loading the new tool shall be acknowledged by FC7. To this end, the parameter "ChgdRevNo" receives the turret number of the new tool that has been inserted.

FC 7 parameters	Values	Comment
Start		Starts task
ChgdRevNo	1	1st turret
Ready		Feedback from FC8/FC6
Error		Feedback from FC8/FC6

3.3.10 Number of replacement tools

Machine data MD17500 MN_MAXNUM_REPLACEMENT_TOOLS can be set to select the maximum number of replacement tools.

Once the set threshold for the number of replacement tools has been reached it is no longer possible to:

- create a new tool with this ID (alarm) or
- assign a tool by renaming an already fully assigned group (alarm).

Alarms

For operation via the HMI, alarm 17192 is output as an indication as soon as the defined limit is violated.

If programming via a part program an additional interpreter alarm is triggered (e.g. 14020 if NEWT fails).

For the setting $MN_MAXNUM_REPLACEMENT_TOOLS = 0, the data is not evaluated; this is the default setting.

Note

Machine data MD17500 MAXNUM_REPLACEMENT_TOOLS has a maximum input value of "32", i.e. a tool group then contains a maximum of 32 tools.

3.3.11 Tool changing errors

If an error is detected by the NCK in the programmed tool preparation (e.g. no tool available, no free position in magazine) program processing is terminated with an alarm.

The operator can assess and rectify various problems without terminating the program.

The following problems can be solved:

- The tool data record is not or not entirely in the NCK.
- The part program contains a programming error.
- No more replacement tools of the tool group in question are present or available (only applies when tool management is active).
- Alarm 22067 or 22069 is output. The tool data record has been loaded into the NCK but is not assigned to a magazine location or the magazine of the tool is not accessible to the tool search (only applies when tool management is active). The tool must be reloaded "manually" (e.g. directly onto the spindle).

Note

The case of illegal "D number" can occur either if there is an error in the part program or the data record for the D number is not in the NCK.

Programming example

```
N10 ...
N100 T="Drill"             ;NCK detects an error
N110 ...
N200 M06                   ;to the extent that the tool change is explicitly program-
                           med in the same program for tool preparation
N210 ...
```

Note

As a rule **M06** is not programmed at the program level of tool preparation but rather in a subprogram, cycle or macro.

Bit 0 of MD22562 $MC_TOOL_CHANGE_ERROR_MODE is used to set at which block a stop is to be made.

TOOL_CHANGE_ERROR_MODE, bit 0=0:

```
N10 ...
N100 T="Drill"          ;NCK detects an error, program stops at this block
N110 ...
N200 M06
N210 ...
```

TOOL_CHANGE_ERROR_MODE, bit 0=1:

```
N10 ...
```

```
N100 T="Drill"          ;NCK detects an error
N110 ...
N200 M06                ;Program stops at this block
N210 ...
```

The error is detected during tool preparation, but is delayed by the NCK. The program continues and stops at M06. Tool preparation has been completed at this point in time for a regular program run. In the event of an error, tool preparation with the correct data can be subsequently repeated. In this situation, the language command GETSELT would supply the value "-1". With this information, e.g. in a change cycle, a branch can be made to fault handling.

The programming error (in block 100 in this example) is corrected by adding the offset to the tool change block:

N200 "T=Drill_1" M06

If a tool change (with M06 programming) is realized by means of a subprogram or cycle program, then the error can be rectified by inserting an overstore block (in the example).

3.3.12 Manual tools (adding tools during machining)

Manual tools are tools whose data are completely available in the NCK, but are not loaded into the magazine. The function is set with MD22562 $MC_TOOL_CHANGE_ERROR_MODE, bit 1=1. The automatically selected tool must be inserted in the machine manually and removed again manually after machining.

Responsibility of the operator

The user must ensure that

- the data record of the tool positioned on the spindle is actually in the NCK and
- that he or she places the tool that corresponds to the data record in the NCK on the spindle.

Tools which are loaded manually during machining are referred to as **"manual tools"**.

Note

The responsibility is on the users themselves to comply with the safety regulations via the PLC program.

Sequence

Internally, the NCK initiates an automatic sequence until the user can perform the tool change with a manual tool. The NCK searches for the selected tool and detects that a suitable tool is not available in the magazine. After a suitable tool is not found in the magazine, then a search is made in the other tools that are not loaded. The tool with the status active is selected from these. If an active tool does not exist, a replacement tool – corresponding to the selected search strategy – is selected.

If a suitable tool is found, then the manual tool can be loaded. The manual tools are identified in the interface to the PLC (VDI) by the **Magazine location no. 1** in the **magazine 9999**. As a consequence, the basic PLC program interprets this as involving a manual tool and sets the interface signals DB72.DBx(n) bit 5 and bit 6 "Replace or load manual tool". The PLC must ensure that a safe state exists in order that the user can perform the manual tool change.

Note

If the manual tool is loaded, alarm 17212: "Channel %1, Manual tool %3, Duplo No. %2, load to tool holder %4" is output. The alarm is confirmed by the tool-change acknowledgement from the PLC

The information alarms can be hidden ($MN_SUPPRESS_ALARM_MASK_2)

Note

It is not permissible that the PLC rejects a manual tool preselected from the NCK (tool rejection, also refer to MD20310 $TC_TOOL_MANAGEMENT_MASK).

The tool state "CHANGEACTIVE" is taken into account corresponding to the setting in "$MC_TOOL_MANAGEMENT_MASK".

A manual tool is always inserted via the spindle change position in DB72, also for the setting $MC_TOOL_CHANGE_MODE = 0.

Magazine location 9999/1 for manual tools

For manual tools, magazine location 1 in Magazine 9999 has a special significance.

Typically, as a result of the magazine configuration that is created on the HMI, the magazine with number 9999 is defined as loading magazine and has at least one magazine location, which has the number 1. As a consequence, in most cases magazine 9999 has a magazine location 1 with the property "loading point".

For the preparation and change command from the NCK or its acknowledgment, normally only locations associated with real magazines and tool holders are involved. However, if magazine location 1 of magazine 9999 is specified as source location of the tool to be loaded or as target location of a tool to be changed out, then it involves a "virtual magazine location address", which under certain circumstances, has nothing to do with the actually available magazine location 1 in magazine 9999. This can then be identified in so much that

- a tool can be loaded into the actual magazine location without an error message being output and
- a tool at magazine location 9999/1 - assuming that magazine 9999 is defined as a loading magazine - is not found when selecting a tool.

Block search, program test

In a block search, there is no difference to a normal tool change. However, the corresponding alarms are not generated.

No change commands are output to the PLC during the block search. If a manual tool needs to be loaded when the NC is first started, then this can be achieved using **magazine location 1** in **magazine 9999** and output of the corresponding alarm.

The data for the tools and magazines has to remain unchanged in the NCK during the **program-test mode** . A manual tool that has been loaded during program-test selection is therefore removed in terms of its data from the tool holder and saved internally. The stored manual tool is loaded back into the tool holder in response to PLC task "Return manual tool from magazine 9999, location 1".

Note

Several tool holders and manual tools can exist in the program test mode because of the technology used for the internal storing.

Boundary conditions

In conjunction with tool selection, tool change and offset selection, only problems associated with the offset block technique can be rectified that have arisen because of programming errors or incorrectly defined data in the NCK.

Manual tools for circular magazines (change with T command)

Manual tools are loaded and removed via the spindle interface in the DB72 - with the exception of a special case. The interface number is the number of the spindle / tool holder that was assigned to the turret. The special case occurs if the tool in the spindle is a manual tool. A new tool, which is located in the turret, is programmed. In this case, the command in DB73 is output in the corresponding turret interface. In this case, the manual tool must first be removed. The tool be loaded into the turret must be swiveled into the machining position. The command then has to be acknowledged.

Unique identification of manual tools

Previous behavior:
A manual tool was solely identified due to the fact that it did not have an owner location. This is then used to derive as to whether (within the scope of a change) it is loaded or removed from the virtual magazine location 9999/1.

This also means that a tool which, for example, was loaded onto the spindle as a result of an operator action at the HMI, is a manual tool. As a consequence, it cannot be placed in the magazine using T0/M06.

The fact that a tool loses its owner location when being transported can be a legal state. This means that it is no longer possible to reliably identify a manual tool.

New behavior:

A manual tool is assigned a unique identifier, $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. By moving this tool between the locations of the buffer magazine - independent of the motion command that initiated this - the "manual tool" state no longer changes. (Exception: see Section, resetting the "manual tool" state)

Note

As a consequence, a tool that was loaded onto the spindle, e.g. via SINUMERIK Operate, is now not a manual tool. As a consequence, it is placed in the magazine using T0/M06.

Setting the "manual tool" state

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

- machine data $MC_TOOL_CHANGE_ERROR_MODE, bit1 is set
- as part of a tool change, an unloaded tool was selected.

Resetting the "manual tool" state

The "manual tool" state of a tool is reset, if

- 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 HMI operator action)
  Writing $TC_MPP6[m,p]=0 to the location of the buffer magazine where the manual tool is currently located. (m=magazine no., p=location no.)
- or the command to select or load a manual tool was interrupted and the tool was still not moved to a magazine location
- or the tool was allocated an owner location. For example, this can be realized with the extended NC language command "GETFREELOC" using a reservation parameter.
- or the tool was brought to a location of a real magazine or the loading magazine.

Explicitly writing the state "manual tool" ($TC_TP8, bit 15)

Further, the "manual tool" state of the tool can be set or reset by writing to the system variables for the tool state ($TC_TP8).

It is not possible to set the "manual tool" tool state, if

- machine data $MC_TOOL_CHANGE_ERROR_MODE, bit1 is not set. Otherwise, Alarm 17242 ("Channel %1, block %2, manual tool cannot be set as the function is not active") is issued.
- It is not possible to set the "manual tool" tool state, if the tool is located
  - in a real magazine location,
  - in a loading magazine,
  - in a buffer magazine location and has an owner location.
  If this is set, it is rejected with Alarm 17218 ("Channel %1, block %2, tool %3 cannot become a manual tool").
  If a tool, which is located in a magazine location, is to become a manual tool, then it must be transported into the buffer or loading magazine (using asynchronous transfer, PI service or MVTOOL language command). Then, using the NC language command "DELMLOWNER", the reservation and the owner location can be deleted. Only then can the "manual tool" state be set.

If the tool state "manual tool" is explicitly reset, then initially an empty location search is not made for the tool and no reservation is made. It is also not allocated an owner location. Only in subsequent operations can an empty location search be made - and real magazine locations reserved.

The state "manual tool" can also be set and reset via the OPI interface (block T/TD, column 8).

Properties of the "manual tool" state

For a manual tool, when the tool is being changed, a search is not made for an empty location, an owner location is not allocated and a reservation is not made. Also for other tool transport types and search operations, for a manual tool, an empty location is not searched for - exception, explicit reservation using the NC language command "GETFREELOC".

The command data set, which the NCK sends to the PLC when a tool is being changed, always contains as target address for a manual tool as old tool, the virtual magazine location 9999/1.

For manual tools, the empty location search strategy "1:1 replacement" is not effective, as the "manual tool" property has priority, i.e.

- if the new tool is a manual tool and the old tool comes from a location in a real magazine, then the transport task sent by the NCK to the PLC is as follows: Old tool back into the real magazine, new tool from virtual magazine location 9999/1.
- And vice versa: If the old tool is a manual tool and the old tool comes from a location in a real magazine, then the transport task sent by the NCK to the PLC is as follows: Old tool to virtual magazine location 9999/1, new tool from the location of the real magazine.

The "manual tool" state is effective via a power on.

When unloading a tool to the virtual magazine address 9999/1, the settings of MD $MN_TOOL_UNLOAD_MASK are not effective, i.e. the corresponding tool states are kept.

Note

Using the NC language command "GETFREELOC" or for the PI service "_N_TMFDPL" or

"_N_TMFPBP", a check is not made as to whether a manual tool is involved. If the NC language command "GETFREE LOC" is parameterized so that also a buffer location is reserved, then as a consequence the tool loses its "manual tool" state.

Note

At power on, the "manual tool" state for tools is reset, which

- are at a real magazine location or the loading magazine, or
- are at a buffer magazine location and have an owner location, or
- are not at any magazine location.

Note

If, when writing to $TC_MPP6[m,p] a manual tool is set at a magazine location, the "manual tool" state is deleted, if

- "m" addresses a real magazine or the loading magazine, or
- "m" addresses the buffer magazine and the tool has an owner location.

Note

If a manual tool should come to the unloading point magazine location 9999/1 for unloading using an asynchronous transfer, the NC language command "MVTOOL" or the PI service "_N_TMMVTL", then this location must be defined and also free. On the other hand, for unloading operations for a manual tool, magazine location 9999/1 does not have to be defined and does not have to be free.

Behavior, NC language commands "TCA", "TCI" and return transport for manual tools

If the language command TCA programs a certain tool, which is not located at a magazine location (also not at an internal magazine location), then this tool is selected and changed in as manual tool.

If the language command TCI is applied to a buffer location where there is a manual tool, then the target location for the movement is the virtual magazine location 9999/1 (this means that the TCI instruction does not bring the manual tool into a real magazine corresponding to the definition of TCI for other tools). Alarm 17215 "Channel %1 tool management: Remove manual tool %3 from buffer %2" is issued. This alarm is acknowledged by the tool change acknowledgement from the PLC.

Users must themselves ensure that the safety regulations are maintained through their PLC programs, such as when loading a manual tool.

The tool state "BACKTRANSFER" for a manual tool is evaluated in the form so that the virtual magazine location 9999/1 is specified in the PLC command for the manual tool.

Alarm 17215 "Channel %1 tool management: Unload manual tool %3 from buffer location %3." is issued. This alarm is acknowledged by the tool change acknowledgement from the PLC.

Users must themselves ensure that the safety regulations are maintained through their PLC programs, such as when loading a manual tool.

Suppressing alarms 17212, 17214, 17215, 17216

The manual tool change request alarms can be suppressed using bit 12 in MD $MN_SUPPRESS_ALARM_MASK_2. Then, using the tool change command data, the PLC must correctly load and remove the manual tool.

Example 1

(milling machine, $MC_TOOL_CHANGE_MODE=1, 1 x spindle, 1 x chain-type magazine, double gripper)

T=MILLER_120"	Task to PLC:	new tool from 9999/1 to 9998/1
		old tool from 9998/1 to 1/10
	"Manual tool" identifier is set	
M06	Information alarm 17212	"...manual tool "MILLER_120" Duplo number 00001 load onto spindle / tool holder 1"
	The information alarm is acknowledged with the final acknowledgment from M06	
...		
T0	Task to PLC:	new tool from 0/0 to 0/0
		old tool from 9998/1 to 9999/1

M06	Information alarm 17212	"...manual tool "MILLER_120" Duplo number 00001 take from spindle / tool holder 1"
	With the final acknowledgment from M06	
	a. The information alarm is acknowledged	
	b. The identifier "manual tool" is deleted	

Example 2

(milling machine, $MC_TOOL_CHANGE_MODE=1, 1 x spindle, 1 x chain-type magazine, double gripper)

The "FACE MILLER_120" tool is loaded directly onto the spindle (e.g. using an HMI operator command) and is then loaded into the magazine using T0/M06.

	Task to PLC:	new tool from 9999/1 to 9998/1
	Identifier "manual tool" is **not** set, no information alarm	
T0	Task to PLC:	new tool from 0/0 to 0/0
		old tool from 9998/1 to 1/x
	As this does not involve a manual tool, the NCK performs an empty location search and places the tool in the magazine.	
M06	No information alarm, as it does not involve a manual tool.	

3.3.13 Tool changes in the NCK using synchronized actions

Overview

At tool change and at loading/unloading it is often necessary to supply the NC cycles with the data for the participating tools.

Usually this is done via the "fast data channel" (dual port RAM) using FC21.

The PLC user program checks the interface in DB71/72/73.

If a new command is pending, the data (new location, old location, T_number,...) are read, preprocessed and supplied to the cycles via FC21. There they are (usually in synchronized actions) read as variable $A_DBB[...] and, for example, magazine movements are derived from them.

To reduce the overhead involved and create simpler mechanisms, most of the data of the tool management interface was mapped onto the NC variables for read access.

This means that all information about the old tool and the new tool can be read directly in the part program or in synchronized actions; the "detour" via the PLC is no longer needed.

The following variables are used for the mapping process:

$AC_TC_FCT	Function no. (command no. of the NCK)
$AC_TC_STATUS	Acknowledgement status from the PLC
$AC_TC_THNO	Tool holder or spindle no. on which the change was executed
$AC_TC_TNO	Internal T no. of the tool to be changed or prepared

$AC_TC_MFN	Source **new** tool: Magazine number
$AC_TC_LFN	Source new tool: Location number
$AC_TC_MTN	Target new tool: Magazine number
$AC_TC_LTN	Target new tool: Location number
$AC_TC_MFO	Source **old** tool: Magazine number
$AC_TC_LFO	Source old tool: Location number
$AC_TC_MTO	Target old tool: Magazine number
$A_TC_LTO	Target old tool: Location number
$AC_TC_CMDT	Trigger variable to output command of the NCK (is set for one IPO)
$AC_TC_ACKT	Trigger variable for acknowledgement of the PLC (is set for one IPO)
$AC_TC_CMDC	Counter for the command output
$AC_TC_ACKC	Counter for the acknowledgements
$AC_TC_MMYN	Owner magazine of the new tool
$AC_TC_LMYN	Owner location of the new tool
	Supplementary variables for multitool
$AC_TC_TOOLIS	Transported tool of the type: 0 = Single tool 1 = Multitool with distance coding "location" 2 = Multitool with distance coding "distance" 3 = Multitool with distance coding "angle"
$AC_TC_MTDIST	Distance of the MT location with the new tool from the reference point. If the multitool has "length" as distance coding, the parameter contains the length in the unit mm or inch as specified by the set dimension system.
$AC_TC_MTNLOC	Number of locations in the MT of the new tool
$AC_TC_MTTN	MT number of the MT location with the new tool
$AC_TC_MTLTN	MT location number with the new tool

Note

The variables are read-only (exception $AC_TC_CMDC and $AC_TC_ACKC). The acknowledgement mechanism remains unaffected (the PLC still continues to acknowledge all commands from the NCK via FC8/FC6 or FC7).

Method of operation

The variables are written:

1. with **each** command from the **NCK** (**CMD**)
2. with **each** acknowledgement from the **PLC** (**ACK**)
3. with **Power On** all are set to value "-1"

The data is retained until it is overwritten by a new command. This means that with commands of the same type, it is not possible to tell from the function number ($AC_TC_FCT) whether a new task is present.

The exceptions are:

$AC_TC_TNO and $AC_TC_THNO

If, for example, the NCK outputs a T preparation, both of these variables are set to "-1" with the first PLC acknowledgement via FC8/FC6 (e.g. status 105).

Note

Scanning should only take place in synchronized actions. Depending on the application, this can then trigger the variables $AC_TC_CMDT and/or $AC_TC_ACKT.

Example 1

Positioning a tool chain onto the old location

Assumption:

The tool chain has 36 locations, is defined as rotary and indexing axis, increments are 10 degrees therefore each graduation corresponds to one magazine location.

Tool_Change_Mode=1, Tool_Change_M-Mode=6

Ids=1 every(($AC_TC_CMDT==1)and(($AC_TC_FCT==2)or($AC_TC_FCT==5))) do $R10=itor($AC_TC_LTO)

...

if ((R10>0)and($A_DBB[x]==5)) pos[U1]=cdc(R10)

endif

The trigger is sent to the command output of the NCK and with command "2" (T preparation) or command "5" (T/M06 in one block) the old location is read out and stored in R10

(itor=IntegerToReal - format conversion if the variable is stored in the R-variable in synchronized actions).

Later in the program, when the enables from the PLC are present (for example as $A_DBB[x]==5), the magazine axis is traversed to the saved position (old location= $AC_TC_LTO).

A magazine movement could also be started as follows (shown here in simplified form):

Ids=1 every(((($AC_TC_FCT==2)or(AC_TC_FCT==5))and ($AC_TC_STATUS==105))and(($AC_TC_LTO>0))) do pos[U1]=cdc($AC_TC_LTO)

With commands "2" and "5" (T preparation or T/M06), with old location>0 and PLC acknowledgement status "105" (serves as enable), the magazine axis is traversed.

Old location>0: If the spindle was empty, there is no old tool and the old location is 0. Therefore, the magazine axis does not need to move.

Example 2

Swiveling a turret

Assumption:

Turret, 6 locations, the turret is defined as an indexing axis, 60 degree increment, corresponds to one tool location, 1xSpindle, Tool_Change_Mode=0

```
Ids=1 every($AC_TC_CMDT==1)and($AC_TC_FCT==4)and($AC_TC_LFN>0) do
$R10=itor($AC_TC_LFN)
...
if ((R10>0)and($A_DBB[x]==5)) pos[B]=cac(R10)
endif
...
```

The NCK is triggered with the command output and with command "4" (change with T command) the new location is read out and saved in R10

(itor= IntegerToReal - format conversion if the variable is stored in the R-variable in synchronized actions).

Later in the program, when the enables from the PLC are present (for example as $A_DBB[x]==5), the turret is traversed to the saved position (new location=$AC_TC_LFN).

The logic operation $AC_TC_LFN>0 prevents a movement from taking place if, for example, T0 was programmed.

3.3.14 Function replacement

Overview

The function allows a TCA, T, M or D function to be replaced by a cycle.

The function is available independent of the tool management and is described in detail in the "Function Manual, Basic Machine, Section 2.11 (mode group, channel, program operation)".

The function is set using the following machine data:

$MN_M_NO_FCT_CYCLE	Enter the M function that is to be replaced, e.g. "6"
$MN_M_NO_FCT_CYCLE_NAME	Name of the cycle that should be executed instead of the M function, e.g. "L6"
$MN_T_NO_FCT_CYCLE_NAME	Name of the cycle that should be executed instead of the T function, e.g. "T_cycle"
$MN_D_NO_FCT_CYCLE_NAME	Name of the cycle that should be executed instead of the D function, e.g. "offset"
$MN_TCA_CYCLE_NAME	Name of the cycle that should be executed instead of the TCA command, e.g. also "T_cycle"

The mode of operation will now be explained using a T function replacement as an example.

Example

If one of the auxiliary functions to be replaced has been programmed, e.g.

N10 G90 G00 Z-100 S3000 T="Miller_20mm" M65

then the NC executes a so-called block splitting and "splits" this NC block into two blocks

a) G90 G00 Z-100 S3000 M65

b) T="Miller_20mm"

The information "Miller_20mm" is transferred as parameter to the cycle.

If the T call is now again programmed in the replacement cycle, then no additional replacement is made.

Machine data $MN_T_NO_FCT_CYCLE_MODE is used to set whether the cycle is executed at the block start or at the block end (or also at the start and end).

The offset handling type is also set: If an offset is programmed (e.g. D3) in the NC block, which is to be split, then it can be set as to whether the offset selection is taken into account or is transferred as parameter to the replacement cycle.

Transfer variables of the T replacement cycle

$C_T	T number of the tool (numerical)
$C_T_Prog	Bool variable that shows whether a T word is available in $C_T
$C_TS	Tool identifier (string)
$C_TS_Prog	Bool variable that shows whether an identifier is available in $C_TS
$C_TE	Address extension of the T word
$C_TCA	Bool variable that indicates whether the TCA replacement is active
$C_DUPLO_PROG	Bool variable that indicates if the Duplo number for the TCA replacement has been programmed
$C_DUPLO	For $C_DUPLO_PROG == TRUE, contains the value of the programmed duplo number
$C_THNO_PROG	Bool variable that indicates if the tool holder / spindle number for the TCA replacement has been programmed
$C_THNO	For $C_THNO_PROG == TRUE, contains the value of the programmed tool holder / spindle number
$C_D	Programmed D no.
$C_D_Prog	Bool variable that shows whether an offset number is available in $C_D
$C_DL	Programmed additive/setup offset
$C_DL_Prog	Bool variable that shows whether an offset number is available in $C_DL
$C_MTL_PROG	Bool variable that shows whether a multitool location is available in $C_MTL (multitool * T=location)
$C_MTL	Programmed multitool position (for T=location)

Example of a T replacement cycle

```
;T replacement cycle
;
;Machine data to be set
;$MN_T_NO_FCT_CYCLE_NAME="T_cycle"
;$MN_TCA_CYCLE_NAME="T_cycle"
;
;
def string[32] Ident
def int T_No,Spi_No,T_prep,T_Spi,TH_No
;
;
```

```
;---------------------------------------------------
;--- T command output ---
;---------------------------------------------------
if(($C_T_PROG==1)or($C_TS_PROG==1)or($C_TCA==1))
  if $C_T_PROG==1                                    ;T=numeric
    if $C_T==0
      if $C_T>=1
        T[$C_TE]=0
      else
        T0
      endif
    else
if $CTE>=1
       if $C_MTL_Prog==true                     ;multitool T=location
          MTL=$C_MTL T[$C_TE]=$C_T
        else
          T[$C_TE]=$C_T
        endif
      else
        if $C_MTL_Prog==true
          MTL=$C_MTL T=$C_T
        else
          T=$C_T
        endif
      endif
    endif
  endif
  ;
  if $C_TS_PROG==1                               ;T=string
    if $C_TCA==1
      if $C_DUPLO_PROG==1
        if $C_THNO_PROG>=1
          TCA($C_TS, $C_DUPLO, $C_THNO)
        else
         TCA($C_TS, $C_DUPLO)
        endif
      else
        if $C_THNO_PROG>=1
          TCA($C_TS, $C_THNO)
        else
          TCA($C_TS)
        endif
      endif
    else
      if $C_TE>=1
        T[$C_TE]=$C_TS
      else
        T=$C_TS
      endif
    endif
  endif
endif
```

```
;
;--------------------------------------------------
;Continuation of the T preparation (machine functions)
;--------------------------------------------------
;
;--------------------------------------------------
;----Tool offset----
;--------------------------------------------------
if (($MN_T_NO_FCT_CYCLE_MODE B_and 'B0')=='B0')
  if $C_D_Prog==true
    D=$C_D
  endif
  if $C_DLProg==true
    DL=$C_DL
  endif
endif
;
msg("")
M17
```

In the cycle, actions can now be easily executed to prepare the tool, scan the gripper, position the magazine data transfer to the PLC, start auxiliary channels, etc.

Note

The TCI language command cannot be replaced!

Transfer variables of the D replacement cycle

$C_D	Programmed D no.
$C_D_Prog	Bool variable that shows whether an offset number is available in $C_D
$C_DL	Programmed additive/setup offset
$C_DL_Prog	Bool variable that shows whether an offset number is available in $C_DL

Transfer variables of the M replacement cycle

$C_T	T number of the tool (numerical)
$C_T_Prog	Bool variable that shows whether a T word is available in $C_T
$C_TS	Tool identifier (string)
$C_TS_Prog	Bool variable that shows whether an identifier is available in $C_TS
$C_TE	Address extension of the T word
$C_D	Programmed D no.
$C_D_Prog	Bool variable that shows whether an offset number is available in $C_D
$C_DL	Programmed additive/setup offset
$C_DL_Prog	Bool variable that shows whether an offset number is available in $C_DL
$C_ME	Address extension of the M word
$C_DL_Prog	Bool variable that shows whether an offset number is available in $C_DL

$C_M_PROG	TRUE if M function was programmed for tool change
$C_M	Value of substituted address M (integer)

There are two different cases:

1. The substitution subprogram for the tool change configured with $MN_M_NO_FCT_CYCLE_PAR was called. $C_M then contains the value $MN_M_NO_FCT_CYCLE [$MN_M_NO_FCT_CYCLE_PAR].
2. In the case of a tool change with M code, only one substitution subprogram was configured for the T and/or D/DL addresses. If the M code for the tool change is programmed together with one of the addresses to be substituted, the value of $MN_TOOL_CHANGE_M_MODE is transferred to the replacement program in $C_M.

If parameters are to be transferred to the M replacement cycle, then this must be set using $MN_M_NO_FCT_CYCLE_PAR = 1

The default setting is "0", i.e. no parameter is transferred.

The parameter transfer refers exclusively to values, which are programmed in this NC block, i.e. in that block where block splitting is performed.

This application is of interest if Tx M06 is always programmed in an NC block. This means that the T selection, change as well as offset selection and deselection can be included in just one single cycle.

Note

Only one function replacement is possible for each NC block.

If, for example, ...T="Miller_20mm" and M06 are programmed in the block, and a replacement cycle applies to both, then this is rejected with an alarm.

Note

Parameters $C_T_PROG, $C_TS_PROG, $C_D_PROG and $C_DL_PROG act as trigger. The T no., D no. ... are valid if these parameters have the value "TRUE" - but only then.

Note

More detailed information on replacing NC language commands is provided in the "Function Manual Basic Functions".

3.3.15 Block search

Block search with calculation

On a block search, selection on reset or start, the tool is selected during preprocessing. In this case the PLC is not allowed to reject the tool (see bit 4 in MD20310). If it does, an alarm is generated. The block search must then be repeated. Use of the active tool can only be prevented from an external source (HMI, PLC).

In block search with calculation the program is generally put into a state where the selected block can be executed. With respect to the tool management function, this means that the tool that should be located in the spindle when the machining block is reached must now be loaded to it.

If another tool is located in the spindle a "replace" command is initiated. In such a case, the signals "Prepare change" (DB72.DBX(n+0).2 and "Execute change" (DB72.DBX(n+0).1 are present at the same time since the auxiliary functions are output together.

Example: $MC_TOOL_CHANGE_MODE=0

Tool "Drill1" is loaded in the spindle. The new search target has T = "Drilling machine 2" as the momentary tool programming.

NCK initiates the tool change. PLC must not reject.

Note

Tool rejection by PLC: If bit 4 of MD20310 $MC_TOOL_MANAGEMENT_MASK is set, then the PLC usually has the possibility to again request preparation for a tool change, yet this time with different parameters, i.e. to reject the tool. This is not possible during block search. In this case, the machine data setting is ignored.

Note

Because the tool change is frequently performed using cycles, a "replace command" generated by the block search must be executed in an asynchronous subprogram (ASUP). Modal and static motion-synchronization action is retained at the beginning of ASUP and is also effective in the asynchronous subprogram. If the asynchronous subprogram is not continued with Repos, the modified modal and static motion-synchronous actions in the main run remain operative.

Alternatively, execution of the NC part program can be stopped by halting feed and read-in, and a fault message "Wrong tool in spindle after the block search" can be generated.

Tool cannot be used

Alarm 22068 is output if the tool to be loaded at the search target cannot be used (it is not available, disabled, ...), the program must be canceled using a reset.

In order that the block search is not canceled, if the tool involved at the search target is not the tool to be loaded, then an attempt is made to also permit a disabled tool (and therefore also an unloaded tool) – assuming that it only has the "enabled" state and therefore has a defined geometry. This is realized before the alarm is generated.

If an additional tool change is programmed, the tool that is underway that cannot be used is not noticed and does not stop the operation. However, if an attempt is made to precisely load this disabled/unloaded/.. tool at the end of the block search after the first START, then this is noticed in the NCK and acknowledged with alarm 22067; this then no longer allows the program to be continued. The PLC can permit a disabled tool to be selected (i.e. by setting the interface signal "Tool disabled ineffective" in the channel DB). However, in a specific case, this must be realized during the block search, i.e. before the NCK outputs Alarm 22068 or 22067. If, for a block search, tool programming is initiated in the NCK, and there is neither a tool that can be used, disabled nor unloaded, then alarm 22068 is output, which then exits the block search.

Example of a search with block splitting effect

```
N100 T="Tool1" M06 D1
N110 SETMTH(1)                   ;Tool holder1 becomes master tool holder
N120 T="ToolZ2" M06 D2           ;Target block: is not yet interpreted
N1000 IF($P_PROG_EVENT ==5)      ;ASUP is started
.............
N1020 SETMTH(2)                  ;Tool holder 2 becomes master tool holder
.............
N1040 ENDIF
N1099 REPOSA
N110 SETMTH(1)                   ;The interrupted main program is continued/started
                                 after the last executable block before the search
                                 target.
                                 ;Tool holder 1 becomes master tool holder again.
```

3.3.16 Block search (SSL) in conjunction with active tool management

The block search has already been described. Here we shall deal with the specific features in conjunction with active tool management.

The block search establishes the start position of the target block. Auxiliary functions programmed in the block search are collected and output in action blocks at the end of the block search.

For now this also applies to the T command and M06. This depends on the setting in machine data 20128 $MC_COLLECT_TOOL_CHANGE

0 = Neither T preparation nor M06 are output.

1 = T preparation and M06 are collected and output (and must be acknowledged to end the block search). Default setting.

The following example shows how to proceed with block search.

Configuration: Milling machine, one spindle

Settings:

$MC_Tool_Change_Mode=1, i.e. change with M06

Block search

Tool change should be subsequently executed:

Situation:

T="Face_80mm" is located in the spindle

Block search to N98 (block search with contour calculation)

Target:

In order to continue in the program:

a. Tool "1537" must be loaded

b. Tool "Drill_6mm" must be prepared

```
...
N10 T="1231"                ;T no. 1
...
N20 M06
...
N30 T="Face_80mm"           ;T no. 2
...
N70 M06
...
N80 T="1537"                ;T no. 3
N90 M06
...
N95 T="Drill_6mm"           ;T no. 4
...
N98
```

Settings:

$MC_Tool_Change_Mode=1

$MC_Collect_Tool_Change=0

$MN_Search_Run_Mode bit 1=1

$MC_Collect_Tool_Change=0 means: **No output** of T and M06 after block search.

Procedure:

- Negative acknowledgement is not required in the PLC.
- The program "Prog_Event.SPF" is started with the last action block.
  The change and preparation must be subsequently carried out.

Note

In order to start the Prog_Event after a block search, settings are required in the machine data.

Prog_Event.SPF

…

def int T_Vor, T_Spi, T_active

…

GETEXET(T_active)	The spindle tool is read from NCK view (blocks N80 and N90) _active=3
GETSELT(T_Vor)	T preparation is read from block N95 T_Vor=4
T_Spi=$TC_MAP6[9998,1]	The actual spindle tool is read T_Spi=2

…

;Load correct tool

if ((T_Spi< >T_active)and(T_active>0))

T=$TC_TP2[T_active]	Preparation of tool "1537"
L6 ;change cycle	Load tool "1537"
Endif	
…	
if T_prep< >T_active	
if T_prep>0	
T=$TC_TP2[T_prep]	Preparation of tool "Drill_6mm" from block N95
Endif	
If t_prep==0	
T0	
Endif	
Endif	

Note

If a change is output by the action blocks (in example block N80 and N90), it is always a command "5", i.e. "Prepare change" and "Perform change" are pending in DB72 at the same time.

If the correct tool is already placed in the spindle (i.e. in the block search example at block N70 and $MC_COLLECT_TOOL_CHANGE=1 is set), the T preparation is issued (from block N30). The setting for bit 12=0/1 in the MD $MC_TOOL_MANAGEMENT_MASK machine data is not evaluated.

Difference between the commands GETEXET and $P_TOOLNO:

GETEXET

Reads the T no. of the spindle tool from the NCK perspective.
Independent of the offset selection.
This has been specifically developed for the block search application.

$P_TOOLNO

Reads the T no. of the active tool.

This does not refer to the "active status" of the tool which is set via the T preparation, instead it refers to the tool whose offset is being calculated. This view of the tools means that a tool doesn't become an active tool until the offset is selected - which is what is read with $P_TOOLNO. This means dependency on MD $MC_CUTTING_EDGE_DEFAULT,

Example:

```
...
N100 T="Countersink"           ;T No. 5
N110 M06
N108 G90 G00 D1 X...
...
N200 T="Drill"                 ;T No. 32
N210 M06
N212 G90 G00 D1 X...
```

Block search to block N200

1. Setting $MC_Cutting_Edge_Default=-2
   GETEXET = 5
   $P_TOOLNO = 5
2. $MC_Cutting_Edge_Default=1
   GETEXET = 5
   $P_TOOLNO = 5

Block search to block N212

1. Setting $MC_Cutting_Edge_Default=-2
   GETEXET = **32**
   $P_TOOLNO = 5
2. $MC_Cutting_Edge_Default=1
   GETEXET = 32
   $P_TOOLNO = 32

3.3.17 System variables for state before block search

3.3.17.1 Description of the variables

The function is available with TMBF, TMMO and TMMG.

To round off the "block search" function, the active tool offset must be available in an ASUP before the block search or test mode when the target block of the block search is reached. In this ASUP, the OEM can implement the steps necessary to place the tools in the state required in the target block. The following system variables that describe the state in the reset state before the block search are available for this functionality:

• `$P_MTHNUM_BEFORE_SEARCH`	Tool holder to which the offset refers. This is normally the master tool holder.
• `$P_D_BEFORE_SSL`	Active offset in the reset state
• `$P_DL_BEFORE_SSL`	Active total and setup offset in the reset state

These system variables are available in the "without magazine management" and the "with magazine management" versions.

The selected and loaded tools on the individual tool holders before the block search are available with the extension of the NC language commands GETSELT - Read the selected T No. (Page 277) and GETEXET - Reading of the loaded T number (Page 279).

In the "**with** magazine management" version, the essential information is already available through the assignment of the tool holders or spindles in the buffer magazine, because the magazine location assignment is not changed by the block search. However:

- The master tool holder
- The selected tool offset "D" and
- The selected insert offset "DL"

that were valid before the block search, are not available.

In the "**without** magazine management" version, there is no information about the state before the block search. As in this case the NCK does not have any data about the location of the tool, it can only draw conclusions about the tool location from the last instructions before the block search.

The state before the block search is the reset state. This means:

- The settings made during the NC reset become the values of the system variables.
- With an NC reset, the master tool holder set during the NC reset determines the active tool and thus the active offset.
- Therefore the settings from the init block are not included in the system variables at the start of the block search.

After reaching the block search target, the system variables take on the values of the corresponding system variables from the NC program, if a new value is set.

Overview of the available system variables for the reconstruction of the selected offset in a ProgEvent program after a block search:

System variable for the state before the block search	System variable for the state during and after the block search	
	Without magazine management	With magazine management
$P_MTHNUM_BEFORE_SEARCH	$P_MTHNUM	$P_MTHNUM
GETSELT(tNo,Th,"S")	$P_TOOLP GETSELT(tNo,Th)	GETSELT(tNo,Th)
GETEXET(tNo,Th,"S")	$P_TOOLNO GETEXET(tNo,Th)	$P_TOOLNO GETEXET(tNo,Th)
$P_D_BEFORE_SEARCH	$P_TOOL	$P_TOOL
$P_DL_BEFORE_SEARCH	$P_DLNO	$P_DLNO

tNo: Tool number, Th: Tool holder

3.3.17.2 Example

The following settings are active:

- The tool base functionality (TMBF) is activated.
- The tool change is performed with M6.
- Cutting edge default=-2

In the reset state, the following state is reached for the tool offsets:

- Master tool holder = 3
- Active T number on master tool holder = 5
- Active D offset = 2
- Programmed T number on master tool holder from last NC program = 6

This results in:

```
$P_MTHNUM_BEFORE_SEARCH =3
```

```
GETSELT(tNo1, 3) with tNo1=6
```

`GETEXET(tNo2, 3) with tNo2=5`

`$P_D_BEFORE_SEARCH = 2`

`$P_DL_BEFORE_SEARCH = 0` (The setup offset is not considered further and behaves in the same way as the D offset selection.)

Sequence:

- The following NC program is to be processed in the block search up to block N220
- The old D offset is selected again in a ProgEvent ASUP in order to safely remove the tool from the tool holder taking the old D offset values into account and
- The new tool is activated with the D offset in the target block

No new tool has been loaded on tool holder 3 up to the target block.

```
.....
N100 SETMTH(2)
N110 T8
N120 M6
N130 D3
N200 T9
N210 SETMTH(1)
N220 X10 G0
```

The meaning of the abbreviations used in the overview is as follows:

MTh: Value of master tool holder ($P_MTHNUM)

SelT(th): Return value of GETSELT for the tool holder "th"

ExeT(th): Return value of GETEXET for the tool holder "th"

DNo: Value of $P_TOOL

The ending S means the value of the corresponding variable "_BEFORE_SEARCH" or with the optional third parameter "S".

MTh	MThS	SelT(2)	SelTS(2)	ExeT(2)	ExeTS(2)	SelT(3)	SelTS(3)	ExeT(3)	ExeTS(3)	DNo	DNoS
Before N100											
*	3	*	*	*	*	*	6	*	5	*	2
After N110 T8											
2	3	**8**	*	*	*	*	6	*	5	*	2
N120 T8											
2	3	8	*	**8**	*	*	6	*	5	*	2
N130 D3											
2	3	8	*	8	*	*	6	*	5	**3**	2
N200 T9											
2	3	**9**	*	8	*	*	6	*	5	3	2
N210 SETMTH											

MTh	MThS	SelT(2)	SelTS(2)	ExeT(2)	ExeTS(2)	SelT(3)	SelTS(3)	ExeT(3)	ExeTS(3)	DNo	DNoS
1	3	9	*	8	*	*	6	*	5	3	2
N220 X10 G0											
1	3	9	*	8	*	*	6	*	5	3	2

* Means that the value for this example is irrelevant or indeterminate.

The ProgEvent program has the following rough structure:

```
N1000 DEF INT TNO1, TNO2
N1010 DEF INT MTH_SAVE, D_NO_SAVE, SELTNO_SAVE, EXETNO_SAVE
N1020 DEF INT SELTNO1_SAVE, EXETNO1_SAVE, TH_OF_D_SAVE
; Save the currently reached state at the end of the block search
N1100 MTH_SAVE = $P_MTHNUM D_NO_SAVE=$P_TOOL TH_OF_D_SAVE =
$P_TH_OF_D
N1110 GETSELT( SELTNO_SAVE, $P_TH_OF_D)
N1120 GETEXET( EXETNO_SAVE, $P_TH_OF_D)
; Collect data on the state before the block search
N1150 GETSELT(TNO1,$P_MTHNUM_BEFORE_SEARCH, "S")
N1160 GETEXET(TNO2,$P_MTHNUM_BEFORE_SEARCH, "S")
N1170 GETSELT(SELTNO1_SAVE,$P_MTHNUM_BEFORE_SEARCH)
N1180 GETEXET(EXETNO1_SAVE,$P_MTHNUM_BEFORE_SEARCH)
; Select old offset again.
N1200 SETMTH($P_MTHNUM_BEFORE_SEARCH)
N1210 T=TNO2
N1220 IF (EXETNO1_SAVE == TNO2)
; Old tool is still on the tool holder
N1230 M6 M=spec; additional information that M6 does not require a
tool change by PLC
N1240 D=$P_D_NO_BEFORE_SEARCH
N1250 X=x_safe Y=y_safe Z=z_safe G0 N1260 ELSE
....
N1300 ENDIF
; Load desired tool in accordance with target block
N1400 SETMTH(TH_OF_D_SAVE)
N1500 T=EXETNO_SAVE
N1510 M6
N1520 D=D_NO_SAVE
; Restore master tool holder in accordance with target block
```

```
N1600 SETMTH(MTH_SAVE)
REPOS
```

MTh	MThS	SelT(2)	SelTS(2)	ExeT(2)	Ex-eTS(2)	SelT(3)	SelTS(3)	ExeT(3)	Ex-eTS(3)	DNo	DNoS
Before N1200:											
1	3	9	*	8	*	*	6	*	5	3	2
After N1200:											
3	3	9	*	8	*	*	6	*	5	3	2
N1210:											
3	3	9	*	8	*	**5**	**5**	*	5	3	2
N1230:											
3	3	9	*	8	*	**5**	**5**	**5**	5	3	2
N1240:											
3	3	9	*	8	*	*	5	5	5	**2**	**2**
N1400:											
2	2	9	*	8	*	5	5	5	5	2	2
N1500:											
2	2	**8**	**8**	8	*	5	5	5	5	2	2
N1510:											
2	2	**9**	**9**	**8**	**8**	5	5	5	5	2	2
N1520 D=D_NO_SAVE(=3)											
2	2	9	9	8	8	5	5	5	5	**3**	**3**
N1600 SETMTH(MTH_SAVE)(=1)											
1	1	9	9	8	8	5	5	5	5	**3**	**3**

* Means that the value for this example is irrelevant or indeterminate.

Note

It is the responsibility of the NC programmer to check whether the specified tool still exists.

3.3.18 Program test

Overview

The "program test" function can be used to traverse a program without axis motion.

All additional data is determined and taken into account. For tool management, this means that the tools are searched and the appropriate values transferred to the PLC interface when the tool is called.

The PLC must acknowledge these tasks without moving the magazine or changing a tool. Special measures are therefore required on the PLC.

The tool management acts in exactly the same way as it would when a program is running. In the case of tools without fixed location codes and acknowledgement, this can result in the PLC data indicating different tool locations to the actual mechanical locations in the magazine. This can be prevented by configuring FC 8 such that a fixed location is simulated for the duration of the program test rather than the calculated empty location applied as a parameter.

The old location of the tool is stored in the function block which handles program testing and returned again to this location in the software (data settings). Any existing spindle tool is also returned to the spindle in the software at the end of the program test or on a reset. This ensures that the magazine assignments in the software match the mechanical assignments after program testing.

For testing programs involving manual tools, see Section "Manual tools (adding tools during machining) (Page 94)".

Example of how to adapt the PLC in test mode

The following program example can be used as a template for adapting the PLC to the program test mode. Only the first channel and a spindle are supported as tool change locations.

The tool is always changed directly into the spindle. The spindle is used as the change position (DB72). Access to the NCK/PLC interface (DB21, DB72) is symbolic. The standard UDTs (UDT 21, 72) are included for this purpose. These are part of the basic program and must be copied to the project and subsequently compiled.

The following must be entered in the symbol table:

Symbol	Address	Data type	Comment
Channel 1	DB21	UDT 21	
SpChPos	DB72	UDT 72	
TC_VAR	DB119	DB119	For testing the tool change

All necessary variables are stored in the instance data block.

If program test mode is **deselected**, no action occurs. The target positions suggested by tool management are confirmed by the PLC.

If program test mode is **selected**, the target positions are defined by the PLC. These correspond to the source positions of the respective tools. The target position is only confirmed and saved by tool management on the first tool change. It is thus possible to undo the first tool change after program test mode is selected.

Two asynchronous transfers are required for this purpose. The first one returns any tool present in the spindle to the magazine. The second asynchronous transfer is intended to return a tool which was loaded in the spindle before program test mode back into the spindle.

Note

The corresponding PLC example is stored in the toolbox. The sample file WZV_PROG.AWL is packed in file WZV_BSP:EXE.

Program test - extended

A setting can be selected with the machine data $MC_TOOL_MANAGEMENT_MASK, bit 20 such that the NCK cannot issue any tool-changing commands to the PLC in the status "Program test active". The NCK outputs its own acknowledgement so that no further data-related tool motion exists.

The default setting is that the Tool_change command is not output.

The following applies for the tool used during program test mode:

The tool status "active" can still be set and the tool status "was in use" is set. This does not have any further detrimental effects since tool monitoring is not normally active in the test mode.

If **bit 20**, value **1** is set, generated commands are output to the PLC. Tool / magazine data can be changed in the NCK depending on the type of acknowledgement by the PLC. If the acknowledgement parameters for the "target magazine" are set to the values of the "source magazine", then the tool is not moved and the data therefore not altered in the NCK.

Exception: The tool status of the tool activated in the test mode can take the status "active".

Note

It may not be derived – to the extent the setting "No tool-change commands to PLC" – that the tool on the spindle in the tool holder during "Program test active" is the active tool.

The search "SERUPRO" search (cross-channel block search) uses the program test.

3.3.19 Several spindles in one channel or TO unit

When using tool management and more than one spindle please note the following:

2 spindles in one channel

Only one tool offset can be active per channel. Spindle 1 defined as master spindle with $MC_SPIND_DEF_MASTER_SPIND = 1. Spindle 2 is a secondary spindle.

The master spindle is spindle no. 1 in each case.

2 channels each of which access the same TO memory have been set in the machine data. One spindle is assigned to each channel. Two spindles are assigned to one magazine in the magazine configuration.

The master spindle is spindle No. 1 for both channels In order to change a tool in spindle No. 2 as well, the second spindle must be defined as master spindle in the second channel before the tool is changed. In the TM system, the spindle number is sent to the PLC. This number is determined from the extended address of T. If this is not programmed, the NCK assigns the master spindle number of the channel in which the program is running.

Each channel has its own master spindle

2 channels each of which access the same TO memory have been set in the machine data. One spindle is assigned to each channel.

Two spindles are assigned to one magazine in the magazine configuration.

In each channel the assigned spindle is defined as the master spindle. Tool change is possible without making any additional definitions in the program.

3.3.20 Decoupling the tool management from the spindle number

The program must specify the location (spindle number on milling machines) at which the tool is to be changed before the tool management can insert a tool.

Machine data **MD20124 $MC_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. Thus the appropriate designation (spindle number or tool holder number) can be used in the event of use.

The following figures show which variable definitions you require for the following variants:

- Working with two spindles in one channel and one TO unit (standard functionality)
- Working with two spindles in one channel (standard functionality)
- Working with two tool holders in two channels (one TO unit)
- Working with two tool holders in one channel

Working with spindle numbers

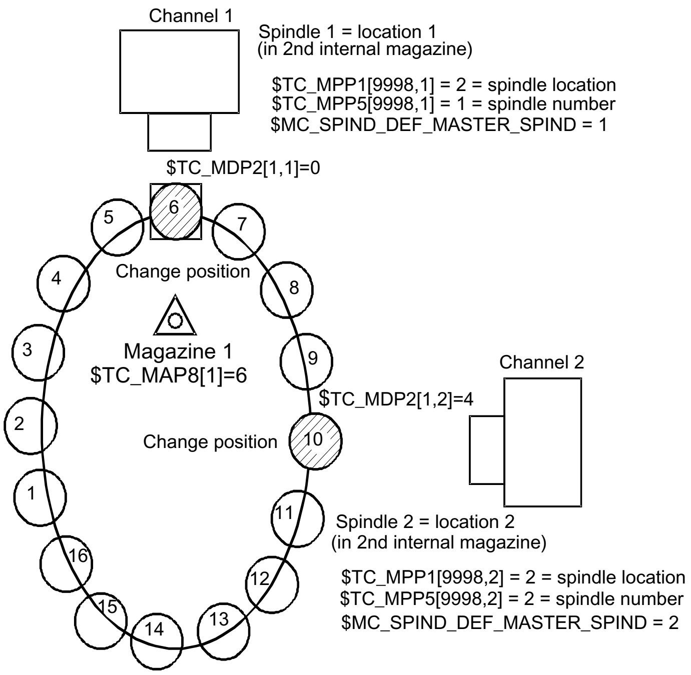

Channel 1
Spindle 1 = location 1
(in 2nd internal magazine)
$TC_MPP1[9998,1] = 2 = spindle location
$TC_MPP5[9998,1] = 1 = spindle number
$MC_SPIND_DEF_MASTER_SPIND = 1
$TC_MDP2[1,1]=0
Change position
Magazine 1
$TC_MAP8[1]=6
Channel 2
$TC_MDP2[1,2]=4
Change position
Spindle 2 = location 2
(in 2nd internal magazine)
$TC_MPP1[9998,2] = 2 = spindle location
$TC_MPP5[9998,2] = 2 = spindle number
$MC_SPIND_DEF_MASTER_SPIND = 2
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

Image 3-14 Two spindles in two channels and one TO unit

Two channels operating with the data of one TO unit (with one magazine). One spindle is defined in each channel.

Spindle 1 in channel 1 has been declared the master spindle with MD SPIND_DEF_MASTER_SPIND=1. Spindle 2 on channel 2 is the master spindle.

Both spindles must be assigned different numbers because the assignment of the spindle to the second internal magazine (buffer magazine) must be unique.

This assignment is realized by $TC_MPP1 (spindle location) and by $TC_MPP5 (spindle number).

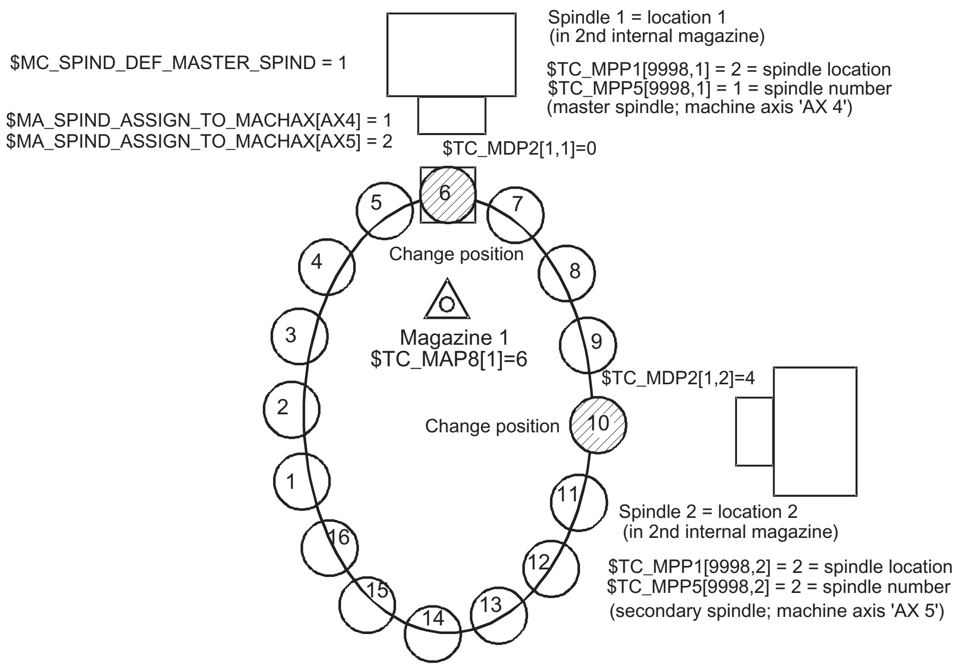

$MC_SPIND_DEF_MASTER_SPIND = 1
$MA_SPIND_ASSIGN_TO_MACHAX[AX4] = 1
$MA_SPIND_ASSIGN_TO_MACHAX[AX5] = 2
Spindle 1 = location 1
(in 2nd internal magazine)
$TC_MPP1[9998,1] = 2 = spindle location
$TC_MPP5[9998,1] = 1 = spindle number
(master spindle; machine axis 'AX 4')
$TC_MDP2[1,1]=0
Change position
Magazine 1
$TC_MAP8[1]=6
$TC_MDP2[1,2]=4
Change position
Spindle 2 = location 2
(in 2nd internal magazine)
$TC_MPP1[9998,2] = 2 = spindle location
$TC_MPP5[9998,2] = 2 = spindle number
(secondary spindle; machine axis 'AX 5')
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

Image 3-15 Two spindles in one channel

Two spindles of a single channel are operating with one magazine.

Spindle 1 was defined as master spindle with SPIND_DEF_MASTER_SPIND = 1.

Spindle 2 is not a master spindle (secondary spindle).

Example of a part program (for a channel with two spindles)

(Requirement: CUTTING_EDGE_DEFAULT=1; i.e. D1 is implicitly active with the tool change M06):

T="Miller" M06	;No address extension programmed -> the master spindle is addressed, i.e. spindle 1 = value of machine data $MC_SPIND_DEF_MASTER_SPIND. ;The tool change takes place in spindle 1. ;The **path is corrected** with the offsets of the "Miller" tool.
...	
T2="Drill" M2=6	;Address extension for the secondary spindle has been programmed. The tool change takes place in the PLC at the tool management interface for spindle 2. ;The **path is still corrected with the offsets of the "Miller" tool**
...	
SETMS(**2**)	;Declares spindle no. 2 as **master spindle**
T="Miller_2" M06	;No address extension programmed → the master spindle is addressed (spindle 2). ;The tool change takes place in spindle 2. ;The **path is now corrected** with the offsets of the "Miller_2" tool
...	

T**1**="Drill_1" M**1**=6	;Address extension for the current secondary spindle has been programmed. ;The tool change takes place in spindle 1. ;The **path is still corrected** with values from tool T="Miller_2".
...	
SETMS	;Declares the spindle defined by $MC_SPIND_DEF_MASTER_SPIND as master spindle
T="Miller_3" M06	;No address extension programmed -> the master spindle is addressed (spindle 1). ;Value of machine data $MC_SPIND_DEF_MASTER_SPIND). Tool change takes place in spindle 1. ;The **path is corrected** with the offsets of the "Miller_3" tool.

Further example (starting conditions as above):

N10 **SETMS**	;Declare spindle no. 1 as **master spindle**
N20 T**2**=3	
...	
N50 M**2**=6	;Address extension for the secondary spindle has been programmed. Tool change takes place on the spindle. ;The **path is corrected with the offsets of "Miller_3"**
...	
N70 D3	;The **path is corrected** with the offsets of the active tool (that was activated before block N10 - here in the example, "Miller_3")
N80 **SETMS**(**2**)	;Declare spindle no. 2 as **master spindle**
T3	
M06	
N90 D2	;The **path is corrected** with the offset of tool "T3".

Note

SETMS does not change the active tool. The new master spindle definition cannot be referenced until the subsequently programmed tool change.

The following applies: Offset selection is active for the last tool loaded onto a master spindle.

Working with tool holder numbers

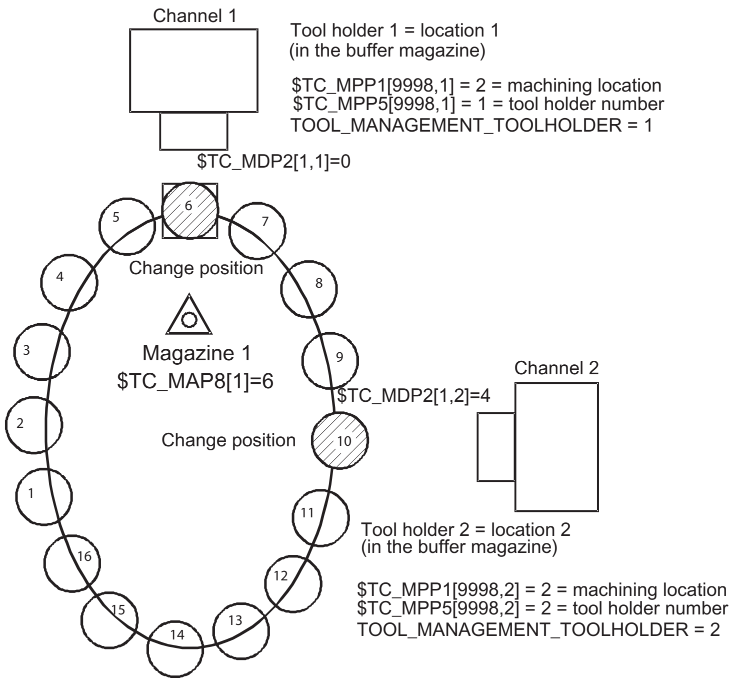

Channel 1
Tool holder 1 = location 1
(in the buffer magazine)
$TC_MPP1[9998,1] = 2 = machining location
$TC_MPP5[9998,1] = 1 = tool holder number
TOOL_MANAGEMENT_TOOLHOLDER = 1
$TC_MDP2[1,1]=0
Change position
Magazine 1
$TC_MAP8[1]=6
Change position
Channel 2
$TC_MDP2[1,2]=4
Tool holder 2 = location 2
(in the buffer magazine)
$TC_MPP1[9998,2] = 2 = machining location
$TC_MPP5[9998,2] = 2 = tool holder number
TOOL_MANAGEMENT_TOOLHOLDER = 2

Image 3-16 Two channels each with one tool holder and one TO unit

(The zero position is located at the change position of the tool holder 1)

Two channels operating with the data of one TO unit (with one magazine). Tool change no longer requires that a spindle number be specified. The address extensions of T and M now refer to the value of machine data MD20124 $MC_TOOL_MANAGEMENT_TOOLHOLDER.

Instead of "spindle location" the general term "tool holder" (tool machining location) is used (spindle is standard). If an address extension is not programmed, the address extension is supplemented by the value of MD20124 TOOL_MANAGEMENT_TOOLHOLDER.

$MC_TOOL_MANAGEMENT_TOOLHOLDER = 0

The previous function remains active (default).

A value greater than zero activates the new function.

$MC_TOOL_MANAGEMENT_TOOLHOLDER > 0

If a tool change is programmed to a buffer location of the type "Tool holder" with $TC_MPP5 = TOOL_MANAGEMENT_TOOLHOLDER, then the offset data defined for this tool corrects the path.

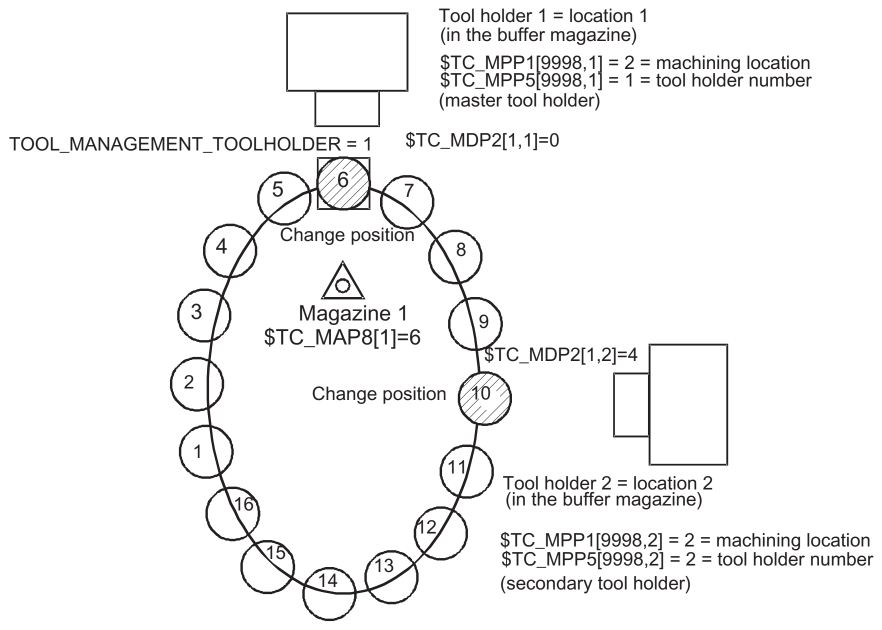

Tool holder 1 = location 1
(in the buffer magazine)
$TC_MPP1[9998,1] = 2 = machining location
$TC_MPP5[9998,1] = 1 = tool holder number
(master tool holder)
TOOL_MANAGEMENT_TOOLHOLDER = 1
$TC_MDP2[1,1]=0
Change position
Magazine 1
$TC_MAP8[1]=6
$TC_MDP2[1,2]=4
Change position
Tool holder 2 = location 2
(in the buffer magazine)
$TC_MPP1[9998,2] = 2 = machining location
$TC_MPP5[9998,2] = 2 = tool holder number
(secondary tool holder)

Image 3-17 One channel with two tool holders

(The zero position is located at the change position of the tool holder 1)

Two tool holders in a channel are operating with one magazine. Tool holder 1 has been declared the master using **TOOL_MANAGEMENT_TOOLHOLDER = 1**. Tool holder 2 is thus the secondary tool holder.

Programming example:

The language command SETMTH (tool holder number) is used to declare one or several tool holders as the master tool holder.

The initial state after power on is defined using machine data $MC_TOOL_MANAGEMENT_TOOLHOLDER; the behavior at NC start and reset using the machine data $MC_START_MODE_MASK and $MC_RESET_MODE_MASK.

SETMTH (tool holder number),

T="Miller" M06	;No address extension programmed -> the master tool holder is addressed (tool holder 1 - value of machine data $MC_TOOL_MANAGEMENT_TOOL-HOLDER). ;The tool change is performed at buffer location 1. ;The **path is corrected** with the offsets of the "Miller" tool.
...	
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)	;Declares tool holder 2 as the **master tool holder**
T="Miller_2" M06	;No address extension programmed -> the master tool holder is addressed (tool holder 2). ;The tool change is performed at buffer location 2. ;The path is corrected with the offsets of the "Miller_3" tool.
...	
T1="Drill_1" M1=6	;Address extension for the secondary tool holder has been programmed. ;The tool change is performed at buffer location 1. ;The **path is not corrected**!
SETMTH	;Declares the tool holder specified in $MC_TOOL_MANAGEMENT_TOOL-HOLDER as the master tool holder
T="Miller_3" M06	;No address extension programmed -> the master tool holder is addressed (tool holder 1 - value of machine data $MC_TOOL_MANAGEMENT_TOOL-HOLDER). ;The tool change is performed at buffer location 1. ;The path **is corrected** with the offsets of the "Miller_3" tool

Note

SETMTH does not change the active tool. The new master tool holder definition cannot be referenced until the subsequently programmed tool change.

The following applies: Offset selection is active for the last tool changed into the active tool holder.

3.3.21 Several spindles/toolholders

Overview

Tool management can work in one channel with more than one tool holder. If several channels of one TO unit are supplied with data, then make sure that the tool holder numbers have different (= unique) numbers in the magazine configuration ($TC_MPP5 of buffer locations of the type ($TC_MPP1) "Spindle"). The spindle numbers of the channels must then be unique as well (if $MC_TOOL_MANAGEMENT_TOOLHOLDER=0).

Example

This example shows how to differentiate between an active tool and a programmed tool.

Spindles 1...4 are defined in channel 1; analog to this, spindle locations 1...4 in magazine configuration 4.

```
SETMS (2)
T12                     ;12 is a programmed tool
M06 D3                  ;12 is an active tool, 3 is an active cutting edge
SETMS (4)
```

```
T22                  ;12 remains an active tool, 22 becomes the programmed tool with
                     respect to spindle=4
T3=33 M3=6           ;T33 is loaded onto the secondary tool holder 3. The tool (T33)
                     is not active.
SETMS (1)            ;Tool holder=1 becomes master spindle, T12 remains active, T22
                     remains programmed
D5                   ;D5=active cutting edge; refers to the active tool, i.e. T12
M00
```

The following situation is given:

Tool holder number	T number	D number
1 master spindle	-	-
2	12 active	5 active
3	33	-
4	22 programmed	-

3.3.22 Several magazines in one channel or one TO unit

Address extension

The NC address T can be programmed with an address extension. The tool management function interprets the programmed address extension as a spindle number or toolholder number. The NC address T without programmed address extension then refers to the main spindle (master spindle).

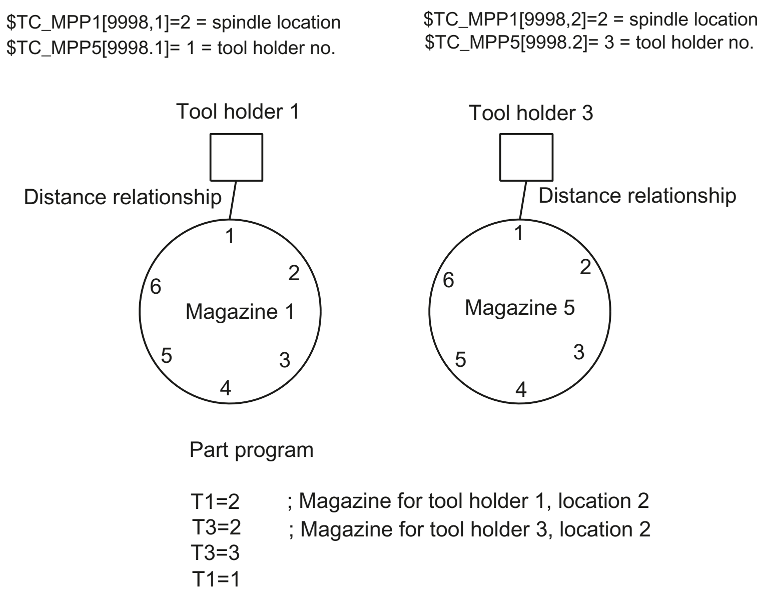

$TC_MPP1[9998,1]=2 = spindle location
$TC_MPP5[9998.1]= 1 = tool holder no.
$TC_MPP1[9998,2]=2 = spindle location
$TC_MPP5[9998.2]= 3 = tool holder no.
Tool holder 1
Tool holder 3
Distance relationship
Distance relationship
Magazine 1
Magazine 5
1
2
3
4
5
6
Part program
T1=2 ; Magazine for tool holder 1, location 2
T3=2 ; Magazine for tool holder 3, location 2
T3=3
T1=1

Image 3-18 T="location" and several magazines in the same channel

The diagram shows the procedure for using more than one magazine in a channel (when programming with T="location" this is usually a turret).

Note

The tool offset is only calculated for the toolholder that is assigned at that point in time to the master spindle or the master toolholder.

3.3.23 Reset and start mode

Fundamentals

The tool offset selection/deselection can be set in the machine data for program end or reset as well as for NC Start.

It is also possible to permanently preset the change for a specific tool, e.g. at NC Start.

The settings are made in the following machine data:

- MD20310 $MC_TOOL_MANAGEMENT_MASK
- MD20110 $MC_RESET_MODE_MASK
- MD20112 $MC_START_MODE_MASK
- MD20122 $MC_TOOL_RESET_NAME
- MD20130 $MC_CUTTING_EDGE_RESET_VALUE

The function and interaction of the machine data are displayed in the following diagram.

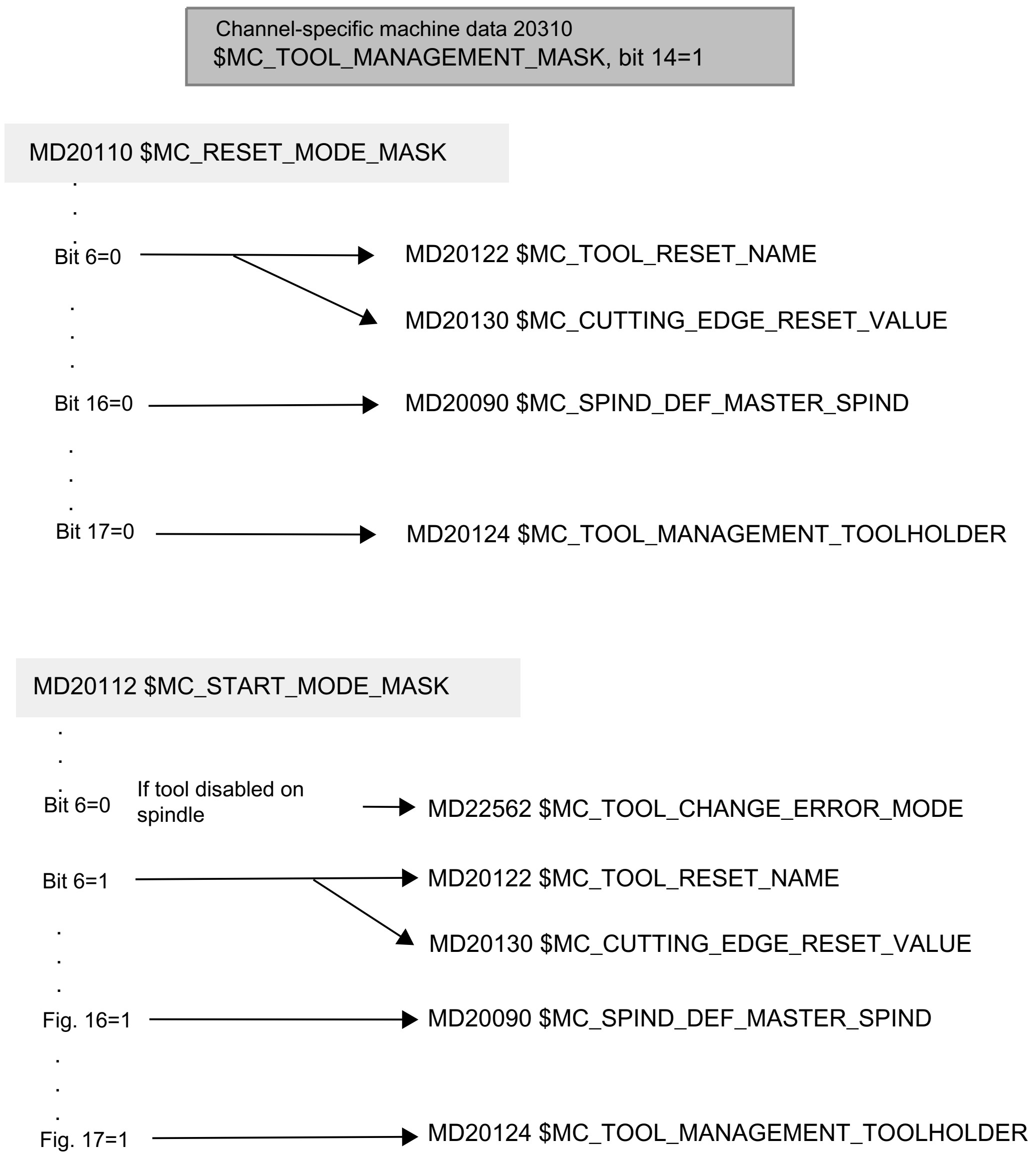

Channel-specific machine data 20310
$MC_TOOL_MANAGEMENT_MASK, bit 14=1
MD20110 $MC_RESET_MODE_MASK
Bit 6=0
MD20122 $MC_TOOL_RESET_NAME
MD20130 $MC_CUTTING_EDGE_RESET_VALUE
Bit 16=0
MD20090 $MC_SPIND_DEF_MASTER_SPIND
Bit 17=0
MD20124 $MC_TOOL_MANAGEMENT_TOOLHOLDER
MD20112 $MC_START_MODE_MASK
Bit 6=0
If tool disabled on spindle
MD22562 $MC_TOOL_CHANGE_ERROR_MODE
Bit 6=1
MD20122 $MC_TOOL_RESET_NAME
MD20130 $MC_CUTTING_EDGE_RESET_VALUE
Fig. 16=1
MD20090 $MC_SPIND_DEF_MASTER_SPIND
Fig. 17=1
MD20124 $MC_TOOL_MANAGEMENT_TOOLHOLDER

Image 3-19 Reset and start mode

MD20110 $MC_RESET_MODE_MASK

Bit	Value	Meaning
0	0	The offset remains unchanged, i.e. after end of part program and reset, the last programmed offset remains active (behavior as with bit 0=1 and 6=1).
	1	Reset mode, i.e. evaluation of bits 4 ..11
2	1	Reset behavior (tool offset) with tool management not active. No effect if tool management active

Bit	Value	Meaning
6	0	Reset behavior corresponds to MD $MC_TOOL_RESET_NAME and $MC_CUTTING_EDGE_RESET_VALUE
	1	Current setting for active tool length compensation is retained beyond reset/end of part program. For active tool management, the tool that is located in the master spindle (general: master tool holder) is selected. If the tool in the spindle is disabled, this state is ignored, there is no selection of a replacement tool! (Replacement tool with Start_INIT only). Activation takes place on the master spindle defined in MD $MC_SPIND_DEF_MASTER_SPIND, or on the master tool holder defined in $TC_TOOL_MANAGEMENT_TOOLHOLDER. The tool in the last programmed master spindle or master tool holder can also be active. Bit 16 or 17 is used for this purpose.

Reset behavior for spindles

Bit	Value	Meaning
16	0	The master spindle is the spindle defined in MD $MC_SPIND_DEF_MASTER_SPIND. The settings in the machine data below refer to this data: • $MC_TOOL_MANAGEMENT_MASK • $MC_RESET_MODE_MASK • $MC_START_MODE_MASK • $MC_TOOL_RESET_NAME • $MC_CUTTING_EDGE_RESET_VALUE
	1	The spindle last programmed with SETMS(x) remains the master spindle after end of program and reset, regardless of the machine data setting. This means that if bits 0/6=1, the offset remains active for the tool which is placed in the spindle. Power on behavior The machine data setting is active after power on. This means the offset for the tool which is placed in the spindles specified in MD $TC_SPIND_DEF_MASTER_SPIND becomes active; the offset value is that of the smallest available D no. for this tool.

Reset behavior for tool holder

Bit	Value	Meaning
17	0	The master tool holder is the tool holder specified in MD $TC_TOOL_MANAGEMENT_TOOL-HOLDER. The settings in the machine data below refer to this data: • $MC_TOOL_MANAGEMENT_MASK • $MC_RESET_MODE_MASK • $MC_START_MODE_MASK • $MC_TOOL_RESET_NAME • $MC_CUTTING_EDGE_RESET_VALUE
	1	The tool holder last programmed with SETMTH(x) remains the master tool holder after end of program and reset, regardless of the machine data setting. This means that if bit 0=1 and bit 6=1, the offset remains active for the tool which is positioned in this tool holder. The machine data setting is active after power on. This means the offset for the tool which is positioned in the tool holder specified in $MC_TOOL_MANGEMENT_TOOLHOLDER becomes active; the offset value is that of the smallest available D no. for this tool.

MD22562 $MC_TOOL_CHANGE_ERROR_MODE

Bit	Value	Meaning
3	0	Change command for a replacement tool is output.
	1	The disabled status of the spindle tool is ignored. The tool becomes active with the last programmed offset.
4	0	Change command for a replacement tool is output.
	1	The spindle tool is set down - "T0" is output.

MD20122 $MC_TOOL_RESET_NAME

Identifier of tool to be loaded

This tool is either loaded when end of program is reached or at reset or Power On if the associated setting is made in MD $MC_RESET_MODE_MASK, or with NC Start if the respective setting was made in MD $MC_START_MODE_MASK.

If there are no entries here ($MC_TOOL_RESET_NAME="") this corresponds to "T0".

MD20130 $MC_CUTTING_EDGE_RESET_VALUE

D number of tool which is to be loaded via $MC_TOOL_RESET_NAME.

This means the tool becomes active with the offset set here.

If no entries are made in this machine data, the behavior corresponds to "D0".

MD20124 $MC_TOOL_MANAGEMENT_TOOLHOLDER

Specifies whether a tool holder number or spindle number is to be specified to define the location of the tool to be loaded.

MD20090 $MC_SPIND_DEF_MASTER_SPIND

Definition of master spindle in channel. The number of the spindle is set.

MD20310 $MC_TOOL_MANAGEMENT_MASK

Bit 14 is used to activate the reset and start behavior. If bit 14 is not set, the settings in machine data $MC_RESET_MODE_MASK and $MC_START_MODE_MASK which are specific to tool management have no meaning.

MD20112 MC_START_MODE_MASK

Bit	Value	Meaning
6	0	The last programmed offset remains active. If the tool is disabled on the spindle, bits 3 and 4 are also evaluated in MD $MC_TOOL_CHANGE_ERROR_MODE.
	1	Start behavior (tool and offset selection) according to MD $MC_TOOL_RESET_NAME and $MC_CUTTING_EDGE_RESET_VALUE

Start behavior for spindles

Bit	Value	Meaning
16	0	The offset that was last selected remains active. It does not matter whether the offset was selected in the part program or via settings in MD $MC_RESET_MODE_MASK. The offset for the tool placed in the master tool holder that was last programmed can also be active (see $MC_RESET_MODE_MASK)
	1	The tool holder specified in MD $MC_Tool_Management_Toolholder becomes active. This means an offset selection refers specifically to this tool holder.

Start behavior for tool holder

Bit	Value	Meaning
17	0	The offset that was last selected remains active. It does not matter whether the offset was selected in the part program or via settings in MD $MC_RESET_MODE_MASK. The offset for the tool placed in the master spindle that was last programmed can also be active (see $MC_RESET_MODE_MASK).
	1	The spindle defined in MD20090 $MC_SPIND_DEF_MASTER_SPIND becomes active. This means an offset selection refers specifically to this spindle.

Note

A power on always triggers a reset ⇒ The settings in MD20110 $MC_RESET_MODE_MASK are active after a power on.

Example 1:

In this example, the tool on the spindle is to remain active after end of program (M02/M30) and reset.

The following applies:

- $MC_TOOL_CHANGE_MODE = 1
- $MC_CUTTING_EDGE_DEFAULT = -2

The following settings need to be made:

$MC_TOOL_MANAGEMENT_MASK	**Bit 14=1**	for reset and start behavior to be active
$MC_RESET_MODE_MASK	**Bit 0=1,** **Bit 6=1**	for the tool offset to remain active
$MC_START_MODE_MASK	**Bit 6=0**	for the tool offset to remain active

NC program: %MPFxxx1

```
N110 T="MILLER_10"
N115 M06                      ;Tool "MILLER_10" is loaded
N120 G90 G00 D2 X...          ;Offset D2 becomes active
…
N850 M30                      ;Offset D2 remains active
```

At the next program start tool "MILLER_10" is active with offset D2.

NC program: %MPFxxx2

```
N10 G90 G00 Z100              ;This block is executed with offset D2
```

Example 2:

The spindle tool is to be removed at the end of program and reset ("automatic T0").

The following applies:

- $MC_TOOL_CHANGE_MODE = 1
- $MC_CUTTING_EDGE_DEFAULT = -2
- One spindle

The following settings need to be made:

$MC_TOOL_MANAGEMENT_MASK**.bit 14 = 1**	Reset and start behavior is activated
$MC_RESET_MODE_MASK**.bit 0 = 1 and bit 6 = 0**	Reset behavior: • TOOL_RESET_NAME • CUTTING_EDGE_RESET_VALUE
$MC_TOOL_RESET_NAME**=""**	Name of the tool that should be loaded at change with reset. If no name is entered here, this has the same meaning as T0
$MC_CUTTING_EDGE_RESET_VALUE **= 0**	The above mentioned tool becomes active with this offset ("0" ≙ D0)
$MC_START_MODE_MASK**.bit 6=0**	The tool offset remains active. In the example: D0

Example 3:

In this example, a specific tool is to be loaded at NC start, e.g. a probe.

The following applies:

- $MC_TOOL_CHANGE_MODE = 1
- $MC_CUTTING_EDGE_DEFAULT = -2
- One spindle

The following settings need to be made:

$MC_TOOL_MANAGEMENT_MASK **Bit 14=1**	Reset and start behavior is activated
$MC_START_MODE_MASK **Bit 6=1**	Start behavior • TOOL_RESET_NAME • CUTTING_EDGE_RESET_VALUE
$MC_TOOL_RESET_NAME**="Probe_1"**	Name of the tool that was loaded with reset/ start. In the example: "Probe_1"
$MC_CUTTING_EDGE_RESET_VALUE**=1**	The above mentioned tool becomes active with this offset, here D1
$MC_RESET_MODE_MASK **Bit 6=0**	Is not relevant for this example

Example 4:

In this example, the tool on the master spindle that was last programmed is to remain active following end of program (M30/M02) and reset.

The following applies:

- $MC_TOOL_CHANGE_MODE = 1
- $MC_CUTTING_EDGE_DEFAULT = -2
- Two spindles
- $MC_SPIND_DEF_MASTERSPIND=1

The following settings need to be made:

$MC_TOOL_MANAGEMENT_MASK.**bit 14=1**	Reset and start behavior is activated
$MC_RESET_MODE_MASK.**bit 0=1** and **bit 6=1**	Leave the tool offset active
$MC_RESET_MODE_MASK.**bit 16=1**	The last programmed master spindle remains active.
$MC_START_MODE_MASK.**bit 6=0**	The current offset remains active.

NC program

Program code	Comment
`N05 SETMS(1)`	`;Spindle becomes master spindle (is also set via MD)`
`N10 T="Tool1"`	
`N15 M06`	`;Change to spindle 1`
`...`	
`N80 SETMS(2)`	`;Spindle 2 becomes master spindle`
`N85 T="Tool2"`	
`N90 M06`	`;Change to spindle 2`
`N95 G90 G00 D2 Z...`	
`...`	
`N230 M30`	`;It is active: Tool2 with offset D2 on spindle 2`

Note

Using MD20310, bit 12 = 1 and the described settings, for each reset a preparation and change command is output to the PLC and must be acknowledged by the PLC. The automatic positive acknowledgement can also be used here.

Note

If a change is triggered by reset mode at POWER ON, the NC is stationary with "No NC ready" until an end acknowledgment has been received for this change.

3.3.24 Repeating a tool change with the same tool identifier

Overview

The behavior for repeated tool changes with identical tool identifier is set using two bits of MD 20310 $MC_TOOLMANAGEMENT_MASK.

Bit 11 = 1	The tool preparation command is also output, if it was already output for the tool. This setting is only effective as long as the prepared tool is still not located on the requesting spindle or tool holder.
Bit 12 = 1	The tool preparation command is also output if the tool is already located in the spindle / tool holder, however it is only issued just one more time.

The default setting (bit 11 and bit 12=0) is selected so that the preparation command is not executed if the tool is already located in the spindle / tool holder – or the same preparation was programmed a multiple number of times.

Exception: Block search

Here the preparation command is always issued even if the tool is already positioned in the spindle.

Repeating the T preparation before loading

	Bit 11 = 0	Bit 11 = 1
`N10 T="Tool1"`	Tool preparation command is output to the PLC	Tool preparation command is output to the PLC
...		
`N20 T="Tool1"`	No output to the PLC (assuming that the state of the tool is unchanged)	Tool preparation command is output to the PLC
`N22 M06`	Change command to the PLC	Change command to the PLC

A typical application is providing a new tool (positioning a chain) and checking before the change using the repeated T call.

New programming for the tool that is still able to be used on the tool holder

Example 1:

	Bit 12 = 0	Bit 12 = 1
`N10 T="Tool1"`	Tool preparation command to the PLC	Tool preparation command to the PLC
`N12 M06`	Tool change command to the PLC	Tool change command to the PLC
`N20 T="Tool2"`	Tool preparation command to the PLC	Tool preparation command to the PLC
`N30 "T=Tool1"`	No output to the PLC. This preparation replaces the preparation from block N20; it is identified that a tool, which can be used from group "Tool1", has already been loaded.	Tool preparation command to the PLC. This preparation replaces (as for bit 12 = 0) the preceding T preparation from N20. The checks on the NCK side are identical. However, as a result of the machine data setting, command output is forced.
`N32 M06`	No change command to the PLC	No change command to the PLC

The preparation command from block N30 was deleted in the NCK; in the programming, it appears as if N10, N12 and N32 are programmed. As the state of the tool "Tool1" on the tool holder has not changed, M06 is not output to the PLC.

Example 2:

	Bit 12 = 0	Bit 12 = 1
`N10 T="Tool1"`	Tool preparation command to the PLC	Tool preparation command to the PLC
`N12 M06`	Tool change command to the PLC	Tool change command to the PLC
`N20 T="Tool1"`	No output to the PLC	Tool preparation command to the PLC

`N30 "T=Tool1"`	No output to the PLC	No output to the PLC (the preparation for the same tool is only repeated once)
`N32 M06`	No change command to the PLC	No change command to the PLC

The preparation command from block N20 was deleted in the NCK by the new T programming in block N30. In the programming, it appears as if N10, N12 and N32 were programmed. As the state of the tool "Tool1" on the tool holder has not changed, M06 is not output to the PLC.

For these two examples, bit 11 has no significance.

New programming for the tool that is still able to be used on the tool holder

`N10 T = "Tool1"`	
`N12 M06`	
`N20 T = "Tool1"`	;Preparation command is output
`N30 T = "Tool1"`	;No command output to the PLC
`N32 M06`	;Change and preparation are output together

N20, N30 and N32 are not output to the PLC.

New programming for the tool that can no longer be used on the tool holder

The time monitoring has, for example, set the tool to the "disabled" state.

`N10 T = "Tool1"`	;Tool preparation command to PLC
N12 M06	;Tool change command to PLC
`N20 T = "Tool2"`	Tool preparation command to PLC
`N30 T = "Tool1"`	;This tool preparation replaces the tool preparation from N20; tool management detects that a tool from group "Tool1" is loaded but that the tool can no longer be used. A replacement tool is searched for in the tool group and the tool preparation command is output to the PLC.
`N32 M06`	;The tool change command N32 is output to the PLC.

Condition for processing a new tool preparation command in the NCK

`N10 T = "Tool1"`	
`N20 T = "Tool2"`	;A command is only processed in the main run if the preceding command from the PLC was acknowledged with "End".

The does not apply if N20 is not output to the PLC. Then the "End" acknowledgement must be present for a new tool preparation command to be output to the PLC.

Condition for processing a new tool change command in the NCK

`N10 T = "Tool1"`	
`N12 M06`	;A command is also processed in the main run if the preceding command from the PLC was not yet acknowledged with "End".

3.4 Search for tool

3.4.1 Strategies for tool searches

The tool search is initiated by the preparation command (T selection). The search begins for a tool to load in the spindle.

In the default setting, tool searches are always performed on a magazine-specific basis, i.e. with this setting for the search strategy, the search is performed in the magazine from which the last change was carried out.

Tool search

The tools with the same identifier (name or Ident) but different duplo numbers are combined to form one tool group. The tool identifier is programmed in the part program with the NC address, i.e. only the tool group is specified during preparation.

In order to move a tool from a physical magazine to a spindle it must have the following characteristics:

- Tool status must be "enabled"
- Tool status must not be "disabled" (exceptions: Language command TCA as well as PLC interface signal "Tool disable inactive")
- Tool status may not be "currently being changed"
- Tool must not already be used by a spindle other than the requesting spindle.
- Tool must be present in the magazine location (except for manual tools)
- This magazine must be linked to the requesting spindle via a distance relationship ($TC_MDP2)
- This magazine must not have the status "disabled".

The explicit tool is requested at the time of the tool call. The request is made for a special spindle (general tool holder); this is the number of the address extension of T. At this point in time, user interface DB72 is written for the relevant spindle and must be evaluated by the PLC user program.

The tool search strategy is defined using the system variable **$TC_MAMP2** for the TO area, with the system variable $TC_MAP10[Mag_No] magazine-specific. With **bit 0** to **bit 2**, a search is made according to the criteria - active tool, lowest duplo No. and the shortest distance. The 'search criterion' for tool wear uses bit 3 to bit 5. Bit 7 is used to set whether the tool search is made on a magazine-for-magazine basis or across all linked magazines.

Setting bit 7=1 means that the search strategies defined using bits 0, 1, 2 start with the search from the 1st magazine of the distance table (the sequence in the distance table is defined using the programming sequence of $TC_MDP2). The standard setting is **bit 7=0**. The search starts in the magazine from which the previous tool that was loaded was taken.

Setting bit 6 means that a search is first made in the magazine currently being considered. This is only effective in conjunction with bit 7=1. The strategies, where the search always starts in the first magazine of the distance table are described in Section "Magazine blocks (Page 253)".

Note

Bit 3 = 1 to **bit 5 = 1** are only significant when the monitoring function is active (defined by $TC_TP9). Otherwise they have no effect on the suitability check.

Note

The tool search strategies apply equally for all magazine types (chain, box-type and circular) but not for the buffer magazine.

Example, tool search routine

A tool change at a spindle shall take place.

The search sequence for the correct tool is as follows:

1. The control checks whether the tool which is called is already located on the spindle.
2. If buffer locations are linked to the spindle (see $TC_MSLR), the control checks whether a suitable tool is already located in one of these.
3. The tool search starts in the 1st magazine of the distance table ($TC_MDP2) according to the selected search strategy.
(Applies only if bit 7 of $TC_MAMP2 = 1; otherwise, the search starts in the magazine from which the last loaded tool was fetched.)
4. If no tool is found in the first magazine, the search is repeated in the next magazine of the distance table.
5. If all the magazines that are linked to the spindle have been searched and no suitable tool found, the search is terminated with an alarm (22069 or 22068).
6. If the function "manual tool" is active, after an unsuccessful search in the magazines - a search is made in the TO unit for an unloaded tool, but a tool that can also be used. If none is found, the search is terminated with an alarm.

Any suitable tool with the programmed identifier found (not disabled) in one of the stages described above will be used.

3.5 Empty location search

3.5.1 Empty location search for a tool – from spindle to magazine

With the T preparation command, a matching empty location is automatically searched for the spindle tool. The location in which the new tool is stored is still occupied at this time and **cannot** therefore be identified as an empty location, the 1:1 interchange is the exception.

Note

Generally, a search is made for an empty location in that magazine from which the current tool in the toolholder was taken.

Fixed location coding

When searching for an empty location for fixed-location coded tools its previous location in the magazine is usually retained.

If the search for an empty location for a fixed-location-coded tool is started with a specific magazine number, that number is ignored. The old tool location is defined as an empty location.

If this number is however an internal magazine number (for a loading or buffer magazine), then the number is explicitly taken into consideration and the fixed location coding is ignored. This case arises when loading/unloading tools.

If a location search for a fixed location coded tool is initiated using a specific magazine number and magazine location number, the fixed location coding is ignored and the specified location checked as a suitable location for the tool. This is used in the HMI function "Relocating".

Variable location coding

Initially, the procedure for an empty location search is the same as that for a fixed location-coded tool. If this check fails, the search for a free location is continued. The search is performed according to the selected search strategy ($TC_MAMP2 / $TC_MAP10). If the search cannot find an available location with the specified location type in this magazine, a new search operation based on the location type hierarchy is started in the magazine. A location is only then considered as a suitable type of location when it applies that "Location type of the location" is larger than "Location type of the tool", whereby the "larger than" relationship is defined by the location-type hierarchy. If no free location is found in this magazine, the search is continued in the next magazine (search strategy).

3.5.2 Search strategy for empty locations

Search strategy

The strategy can be defined with the magazine configuration according to which the search is made in magazines of TO units for an empty location. If it involves a box-type magazine, then the search is executed according to the default strategy (forwards search starting at the first location number).

Possible strategies are listed in the table.

$TC_MAMP2	Search strategies	Meaning
Bit 8 = 1 256	Forwards search	The search takes place from location No. 1 in ascending order.
Bit 9 = 1 512	Forwards search	The search takes place from the current location at the change position in ascending order.
Bit 10 =1 1024	Backwards search	The search takes place backwards starting from the last location No.
Bit 11 = 1 2048	Backwards search	The search takes place from the current location at the change position backwards.
Bit 12 = 1 4096	Symmetrical search	The search starts at the current location number at the change position (1st location left, 1st location right, 2nd location left, 2nd location right, etc.).
Bit 13 = 1	1:1 exchange	If the location type and size of the old and new tool are the same, the magazine location of the "new" tool to be loaded is transferred to the "old" tool to be unloaded – and vice versa. The 1:1 interchange acts in addition to other selected search strategies. If possible, the 1:1 exchange is treated as a priority.
Bit 14	Magazine search strategy	=0 First search the magazine with a search run across the location types of the hierarchy of the tool type. If no location is found, then a search run is performed across the magazines. =1 Search the location type of the tool in the magazine. If nothing is found, search run across the magazines. If a location is still not found, then a search run is repeated across all magazines with the next location type from the type hierarchy.

Note

The empty location search strategy is set TO-specific using $TC_MAMP2, magazine-specific using $TC_MAP10. If the magazine-specific search is not set, when the NCK boots, it copies the value from MAMP2 to MAP10.

Definition of the current magazine position

The current magazine position at the change location is stored in magazine parameter (system variable) $TC_MAP8. The value is automatically updated by the PLC acknowledgement of a command, if the new tool is moved. If the magazine or tool is moved without a task from the NCK, the user must correct the actual position. An NC cycle or also the PLC can write to parameter $TC_MAP8. Either by writing the OPI variable (selected with NCVAR selector block TM; variable magNrPaces) or with the FC 8/FC 6 (with the parameters TaskIdent=4 TasIIdentNo=channel number, status=5, OldToolMag=9998,OldToolLoc=1). The current position is parameterized (referred to spindle) in NewToolMag and NewToolLocspindle.

3.5.3 Assigning a hierarchy to the location types

If a tool is to be loaded from the spindle into the magazine or a tool is to be loaded into the magazine, then the location type decides which magazine locations are available for selection, i.e. $TC_TP7 and $TC_MPP2 must be defined and match or $TC_MPP2=0, i.e. every tool type fits in this location.

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

Example:

A tool with location type B is to be placed in the magazine or a search made for a free location with type B.

The following location type hierarchy applies: A < B < C < D

The location type hierarchy is set centrally for the entire TO unit in parameter $TC_MAMP2 with bit 14.

The search is set in $TC_MAMP2 or in $TC_MAP10[Magazine-No.] with bit 14.

Procedure:

Bit 14=0

First a check is performed to see whether there is a location with type B in the magazine to be searched. If there is not, the search is continued for a type C or D location in the same magazine. In doing so, only the validity of the location types is tested. If the empty location is of type C, then the search condition is fulfilled, also if the empty location is of type D. This means that no search is made for the next largest location type. If a location is not found, the search is continued in the next, connected magazine.

Bit 14=1

First a check is performed to see whether there is a location with type B in the magazine to be searched. If not, then a search is made for precisely this location type in all of the linked magazines. If a location is still not found, then a search run is made across all magazines with the next larger location type from the type hierarchy.

Several such hierarchies can be set up in one TO unit. A location type may only be entered into one hierarchy.

Example: def int A,B,C,D,E

The following location type relationships are defined:

First hierarchy

$TC_MPTH[0.0] = A
$TC_MPTH[0.1] = B
$TC_MPTH[0.2] = D

Second hierarchy

$TC_MPTH[1.0] = C
$TC_MPTH[1.1] = E

Type A < type B < type D apply in the first hierarchy and type C < type E in the second hierarchy. In the first hierachy, location type A is the smallest (and fits on all other location types), location type D is the largest.

Both the number of different hierarchies as well as the number of entries in a hierarchy can be set using a machine data (18078 $MN_MM_MAX_NUM_OF_HIERARCHIE, 18079 $MN_MM_MAX_HIERARCHIE_ENTRIES).

3.5.4 New type of location type hierarchy

It is now possible to define location types from 1 to 8. A table showing possible alternative location types can be added to each of these location types. The location types used in the tables can have values from 1 to 32000. This empty location search is set centrally for the entire TO unit by setting bit 15 in $TC_MAMP2.

The same location types can be entered into various tables.

A location type hierarchy can comprise a maximum of 8 additional location types. The location type hierarchy is defined using the already described system parameter $TC_MPTH[n,m] with n, m = 0,....7. A definition is made that the location type hierarchy of the designated location type 1 is defined with $TC_MPTH[0,m]. In the NCK, it is not monitored which location types are entered into the location type hierarchy.

Boundary conditions

This type of hierarchy formation of magazine location types is valid for the TO unit in which the system parameter $TC_MAMP2 is correspondingly set.

Note

If the location type hierarchies ($TC_MPTH) are defined before the new hierarchy is set in $TC_MAMP2 with bit 15 = 1, this results in:

a. Deletion of the hierarchies ($TC_MPTH[x,y]=9999)

b. Alarm 17255 "Magazine location hierarchies have been deleted

Examples

There are three tools with the T and location type numbers = 100 and 1, 200 and 2, 600 and 6. The following definitions are made.

```
$TC_MPTH[ 0, 0 ] = 10        ; hierarchy of the designated location type = 1
$TC_MPTH[ 0, 1 ] = 21
$TC_MPTH[ 0, 2 ] = 32
$TC_MPTH[ 0, 3 ] = 43
$TC_MPTH[ 0, 4 ] = 54
$TC_MPTH[ 0, 5 ] = 65
$TC_MPTH[ 0, 6 ] = 76
$TC_MPTH[ 0, 7 ] = 87

$TC_MPTH[ 1, 0 ] = 1         ; hierarchy of the designated location type = 2
$TC_MPTH[ 1, 1 ] = 10
$TC_MPTH[ 1, 2 ] = 111

$TC_MPTH[ 5, 0 ] = 1         ; hierarchy of the designated location type = 6
$TC_MPTH[ 5, 1 ] = 10
$TC_MPTH[ 5, 2 ] = 111
$TC_MPTH[ 5, 3 ] = 222
```

The location types 1, 10 and 111 are included in several hierarchies.

A search is now made for an empty location for tool with T No. = 200 and location type No. = 2. If no type 2 location is empty, then the search is continued by evaluating the type hierarchy $TC_MPTH[1, m].

T No. = 200 also fits into a location, type 1, 10 or 111.

T No. = 600 with location type No. = 6 - in addition to the preferred location with type = 6, it also fits in location types 1, 10, 111 or 222.

On the other hand, T No. = 100 with location type number = 1 does not fit in locations of type = 111 or 222, but only in locations of the preferred type 1, or 10, 21, 32,.... 87.

System variables

AUTOHOTSPOT

3.5.5 Search procedure for empty locations

The following apply as criteria for the empty location search:

- Location type must coincide with location type of tool. A hierarchy is taken into account.
- Check the tool size.
- Location must have the status "free".

- Location must not be "disabled".
- Magazine must not be "disabled".

The essential search criterion for the empty location search is the magazine location type. The magazine location type must match the magazine location type entered in the tool-specific data ($TC_TP). The magazine is searched. Each location is checked. If a suitable location is found the search is terminated.

If a matching location is not found, then a check is made whether there is a magazine location type hierarchy for the magazine-location type that is entered in the tool. If there is none, the next magazine is taken if there are further magazines available. If there is a defined hierarchy, then the search routine is repeated starting at the magazine that has just been searched. If this search is also unsuccessful, the search moves to the next magazine, assuming another one is available.

Note

With oversized tools, the location types of the adjacent location are not considered.

3.5.6 Search strategy, 1:1 exchange (old for new)

With this search option, the magazine location of the "new" tool (tool to be loaded) is made available as the empty location for the ’old’ tool (tool to be unloaded).

It is not assumed that the "new" tool is stored in the magazine location. It only needs to have been loaded (it may be located on a gripper, for example). If the location in question is not suitable for the "old" tool, then another appropriate empty location is sought.

Description of function

The new search strategy for an empty location is preset in the already existing bit-coded system variable **$TC_MAMP2** with **bit 13**.

If 2 tools (new and old tool) are exchanged 1:1, then the two tools are marked. To do this, bit 15 is set to 1 in the tool status ($TC_TP8).

Boundary conditions

With this empty-location search strategy, the NCK checks a magazine location that at the point of time of making the check is normally identified as still occupied by the "new" tool or is still "reserved for tool from buffer location". This location is defined as an empty location for the "old" tool if the check gives a positive result.

If the new or old tool is coded as a fixed location tool or the tool size or the location type are not identical, then the strategy is not used.

Note

The PLC program has to execute the tool transportation operations in the correct sequence for the tool change:

Remove "new" tool from the magazine location

Bring the "old" tool to the magazine location

Otherwise damage may occur to the machine or tool.

The 1:1 exchange can be set as the only search strategy for the buffer magazine (9998).

The empty location search strategy is only effective within tool changes programmed in the part program. PI services or language commands for an empty location search cannot use this.

Example

This strategy is especially suitable for use with double grippers and tools of the same type (same size and same location type).

The already defined system variable $TC_MAMP2 includes an additional setting option for the new empty-location search strategy.

Bit	Value	Meaning
0		
...		Tool search strategy
7		
8		
...		Search strategy for empty location
13		The magazine location for the "new" tool to be loaded is transferred to the "old" tool to be replaced and vice versa. Precondition is that the **tool sizes and location types of the tools match** or are compatible in terms of location hierarchy. The location of the "new" tool is detected as empty location for the "old" tool even if the "new" tool is still positioned at this location at the time the check is performed. The tool transport must be designed so that the "new" tool is first removed from the magazine location before the "old" tool is taken to it. Otherwise, depending on the sequence of the mechanical tool transport operations, the machine could be damaged. The type of empty location search is determined via the bits 8 through 12. It is not possible for tool change to take place if the "old" tool does not have a magazine location assigned to it. The tool search strategy is then determined via bits 8 to 12.

3.5.7 Tool search in wear group

Overview

If "Wear group" function is used:

In the case of existing tool-search strategies, the search refers only to the active wear group, i.e. only those tools are considered during a search within a tool group that are at magazine locations of the active wear group.

Tools in magazine locations with wear group number 0 are also checked for suitability.

If there are no spare tools available, then all $TC_MPP5 parameters of the current groups are negated and all locations are individually disabled by this. $TC_MAP9 is also negated (wear group disabled). All active tools are reset if this response has been configured via $TC_MAMP3 (bit 1 = 1).

The next wear group is called ($TC_MAP9 is assigned the number of the next wear group that can be activated).

If no further groups are available the search is terminated with an alarm. In such a case, the disabled tools should be replaced, if necessary. In order to enable the wear groups again, the wear group numbers of the magazine locations must be again set to values > 0.

Search strategies

There are two search strategies for finding the next available wear group that can be activated:

- Starting from the lowest magazine location number, the replacement tools are searched through location by location according to the way they are sorted internally (time-optimized search).
  The wear group that is the subject of the search is found by searching for the first tool that is assigned to a wear group that can be activated.
- A search is made for the wear group with the lowest enabled wear group number (the first that can be activated).

Search in several magazines

The magazine definition for a machine defines whether the search is to be performed in one or several magazines.

If the search is conducted in several magazines while several wear groups are being used, always make sure that a wear group can only ever be assigned to one magazine.

The search is conducted according to the following priorities:

1. The search is performed in a magazine according to the configuration and strategy.
2. The search is performed in the active wear group.
3. The set tool-search strategy is taken into consideration.

Activation

In order to work with wear groups, the magazine locations must be assigned to wear groups via system variable **$TC_MPP5** and the function must be activated via the machine data.

In addition, the number of the wear group with which machining is to commence must be assigned to system variable **$TC_MAP9** of the magazine to be selected (value > 0).

For the configuration of the machine, it is defined by **$TC_MAMP3** how the tool status shall change when switching from one wear group to the next (defaulted is an unchanged tool status).

Note

The PLC signal "tool selection from disabled tools" has no effect when resetting the tool state.

Example: Tool search in wear group

$TC_MAMP3 = 3 - change "active" status of tools

Target

- The tools must be set to "active" when a wear group is activated.
- When a wear group is disabled all the tools contained in that wear group should also be deactivated.

Specifications

- Circular magazine number 1 (6 locations)
- The circular magazine is to be divided into two parts:
  Locations 2 and 3 form wear group 1.
  Locations 4, 5, 6 and 1 form wear group 2.
- $TC_MAP9 = 1 (wear group 1 is "active")

Assignment to the wear group is achieved by:

$TC_MPP5[1,2] = 1

TC_MPP5[1,3] = 1

$TC_MPP5[1,4] = 2

$TC_MPP5[1,5] = 2

$TC_MPP5[1,6] = 2

$TC_MPP5[1,1] = 2

The tools with T=10 and T=11 are assigned to wear group 1. As wear group 1 was activated, tools T=10, 11 were therefore also set to "active" (via $TC_MAMP3, bit 0=1).

Note

Language command SETTA (see Section "SETTA - Activate tool from wear group (Page 299)") can also be used to set these tools to active.

Tool assignment:

```
$TC_MPP6[1,2] = 10         ;T=10 has identifier "Tool1"/duplo no.=1 tool state "ac-
                           tive"
$TC_MPP6[1,3] = 11         ;T=11 has identifier "Tool2"/duplo no.=1 tool state "ac-
                           tive"
$TC_MPP6[1,4] = 12         ;T=12 has identifier "Tool1"/duplo no.=2
$TC_MPP6[1,5] = 13         ;T=13 has identifier "Tool2"/duplo no.=2
$TC_MPP6[1,6] = 14         ;T=14 has identifier "Tool1"/duplo no.=3
$TC_MPP6[1,1] = 15         ;T=15 has identifier "Tool2"/duplo no.=3
```

$TC_MAMP2 = 1

The active tool is to be searched for. If none is available, the next possible tool is to be located.

This tool search strategy is superimposed by a check for the number of the active wear group. That is to say only those tools are considered during the search for a tool with the status "active" that are at magazine locations and that have the number of the currently activated wear group.

T="Tool2"

Tool group "Tool2" consists of tools

T=11, 13, 15.

T=11 is positioned in a location of the active wear group (No. 1) and is "active". The result of the tool search is T=11.

Machining is continued. T=11 is "disabled" during machining.

T="Tool1"

Wear group 1 is still active. T=10 is identified as active and suitable for use.

T="Tool2"

The tool group of identifier "Tool2" now has no active tool (has been disabled) and a new tool has not yet been set to "active". This step is not taken until "Tool2" is reprogrammed. The tools of the group are examined. In the locations of wear group 1, which is still active, there is no tool with identifier "Tool2" or any other suitable tool.

This condition causes the next wear group (2) to be activated. Wear group 1 is now no longer the active wear group. The status of the tools in wear group 1 has been reset (not "active"), as configured by **$TC_MAMP3, bit 1=1**.

The tool search is now performed exclusively in wear group 2. Its tools were set to "active" when the wear group was activated (one tool from each tool group in the wear group because setting of $TC_MAMP3, bit 0=1).

The turret is now assigned as follows:

```
$TC_MPP6[1,2]=10              T=10 has identifier "Tool1"/duplo no.=1 ;tool state "not active"
$TC_MPP6[1,3]=11              T=11 has identifier "Tool2"/duplo no.=1 ;tool state "disabled"
$TC_MPP6[1,4]=12              T=12 has identifier "Tool1"/duplo no.=2 ;tool state "active"
$TC_MPP6[1,5]=13              T=13 has identifier "Tool1"/duplo no.=2 ;tool state "active"
$TC_MPP6[1,6]=14              T=14 has identifier "Tool1"/duplo No.=3
$TC_MPP6[1,1]=15              T=15 has identifier "Tool2"/duplo No.=3
```

In the example **T=13** is now taken as the next available tool "Tool2".

Note

The tool search only then generates an alarm when no further spare tool available in the tool group with the given identifier is found and no further wear group can be activated.

Control system response

Control behavior at Power On, mode group change, reset, block search and Repos is described below.

Configuration $TC_MAMP3, bit 0=1 (activate internally)

At Power On, the NCK checks whether the value of $TC_MAP9 > 0, i.e. whether a wear group has been selected. In this case the tools of that wear group are checked again and the value for $TC_MPP5 of each location in question is set to positive. In addition, the status of the tool in the location is set to "active".

Configuration $TC_MAMP3, bit 1=1 (disable internally)

At Power On, the NCK checks whether $TC_MAP9 is negative, i.e. a wear group has been disabled. In this case the tools of the disabled wear group are checked again and the value for $TC_MPP5 of the location in question is set to negative. The "active" status of the tool in the location is reset.

3.6 Loading

3.6.1 Loading sequence

Overview

Loading generally involves transporting a tool from a loading point (location in the loading magazine) into a real magazine. In so doing, the tool is allocated an empty location (the location where it was loaded). Only then, from the perspective of the NCK, is the tool considered to have been loaded. This means that a tool that was loaded onto the spindle – e.g. via SINUMERIK Operate – is considered as not having been loaded.

Loading sequence

In the 1st step, a search is made for a suitable empty location for the tool to be loaded. In the 2nd step, the tool whose data must have been completely created (tool data and at least one cutting edge), is placed at a loading point. In the 3rd step, the tool is transported.

Note

SINUMERIK Operate automatically handles the loading. The tool data is completely generated. The tool is placed at the loading point and tool transport initiated. This sequence does not change, even if the tool is visually created in the magazine list via the user interface.

3.6.2 Function of the PLC when loading?

Loading sequence

When loading, the PLC is informed by the NCK about magazine and location numbers.

When a tool is loaded, the target address is the magazine and the loading location for the tool (DB71.DBW (n+24) and (n+26). This target address is communicated to FC8/FC6 as parameters **"NewToolMag"** and **"NewToolLoc"** and "Status = 1" or "Status = 10" once the load operation has been successfully completed. Parameters "OldToolMag", "OldTool-Loc" must be supplied with 0. The number of the active interface identifies the loading location.

The loading procedure is performed as follows:

1. A request is sent to the PLC to load the tool. The information is transferred to the PLC in DB71.
   Example:
   Data in DB71 when loading for the 2nd interface,
   (location 5 in magazine 1 is to be loaded from loading location 2)

DB71.DBX0.1 = 1	Interface 2 active
DB71.DBX34.0 = 1	Command: Loading

DB71.DBW50 = 9999	Magazine no. of the loading magazine
DB71.DBW52 = 2	Location no. within the loading magazine
DB71.DBW54 = 0	Magazine no. Source for unloading
DB71.DBW56 = 0	Location no. Source for unloading
DB71.DBW58 = 1	**Magazine no. target for loading**
DB71.DBW60 = 5	**Location no. target for loading**

1. Generally, the PLC would now move "location 5" from "magazine No. 1" (into which the tool should be loaded) to "loading location 2" and execute the load operation.
2. If the tool is physically in the magazine, FC8 must be called from the user program and the loading operation acknowledged.

Example of FC8/FC6 call when loading

FC 8 parameters	Values	Comment
Start	1	Starts task
TaskIdent	1	DB 71 interface
TaskIdentNo	2	No. of active interface
NewToolMag	1	Mag. No. 1
NewToolLoc	5	Location No. 5
OldToolMag	0	During loading = 0
OldToolLoc	0	During loading = 0
Status	1/10	Operation completed
Ready		Feedback from FC8
Error		Feedback from FC8

For loading and unloading, as well as for relocating the acknowledgement status = 10 is available. It has the same effect as status =1 (final acknowledgment; however it does not change the magazine position ($TC_MAP8). For every final acknowledgment (if the tools were moved) the magazine position is always aligned.

In this specific case, loading from location 5 into magazine 1, with the final acknowledgment (status_1) the magazine position is set to 5 (possibly existing offsets are not taken into consideration). However, this is not always desirable (e.g. when equipping a turret), as HMI Operate derives this display from the actual magazine position.

Problems during loading

A tool cannot be loaded. Check the following:

- Is the location type correct?
- Is a suitable empty location available?
- Has the number of tools enabled in the NCK (MD18082) been reached?
- Does the tool size include a "0", e.g. "1011"? (this is not permitted).

Alarms on the operator panel:

- No suitable empty location available
- "Create tools" command cannot be output to the NCK

3.6.3 Loading by directly assigning the T number

By directly writing to parameter $TC_MPP6[magazine No.,location No.] = T, a tool can be set to a magazine location (whether it is a real magazine, buffer or loading magazine). For example, for oversized tools, the NCK automatically reserves adjacent locations.

The NCK checks whether the tool fits at this location (location type, free adjacent locations, ...), if not, then an alarm is generated.

This does not involve an associated command output to the PLC.

3.6.4 Load tools via a part program

T number

The data required for a tool can also be loaded via a part program.

There are two possibilities to get the T number that addresses the data. You can:

- assign the T number yourself or
- allow the NC to assign the T number
  (via the command NEWT(...), see index entry).

The other data can be addressed by the T number determined in this way. Otherwise the T number can be assigned by the user (refer to the following example):

Example

```
DEF INT TNo
TNo=NEWT("test",1)
$TC_TP3[TNo]=2                ;Size left
$TC_TP4[TNo]=2                ;Size right
$TC_TP5[TNo]=1                ;Size top
$TC_TP6[TNo]=1                ;Size bottom
$TC_TP7[TNo]=2                ;Location type
$TC_TP8[TNo]=2                ;Tool status
$TC_TP9[TNo]=0                ;Type of monitoring
$TC_TP10[TNo]=0               ;Substitute tool strategy
$TC_TP11[TNo]=0               ;Tool info
$TC_DP1[TNo,DNo]=120          ;Tool type: (all of the offset data is provided here)
$TC_MPP6[MagNo,LocNo]=TNo     ;Tool with the determined T number is written/loaded
                              to the location
```

The tool described here also occupies adjacent locations. The tool management automatically assigns/reserves the adjacent locations.

It is also possible to delay assignment of a tool to a location, in which case the command $TC_MPP6 is not required. After execution of the part program the tools are contained in the tool list and can be loaded at a later time.

3.6.5 Retroload tool data

Procedure

When tool data is "reloaded" this means that the offset data is not entered or loaded until after the tool loading operation.

- The tools are already located in the magazine, both physically as well as their data, i.e. the "Tool ←→ Location" assignment has taken place.
- There is either no tool offset data in the NC or it is no longer up to date.

The offset data is assigned via the part program, i.e. the existing data is overwritten. If not already known, the internal T number of the particular tool first has to be determined in the "retroload" program to do this.

The internal T number is the tool number that the NC works with. It is unique and describes a tool. All parameters of this tool are addressed by this T number.

The T number can be assigned either by the operator when creating a tool or by the NC.

If the operator knows the T number (e.g. specified by the entries made at the measuring station), then this number can be retrieved in the retroload program.

If the T number is not known, then it has to be determined for each tool to be retroloaded and be supplied from a variable. This reduces the overhead for the user and also reduces the scope for errors.

Create the retroload program

The tool is measured at a measuring station and the measured data stored. For this purpose, the tool must already be defined, i.e. by both an identifier ("Drill 12 mm" or "Miller 23" in the following) and the relevant duplo number. (The combination of tool identifier and duplo number uniquely defines the tool.) The internal T number of this tool is determined prior to the data record using the command GETT("Identifier", duplo No.) and saved a variable ("T No." here). The data required for the tool is written and then the entire program is transferred to the NC where it is processed.

Only the variables for which data is entered have to be written. The first tool in the following example contains all the data, the second tool only contains the relevant data.

The T number does not have to be determined in the retroload program if already specified during loading because the data can then be assigned directly.

For example, for tool "1", writing length L1 looks like this:

```
$TC_DP1[1,1]=120          ;Tool type
```

```
$TC_DP3[1,1]=67.032      ;Length
```

Program for retroloading tool offset data

```
DEF INT Tno                       ;Definition of variable TNo
tl1:
TNo=GETT ("Drill 12mm",1)
if TNo==-1 goto tl2
$TC_DP1[TNo,1]=120                ;Tool type
$TC_DP2[TNo,1]=0
$TC_DP3[TNo,1]=67.032             ;Length1
$TC_DP4[TNo,1]=0
$TC_DP5[TNo,1]=0
$TC_DP6[TNo,1]=24                 ;Radius
$TC_DP7[TNo,1]=0
$TC_DP8[TNo,1]=0
$TC_DP9[TNo,1]=0
$TC_DP10[TNo,1]=0
$TC_DP11[TNo,1]=0
$TC_DP12[TNo,1]=0
$TC_DP13[TNo,1]=0
$TC_DP14[TNo,1]=0
$TC_DP15[TNo,1]=0
$TC_DP16[TNo,1]=0
$TC_DP17[TNo,1]=0
$TC_DP18[TNo,1]=0
$TC_DP19[TNo,1]=0
$TC_DP20[TNo,1]=0
$TC_DP21[TNo,1]=0
$TC_DP22[TNo,1]=0
$TC_DP23[TNo,1]=0
$TC_DP24[TNo,1]=0
$TC_DP25[TNo,1]=0
$TC_MOP1[TNo,1]=0
$TC_MOP2[TNo,1]=0
$TC_MOP3[TNo,1]=0
$TC_MOP4[TNo,1]=0
tl2:                              ;Next tool
TNo=GETT ("Miller23",2)
if TNo==-1 goto error             ;Possible error routine if tool does not exist
$TC_DP1[TNo,1]=120
$TC_DP3[TNo,1]=82.51
$TC_DP6[TNo,1]=25
Fault:                            ;Error
:
:
```

```
M17
```

3.7 Unloading

3.7.1 Overview

Unloading removes the tool from the magazine. It always involves transporting a tool from a real magazine into the loading magazine. In so doing, the owner location of the tool as well as all reservations made in the real magazine are deleted. The job is initiated via PI service (initiated via the HMI user interface) or language command. In so doing, NCK generates a CMD1, this means that the DB71 is used.

3.7.2 PLC function when unloading

Overview

When unloading, here in the example, this is initiated by an HMI operator action – the target address of the tool is specified to FC8/FC6 with the identifier of the loading/unloading point (DB71.DB(n+16) and DBW(n+18), start address "n" is included in the interface list). The target address is communicated to the FC8/FC6 as parameters "**OldToolMag**", "**OldToolLoc**" and "Status = 1" or "Status = 10" once the unload operation has been successfully completed. The "NewToolMag" and "NewToolLoc" parameters must be assigned the value zero.

Unloading sequence

Unloading is controlled via DB71. The unloading sequence is as follows:

1. The PLC receives the command to unload the selected tool. The information is transferred to the PLC in DB71. Example of the data in DB71 when unloading for the 2nd interface Location 7 of magazine no. 1 must be unloaded at loading location 2.
   Example:

DB71.DBX0.1=1	;Interface 2 active
DB71.DBX34.1=1	;Command: Unloading
DB71.DBW50=9999	**Magazine no. of the unloading point**
DB71.DBW52=2	**;Location no. of the unloading point**
DB71.DBW54=1	;Magazine no. for unloading
DB71.DBW56=7	;Location no. for unloading
DB71.DBW58=0	;Magazine no. target for loading
DB71.DBW60=0	;Location no. target for loading

1. Generally, the PLC must now position "Location 7" from "Magazine No. 1" (from which the tool should be unloaded) to the "loading/unloading point 2" and then unload the tool.
2. When the tool is removed from the magazine, FC8/FC6 must be called from the user program and the unloading operation acknowledged.

Example: Call FC8/FC6 when unloading

FC8 parameters	Values	Comment
Start		Starts task
TaskIdent	1	DB71 interface
TaskIdentNo	2	No. of active interface
NewToolMag	0	During unloading = 0
NewToolLoc	0	During unloading = 0
OldToolMag	9999	Mag. No. 9999
OldToolLoc	2	Location no. 2
Status	1/10	Operation completed
Ready		Feedback from FC8
Error		Feedback from FC8

The PLC user program then has to traverse the magazine to the correct unloading point and execute unloading. If the tool comes via a buffer location (gripper, loader) to the unloading point or station, then the NCK is to be notified of each position change by means of the FC8/ FC6 with status 104, 105. Status "1" is not set via FC8 until the tool is in the specified unloading point/station. This means that the unload operation has been completed; the magazine position is also automatically aligned. In this example, do not take into account possibly existing offsets at location_7.

Positioning for unloading (with OP177 and SINUMERIK Operate)

This means that the HMI sends two tasks to the NCK (and this in turn to the PLC). First positioning (which is described in the following), then unloading.

When a magazine is being **positioned** at a loading magazine, the target address is stored in DB71.DBW(n+16) and DBW(n+18). This target address is passed to FC8/FC6 as parameters "NewToolMag" and "NewToolLoc" and "Status" = 1 once the magazine has been successfully positioned. Parameters "OldToolMag", "OldTool-Loc" must be set to 0.

The magazine and magazine location to be positioned are stored in DB71.DBW(n+20) and DBW(n+22). Positioning here only concerns magazine positioning a free location or a location with a tool to a loading/unloading station. The number of the active interface identifies the loading magazine (location No.).

Example: Positioning for unloading

FC8 parameters	Values	Comment
Start		Starts task
TaskIdent	1	DB 71 interface
TaskIdentNo	2	No. of active interface
NewToolMag	**9999**	**Mag. No. 9999**
NewToolLoc	**3**	**Loc. No. 3**
OldToolMag	0	During positioning = 0
OldToolLoc	0	During positioning = 0
Status	1	Operation completed

3.7.3 Unloading by deleting the T number from the magazine location

By directly writing to parameter $TC_MPP&[magazine No.,location No.] = 0, a tool can be removed from this magazine location (whether it is a real magazine, buffer or loading magazine). All reservations of the removed tool are automatically deleted.

This does not involve an associated command output to the PLC.

3.8 Relocating tools and positioning the magazine

3.8.1 Relocate (task from TM system)

Basics

When **relocating**, the target address is the magazine and the loading location for the tool to be relocated (DB71.DBW(n+24) and DBW n+26). The source of the tool is specified in DB71.DBW(n+20) and DBW(n+22). The target address is passed to FC8/FC6 as parameters "NewToolMag" and "NewToolLoc" and status = 1 when relocation has been successfully completed. Parameters **"OldToolMag"** and **"OldToolLoc"** must be set to zero because the tool management recognizes the location of the old tool.

All relocation tasks are handled via the 1st interface.

Example of relocating a tool

FC 8 parameters	Values	Comment
Start		Starts task
TaskIdent	1	DB71 interface
TaskIdentNo	1	Channel No. for tool management
NewToolMag	2	New magazine No.
NewToolLoc	17	New location no.
OldToolMag	**0**	**Old magazine number not used**
OldToolLoc	**0**	**Old location number not used**
Status	1	Operation complete

3.8.2 Relocation by the PLC

Task from PLC

The PLC can also issue a task to tool management to relocate a tool. This is done by notifying a new location for the tool to the tool management. The FC8/FC6 block (TaskIdent := 4) is called with the following parameters:

- Old magazine no. (OldToolMag)
- Old location no. (OldToolLoc)
- New magazine no. (NewToolMag)
- New location no. (NewToolLoc)

Example 1

The tool in magazine No. 1, location No. 5 is to be relocated to magazine No. 2, location No. 17. The PLC takes responsibility for ensuring that the location type is correct for the transfer. This example for calling an FC8 does not take into consideration any checkback signal to tool management for intermediate positions of tools.

FC8 parameters	Values	Comment
Start		Starts task
TaskIdent	4	Task from PLC
TaskIdentNo	1	Channel No. for tool management
NewToolMag	2	New magazine No.
NewToolLoc	17	New location No.
OldToolMag	1	Old magazine No.
OldToolLoc	5	Old location No.
Status	1	Operation completed

Example 2

Example: The tool is to be relocated from mag. No. 1, location No. 5 via grippers 3 and 4 to mag. No. 2, location No. 17.

FC8/FC6 must be called four times in this procedure. Only the important parameters are listed. All other parameters are as for the example above.

The tool is transported in 4 steps:

1. From magazine 1, location 5 to gripper 3 (location no. 4)

FC8 parameters	Values	Comment
Start		Starts task
TaskIdent	4	Task from PLC
TaskIdentNo	1	Channel No. for tool management
NewToolMag	9998	New magazine No.
NewToolLoc	4	New location No.
OldToolMag	1	Old magazine No.
OldToolLoc	5	Old location No.
Status	1	Operation completed

2. From gripper 3 to transfer location 2, location no. 6

FC8 parameters	Values	Comment
Start		Starts task
TaskIdent	4	Task from PLC
TaskIdentNo	1	Channel No. for tool management
NewToolMag	9998	New magazine No.
NewToolLoc	6	New location No.
OldToolMag	9998	Old magazine No.

OldToolLoc	4	Old location No.
Status	1	Operation completed

3. From transfer location 2, location no. 6 to gripper 4, location no. 5

FC8 parameters	Values	Comment
Start		Starts task
TaskIdent	4	Task from PLC
TaskIdentNo	1	Channel No. for tool management
NewToolMag	9998	New magazine No.
NewToolLoc	5	New location No.
OldToolMag	9998	Old magazine No.
OldToolLoc	6	Old location No.
Status	1	Operation completed

4. From gripper 4, location no. 5 to magazine 2, location 17

FC8 parameters	Values	Comment
Start		Starts task
TaskIdent	4	Task from PLC
TaskIdentNo	1	Channel No. for tool management
NewToolMag	2	New magazine No.
NewToolLoc	17	New location No.
OldToolMag	9998	Old magazine No.
OldToolLoc	5	Old location No.
Status	1	Operation completed

Note

TaskIdent 5 may be programmed only for a tool transfer (magazine → buffer location). Otherwise an error message is output, even though the tool is transferred.

The reservation is automatically reset when the tool is transferred from the buffer back into the magazine.

3.8.3 Positioning the magazine

Overview

When positioning, a traversing block is sent to the PLC by the tool management. Bit 3 is set in DB71.DBB(n+0) "Position to the loading point". The magazine No. and the location No. (as target) are transferred in the parameters DB71.DBW(n+20) and (n+22) during positioning.

The PLC then has to move this location to the loading point. The number of the loading point is entered in DB71.DBW (n+18) or determined by the number of the interface. If the PLC has moved the magazine location to the loading point, FC8/FC6 must be called and the operation acknowledged with status 5 "Position changed".

Example:

Location 5 in magazine 1 (source) must be moved to the loading magazine 2 (target).

DB71.DBX0.1	=1	Interface 2 active	
DB71.DBX34.3	=1	Positioning has been initiated	(n+0)
DB71.DBW50	=9999	Magazine No. of the loading magazine	(n+16)
DB71.DBW52	=2	Location No. of new loading magazine	(n+18)
DB71.DBW54	=1	No. of the magazine to be positioned	(n+20)
DB71.DBW56	=5	No. of the location to be positioned	(n+22)
DB71.DBW58		Magazine No. to which positioning is to be carried out	(n+24)
DB71.DBW60		Location No. to which positioning is to be carried out	(n+26)

Parameters "OldToolMag" and "OldToolLoc" in FC8/FC6 are not required for positioning because only the PLC requires the information for traversing the magazine. The values for NewToolMag and NewToolLoc are from DB71(n+24 and n+26). The PLC has to execute the positioning task and acknowledge it with an FC8/FC6 call as follows:

Example: FC8/FC6 call for positioning

FC 8 parameters	Values	Comment
Start		Starts task
TaskIdent	1	DB71 (loading/unloading, positioning, relocating)
TaskIdentNo	2	No. of active interface
NewToolMag	9999	Magazine no. 9999 (loading magazine)
NewToolLoc	2	Location 2
OldToolMag	0	During positioning = 0
OldToolLoc	0	During positioning = 0
Status	5	Operation completed
Ready		Feedback from FC8
Error		Feedback from FC8

3.9 Tool monitoring (workpiece count, tool life, wear)

3.9.1 Monitoring types

Quantity

The quantity monitoring, initiated by the language command SETPIECE, considers the cutting edges of all of the tools that have been used. In this case, it must be observed that there can be several spindles.

Service life

Tool life monitoring is only performed on the tool edge that is currently in use. As soon as the path axes are moved (with the exception of G00), then monitoring time of the active cutting edge is decremented. If the tool life for a cutting edge expires during machining, the tool is disabled as a whole.

Wear

As is the case for the time and workpiece count (quantity) monitoring, the prerequisite for using the wear monitoring function is that tool monitoring is enabled in the machine data. In addition, wear monitoring must be enabled via an MD. The wear parameters of the cutting edge correspond to the local offsets (additive offset parameters), see Section "Location-dependent offsets, fine (additive offsets) (Page 231)".

Service life, quantity

The monitoring value runs from a value greater than 0 to 0. The monitoring setpoint has no affect on the monitoring, this value is only evaluated when reactivating a tool.

Tool life, workpiece count and wear

The monitoring type for the tools is generally set when creating or loading. You can change the monitoring type at any time by changing the setting for system variable MD $TC_TP9.

The tool management monitors a tool for tool life, workpiece count, tool wear state or additive offset with prewarning limit.

The monitoring counter triggering the tool status change depends on the system variable $TC_TP9 (= type of monitoring):

- $TC_TP9 = 0 → No monitoring
- $TC_TP9 = 1 → Time-monitored tool
- $TC_TP9 = 2 → Workpiece count-monitored tool
- $TC_TP9 = 4 → Wear-monitored tool
- $TC_TP9 = 8 → Additive offset

Several monitoring types can be activated simultaneously for one tool. Of the monitoring types, only wear monitoring and additive offset monitoring are mutually exclusive.

If the monitoring criterion (tool life/workpiece count and wear) for a tool that is currently located in the spindle expires, it remains in use. Machining is not automatically interrupted to replace the tool with a replacement tool. The tool is not disabled until the next time it is selected. Since it is no longer "available", a search is made for a replacement tool and the replacement tool is then loaded into the spindle. The tool change has to be organized by the PLC or come from the NC cycle.

The monitoring counters count from a set value > 0 down to zero. Once a monitoring counter has reached the value ≤ 0, the limit value is considered to have been reached. When a tool's cutting edge (one of maximum 12 cutting edges) has reached its limit value, the whole tool is set to status "disabled".

Prewarning limit reached

If a cutting edge has reached its prewarning limit, then the whole tool is set to the status "Prewarning limit reached" (SLTD_SUSPENDED ($TC_TP8[i]=4)).

At the same time a message is issued to inform the operator that a replacement tool may be required. If an operator action resets the monitoring counter from zero or the prewarning limit back to a value > 0, or > prewarning limit, then the tool status changes automatically to reflect the change in the data. This allows the operator to selectively withdraw a "disabled" status caused by the tool expiring when its monitoring limit was reached.

If the tool has several cutting edges, all of the cutting edges must be beyond the monitoring limits.

Tool monitoring alarms

When the prewarning limit or the monitoring limit of a tool is reached, one of the alarms 6010, 6011, 6012, 6013 (cancel clear delete condition) is output for information purposes.

With the NC language command SETPIECE(...) or PI command _N_TMPCIT (= other workpiece counters) it is possible for several tools to reach a limit value and therefore result in multiple alarms being issued.

No alarm is output if a limit value is reached as the result of data manipulation via variable service (OPI).

Check monitoring status

A check can be made during program execution by issuing the programmed tool change command (e.g. "M06" for milling) without a T call to see whether a monitoring function has responded. If it has, tool life management will search for a replacement tool and a request for a tool change issued.

Enable memory and function

In general, in machine data

- MD18080 $MN_MM_TOOL_MANAGEMENT_MASK and
- MD20310 $MC_TOOL_MANAGEMENT_MASK

at least the bits 0 and 1 (3) must be set. This prepares the memory for the monitoring data and enables the function.

Enabling tool life monitoring

For the tool life monitoring, in addition, in the channel-specific MD20320 $MC_TOOL_TIME_MONITOR_MASK, the spindle (the tool holder) or the spindles must be specified, for which a tool life check should be performed. This machine data is bit-coded.

Example: MD20320 $MC_TOOL_TIME_MONITOR_MASK

- Value = 1, only spindle number 1
- Value = 2, only spindle number 2
- Value = 3, spindle numbers 1 and 2

3.9.2 Tool life monitoring

Monitoring of tool cutting edge

Tool life monitoring is only performed on the tool edge that is currently in use. The spindle (tool holder) must have been activated beforehand (MD20320 $MC_TOOL_TIME_MONITOR_MASK = spindle no.).

If MD20124 $MC_TOOL_MANAGEMENT_TOOLHOLDER is > 0, then for MD 20320, the tool holder number is selected and not the spindle number.

Service life	Data backup during unloading takes place in milliseconds.
Disabled	If the remaining tool life is less than or equal to 0, the tool is set to "disabled". After the next tool change it is no longer used.
Monitoring from the NCK	The residual tool life is decreased whenever one of the 3 path axes is traversed at machining feedrate (e.g. G01). G00 traversing blocks are not "counted".
Monitoring from PLC	The user can start and stop the time monitoring using PLC signal "Time monitoring active" (DB21DBX1.3). Machine data 20310 is used to select whether this control type is active.
Prewarning limit	Entered during loading or via part program with $TC_MOP1=50. When the prewarning limit has been reached, the tool is assigned the status "Prewarning limit reached" (display in the magazine list).
Special case, limit values	The tool life of a tool expires while it is in use. A check is made as to whether this disabled tool is re-programmed by a change operation (e.g. M06 without T word), and whether the monitoring time has already elapsed. If yes, a replacement tool is located.

$A-MONIFACT factor

By entering a channel-specific factor which is set before a tool is used for the first time, it is possible to monitor the different degrees of tool wear resulting from machining different types of workpiece material. The value is multiplied by the current time unit before the time value of the cutting edge is decremented. The write operation is performed synchronously with the main run.

Start and stop the tool life decrementation

Tool life monitoring runs when geometry axes are not traversed with **G00** (default setting).

The user can start and stop the time monitoring using PLC signal "Time monitoring active" (DB21 DBX1.3).

Which type of control is active is set via MD20310 $MC_TOOL_MANAGEMENT_MASK, bit 17. The default setting (bit 17=0) is standard; i.e. traversing blocks not equal to G00 will increment the time counter.

Time monitoring hierarchy

The combination of system variable $A_MONIFACT and function "Program test active" produces the following nested time monitoring structure:

Machine data MD20310 $MC_TOOL_MANAGEMENT_MASK defines the monitoring control via G00 or via a PLC signal. Tools on spindles that are activated by machine data MD20320 $MC_TOOL_TIME_MONITOR_MASK, are time-monitored.

The VDI signal "Program test active" switches the momentarily valid time monitoring on or off; i.e. "Program test active" has a higher priority than the actual time monitoring.

When time monitoring is active, the real time (as defined by the internal clock) is multiplied by the factor $A_MONIFACT and the result subtracted from the current time count of a tool edge mounted on the spindle.

3.9.3 Workpiece count monitoring

Changing the number of workpieces

The number of workpieces can be changed by:

- Operator action at the HMI
- With a part program command (SETPIECE)
- PI service (TMPCIT) by PLC or HMI-OEM

Workpiece counter per spindle

Every spindle has a "memory" for the cutting edges used on it. With program command SETPIECE (1) the workpiece counter for the cutting edges that are used on the main spindle is decremented by 1. The workpiece counter of each spindle can be addressed individually.

The workpiece counter must count all the tools that are used to produce a workpiece. Take into account that the machine may have several spindles and that different tools can be used simultaneously.

The cutting edge of a tool is only counted once per spindle.

The part program programmer who programmed **SETPIECE** can program the parameter as a function of the material.

SETPIECE (factor * no. of workpieces)

Like the factor for time monitoring, this function allows a workpiece count that depends on the process, the workpiece material or other factors.

Workpiece count can be deactivated via the channel DB.DBX29.5.

Monitoring from the NCK	The workpiece counter is decremented each time that SETPIECE is programmed. When the prewarning limit is reached, an information alarm is output. When the counter reaches the value=0, the tool is disabled and an alarm is output. The next time the tool is called, the replacement tool is loaded at change.
Set workpiece counter	Entered when creating or loading the tool or via the part program with e.g. $TC_MOP4=500.
Decrement number of work-pieces	The number of workpieces must be decremented at the relevant point in the part program with the NC language command SETPIECE (x, y) (e.g. SETPIECE(1) → workpiece counter for main spindle tools is de-cremented by 1). The function for decrementing the workpiece count is activated from die PLC program using a PI command (TMPCIT).
Disabled	When the workpiece count has reached zero the tool is disabled.
Prewarning limit	Entered when creating or loading the tool or via part program with e.g. $TC_MOP3=50. When the prewarning limit has been reached, the tool is assigned the status "Prewarning limit reached".
Special case, limit values	It is not possible to realize a workpiece count for any number of cutting edges simultaneously! If the monitoring function has been enabled and activated by machine data, then all spindles can be monitored together at a time = "Number of cutting edges in the TO area" (= MD) for the number of cutting edges. An edge of a tool is only counted once per spindle.

3.9.4 Wear monitoring

The wear monitoring function is only available if the "Tool monitoring" system has been enabled via machine data.

In addition, wear monitoring must be enabled using machine data (MD18080 $MN_MM_TOOL_MANAGEMENT_MASK, bit 5).

Definition

$TC_TP9 = 4; Wear monitoring is active for the tool.

$TC_TP9 = 8 can be set to select the "Additive offset" monitoring function if this is required.

$TC_TP9 = 4

The wear parameters for a tool edge are defined with system variables $TC_DP12, ..., $TC_DP20.

These are assigned directly to the edge geometry values TC_DP3, ...,$TC_DP11.

$TC_DP10 and $TC_DP11 describe "angles". The other parameters stand for the tool edge lengths and radii.

Only these values are included in the monitoring, i.e. wear parameters $TC_DP19 and $TC_DP20, which are analogous to system variables $TC_DP10 and $TC_DP11, are not taken into account.

Note

Wear monitoring is performed automatically by the NCK when the user changes the cutting edge offsets.

Note

Wear monitoring does not monitor every single value but rather only the largest absolute value of each of these maximum seven wear parameters ($TC_DP12, ..., $TC_DP18).

$TC_TP9 = 8

Wear parameters (system variables) of the cutting edge can be found in the **additive offset parameters**.

Analog to wear, the following system variables are monitored for the additional offsets that are dependent on the location (location-specific offsets) of the cutting edge:

- $TC_SCP12, ... $TC_SCP18
  first additive offset for the cutting edge (if defined)
- $TC_SCP22, ... $TC_SCP28
  second additive offset for the cutting edge (if defined) etc. for the other additive offsets for the cutting edge

  Note

  Wear monitoring does not monitor every single value but rather only the high contribution of each of these maximum seven additive offset parameters * number of defined additive offsets for the cutting edge ($TC_SCP12, …, $TC_SCP18, $TC_SCP22, …, $TC_SCP28, …).

Most tool geometries are described by a subset of the named data records.

If a parameter is changed (written), the NCK then checks whether the new value is higher than any of the other parameters and, if necessary, this value is subtracted from the wear setpoint. The result is the new actual value for the wear.

Analogous to other monitoring variables, the actual wear runs from the positive setpoint towards zero.

Monitoring parameters (system variables)

$TC_MOP15	Wear setpoint or additive offset value
$TC_MOP5	Wear pre-warning limit or additive offset pre-warning limit
$TC_MOP6	Wear value or additive-offset setpoint

The physical quantity of the new monitoring parameters is "Length". The unit is the same as for the wear values.

Wear monitoring can be deactivated via the channel DB.DBX29.6.

The signal only acts on changes in wear data that occur during execution of the NC program. The PLC signal is suppressed if this data is changed via the OPI (e.g. during HMI operation).

Reset to setpoints

Resetting the actual values of wear and additive offset "fine" means that all the parameters for wear offset and additive offset used for monitoring are set to zero.

Example

For the tool with T No.=3, the wear monitoring is active and

```
$TC_MOP5[3,1]=0.002        ;= Wear prewarning limit
$TC_MOP6[3,1]=0.003        ;= Wear actual value
$TC_MOP15[3,1]=0.007       ;= Wear setpoint
```

It was already set

```
$TC_DP12[3,1]=0.004         ;= Wear component 1
$TC_DP13[3,1] =+0.00        ;= Wear component 2
```

Wear component 3 is now set

$TC_DP14[3,1] := -0.006.

Thus the maximum absolute value is given for the wear components = 0.006.

The resulting new actual value is

$TC_MOP15[3,1] - 0.006 = 0.001 = $TC_MOP6[3,1].

The prewarning limit has been reached.

Note: The wear components can be negative or positive - or a mixture of each.

Note

The VDI signal "Activate program test" has no effect on wear monitoring since new wear values are only entered during machining and not during the program test (provided that wear was not changed by the machining program itself).

3.9.5 Signals to and from the PLC

Overview

Previously, an alarm message was output as soon as the prewarning limit or limit value was reached. Alarms **6410** and **6411** are output when the prewarning limit is reached and **6412** and **6413** when the limit value is reached. Alarms 6410 and 6412 are triggered via the OP interface and alarms 6411 and 6413 via the NC program. The alarm texts identify the affected tool via the tool ID, duplo number and D number.

The following information is returned to the channel interface for one OB1 cycle (internal T numbers):

- Prewarning limit reached
- Limit reached

A strobe signal is set for one PLC cycle (DB channel.DBB344) which indicates that new data is available.

VDI signal "Warning limit reached" channel DB.DBD348

If a tool reaches its prewarning limit with tool life, workpiece count or wear monitoring, the internal T No. of the tool is entered here and the associated strobe signal is set.

VDI signal "Limit value reached" channel DB.DBD352

If the tool life, workpiece count or wear value has expired for a monitored tool, the internal T No. of the tool is entered and the associated strobe signal is set.

Note

If machining is being performed with tools that are monitored for their workpiece count, it is possible for several tools to reach their prewarning limit or limit value simultaneously (SETPIECE is programmed at the end of program).

In this case, only the T No. of the tool that was last programmed is output.

VDI signal "T number of new replacement tool" - channel DB.DBD356

The transition to a new replacement tool is initiated by the tool status "was in use". This means that when searching for a tool in the NCK, if a (usable) tool was found that had still not been used, then it is evaluated as being the first selection and the interface signal set.

This process state change is output to the PLC via the T number of the replacement tool.

The action of the operator changing the tool status does not cause any change to the signal.

VDI signal "Last replacement tool in the tool group" - channel DB.DBD360

If, during the tool change when searching for a tool in the NCK a tool is found, and at this point in time there are no further replacement tools available for the programmed spindle / tool holder, then this is evaluated as the "Last replacement tool found in the tool group".

If there is only one tool (i.e. there is no replacement tool) it is also a tool group. When this tool is programmed, the interface signal is set immediately.

This process state change is output to the PLC via the T number of the replacement tool.

The action of the operator changing the tool status does not cause any change to the signal.

Note

For tool groups containing many tools, the function increases the time required in the NCK for the main run when the tool is selected.

The function must be enabled **MD20310 $MC_TOOL_MANAGEMENT_MASK, bit 18=1**.

Disable monitored tools - PLC-controlled by the VDI signal

In earlier versions, a tool assumed the status "disabled" as soon as the actual value of the active monitoring function reached the value zero. A tool being used for machining that is set to "disabled" remains operational until the next tool change takes place. After that the tool can no longer be used.

The PLC can also determine when a disabled tool can no longer be used, i.e. when the "disabled" status is taken into account in the tool search.

- With the **VDI signal "Tool disable not active"** = 1
  (channel DB.DBX29.7 = 1) the NCK does not take the tool status "disabled" into account during the tool search.
- With the **VDI signal "Tool disable not active"** = 0
  (channel DB.DBX29.7 = 0) the NCK takes the tool status "disabled" into account during the tool search.

The bit is channel-specific.

"Search for active tool" strategy

This search strategy can ensure that a machining operation is not performed with different tools from the same tool group.

When the tool is disabled, a monitoring function and the set VDI signal "Tool disable not active" mean that the status "active" is **not** canceled.

This **tool is therefore assigned** the status **"active"** and **"disabled"**.

If the required machining operation is terminated without a tool change, the status of all disabled tools must be checked. A PI service (_N_TMRASS, in PLC TMRASS) is provided for this. This service can be used to cancel the status "active" for all tools that have been disabled (e.g. by the PLC program at the end of the program).

The other tool-search strategies

A disabled tool can still be used with the other tool-search strategies when the VDI signal "Tool disable not active" (channel DB.DBX29.7 = 1) is set. The tool selected solely depends on the search strategy.

The **search strategy** therefore takes **precedence over** the **VDI signal "Tool disable not active"**. Both the last tool to be disabled or any other disabled tool can be selected.

Another tool which is not disabled might also exist, but is not selected because of the search strategy!

TO unit active in several channels

If a TO unit is assigned to several channels (tool and magazine data are "visible" in several channels), then the setting of the channel-specific VDI signal "Tool disable" is effective in each channel.

3.10 Tool monitoring without active tool management

3.10.1 Tool monitoring overview

General information

Note

The function is designed for simple applications, only 1-channel systems are supported

The tool monitoring without active tool management is activated via the following machine data:

- MD18080 $MN_MMTOOL_MANAGEMENT_MASK, bit 1=1
- MD20310 $MC_TOOL_MANAGEMENT_MASK, bit 1=1

Note

Monitoring without active tool management

The "Monitoring without active tool management" function (TMMO) is set with the same option date as the tool management (TMMG).

The tool monitoring without active tool management function allows the following monitoring types of the active cutting edge of the active tool:

- Monitoring the **tool life**
- Monitoring the **workpiece count**
- Monitoring the **wear**
- Monitoring the **additive offset**

The function allows tool identifiers to be used and it is also possible to use replacement tools.

Monitoring counter

Monitoring counters exist for each monitoring type. These count from a set value > 0 down to zero. When a counter has decremented to a value of ≤0, the limit value is reached. A corresponding alarm is issued. The same applies if a cutting edge of a tool has reached its set prewarning limit.

Note

The monitoring function cannot be used with the "flat D numbers" function.

State and type of tool monitoring

System parameter $TC_TP8 can be used to determine the state for the particular tool, $TC_TP9 to determine the monitoring function type.

Contrary to the magazine management (see Section "Magazines (Page 29)"), in the tool monitoring function without active tool management, only the following states are of significance:

- $TC_TP8[t] - State of the tool with the number t
  Bit 0 = 1: Tool is active
  Bit 0 = 0: Replacement tool
  Bit 1 = 1: Tool is enabled
  Bit 0 = 0: Tool is not enabled
  Bit 2 = 1 Tool is disabled
  Bit 2 = 0: Tool is not disabled
  Bit 3: Reserved
  Bit 4 = 1: Prewarning limit reached
  Bit 4 = 0: Prewarning limit not reached
- $TC_TP9[t] - Type of the monitoring function for the tool with the number t
  = 0: No monitoring
  = 1: Time-monitored tool
  = 2: Workpiece count-monitored tool
  = 4: Wear-monitored tool
  = 8: Additive offset

Several monitoring types are possible for a tool. Wear and additive offset monitoring cannot be simultaneously selected (only alternatively).

System variables for active tool

The active cutting edge of the active tool can be determined using an existing system/OPI variable:

- $P_TOOL / actDNumber: Active tool offset D
- $P_TOOLNO / actTNumber: Active tool number T

3.10.2 Tool life monitoring

Overview

Tool life monitoring is done for the tool cutting edge that is currently in use (active cutting edge D of the active tool T).

As soon as the path axes traverse (G1, G2, G3, ... but not for G00), the residual tool life ($TC_MOP2[t,d]) of this tool cutting edge is updated. If the remaining tool lifetime of a tool cutting edge expires during a machining operation, an alarm is output. The tool changes to the "disabled" condition and cannot be programmed as long as the "disabled" condition is present. The operator must intervene and ensure that an operational tool for machining is provided.

Note

As standard, the time is counted if the geometry axes are traversed - but not equal to G00. As an alternative, using MD $MC_TOOL_MANAGEMENT_MASK, bit 17, the time monitoring can be started and stopped from the PLC.

The channel-specific system parameter $A_MONIFACT allows the clock to be either run slower or faster.

Time monitoring and reference to the spindle or to the tool holder

For MD $MC_T_M_ADDRESS_EXT_IS_SPINO = TRUE, only the time for the active tool is counted whose programmed spindle number has a value that is set is by selecting the bit in MD $MC_TOOL_TIME_MONITOR_MASK.

The value of MD $MC_TOOL_TIME_MONITOR_MASK has no significance, if MD $MC_T_M_ADDRESS_EXT_IS_SPINO = FALSE. In this case, the time is counted and monitored for each active cutting edge.

3.10.3 Workpiece count monitoring

General

The active cutting edge of the tool that has been loaded at change is monitored for the workpiece count without reference to the spindle number (standard).

Monitoring the workpiece count registers all the tool edges that are used to produce a workpiece. If the workpiece count is changed by an operator input, the monitoring data of all the cutting edges that have become active since the last workpiece count is adjusted. It is taken into account that several tool cutting edges can be simultaneously used.

The workpiece count can be updated by the PLC, using the PI command _N_TMPCIT or from the NC program using the language command SETPIECE.

Workpiece count monitoring and reference to the spindle or to the tool holder

If MD $MC_T_M_ADDRESS_EXT_IS_SPINO = TRUE is used, the address extension of T is interpreted as spindle number. The workpiece counter monitoring is then realized separately for the spindles; further, the SETPIECE command requires the "spindle number" parameter.

If MD $MC_T_M_ADDRESS_EXT_IS_SPINO = FALSE is used, the parameter "spindle number" in SETPIECE and/or the analog PI service is ignored.

The workpiece count monitoring can be activated and deactivated by the PLC using an interface signal.

3.10.4 Wear monitoring

The wear monitoring must be enabled using MD $MN_MM_TOOL_MANAGEMENT_MASK, bit 5. As default, the function is not activated.

The wear parameters for a cutting edge are defined with system variables $TC_DP12 ...20. The highest value of parameter $TC_DP12 ...18 - responsible for the lengths and radii of the cutting edge - is monitored.

The wear monitoring can be activated and deactivated by the PLC using an interface signal.

3.10.5 Monitoring the additive offset

When using the "additive offset" function, using $TC_TP9=8 for the wear monitoring, additive offset monitoring can be selected. Instead of wear parameters $TC_DP12 ... 18, the additive offset parameters $TC_SCP12 ...18 are monitored. This corresponds to a defined 1st additive offset of the cutting edge. In turn, only the highest value of the additive offset parameter of all defined additive offsets is monitored.

3.10.6 Working with replacement tools

If replacement tools are used, it is necessary to define the rules relating to the transition to the replacement tool.

When changing a tool, only the name of the tool group is specified in the part program. This comprises several tools, that differ by their duplo number. The NCK selects, if available, the tool with the "active" state. If there is no tool with the "active" state, the next tool that can be used is selected whose value of the system variable $TC_TP10 regarding the other tools in the tool group is the lowest. The user allocates the parameter values so that tools are used in the required sequence. If values are not allocated, then the NCK automatically selects some sort of suitable tool.

3.10.7 Examples

Precondition

Activating the tool monitoring without active tool management for the option "tool monitoring":

MD18080 $MN_MM_TOOL_MANAGEMENT_MASK = 0x02
MD20310 $MC_TOOL_MANAGEMENT_MASK = 0x02

Service life monitoring for tool 2, cutting edge 1 in the NC program

```
$TC_TP9[2,1]=1                ;Activation of the service life monitoring
$TC_MOP1[2,1]=100             ;Prewarning limit in minutes
$TC_MOP2[2,1]=245             ;Residual tool life in minutes
$TC_MOP11[2,1]=800            ;Setpoint service life in minutes
```

Service life monitoring for the active tool with active D number in the NC program

```
$TC_TP9[$P_TOOLNO, $P_TOOL]=1           ;Activation of the service life monitoring
$TC_MOP1[$P_TOOLNO, $P_TOOL]=200        ;Prewarning limit in minutes
$TC_MOP2[$P_TOOLNO, $P_TOOL]=602        ;Residual tool life in minutes
$TC_MOP11[$P_TOOLNO, $P_TOOL]=700       ;Setpoint service life in minutes
```

3.10.8 NC language commands

General

The NC language commands, which apply for tool monitoring without tool management, are only listed here. A detailed description is provided in this documentation under the appropriate index entry.

- SETPIECE
  Decrement workpiece counter.
- RESETMON
  Language command for setpoint activation
- $A_MONIFACT
  Read factor for tool life monitoring.
- NEWT
  Create new tool.
- DELT
  Delete tool.
- GETT
  Read T number
- GETACTT
  Read the active internal T No.

The associated PI commands can also be used:

- _N_TMCRTO
- _N_TMCRTC
- _N_TMGETT
- _N_TRESMO

3.11 Variants of D-number assignments

3.11.1 Relative D No. for each T - standard

D numbers ranging from 1 to max. 12 are available for every T = "identifier" (with TM) or for every T number (without TM). These D numbers are assigned directly to the tool cutting edges.

An data record ($TC_DPx[t,d]) belongs to each D number (= cutting edge number).

D0 is the offset deselection code.

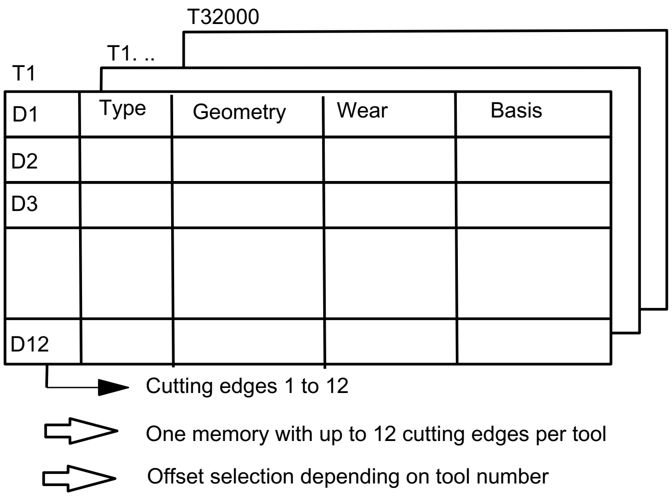

T32000
T1. ..
T1
D1
Type
Geometry
Wear
Basis
D2
D3
D12
Cutting edges 1 to 12
One memory with up to 12 cutting edges per tool
Offset selection depending on tool number

Image 3-20 Structure of the tool offset memory

3.11.2 Absolute D No. without reference to the T number (flat D No.)

Independence between D number and T number can be selected as an alternative to relative D numbers in systems without tool management.

The reference of T number, cutting edge and offset by the D number is defined by the user.

The range of D numbers is between 1 and 32000. D0 is the offset deselection code.

Note

The T number is always output to the PLC with an extended address (= spindle or tool holder number) with this type of tool offset.

Uniform system support is not available for this function.

(SINUMERIK Operate does not support the function)

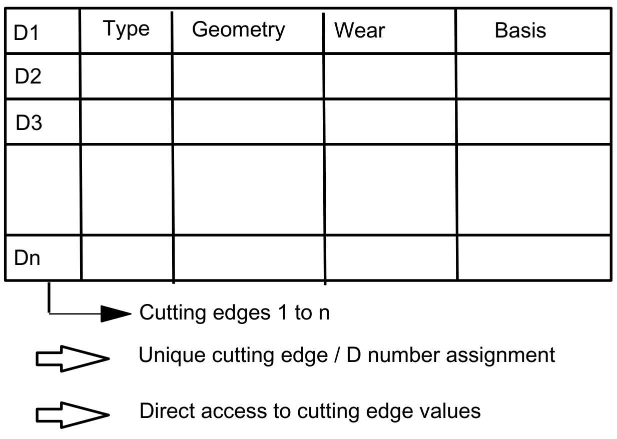

D1
D2
D3
Dn
Type
Geometry
Wear
Basis
Cutting edges 1 to n
Unique cutting edge / D number assignment
Direct access to cutting edge values

Image 3-21 Structure of the tool offset memory

3.11.3 Free selection of D numbers for every T

D numbers can be freely assigned to tool edge numbers in systems with and without tool management. A maximum of 12 cutting edges are possible for each tool "T". The upper limit for the D numbers used is limited by the machine data.

This assignment option is an extension of the process to relative D numbers.

With this setting ($MN_MM_MAX_CUTTING_EDGE_NO > $MN_MM_MAX_CUTTING_EDGE_PER_TOOL) additional program commands can be used that make a check for unambiguous assignment of D numbers to T numbers or identifiers possible. With this setting, the cutting edge number $TC_DPCE[T-No,D-No] is also created

The same D numbers shall be assigned in each case for the cutting edges for duplo tools (same identifiers).

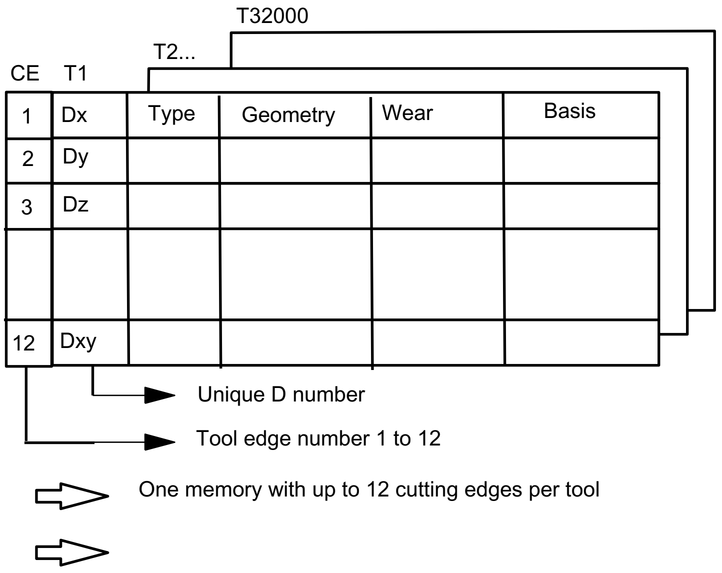


Image 3-22 Structure of the tool offset memory

Note

Uniform system support is not available for this function.

Machine data for user (unique) assignment of D numbers

$MN_MAX_CUTTING_EDGE_NO = Maximum permissible D number

Example:

```
$MN_MAX_CUTTING_EDGE_NO=1              A maximum of 1 offset (D1) can be defined per
                                       tool.
$MN_MAX_CUTTING_EDGE_NO=9999           Tools can be assigned unique D numbers as
                                       follows:
                                       T1 with D1, D2, D3
                                       T2 with D10, D20, D30
                                       T3 with D100, D200, D300
```

$MN_MAX_CUTTING_EDGE_PER_TOOL =Assignment of tool edges per tool

Example:

```
$MN_MAX_CUTTING_EDGE_PER_TOOL=1        Only tools used with 1 cutting edge
$MN_MAX_CUTTING_EDGE_PER_TOOL=12       Up to 12 cutting edges per tool.
```

Check for uniqueness (CHKDNO)

The NC language command **CHKDNO** checks the D numbers assigned within the NCK for uniqueness. The D numbers of all tools defined within a TO unit may not occur more than once. No allowance is made for replacement tools.

Check within the magazine (CHKDM)

Exactly like CHKNO, the NC language command **CHKDM** checks the D numbers assigned within the NCK for an activated tool management for uniqueness. This check function can be restricted to individual magazines.

D number to T number (GETACTTD)

Using the NC language command **GETACTTD** if tool management is active, the T number in which the D number occurs can be searched for.

Precondition for this is that the D numbers have been uniquely assigned in the TO unit being considered.

GETDNO, SETDNO when re-equipping

The NC language commands **GETDNO** and **SETDNO** permit reading and writing the offset number D for a specified cutting-edge number CE.

GETDNO (T, CE): Read the D number for the cutting edge CE of the tool T

SETDNO (T, CE, D): Set the D number for the cutting edge CE of the tool T

$TC_DPCE[T, D]=...: Assignment of tool edge number CE to offset number D

Example:

Rename cutting edge CE=3 from D2 to **D17**

- With initial situation:
  Internal T number 1
  D number: 2
  Tool 1 cutting edge with:
  $TC_DP2[1, 2]=120 ;tool radius T1, D2: 120 mm
  $TC_DP3[1, 2]=5.5 ;tool radius T1, D2: 5.5 mm
  $TC_DPCE[1, 2]=3 ;cutting edge number T1, D2: 3
  (programming: T1,...D2)
- using variable definition:
  DEF INT DNoOld, DNoNew=17
  DnOld=**GETDNO** (1, 3) ;value 2 is read in DnOld
  SETDNO (1, 3, DNoNew) ;the new D no. is assigned to the cutting edge
- The new D value 17 is assigned to cutting edge CE=3
  $TC_DP2[1,**17**]=120
  $TC_DP3[1, **17**]=5.5
  $TC_DPCE[1, **17**]=3

3.11.4 Location-dependent offsets (additive offsets)

Overview

Location-dependent offsets are a generalized form of wear. They are part of the cutting edge data. The parameters of the additive offset refer to the geometrical data of a cutting edge.

Location-dependent offset can in general be used, i.e. with active/inactive tool management; with flat D-number function.

To meet the requirements of special machine operating modes, the location-dependent offsets can be subdivided into the following categories by setting the appropriate machine data:

- Location-dependent offset, fine
- Location-dependent offsets, coarse = setting-up offset

The purpose of the setting-up offset is to allow the operator to set values prior to the machining operation. These values are stored in their own memory in the NCK, the operator can access the location-dependent fine offsets via the HMI. Location-dependent offsets "fine" and location-dependent offsets "coarse" are added NCK-internally and then act like the additive offset itself.

Several location-dependent offsets can be defined per D number. Machine data define the absolute number of location-dependent offsets, the maximum number of location-dependent offsets per cutting edge and specify which additive offsets are active after the end of program or when the Reset key is pressed.

Applicable only when tool management is active:

MD18112 $MN_MM_KIND_OF_SUMCOR can be set to define which additive offset must be operative if a tool is assigned the "active" status in the part program in the coarse of a programmed tool change:

- Additive offset values "fine" of the tool cutting edges remain unchanged or
- Additive offset values "fine" of the tool cutting edges are set to the value 0

The function is enabled by setting bit 8 = 1 in machine data $MN_MM_TOOL_MANAGEMENT_MASK.

DL - programming the additive/setting-up offset

Programming the additive offset is always relative to the active D number and is executed using the command

DL ="n"

The additive offset with the relative number "n" with respect to the active D number is activated by this. This means that the additive offset "n" is added to the wear of the active D number.

The additive offset is deselected with the command

DL = 0

Configuration of additive/setting-up offset

$MN_MM_KIND_OF_SUMCORR, bit 4=0

Corresponds to the default setting; only one data record of additive offset available per DL number. In this case, the term "additive offset" merely refers to the data represented by $TC_SCPx.

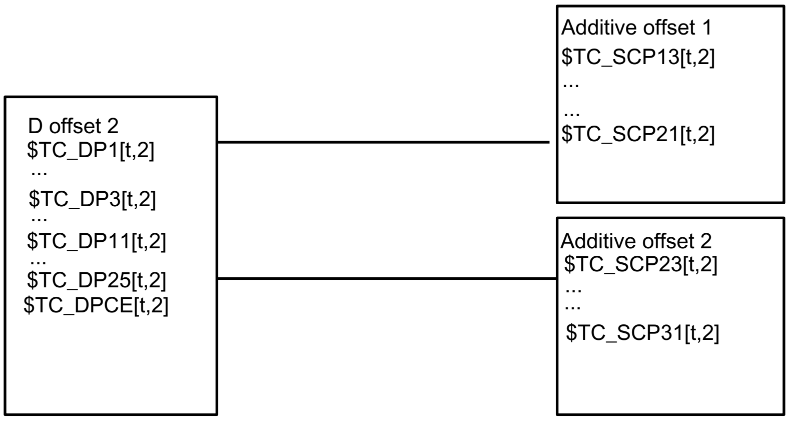


Image 3-23 $MN_MM_KIND_OF_SUMCORR, bit 4=0

The data in the diagram is used when programming (the tool with T=t is active):

```
D2              ;Cutting edge offsets
                ;i.e. $TC_DP3,...$TC_DP11 + wear ($TC_DP12,...$DP29) + adapter dimen-
                sion
                ...
DL=1            ;Additive offset 1 is added to the existing offsets of D2
                ;i.e. $TC_SCP13,...$TC_SCP21
                ...
DL=2            ;Additive offset 1 is no longer added to offset D2, but additive off-
                set 2 instead
                ;i.e. $TC_SCP23,...$TC_SCP31
                ...
DL=0            ;Deselection of additive offset; only the data of D2 is still effec-
                tive
```

$MN_MM_KIND_OF_SUMCORR, bit 4=1

Setting-up offsets are available. The general term "additive offset" refers to a combination of the "fine" additive offset, represented by $TC_SCPx, and the additive offset, represented by $TC_ECPx. There are two data records for one DL number. The additive offset equals the product of the corresponding components $TC_SCPx + $TC_ECPx.

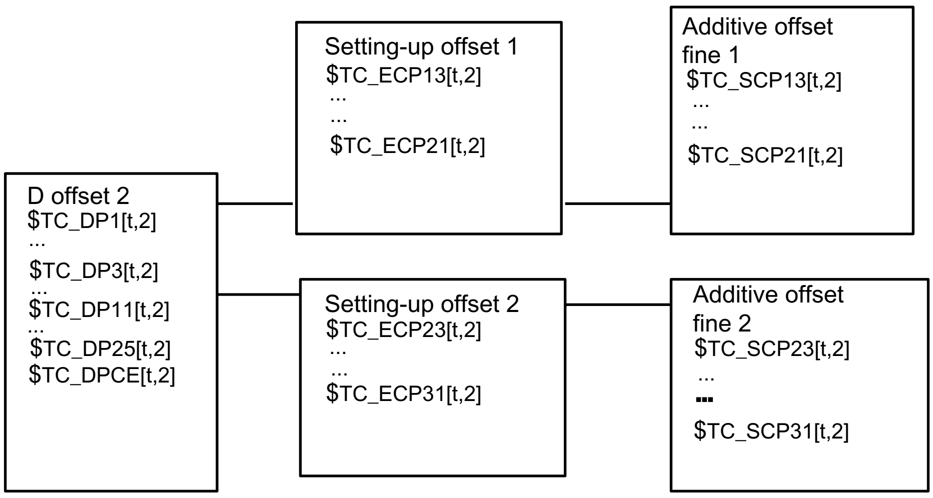


Image 3-24 $MN_MM_KIND_OF_SUMCORR, bit 4=1

The data in the diagram is used when programming (the tool with T=t is active):

```
D2              ;Cutting edge offsets
                ;i.e. $TC_DP3,...$TC_DP11 + wear ($TC_DP12,...$DP29) + adapter dimen-
                sion
                ...
DL=1            ;Additive offset 1 is added to the existing offsets of D2
                ;i.e. $TC_ECP13 + $TC_SCP13 ,...$TC_ECP21 + $TC_SCP21
                ...
```

```
DL=2            ;Additive offset 1 is no longer added to offset D2, but additive off-
                set 2 instead
                ;i.e. $TC_ECP23 + $TC_SCP23,... $TC_ECP31 + $TC_SCP31
                ...
DL=0            ;Deselection of additive offset; only the data of D2 is still effec-
                tive
```

The new NC language command DELDL can be used to delete location-dependent offsets from cutting edges.

3.12 Adapter data

3.12.1 Overview

Purpose

With the system variables **$TC_DP21, $TC_DP22** and **$TC_DP23**, the standard data record for the tool offset offers the option of entering the dimensions for an adapter (length1, length2 and length3).
This data is defined offset-specific.

If tool management is active the additional adapter data can also be assigned to specific magazine locations.

This function is used for adapters that are fixed to a magazine location for a long period and used by different types of tool.

In individual cases, it is also possible to use identical adapters on several magazine locations. To do this it makes sense to define and store the adapter data records separately from the magazine locations.

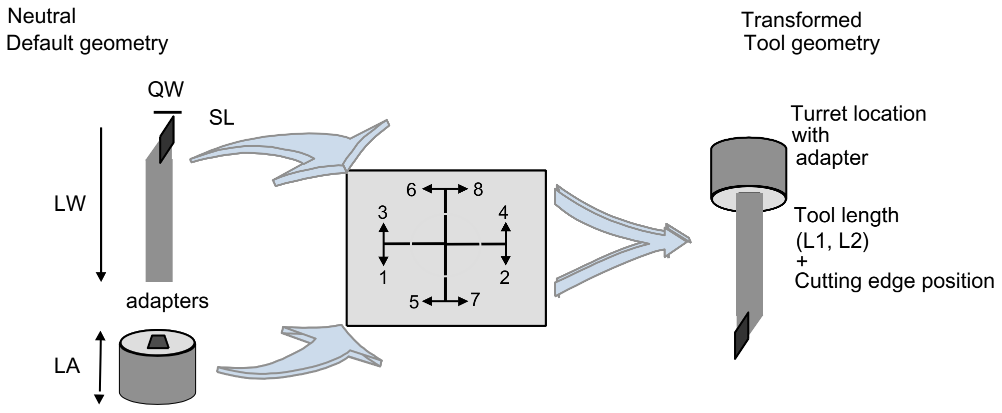


Image 3-25 Adapter transformation

Adapter transformation

Adapter data "adapter transformation" allows fixed orientation of the tool on the adapter or orientation of the adapter including its tool with reference to the machine.

This function can be used as an alternative to the previous one. If adapter data are used, system variables $TC_DP21, $TC_DP22 and $TC_DP23 have a different reference and are therefore only formally part of the cutting edge data record in the NCK.

3.12.2 Description of function

The adapter data function must be enabled via machine data (MD18104: $MN_MN_MM_NUM_TOOL_ADAPTER).

In order to activate the setting, bit 7 must be set in MD18080 MM_TOOL_MANAGEMENT_MASK.

Definitions

The machine data can be used to set two definition types of adapter data:

- One adapter data record is assigned to each magazine location as standard.
- Adapter data records can be defined independently of magazine locations. The magazine locations are then assigned as an additional step.

The magazine location is the reference point for adapter **and** tool. Both are assigned to the magazine location.

The following elements are implemented when programming D numbers in the part program:

- The offset must be assigned to a real tool.
- The tool is assigned to a magazine location.
- It is possible to assign an adapter to the magazine location, for which a transformation (orientation) of the tool it contains can be defined.

Thus, the working offset can be clearly computed and the tool path accordingly adjusted.

If an additive offset is programmed, then its value refers to the active D offset.

3.12.3 Activation

Preconditions

- In order to be able to use magazine location oriented adapter data, machine data **MD18104 $MN_MM_NUM_TOOL_ADAPTER** must have a value that is not equal to zero.
- Adapter data records must be defined.
- If the values of the machine data are > 0 ,the adapters must be linked to the magazine locations or assigned to them (can be automated via the HMI or using a cycle).

As a result, the adapter data including the defined transformations are always taken into account for the tool located on the magazine location in question. The working offset is calculated including the transformation and the adapter data.

The offset data can then be displayed as follows:

- Geometrical data for the tool (system variable $TC_DP3,...DP11); designated as neutral default geometry
- Non-transformed working offset (sum of the values from tool geometry, wear, additive offset, tool base dimension or adapter)
- Non-transformed working offset (transformation of the sum of the values from tool geometry, wear, additive offset) and tool base dimension of the adapter).

The quantities to be transformed can be selected via machine data. The mode of transformation of the additive offset can be set.

Magazine-location-related adapter data records

Create new

MD18104 $MN_MM_NUM_TOOL_ADAPTER = -1:

One magazine location and one adapter data record are created. Default values are stored in the adapter data record which is automatically linked to the magazine location.

It is not possible to create a new free adapter at this point. The adapter numbers are assigned automatically (1 ... max. number of available magazine locations).

Delete

If an adapter data record is linked to a magazine location (MM_NUM_TOOL_ADAPTER = -1), it cannot be deleted.

Note

Only the setting $MN_MM_NUM_TOOL_ADAPTER = -1 is currently supported by HMI Operate.

Free adapter data records

Create new

MD18104 $MN_MM_NUM_TOOL_ADAPTER > 0:

The adapter data can be created freely. Adapter data can be created by the user with a write operation to a non-existent data record.

$TC_ADPTi[n] = value;i = T, 1 2, 3, ..., n (number of the adapter)

If data record n does not yet exist and the maximum number of adapter data records that have already been defined is less than the value of MD18104 **$MN_MM_NUM_TOOL_ADAPTER**, a new adapter data record is created and assigned specific values.

The value "value" is assigned to parameter i. The following applies, 0 < n ≤ 3 2000. The index value 0 is reserved.

Note

The adapters must be assigned explicitly to the magazine locations for MM_NUM_TOOL_ADAPTER > 0.

Delete

If, MD18104 MM_NUM_TOOL_ADAPTER has the value > 0, then the adapter data can be deleted as required provided it is not assigned to a magazine location.

$TC_ADPTT[n] = -1

Adapter data record n is deleted and the memory becomes free again.

Deleting an assigned adapter data record:

The assignment to the magazine location must be undone first. You can only do this if the magazine location is empty. An alarm is issued if deletion fails.

Please proceed as follows:

- Remove the tool from the magazine location (unload, relocate).
- Remove the adapter from the magazine location.
- Delete the adapter data record (with $TC_ADPTTi[n] = -1).

Adapter data record n is deleted and the memory becomes free again.

Deleting all adapter data records

For **MM_NUM_TOOL_ADAPTER > 0**, you can delete the adapter data if it is not assigned to a magazine location:

$TC_ADPTT[0] = -1

All non-assigned adapter data of the TO units is deleted. If you want to delete assigned adapters, you must first undo the assignment of those adapters to magazine locations. An alarm is issued if deletion fails.

Read/write adapter parameters

You can modify adapter data at any time even if an adapter is assigned to a magazine location and/or a tool is located in the magazine location with the adapter.

Magazine location assignment/release

For **MM_NUM_TOOL_ADAPTER > 0**, an adapter record must be assigned to a magazine location explicitly:

$TC_MPP7[m,p] = "adapter no."

Adapter number "adapter no." is assigned to magazine location p of magazine m. For "adapter no." = 0 any previous assignment is removed.

Note

An assignment can only be established or released if no tool is assigned to the magazine location.

Determination of the adapter

When determining the relevant adapter, a distinction is made between a turret and the other magazines.

The following applies for a turret:

The adapter is used on the owner location (i.e. the source location, whereby the tool only leaves the magazine location in the data) of the selected tool.

The following applies for chain and box-type magazines:

The adapter of the current magazine location containing the tool (usually the spindle) is used.

Example:

A tool magazine is changed from Magazine_1/Location_1 into the spindle. The adapter from Magazine_location_1 is not read now, but rather from the spindle location. This means that in the change cycle, the adapter data from the magazine location must be copied to the spindle location.

Example of an adapter transformation

A turning tool with lengths L and Q is described below.

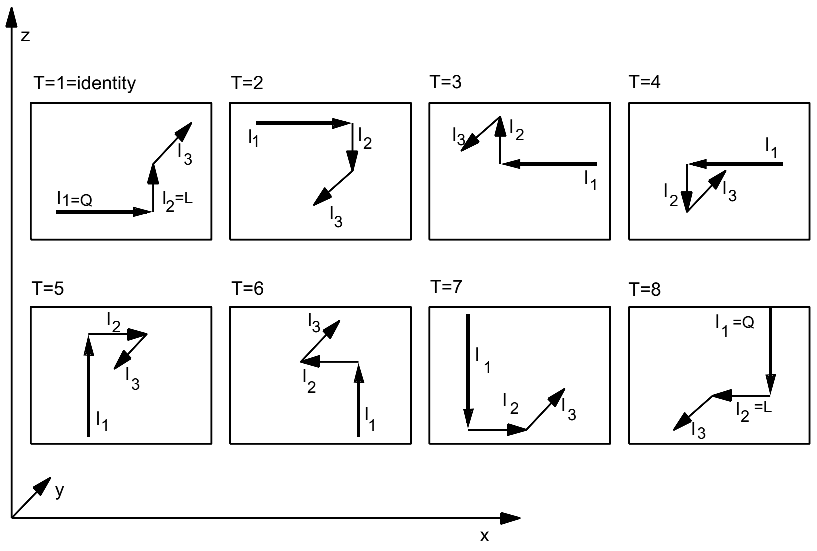


Image 3-26 The 8 defined transformations (T = 1...8) for the adapter with G 18 and for a turning tool. The assignments of tool lengths l1, l2, l3 are shown on the geometry axes x, y, z.

Transformations for numbers 1 to 8 are defined. Number 1 is the identity: no transformation of input data.

Other transformations can be implemented. The available transformations are designed initially for turning tools. These are typically defined by **Q=l_1=$TC_DP3** and **L=$l_2$=$TC_DP4**.

The transformation numbers correspond to the transformations shown in the table. In general:

Length1$_t$, length2$_t$, length3$_t$ = f(length1, length2, length3) = f(l_1,l_2,l_3) = f(Q,L,l_3)

Transformation number	GEO			
	X	Y	Z	Turning tool with G18 transformation
1	+L1	+L3	+L2	-
2	+L1	-L3	-L2	ROT X 180
3	-L1	-L3	+L2	ROT Z 180
4	-L1	+L3	-L2	ROT X 180 Z 180
5	+L2	-L3	+L1	ROT Y 90 Z 180
6	-L2	+L3	+L1	ROT Y -90

7	+L2	+L3	-L1	ROT Y 90
8	-L2	-L3	-L1	ROT Y -90 Z 180

L1, L2 and L3 are working offsets of the tool prior to transformation with or without adapter (depending on machine data settings). They are assigned to the geometry axes during compensation.

Note

In turning, L and Q are also used to describe a tool. In the above table, l_1 corresponds, for example, to variable Q (or x direction) and l_2 to variable L (or z direction), assuming the plane G18 is selected (default setting for turning machines).

As standard, activation of an offset is calculated as follows:

Offset = D offset + x_i(e.g. wear, additive offset)

Length1	\$TC_DP3 + x_i
Length2	\$TC_DP4 + x_{i+1}
Length3	\$TC_DP5 + x_{i+2}
Length4	\$TC_DP6 + x_{i+3}

The adapter transformation then acts on the transformed tool offset values and is added to the transformed offset values.

The transformation number of the adapter causes a transformation of the tool (the cutting edges) located in this adapter (orientation according to the transformation number).

Working offset = f(offset) + adapter dimension of the magazine location

aLength1	Length1_t + \$TC_ADPT1
aLength2	Length2_t+ \$TC_ADPT2
aLength3	Length3_t + \$TC_ADPT3
ARadius1	Radius1

Depending on the programmed plane selection G17, G18, G19, these values are added to the geometry axes.

G17, G18, G19 - plane selection (declarations)

The following agreements (different for machining and milling tools) apply for assigning tool-length parameters of the tools to the geometry axes:

Machining plane	System variables for tool length description		
	\$TC_DP3(l_1)	\$TC_DP4(l_2)	\$TC_DP5(l_3)
G17 Milling Turning	Z Y	Y X	X Z

G18 Milling **Turning**	Y **X**	X **Z**	Z **Y**
G19 Milling Turning	X Z	Z Y	Y X

Transformation of cutting edge position

The cutting edge position described by system variable $TC_DP2 is also transformed.

Transformations for the cutting edge position are performed as shown in the table below:

Transformation number	Cutting edge position								
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1	1	2	3	4	5	6	7	8	9
2	2	1	4	3	7	6	5	8	9
3	4	3	2	1	5	8	7	6	9
4	3	4	1	2	7	8	5	6	9
5	1	4	3	2	6	5	8	7	9
6	4	1	2	3	8	5	6	7	9
7	2	3	4	1	6	7	8	5	9
8	3	2	1	4	8	7	6	5	9

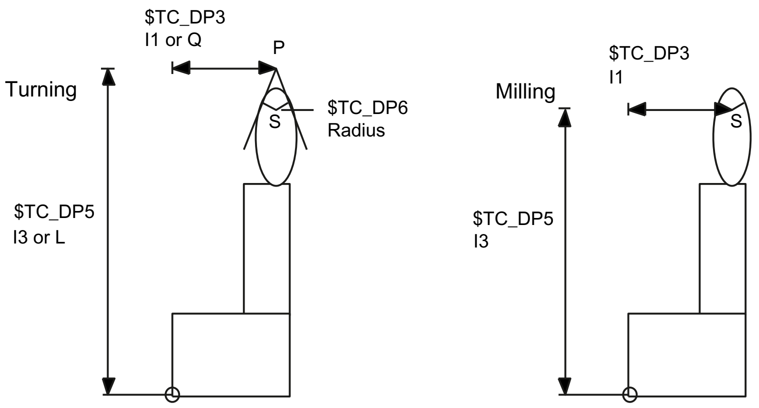


Image 3-27 Turning and milling tools - relationship between cutting edge position and radius compensation

Turning tool geometries (I_1, I_3 or L, Q) are described in terms of P, the point of approach at the workpiece. However, the center point of cutting edge S with reference to the tool nose radius must be known for radius compensation.

This center point can only be accurately calculated if the cutting edge position is known. Point S can then be derived from point P.

The position of the tool to the workpiece coordinate system is defined using the cutting edge position (values 1 ... 8). Cutting edge position 9 corresponds to S = P.

Note

The cutting edge position is only used for turning tools because their geometry is described with reference to P and not with reference to S as is the case for milling tools.

Adapter transformation for tools with three length components

The transformations defined here constitute a subset of all conceivable transformations. Only certain discrete values are considered here - in particular those that meet the requirements for turning tools (2 length components only).

System variables $TC_DP21 ... 23 and $TC_ADPT

If the function "Adapter" is active, then there is no further cutting-edge specific data for the "Base adapter dimension".

In order to keep cycles that operate with adapter data compatible, the following rules apply:

If a tool is at a magazine location with an adapter and the adapter data can be accessed by system variable $TC_DP21...23, then the adapter parameters of the location can be read and written.

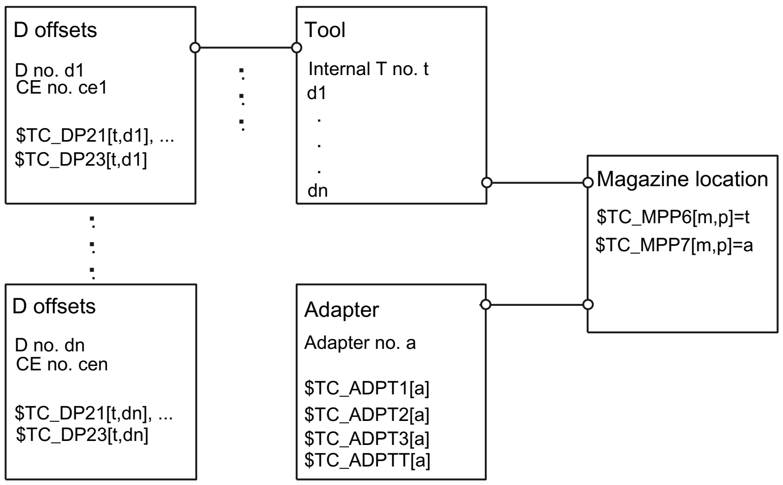


Image 3-28 Fig. 3-30 $TC_DP21, ...23 - Contents for an active "Adapter" function

Specifications:

- Tool t
- Magazine location p
- Magazine m
- Adapter a
- Tool with D offsets d_1, ... d_n

The adapter is assigned to the magazine location. If, for example, system variable $TC_DP21[t,d$_1$] is read or written in the part program, the programming accesses system variable $TC_ADPT1[a] of the adapter, i.e. the same machine data is accessed for all d$_1$, ... d$_n$.

If the assignment of the tool to the magazine location is released or the adapter is removed from the magazine location, no more data can be assigned to the parameters. A read operation returns the value 0, a write operation does not change the data (nor does it generate an alarm).

Transformed and non-transformed offset values

The values included in the path offset are usually the transformed working offsets.

It can generally be said that the data that describes a tool is subject to transformation. The transformation of the adapter is communicated to the tool (orientation in which it is positioned in the adapter). The adapter data itself is not transformed.

Data transfer to the NCK

You need to declare how the data is transferred to the NCK.

- You can transfer the data via the part program by programming the system variable $TC_... The parameters are defined as non-transformed values.
- The transfer can take place via the OPI using the variable services. In this case, the data can be transferred either as transformed or non-transformed values.

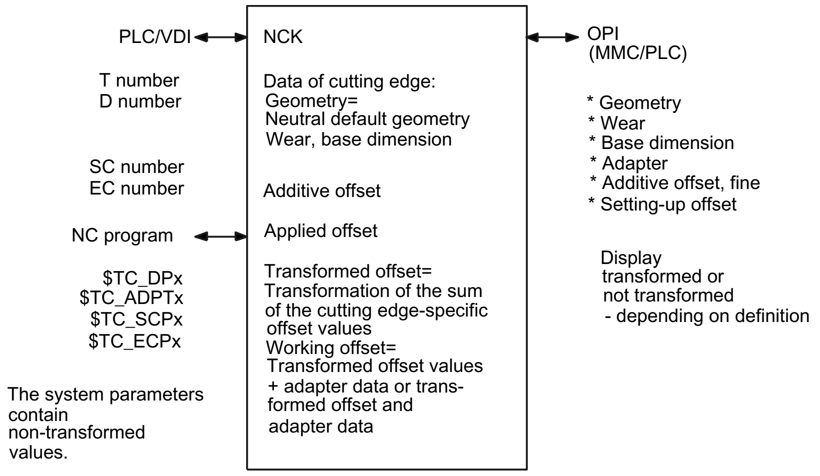


Image 3-29 Geometry of a tool edge and applied offsets

Boundary conditions

When using the function (magazine-location-oriented) "adapter data" the user must ensure that the old data records of all the data records with edge-specific adapter data are adapted to the requirements of the new function.

However, using the described edge-specific adapter parameter definition (system variables $TC_DP21,...23), it is possible that all old data is converted to the adapter data function by the NCK.

The function "Adapter data" excludes the existence of the cutting-edge specific data "base / adapter dimension".

The function "adapter data" is better suited to the applications of an adapter because it defines the adapter as part of the magazine location and not as part of the tool or cutting edge.

Example 1

Specifications:

- MM_NUM_TOOL_ADAPTER = -1
- MM_NUM_MAGAZIN_LOCATION = 20
- One chain with 16 locations, magazine number = 1
- Two grippers
- One spindle
- One loading and unloading point
- Assignment

When creating the 20 locations in all, 20 adapters should be assigned, i.e. exactly one adapter assigned to each location.

Note

It does not matter if the real locations are not actually fitted with an adapter. Preassigned adapter locations have no effect on the offset. When equipping a location with a real adapter make sure that the appropriate values are assigned to the adapter data.

The transformation number of the adapter in location 3 of the chain magazine (No. 1) is to be changed to the new value 8:

$TC_ADPTT[$TC_MPP7[1,3]] = 8	;$TC_MPP7 contains the number of the adapter at the new magazine location

Once adapter data records have been automatically generated and assigned, operations such as undoing an assignment, renewed definition of an assignment and deletion of an adapter data record are possible.

Example 2

Specifications:

- MM_NUM_TOOL_ADAPTER = 4
- MM_NUM_MAGAZIN_LOCATION = 20
- One chain with 16 locations
- Two grippers

- One spindle
- One loading and unloading point

There are 4 different adapter geometries in this case. Adapters must be configured for the chain only.

Assignment

These locations (20 in total) are initially created without adapters. Locations 1 to 4 of the chain are equipped with adapters of the same geometry (here adapter 1). 4 chain locations are to be equipped with adapters with the same geometry.

First, you must define the 4 adapter data records. Now you assign them:

$TC_MPP7[1,1] = 1	$TC_MPP7[1,13] = 4
$TC_MPP7[1,2] = 1	$TC_MPP7[1,14] = 4
$TC_MPP7[1,3] = 1	$TC_MPP7[1,15] = 4
$TC_MPP7[1,4] = 1	$TC_MPP7[1,16] = 4
….	

In this way you can assign one adapter data record to several magazine locations.

Note

If you want to delete an adapter data record with a multiple assignment you must make sure that you first undo all the adapter assignments.

3.12.4 Transformed data of the active tool $P_ADT[n]

A new system variable is introduced that reads the offset parameters of the active tool offset transformed according to the tool adapter $TC_DP1,... etc. Refer here to the system variable $P_AD (Section "Additional language commands (Page 350)") that reads the non-transformed parameters.

$P_AD and $P_ADT have the same meaning without the function "Tool adapter" - as sub-function of the function TMMG. In other words, the system variable application is only meaningful within the scope of the TMMG function.

With active function "Tool adapter", the $P_ADT provides when reading the offset parameters, transformed values of those parameters that are subject to the tool-adapter transformation in the event that the active tool is on a tool adapter at the point in time the parameters are read. The parameters not subjected to the transformation still provide the same values during reading as $P_AD.

When writing, the transformed parameter values subject to the tool-adapter transformation are accordingly transformed back by the NCK and are subsequently saved in a non-transformed form in the NCK. Non-transformed values are still written with $P_AD.

3.13 Power failure while tool command is in progress

If a power failure occurs during an action requested by tool management, defined strategies are executed by the PLC or special part programs, in order to establish a defined and consistent status on the machine and the tool management system. These strategies are machine-specific. SINUMERIK control systems thereby support the following measures:

Buffered data

All tools, magazine data as well as all magazine assignments are buffered.

Data initialized for control "Power On"

The following data is set to zero:

- Tool status "Tool change in progress"
- Magazine status "Motion is active"
- Magazine location status "Reserved for tool to be loaded"
- The PI-command status with respect to magazine operations like e.g. "Motion is active".
- The tool status selected for 1:1 exchange

Task of the manufacturer's configuration

The PLC must send the last unacknowledged FC 7 or FC 8/FC 6 prior to power failure (READY did not change to TRUE before power off) back to the NC when the power supply is restored. The function "Asynchronous transfer" is used for tool transfer in FC 8/FC 6.

Without receiving the request from the tool management system, the PLC initiates a relocation of tool data from one location to another. For example, relocation of tool data from gripper to magazine if the tool needed to be returned manually to the magazine when the tool change operation was aborted.

Changes in position of the tools involved must be communicated via FC 8. The NC then updates the data for this tool in the tool management.

Further strategies may be necessary, e.g. if a tool change was interrupted. Tools stored in the buffer must be returned to the magazine for this purpose.

3.14 PLC description

3.14.1 Interfaces

Overview

The interfaces in the PLC consist of data blocks that are updated by the basic program. Tasks such as load tool or prepare tool change are stored in the data blocks with the source and target for the particular tool. For the interfaces for spindle or turret, in addition to the tool numbers (internal number, which is assigned by the NCK when loading), tool size and tool status are transferred.

If the position of the tool changes (e.g. from magazine to gripper...), the new positions must be transferred to the tool management on the NCK. Three function blocks **FC6** (TM_TRANS2), **FC7** (TM_REV) and **FC8** (TM_TRANS) are provided for this purpose. The PLC programmer can call these blocks and supply them with the required parameters.

If a magazine or a turret is not driven by an auxiliary axis, the shortest direction of rotation can be calculated with **FC22** (TM_DIR) and the positioning time optimized. FC18 is available if positioning is performed using an auxiliary axis of the 840D sl.

Commissioning tool management

The precondition for commissioning the PLC is that the NC commissioning has been completed. The machine data must be correctly and completely set and the magazine configuration loaded. The basic program sets-up the user interface (DB 71 ... DB 73, DB1071 ... DB1073) as well as the internal data block (DB74) when it boots. For information about this, which magazine, number of loading points, etc. is in DB 4 or is automatically entered by the NCK (see Section "Load the machine manufacturer PLC blocks (Page 215)").

If tool management is configured using HMI Adv, the entry in the DB is made using the softkey "Create PLC data"; DB4 can also be "manually" supplied (see Section "Signal description of the PLC interface (Page 509)").

Note

If data blocks DB71...DB74 are already available, then the basic program first automatically deletes them, and then the new DB is set-up.

Data blocks DB1071 - DB1073 are automatically set-up by the basic program for the multitool. This means that the blocks are also set-up, if the "Multitool" function has not been set using the appropriate machine data.

Overview of data blocks

Block number	Length in byte	Meaning
DB71	4 + 30 bytes * B	Interface for loading/unloading points
DB72	4 + 48 bytes * W	Interface for spindle as change position
DB73	4 + 44 bytes * R	Interface for tool turrets as change position
DB1071	20 byte * B	Extended data for the multitool
DB1072	50 byte * W	Extended data for the multitool
DB1073	50 byte * R	Extended data for the multitool
DB74	Length depends on configu-ration	Internal data block for tool management

B = number of loading locations

W= Number of spindles as change positions

R= Number of turrets

DB71 to DB74 occupy approximately 550 bytes for simple configurations of magazines, buffers and loading/unloading points.

There is one interface (data record) per data block for each loading/unloading point, spindle and turret. The data blocks are assigned to the respective tasks.

DB71

DB71 assumes the functions of loading and unloading, positioning and relocating. The relocating and positioning functions (no matter from which NC channel) are generally mapped on the 1st interface.

DB72

DB72 is the interface for changing tools into the spindle. This change procedure also includes preparation of the tool. (Setting: $MC_TOOL_CHANGE_MODE=1)

DB73

DB73 is the interface for tool changes with a circular magazine. (Setting: $MC_TOOL_CHANGE_MODE=0)

DB1071

Extended data for the multitool
Loading/unloading, relocating and positioning. The structure is analog to DB 71.

DB1072

Extended data for the multitool
T preparation and M06. The structure is analog to DB 72.

DB1073

Extended data for the multitool
Change with T command. The structure is analog to DB 73.

DB74

DB74 is an internal tool management data block used for communication control. You must not write to this data block.

For all the interfaces listed here, source and target positions are available for the tools associated with the machining operation.

FB15 is called in the basic program for communication between the NCK and the PLC when tool management is active. This block informs the user interfaces (DB71 to DB73 and DB1071 to DB1073) if a tool management function is activated via the part program or operator input.

Interfaces within DB71 to DB73

A bit field for the active and passive status of each interface is contained in bytes 0 and 1 of the respective data block (DB71 to DB73). DBX 0.0 represents the first interface, DBX 0.1 the second, etc. A total of 16 interfaces can be addressed. If one of these bits is set to the value = 1 by the tool management, the associated interface is activated. If set to 0, the interface may not be processed by the user.

Principle of the interfaces DB71-73

No. 8	No. 7	No. 6	No. 5	No. 4	No. 3	No. 2	No. 1
No. 16	No. 15	No. 14	No. 13	No. 12	No. 11	No. 10	No. 9
1st interface							
2nd interface							
....							
14th interface							
15th interface							
16th interface							

If the value = 1, the user must evaluate the commands at this interface and initiate the necessary actions (e.g. position magazines, change tools, etc.). After activation, user programmers can also write access this interface (e.g. in order to delete the "prepare change" bit). Communication with the tool management is realized exclusively via FC8 or FC7 or FC6, not via DB 71, 72, 73, ... (the only exception is the fast acknowledgement). For each change to tool positions and/or status information from an interface request, FC8/FC6 should be called with these modified values.

Tasks from NCK tool management

Tasks	Interface	Acknowledgement	Applications, special features
Loading	DB71	FC8/FC6, TaskIdent = 1, TaskIdentNo = interface no.	NewToolPlace = target destination of the tool NewToolMag = target magazine OldToolPlace = No. of the loading or unloading point OldToolMag = 9999 or: OldToolMag/OldToolPlace = 0 Status = 1
Unloading	DB71	FC8/FC6, TaskIdent = 1, TaskIdentNo = interface no.	NewToolPlace = No. of the loading or unloading point NewToolMag = 9999 OldToolPlace = Location of the tool to be unloaded. OldToolMag = Magazine No. of the tool to be unloaded (real or also buffer magazine) or: OldToolMag/OldToolPlace = 0, status = 1

Relocating	DB71	FC8/FC6, TaskIdent = 1, TaskIdentNo= 1	NewToolPlace = target position of the tool
			NewToolMag = target magazine
			OldToolPlace source location
			OldToolMag = source magazine
			Status = 1

Positioning	DB71	FC8/FC6, TaskIdent = 1, TaskIdentNo = interface no.	Positioning at the loading magazine corresponding to the interface number - any positioning, always via interface 1
			NewToolPlace = LMG or BL
			NewToolMag = 9999 or 9998
			OldToolPlace = magazine location to be positioned
			OldToolMag = magazine to be positioned
			Status = 5

Prepare change for tool in spindle	DB72	FC8/FC6, TaskIdent = 2, TaskIdentNo = interface no.	Positioning NewTool to change position, OldTool remains in spindle to complete status 1, in order that the change command can be output.
			OldToolPlace = BL (spindle)
			NewToolPlace = NewTool location

Change in spindle	DB72	FC8/FC6, TaskIdent = 2, TaskIdentNo = interface no.	OldTool is unloaded (gripper or directly into magazine), NewTool is loaded to spindle. Status 1 required, so that part program processing is continued.
			NewToolPlace = BL (spindle)
			OldToolPlace = OldTool location

Without NCK Command: Return Old-Tool to magazine		FC8/FC6, TaskIdent = 4, TaskIdentNo = channel	The OldTool may need to be transferred asynchronously to the location specified in the prepare change command to move the tool from the gripper to the magazine.

Tasks	Interface	Acknowledgement	Applications, special features
Change with turret	DB73	Normally FC7 or FC8/FC6, TaskIdent = 3, TaskIdentNo = turret no.	When turret has finished swiveling, FC 7 is called with turret No. As parameter ChgdRevNo.

LMG: Loading point

BL: Buffer location

T: Tool

NewToolPlace: FC8/FC6 parameter NewToolMag, NewToolLoc

OldToolPlace: FC8/FC6 parameter OldToolMag, OldToolLoc

Position changes of tools without task from NCK

Tasks	Acknowledgement	Applications, special features

Asynchronous transfer	FC8/FC6, TaskIdent = 4,
	TaskIdentNo = channel no.
	Status = 1
		General tool transport from the PLC
		NewToolMag/NewToolLoc = target
		OldToolMag/OldToolLoc = source

Asynchronous transfer with location res-ervation for tool transport from a real magazine into the buffer	FC8/FC6, TaskIdent = 5,
	TaskIdentNo = channel no.
	Status = 1
		Tool transport from PLC
		NewToolMag/NewToolLoc = buffer
		OldToolMag/OldToolLoc = source in the real magazine

Asynchronous transfer without tool movement to align the magazine posi-tion	FC8/FC6, TaskIdent = 4,
	TaskIdentNo = channel no.
	Status = 5
		Communicates the actual magazine po-sition (e.g. after turret cycles) to the NCK
		NewToolMag/NewToolLoc = actual lo-cation
		OldToolMag/OldToolLoc = change or loading point

Further interfaces in the channel interfaces for the ToolMan function

DBD348	T number for tool pre-alarm limit
DBD352	T number for tool limit value
DBD356	T number of the new replacement tool
DBD360	T number of the last replacement tool

Change bits in DBB344

The information can be evaluated within one OB 1 cycle on the basis of a change bit. The PLC can use this information to derive appropriate measures.

Note

The last replacement tool monitoring must be set in machine data $MC_TOOL_MANAGEMENT_MASK.

Other signals are as follows:

	Channel DB	
Tool missing	DBX317.7	From NCK
Do not disable tool (ignore disabled state)	DBX29.7	to NCK
Deactivate wear monitoring	DBX29.6	to NCK
Deactivate workpiece counter	DBX29.5	to NCK
Activate time monitoring by PLC The function must be set in machine data $MC_TOOL_MANAGEMENT_MASK	DBX1.3	to NCK

Additional interfaces in the NCK interface (DB10) for the tool management function

DB10.DBX105.0	Abort of all tool commands in the NCK The signal is intended for commissioning. While the negative acknowledgement (status 3) specifically interrupts a command, with DB10.DBX105.0, all NCK commands are interrupted, i.e. in all channels.

3.14.2 Definitions of acknowledgement status

Magazine identifier

Real magazines

The number range from 1 ...9997 is available for the real magazine, defined when generating the magazine configuration (the numbering must not be ascending).

Buffer magazine

The buffer magazine always has the number 9998. The number of locations in the buffer magazine is defined using the magazine configuration. The locations with the "spindle / tool holder" identifier correspond to an interface

Loading magazine

The loading magazine always has the number 9999. A magazine location corresponds to a loading point (defined when generating the magazine configuration), every loading point corresponds to an interface (loading point 1 = interface 1, etc.). Loading point 1 has a special significance. Every relocation operation, unloading and loading a manual tool and manual loading, unloading is generally managed via this as basis.

Status value 1 - 10

The status information 1 to 10 (current upper limit 10) leads to the command being terminated. As a consequence, the "active bit" of the interface is reset to 0 and the operation is completed.

Status value > 100

When acknowledging this status information (FC8/FC6), the "active" bit of the relevant interface remains at "1". Further processing is required by the user program in the PLC (e.g. continuation of magazine positioning). This item of status information is generally used to transfer changes in position of one or both tools while the operation is still in progress.

Synchronization

There are various methods by which the PLC and NCK can be synchronized. Synchronization is forced by MD20310 $MC_ TOOL_MANAGEMENT_MASK using bits 5, 6, 7, 8 and 19. For internal communication between the PLC and NCK, the devices wait for an acknowledgement after each command. We distinguish between two types of acknowledgement:

- Transport acknowledgement
  The transport acknowledgement indicates to the NCK that the issued command has been accepted by the basic PLC program. Before a new command is output, the system checks whether the previous command was accepted. If this is not the case, no output is made. The NCK waits for the acknowledgement before a new command is output.
- End acknowledgement
  Status feedback signal of the PLC for an NCK command that has been accepted with acknowledgement status 1, 10 or acknowledgement status 3.

Output of the command

Synchronization of the NCK and PLC is implemented in three steps:

- The interpolation task from NCK has prepared a command on the basis of the programming and outputs this to the NCK-internal image of the VDI interface.
- The NCK-internal image of the VDI interface is transferred in the same cycle to the VDI.
- The basic PLC program accepts the command from the VDI interface.

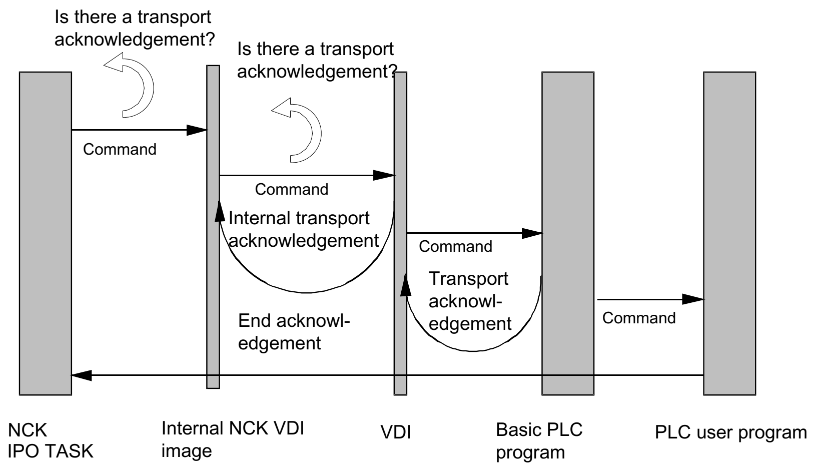

Is there a transport acknowledgement?
Is there a transport acknowledgement?
Command
Command
Internal transport acknowledgement
Command
Transport acknowl-edgement
Command
End acknowl-edgement
NCK IPO TASK
Internal NCK VDI image
VDI
Basic PLC program
PLC user program

Image 3-30 Transport and end acknowledgement

Acknowledgement of output commands

Acknowledgements from the basic PLC program and from the VDI are returned while the command that has been output is being executed.

- The basic PLC program outputs the transport acknowledgement to the NCK once the command has been accepted.
- An internal transport acknowledgement is issued after the internal VDI image has been transmitted.

The PLC user program can only process one command at a time. It determines how long it takes until a command is processed. If the NCK provides the command faster than can be processed by the PLC user program, then the NCK is put into the waiting state.

The NCK can also output commands which do not originate from the part program over the interface. Included here are PI services that are synchronously superimposed over the part program processing.

Completion of the command

Depending on how bits 5-8 and 19 of the MD20310 $MC_TOOL_MNAGEMENT_MASK are set, the command output is considered to have been completed at various points in time.

- If bit 5 (or bit 6 for the secondary spindle) is set, the command output is completed when the internal transport acknowledgement and the transport acknowledgement are available. The command has been accepted by the basic PLC program.
- If bit 7 is set (or 8 for the secondary spindle), this means that the command output is only completed when the end acknowledgement is received from the PLC.
- If the bits are not set, then the output of the command is considered as being completed when the NCK has output the command to the NCK-internal VDI image. This is the default setting.

Note

From the viewpoint of the tool-change command, the block change can take place as soon as the NCK has output the command.

The set bit 19 in conjunction with set bits 5-8 permit a block change to be prevented as long as the extended acknowledgements are not present.

Changing the acknowledgement data

When acknowledging a command from the NCK, the PLC can change the parameters of this command in the acknowledgement data.

The following sequence is implemented to allow acknowledged commands from the PLC to be assigned in the NCK:

- Using the ID number in the acknowledgement command (this is tracked by the basic program), the tool in the NCK is determined.
- The data of the current tool location are obtained from the tool.
- This current tool location is checked against the address specified in the command.

- If this data does not match, then after the acknowledgement data of the PLC has been checked, it is accepted in the NCK data management.
- Acknowledgement of the command in the NCK is continued.

Note

If the tool to be loaded at change is transported from the magazine to the tool holder in multiple individual steps, the PLC acknowledgement number 105 applies.

With the PI command _N_TMMVTL or the analog language command "MVTOOL", a tool in status "being changed" cannot be moved.

The following applies for loading, relocating, reloading and positioning: The PLC must not change the target positions specified by the NCK for the NewTool as they have to be identical with those in the NCK.

Example 1

A tool is being prepared and already with the preparation is to be brought into gripper_1 (preparation). The change moves the new tool from gripper_1 to the spindle and moves the old tool into gripper_2. This completes the change.

The NCK command looks like this:
Bring the new tool (T No.6) from magazine_3, location_6 to the spindle and the old tool from the spindle (9998/1) into magazine_3, location_11.

ID:00000/00000-------- CMD:00002

NewTool:
from M: 00003 P: 00006 **to M**: **09998** P: **00001** TNo: 00006 Spindle: 00001

OldTool:
from M: 09998 P: 00001 **to M**: **00003** P: **00011**

As a result of this change sequence, this task is not executed like this. When acknowledging, the PLC program must change the data shown in bold.

There are two options to do this:

Case_1
The PLC synchronously acknowledges the preparation command

ID:00000/00001-------- ACK:00002 St: 00001

NewTool:
from M: 00003 P: 00006 to M: 09998 P: 00002

OldTool:
from M: 09998 P: 00001 to M: 09998 P: 00001

Case_2
The PLC asynchronously transports the prepared tool to the gripper and now synchronously acknowledges the preparation command to the end.

ID:00000/00001-------- ACK:00008 St: 00001

NewTool:
from M: 00003 P: 00006 to M: 09998 P: 00002

OldTool:
from M: 00000 P: 00000 to M: 00000 P: 00000

ID:00000/00002-------- ACK:00002 St: 00001

NewTool:
from M: 09998 P: 00002 to M: 09998 P: 00002

OldTool:
from M: 09998 P: 00001 to M: 09998 P: 00001

In both cases, using the end acknowledgement, the NCK checks the tool data acknowledged by the PLC against the data in the command - and after an internal command assignment, corrects the command data, in order that additional acknowledgements are made with valid data (new tool located on 9998/2). The source file ("from") for the old tool and the new tool are managed by the basic program, a change is not possible.

3.14.3 Simplified acknowledgement of tool management commands

Overview

Previously, it was only possible to acknowledge tool management commands via the user program with FC8/FC6 or FC7. Now, for the standard acknowledgement with the end status per interface, there is a bit available within DB71, DB72, DB73 which can be used to acknowledge the command using the cyclic basic program. For this purpose, acknowledgement bits are defined in DB71, DB72, DB73 in DBW2, which the user program must set to acknowledge an active command for one PLC cycle. When the acknowledgement bit is set for the pending task, then end acknowledgements are generated.

The acknowledgement parameters used are described in the following table:

Function	New tool	Old tool	Status
Loading	Target position (n+24, n+26)	0	1
Unloading	0	Specified loading point	1
Relocating	Target position (n+24, n+26)	0	1 or 6
Positioning	Target position (n+24, n+26)	0	5
Prepare change	Actual position or 0	Actual position or 0	1
Change	Target position (n+16, n+18) or 0 (for T0)	Target position (n+24, n+26)	1
Change with preparation	Target position (n+16, n+18) or 0 (for T0)	Target position (n+24, n+26)	1
Change with turret	Target position (n+16, n+18) or 0 (for T0)	Target position (n+24, n+26)	1

If active commands with fault are to be acknowledged, this can also be realized using the acknowledgement bit in DBW2. In addition, for the particular task, in DBX(n+1.0) the value 1 must be connected in parallel to set the acknowledgement bit. The acknowledgement parameters are listed in the following table.

For DBX(n+1.0), n is the start address of the interface in the particular data block.

Function	New tool	Old tool	Status
Loading	Current position	0	3
Unloading	0	Current position	3

Function	New tool	Old tool	Status
Relocating	Current position	0	3
Positioning	Current position	0	3
Prepare change	Actual position or 0 (for T0)	Actual position or 0	3
Change	Actual position or 0 (for T0)	Actual position or 0	3
Change with preparation	Actual position or 0 (for T0)	Actual position or 0	3
Change with turret	Actual position or 0 (for T0)	Actual position or 0	3

When processing an acknowledgement using the acknowledgement bits, the acknowledgement bit and also DBX(n+1.0) are automatically reset by the basic program.

Note

Before setting the acknowledgement bit, additional intermediate acknowledgements (with the status > 100), are realized from the user program via the FC8/FC6 (e.g. re-storing in the gripper).

Relocating

If a tool is relocated from the real magazine into the spindle or to a buffer location, acknowledgement is with Status_6, i.e. the source location in the real magazine is reserved for the tool.

3.14.4 Diagnostics for communication between NC and PLC

General

The NCL-PLC communication can be logged in a file as part of the tool change procedure.

Note

The diagnostics data is saved with the reset or cancel key.

Precondition

- Bit 13 of MD20310 $MC_TOOL_MANAGEMENT_MASK must be set.
- The trace is a ring buffer (_NTCTRA'xx'MPF, with 'xx' = channel number 01, 02, ...). Free memory space must be available on the NC for saving the data. The number of files in the file system must be below the maximum number of files.
- The trace size can be set using the machine data $MN_TOOL_MANAGEMENT_TRACE_BUFFER_SZ[Index] = number of commands
  Index 0 = buffer size of the IPO trace (prepare, change commands and acknowledgements, asynchronous transfers).
  Index 1 = buffer size of the preparatory trace (commands and acknowledgements due to PI services).
  The Trace representation with index 0 also assumes the commands and acknowledgements due to PI services, the trace file with index 1 is then irrelevant.

Example

Milling machine with a chain magazine, a spindle and double gripper (any tool is located on the spindle)

```
T="Miller_30mm"        - The tool remains at its location, after positioning the mag-
                       azine, the preparation is acknowledged with "end"
M06                    Step 1 (is acknowledged with Status_105)
                       - Old tool comes from the spindle into Gripper_1
                       - New tool comes from the magazine into Gripper_2
                       Step 2 (is acknowledged with Status_1 (end))
                       - New tool comes from the Gripper_2 into the spindle
                       - Old tool comes from the Gripper_1 into the magazine
```

The recorded trace (**TCTRA'xx'_MPF)**, looks like this:

ID:00002/00002 ------------ CMD:00002
NewTool: from M: 00003 P: 00006 to M: 09998 P: 00001 TNo: 00006 Spindle: 00001
OldTool: from M: 09998 P: 00001 to M: 00003 P: 00002

ID:00002/00003 ------------ ACK:00002 St: 00001
NewTool: from M: 00003 P: 00006 to M: 00003 P: 00006
OldTool: from M: 09998 P: 00001 to M: 09998 P: 00001

ID:00003/00003 ------------ CMD:00003
NewTool: from M: 00003 P: 00006 to M: 09998 P: 00001 TNo: 00006 Spindle: 00001
OldTool: from M: 09998 P: 00001 to M: 00003 P: 00002

ID:00003/00004 ------------ ACK:00003 St: 00105
NewTool: from: M: 00003 P: 00006 to M: 09998 P: 00003
OldTool: from M: 09998 P: 00001 to M: 09998 P: 00002

ID:00003/00005 ------------ ACK:00003 St: 00001
NewTool: from: M: 09998 P: 00003 to M: 09998 P: 00001
OldTool: from M: 09998 P: 00002 to M: 00003 P: 00002

Explanation

The ID number is used by the NCK to assign acknowledgements. The first number specifies the command ID of the output command. The second number specifies the command identification with which the PLC had acknowledged the command.

Definition of command identification: Every command via the tool management interface from the NCK to the PLC is allocated a unique identification number. PLC (the basic program, not the user program) acknowledges a command with this command number. This means that it is possible to assign the acknowledged command to the command that was output.

Every NCK command (CMD) has the following structure:

1. The command ID (this is incremented at each command output)
2. The command no. (CMD preparation, change, ...)
3. The transport for the tool to be loaded at change (from - to)
4. The internal T No. of the tool to be loaded at change
5. The number of the requesting spindle or tool holder
6. The transport for the tool to be replaced (from - to)

Every PLC acknowledgement (ACK) has the following structure:

1. From the ID No. The ID of the NCK is returned + a PLC ID
2. The command no. (NCK sends a CMD:00002, PLC responds with an ACK:00002)
3. The acknowledgement status (see FC8/FC6 description)
4. The transport for the tool to be loaded at change (from - to)
5. The transport for the tool to be replaced (from - to)

List of values and meanings for CMD and ACK

CMD	Explanation
1	A tool is transported from ... to ... . Loading, unloading, relocating, positioning.
2	Tool change is to be prepared (setting MD22550 = 1)
3	Tool change is to be executed (setting MD22550 = 1)
4	Tool change is to be prepared and carried out (setting MD 22550 = 0)
5	Tool change is to be prepared and carried out (setting MD 22550 = 1)

ACK	Explanation
1	Tool is/was transported. Loading, unloading, relocating, positioning. FC8/FC6 - parameter TaskIdent = 1
2	Tool is/was prepared (setting MD22550 = 1) FC8/FC6 - parameter TaskIdent = 2
3	Tool is/was executed (setting MD22550 = 1) FC8/FC6 - parameter TaskIdent = 2
4	Tool change is/was prepared (setting MD22550 = 0) FC8/FC6 - parameter TaskIdent = 3

ACK	Explanation
5	Tool change is/was prepared (setting MD22550 = 1) FC8/FC6 - parameter TaskIdent = 2
7	Terminate canceled tool command DB10.DBX105.0=1
8	Tool was transported. If a tool is present at the source address, its data is transported to the target address. Otherwise, only the current magazine position is changed. If the tool transport is from a real magazine, the location to which the source address points is reserved. FC8/FC6 - parameter TaskIdent = 5
9	Tool was transported. If a tool is present at the source address, its data is transported to the target address. Otherwise, only the current magazine position is changed. FC8/FC6 - parameter TaskIdent = 4

3.14.5 Function blocks

Overview

Block number	Meaning
FC7	Transfer block for tool change with turret
FC8	Transfer block for tool management, call at position and status changes
FC6	Transfer block for tool management, call at position and status changes. FC 6 is required for the "multitool" function. The block corresponds to the FC8. It has an additional parameter "MultitoolPosition". The FC6 includes the complete FC8 functionality, so that the FC6 can completely replace the FC8.
FC22	Direction selection for shortest path

All function blocks are described in: /FB1/ P3, Basic PLC Program

Additional PLC services

In addition to the function blocks given above, there are further PLC services available for more complex requirements on the part of the PLC user program to influence tool management. These services are possible using FB2, FB3, FB4 and FB7 (read and write variables or PI services). A description of these function blocks is part of the basic PLC program description. The PI services (program instances) of the tool management are also described in the basic PLC program description regarding FB4 and FB7. The tool management variables are described in the lists in the section on variables.

Commissioning 4

4.1 Input of the machine data

General machine data

Machine data for memory partitions, assignment of channels to TO units have to be set for tool management. Also, memory will be needed in the battery-buffered RAM. When "memory-influencing" machine data is changed, i.e. at next power on, restart or cold restart (reboot), this memory area is deleted and configured again. Therefore, data must be backed up prior to reset/cold restart.

Order when releasing memory using the machine data

Option: More than three magazines (if required)

Activate the function "Tool management" (tool monitoring+magazine management)

MD18080	MM_TOOL_MANAGEMENT_MASK Activate the memory for tool management

MD17500	MAXNUM_REPLACEMENT_TOOLS Maximum number of replacement tools
MD17510	TOOL_UNLOAD_MASK Behavior of tool data at unloading
MD17515	TOOL_RESETMON_MASK Behavior of tool data at RESETMON
MD17520	TOOL_DEFAULT_DATA_MASK Create new tool: Data default setting
MD17530	TOOL_DATA_CHANGE_COUNTER Identifying tool data change for HMI
MD17540	TOOLTYPES_ALLOWED Permitted tool types
MD18074	MM_TOOL_MANAGEMENT_TRACE_SZ Maximum size of the tool management diagnostics ring buffer
MD18075	MM_NUM_TOOLHOLDERS Max. number of tool holders
MD18076	MM_NUM_LOCS_WITH_DISTANCE Max. number of magazine locations per TOA with distance connections
MD18077	MM_NUM_DIST_REL_PER_MAGLOC Max. number of magazines in the distance table of a magazine location

MD18078	MM_MAX_NUM_OF_HIERARCHIES
	Maximum number of definable hierarchies for magazine location types
MD18079	MM_MAX_HIERARCHY_ENTRIES
	Maximum permissible number of entries in a magazine-location-type hierarchy

Definition of number of magazines and magazine locations

MD18084	MM_NUM_MAGAZINE
	Maximum number of magazines that NCK can manage (minimum of 3 magazines). Buffer and loading magazines must be added (i.e. for 2 real magazines, "4" must be specified)!
MD18086	MM_NUM_MAGAZINE_LOCATION
	Number of magazine locations that the NCK can manage. Buffer and loading locations must be added!

Definition of tools and tool cutting edges

MD18082	MM_NUM_TOOL
	Number of tools to be managed by the NCK
MD18100	MM_NUM_CUTTING_EDGES_IN_TOA
	Number of cutting edges in NCK, tool offsets per TOA block
MD1810:	MM_MAX_CUTTING_EDGE_PERTOOL
	Maximum number of cutting edges (D offset) per tool (per T number)

Options for providing additional user data for magazines, magazine locations, tools and tool edges

MD18090	MM_NUM_CC_MAGAZINE_PARAM
	Number of additional magazine data $TC_MAPCx[n]
MD18091	MM_TYPE_CC_MAGAZINE_PARAM
	Type definition for magazine-oriented user data
MD18092	MM_NUM_CC_MAGLOC_PARAM
	Number of additional magazine location data $TC_MPPCx[n,m] generated
MD18093	MM_TYPE_CC_MAGLOC_PARAM
	Type definition for magazine location-oriented user data
MD18094	MM_NUM_CC_TDA_PARAM
	Number of additional tool-specific data per tool $TC_TPPCx[t] generated
MD18095	MM_TYPE_CC_TDA_PARAM
	Type definition for tool-oriented user data
MD18096	MM_NUM_CC_TOA_PARAM
	Number of additional data per tool edge $TC_DPCx[t,d] generated
MD18097	MM_TYPE_CC_TOA_PARAM
	Type definition for cutting edge-oriented user data

MD18098	MM_NUM_CC_MON_PARAM Number of additional monitoring data per tool edge $TC_MOPCx[t,d] generated
MD18099	MM_TYPE_CC_MON_PARAM Type definition for monitoring-related user data

Channel-specific machine data

Enabling of channel-specific functions for tool management.

MD20310	TOOL_MANAGEMENT_MASK Channel-specific activation of tool management

Specification of spindle number for tool life monitoring

MD20320	TOOL_TIME_MONITOR_MASK Activation of tool life monitoring for the spindle specified here (tool holder number)

Tool change turret or spindle.

MD22550	TOOL_CHANGE_MODE Change with M06 or with T selection
MD22560	TOOL_CHANGE_M_MODE M function for tool change

Cutting edge selection after tool change.

MD20270	CUTTING_EDGE_DEFAULT Offset selection or deselection at tool change

Definition of tool with which tool offset is to be selected as a function of MD20110 and MD20112 during power-up and reset.

MD20122	TOOL_RESET_NAME Definition for selection of tool length compensation

Definition of the active tool holder number.

MD20124	TOOL_MANAGEMENT_TOOLHOLDER Definition of the active tool holder number

Assignment of TO units to channels.

MD28085	MM_LINK_TOA_UNIT Allocation of a TOA range to a channel

Definition of initial setting for control after boot, reset, end of part program in relation to G code, tool length compensation and transformation.

MD20110	RESET_MODE_MASK Definition of the control basic setting. Relevant bit = 0: The actual value remains valid.

Note

For machine data 20310 TOOL_MANAGEMENT_MASK and 18080 MM_TOOL_MANAGEMENT_MASK, bits 0-3 must always be set the same.

4.2 Load the machine manufacturer PLC blocks

4.2.1 Overview

Overview

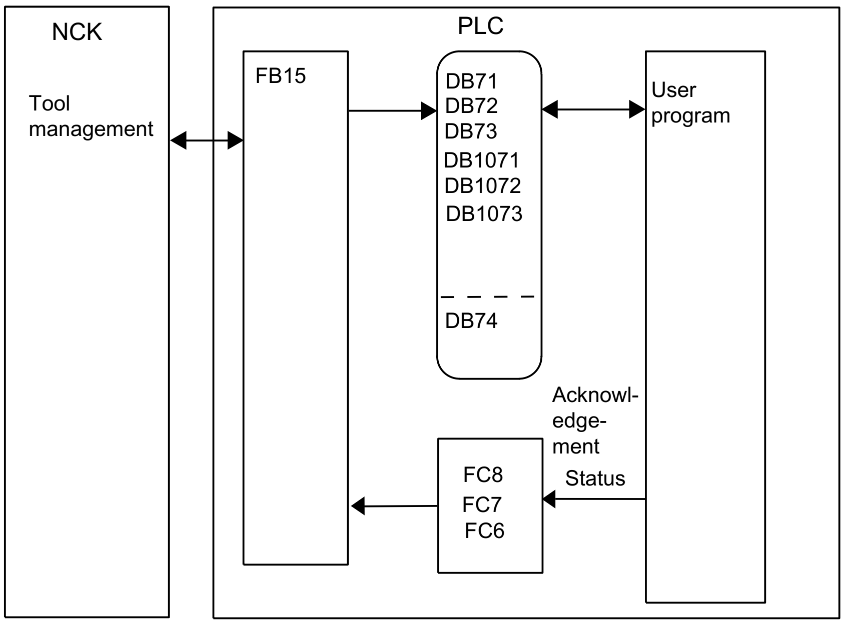


Image 4-1 Commissioning the PLC Program

The basic program supplies the tool management interfaces (data blocks DB71-DB73) with information for the new and old tool. The user must process this data from the active interface in the user program and ensure that the tools (old and new) are placed on the respectively associated positions (magazine, location). In order for the tool management to always know where a tool is located, each time a tool changes location the new location must be transferred to the tool management via the FC7 or FC8/FC6 acknowledgement status.

4.2.2 Create PLC data

Commissioning tool management

If the magazine configuration was created (for all channels involved), the PLC data must be generated. This is done as follows

- a) Manually, by pre-assigning DB4
- b) By pressing the "Generate PLC data" softkey on HMI-Advanced if the magazine configuration was generated with it.
- c) The PLC user interface is set-up automatically
  During run-up, the NCK checks the magazine configuration in all channels and transfers the magazine description via the VDI interface directly to the basic PLC program.
  The following sequence is defined:
  Is the magazine description from the NCK available (and no entry in DB4)?
  - This is used and the user interface set-up accordingly; the magazine description is NOT copied to DB4

  Is a magazine description available in DB4?
  - This is used and the user interface set-up accordingly

  This means that the entry in DB4 has priority. This ensures full compatibility.
  If there is neither a magazine description from the NCK nor one in DB4, the user interface is set-up with the maximum configuration (16 interfaces each).

FC8/FC6 (FC7 for circular magazines), TM_TRANS / TM_TRANS2 (transfer blocks) and if required FC22 TM_DIR (selection of direction) must still be loaded and called by the user program.

When commissioning has been completed, the next time the PLC is booted the following data blocks are set up for the user (user interfaces for tool management) in addition to a data block for the tool management FCs. The lengths of the data blocks are derived from the commissioning parameters in tool management.

Example of chain magazine

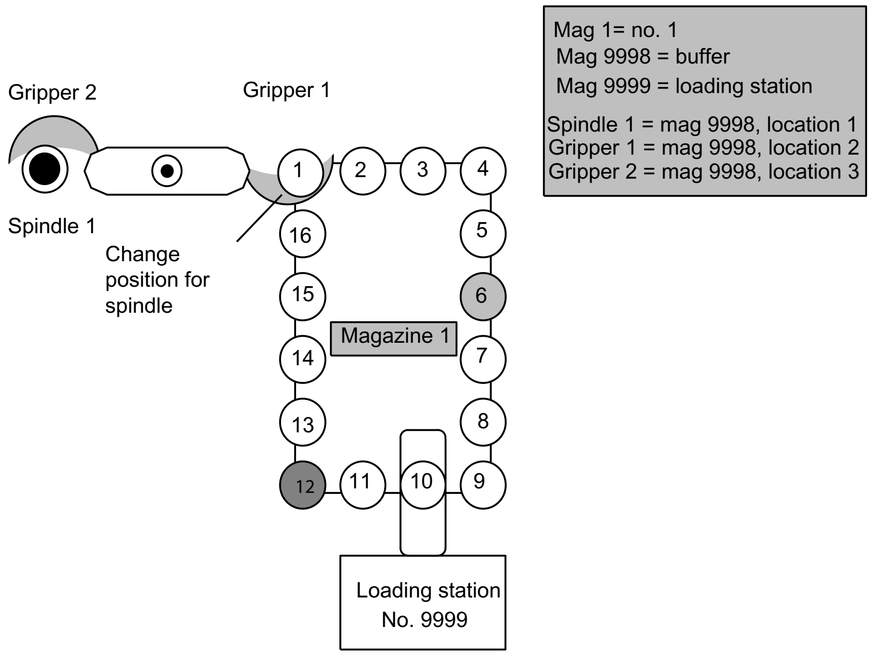

Mag 1= no. 1
Mag 9998 = buffer
Mag 9999 = loading station
Spindle 1 = mag 9998, location 1
Gripper 1 = mag 9998, location 2
Gripper 2 = mag 9998, location 3
Gripper 2
Gripper 1
Spindle 1
Change position for spindle
Magazine 1
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Loading station No. 9999

Image 4-2 Example of a magazine with gripper and loading station

"Drill120" is placed in location 6 and location 12 is reserved for the spindle tools to be exchanged.

Execution example for tool change

1. Part program contains T="Drill120"
   Output to the PLC:
   "PREPARE CHANGE" DBB(n+0) bit 2=1
   (bring new tool from Mag1, location 6 to Mag9998, location 1 and bring old tool from Mag9998, location 1 to Mag1, location 12).
2. Location 6 is moved to the point of change.
3. The tool is taken from location 6 and placed into gripper 1. The user program resets "PREPARE CHANGE" DBB(n+0) bit 2 to zero. The new position (9998, 2) of the new tool ("Drill120") is signaled via FC 8 with status 1. The old tool remains at position 9998,1. Bit 0.0 in DB72 is reset by FC8/FC6.
   The magazine is moved with location 12 to the change position for the old tool to be placed into it.
4. M06 is executed in the part program
   Output to PLC. "CHANGE" DBB(n+0) bit 1=1
   No new tool positions are entered in the interface with output of the M06 command. If required, they can be later made by the user program at change of position.

5. The PLC user program carries out the tool change and brings the tool into the spindle. During this process, the old tool is removed from the spindle and placed into gripper 2. The new tool in gripper 1 is placed into the spindle. FC8/FC6 acknowledges with status 105 (position of the new tool: 9998, 1; position of the old tool: 9998, 3).
6. The (old) tool is returned from gripper 2 to the new magazine location 12. This is acknowledged using FC8/FC, status 1 (position of new tool: 9998, 1; position of old tool 1, 12). This represents the end of the tool change procedure. Bit 0.0 in DB72 is reset by FC8/FC6.

Note

The timing of the tool change can be optimized by applying the following strategy for further processing in the part program:

In step 5, use status 1 with FC8/FC6 instead of status 105. The old tool is then returned to storage in step 6 with the asynchronous FC8/FC6 transfer function (status 1, OldToolMag=9998, OldToolMag=1, NewToolLoc=12).

4.2.3 Description of the test blocks

Overview of test blocks

Block No.	Structure	Meaning
FC40	Subprogram	Preparation of the data for a change with gripper via asynchronous transfer
FC41	Block to be called in OB1	Global functions (task control, check commands, H decoder, ...)
FC42	Subprogram	Supply of data for FC 8/FC 6 if a task is active
DB62	Data for active tasks, control parameter	
DB63	Data for FC22	
DB64	Data for asynchronous transfer	

Test blocks for tool management

To test the tool management function, blocks FC40, FC 41, FC 42 and data blocks DB62, DB63 and DB64 must be loaded to the PLC. FC41 (without parameters) must also be called in the organization block 1 (OB1). The following overall procedure is implemented by integrating these blocks:

1. The tool management function is activated (acknowledgement of tasks) by programming H9001 in the first channel (and deactivated with H9000).
   The system can also be activated by setting data bit DB62.DBX15.7. The initial setting when the PLC is rebooted is H9000. The other functions can only be used once the system has been activated via H9001.
2. The direction selection function (FC22) can be activated with the machine control panel (MCP) above the rapid traverse override key (the normal MCP connected via FC19 or FC25). Data must be written to DB63 (e.g. via the variable status) before the function is activated.
   Structure of data block DB63:
   Input parameters
   DBW0 = Magazine number
   DBW2 = Position setpoint
   DBW4 = Actual position
   DBW6 = Offset for special positioning
   Output parameters
   DBW8 = Differential position (shortest path)
   DBB10 = Rotation in CW direction == 1
   DBB11 = Rotation in CCW direction == 1
   DBB12 = Position reached
   DBB13 = Error == 1
   If an error (e.g. parameterizing error) occurs, the LED for the key lights up.
3. Every user interface (DB71 to DB73) is checked for active status by block FC41.
   If an interface is active, a transfer with new positions (usually target positions) and status information "1" (completed) is passed to the NCK immediately.

4. If H9003 is programmed in the first channel (corresponding to data block DB62.DBX15.6 set), the transfer operation described under point 3 is only executed after operating the MCP key above the minus-direction key.
   As a result, it is possible to intervene in the transfer values via a status function. The function is deactivated via H9002 (default setting). The transfer blocks are provided in data block DB62.
   Input parameters:
   DBB0 = Task identifier (1, 2, 3)
   DBB1 = Task number
   (only make changes in DBW2 to DBW10)
   DBW2 = Magazine for new tool
   DBW4 = Location for new tool
   DBW6 = Magazine for old tool
   DBW8 = Location for old tool
   DBW10 = Status information (see description of FC8/FC6)
   Output parameters:
   DBW12 = Error has occurred
   If an error occurs, the LED for the activation key lights up.
   The following functions are implemented for command acknowledgement in DB71, DB72, DB73:
   Load/unload, relocate:
   The required target positions are acknowledged with status 1 via FC8/FC6.
   Position:
   The required target position is acknowledged with status 5 via FC8/FC6 because the tool remains in the magazine.
   Prepare change (spindle interface):
   "New tool" remains at the original location, "Old tool" remains in the spindle.
   Special treatment is implemented for T0 or empty spindle.
   Acknowledgement is via FC8/FC6 with status = 1.
   Change (spindle interface):
   "Old tool" is transferred to allocated magazine location, "New tool" is loaded into the spindle.
   Acknowledgement is via FC8/FC6 with status = 1.
   Special treatment is implemented for T0 or empty spindle.
   Change (turret interface):
   Acknowledgement is via FC7.
   Optionally with DB62.DBX15.4 = 1 acknowledgement is via FC8/FC6 with status = 1.

5. Values other than zero can be set in DB62.DBW20 and DB62.DBW22. DB62.DBW20 means the spindle number and DB62.DBW22 the buffer number of a gripper assigned to the spindle.
   It is thus possible to automatically allow for a gripper located *between a spindle* and a *magazine* in the acknowledgement.
   The following sequence is implemented (only for spindle as change position, M06 setting as change command):
   When preparing, the behavior is identical to "normal operation".
   The "new tool" remains in the magazine, the "old tool" remains in the spindle. The "old tool" must continue to machine.
   On the change command:
   "New tool" goes into the spindle, "old tool" goes into the gripper. An asynchronous transfer is used to move the "old tool" to the suggested magazine location. A manual acknowledgement is required for this purpose.
6. Asynchronous transfer (changes in a tool location can be communicated without an NCK task)
   DB64 can be used to communicate a change in position of a tool to the tool management function in the NCK.
   The position of the tool was changed by the PLC. Entries must be made in DB64 (e.g. via variable status). Asynchronous transfer can then be subsequently started via DB64.DBX14.0 = 1.
   Asynchronous transfer with location reservation can be selected via the data DB64.DBX15.4 = 1.
   This corresponds to TaskIdent = 5.
   For a value of zero in the specified data, TaskIdent = 4 is activated.
   Input parameters:
   DBB1 = Associated NC channel number
   DBW2 = Original magazine of the tool
   DBW4 = Original location of the tool
   DBW6 = Target magazine of the tool
   DBW8 = Target location of the tool
   DBW10 = Status information (see description of FC8/FC6)
   Only status = 1 and status = 5 are permitted.
   Output parameters:
   DBW12 = Error has occurred

Note

If incorrect values are communicated from the NCK, the following error signals causing PLC stop are output and either displayed via the HMI or entered in the diagnostics buffer of the PLC.

The test blocks only use FC7 and FC8/FC6. The multitool function is presently not supported by the test blocks.

Alarm 400604:

In function 4 the specified magazine is not a turret.

Remedy: Machine data (tool change with M06 command)

4.2.4 Delete pending tasks

General

The communication initiated by the NC yet interrupted by the PLC task "Delete pending task" (DB10.DBX105.0) can be terminated by the PLC during commissioning.

The function cancels all pending tool management tasks from the NCK (compare NC switch-on). The NC tool management is reset in a defined manner. This is intended to be a pure commissioning function. If a command must be cancelled in regular operation, then this is done using FC8/FC6 and acknowledgement status "3" (negative acknowledgement).

This function enables direct intervention by the operator to, for example, take a tool out of the gripper where a change is just about to take place, or if there is no acknowledgement from the PLC program.

Note

Please ensure that the data consistency in the NC is maintained.

Boundary condition

The "Delete pending task" function can be activated only if the NC is in the "Channel not active" state.

Programming 5

5.1 Overview of OPI and system variables

Fundamentals

All the data required for the purpose of data management (e.g. to define a magazine or load a tool...) is stored in the NCK. 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 variables selector.

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 in the diagram corresponds to the OPI numbering sequence.

Note

System variables are available for OEM Siemens data. However, they are not described here because they are not meaningful at present.

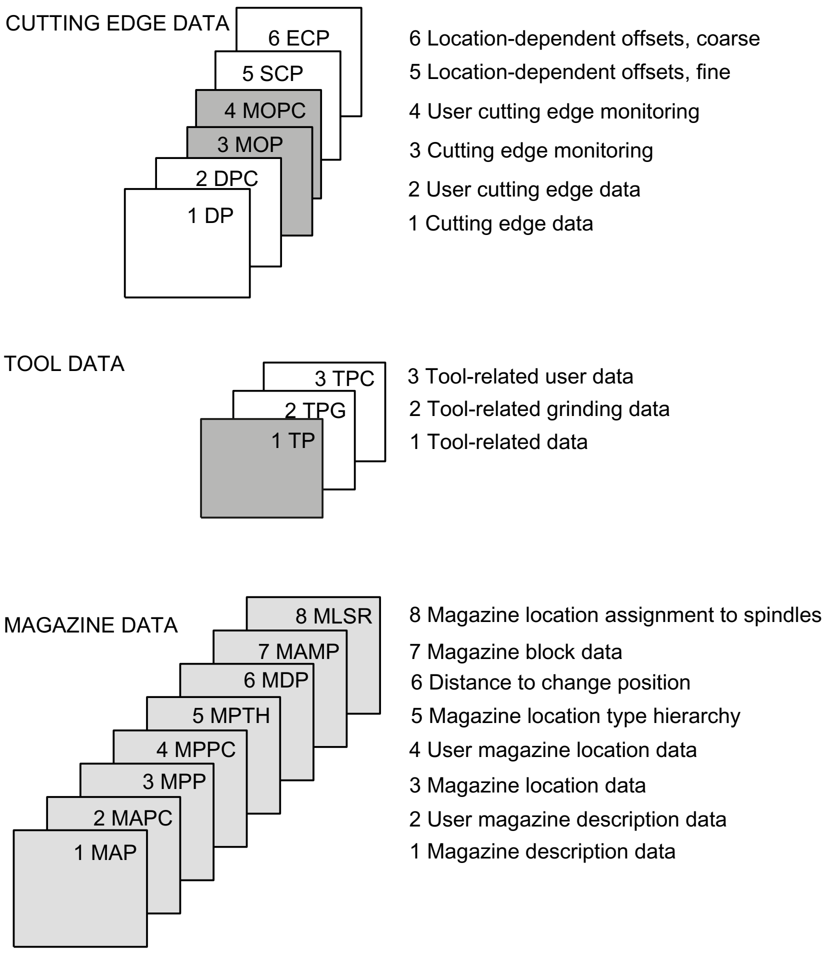

CUTTING EDGE DATA
6 ECP
5 SCP
4 MOPC
3 MOP
2 DPC
1 DP
6 Location-dependent offsets, coarse
5 Location-dependent offsets, fine
4 User cutting edge monitoring
3 Cutting edge monitoring
2 User cutting edge data
1 Cutting edge data
TOOL DATA
3 TPC
2 TPG
1 TP
3 Tool-related user data
2 Tool-related grinding data
1 Tool-related data
MAGAZINE DATA
8 MLSR
7 MAMP
6 MDP
5 MPTH
4 MPPC
3 MPP
2 MAPC
1 MAP
8 Magazine location assignment to spindles
7 Magazine block data
6 Distance to change position
5 Magazine location type hierarchy
4 User magazine location data
3 Magazine location data
2 User magazine description data
1 Magazine description data

Image 5-1 Overview of cutting edge, tool and magazine data

The identifiers (DP,...PP,...MAP,...) 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 also even if tool management is not active.

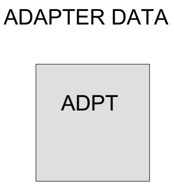


Image 5-2 Adapter data

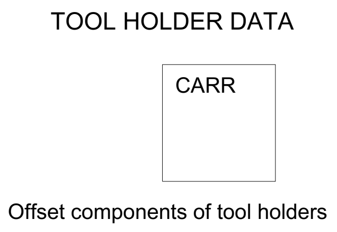


Image 5-3 Tool holder data

5.2 Cutting edge data

5.2.1 Cutting edge data

Cutting edge data

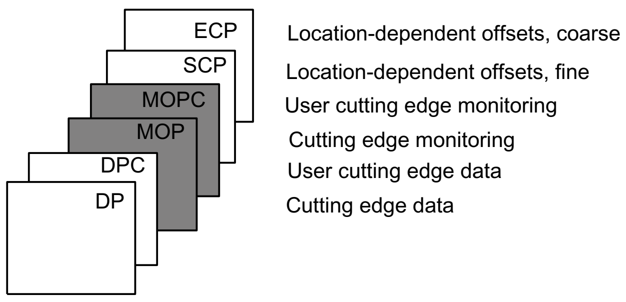

ECP
SCP
MOPC
MOP
DPC
DP
Location-dependent offsets, coarse
Location-dependent offsets, fine
User cutting edge monitoring
Cutting edge monitoring
User cutting edge data
Cutting edge data

Image 5-4 Overview of cutting edge data

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 the NC program.

5.2.2 Cutting edge parameters

$TC_DPx[t,D]

Cutting edge parameters for geometry, technology and tool type.

Parameters:

- x: Parameters 1...25
  The maximum value for x is displayed in the OPI variable "numCuttEdgeParams" in block Y.
- t: T number 1...32000
- D: Cutting edge number 1...12 or D number

Depending on the tool type, up to 25 cutting edge parameters can be programmed for each tool cutting edge.

References: /FB1/ W1, Tool Offset

OPI block TO

Address calculations:

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

Tool offset parameters (system variables)					
NCK identifier	Type	Designation	OPI variables	Type	Default[2)]
$TC_DP1	INT	Tool type	edgeData	REAL	9999
$TC_DP2	Double	Cutting edge position*	edgeData	REAL	0
$TC_DP3	Double	Geometry length 1	edgeData	REAL	0
$TC_DP4	Double	Geometry length 2	edgeData	REAL	0
$TC_DP5	Double	Geometry length 3	edgeData	REAL	0
$TC_DP6	Double	Radius geometry	edgeData	REAL	0
$TC_DP7	Double	Geometry - corner radius (tool type 700; slotting saw)	edgeData	REAL	0
$TC_DP8[1)]	Double	Geometry length 4 (tool type 700; slotting saw)	edgeData	REAL	0
$TC_DP9[1)]	Double	Geometry length 5	edgeData	REAL	0
$TC_DP10[1)]	Double	Geometry - angle 1	edgeData	REAL	0
$TC_DP11[1)]	Double	Geometry – angle 2 for tapered milling tools	edgeData	REAL	0
$TC_DP12	Double	Wear - length 1	edgeData	REAL	0
$TC_DP13	Double	Wear - length 2	edgeData	REAL	0
$TC_DP14	Double	Wear - length 3	edgeData	REAL	0
$TC_DP15	Double	Wear - radius	edgeData	REAL	0
$TC_DP16	Double	Wear - slot width b/rounding radius	edgeData	REAL	0
$TC_DP17	Double	Wear - projection k	edgeData	REAL	0
$TC_DP18	Double	Wear - length 5	edgeData	REAL	0
$TC_DP19	Double	Wear - angle 1	edgeData	REAL	0
$TC_DP20	Double	Wear - angle 2 for conical milling tools	edgeData	REAL	0
$TC_DP21	Double	Adapter length 1	edgeData	REAL	0
$TC_DP22	Double	Adapter length 2	edgeData	REAL	0
$TC_DP23	Double	Adapter length 3	edgeData	REAL	0
$TC_DP24	Double	Clearance angle	edgeData	REAL	0
$TC_DP25	Double	**ManualTurn**: Cutting speed **ShopMill**: Status value of tool of type 1xx and 2xx.	edgeData	REAL	0

Tool offset parameters (system variables)					
NCK identifier	Type	Designation	OPI variables	Type	Default[2)]
$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.	-	-	0
$TC_DPH [t,d]	INT	H parameter (Y / extraCuttEdgeParams), Bit0=1	-	-	0
$TC_DPV	Double	Tool cutting edge orientation	-	-	-
$TC_DPV3		L1 component of the tool cutting edge orientation	-	-	-
$TC_DPV4		L2 component of the tool cutting edge orientation	-	-	-
$TC_DPV5		L3 component of the tool cutting edge orientation	-	-	-
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 NCK system variable:

- $TC_DP11 is a tool OEM parameter in so far as NCK does not evaluate the contents of the value.
- $TC_DP11 is an NCK system variable in so far as when accessing via $P_ADT[n], with n=11, NCK 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.

5.2.3 User cutting edge data

$TC_DPCx[t,D]

Up to 10 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...10

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+parameterNo.

Calculation of column: T number

User-related cutting edge data					
NCK identifier	Type	Designation	OPI variables	Type	Default setting
$TC_DPC1	Double	CC_Cutting edge parameter 1	edgeData	REAL	0
...	Double	...	edgeData	REAL	0
$TC_DPC10	Double	CC_Cutting edge parameter 10	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.

5.2.4 Cutting edge-related tool monitoring

$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+parameterNo.

Calculation of column: T number

Tool management monitoring data					
NCK identifier	**Type**	**Designation**	**OPI variables**	**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

Tool management monitoring data					
NCK identifier	Type	Designation	OPI variables	Type	Default setting
$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

5.2.5 User cutting-edge monitoring

$TC_MOPCx[t,D]

Tool monitoring user data (edge-specific)

Up to 10 additional tool monitoring parameters can be programmed for each cutting edge. Setting with MD18098 MM_NUM_CC_MON_PARAM and enable with MD18080 MM_TOOL_MANAGEMENT_MASK (set bit 2)

X = Parameter 1....10

T = T_Number 1....32000

D = Cutting edge number 1....12

D = D number

OPI block TUS

Calculation of line: (d-1)*numCuttEdgeParams_tus+parameterNo.

Calculation of column: T number

Tool monitoring user data (edge-specific)					
NCK identifier	Type	Designation	OPI variables	Type	Default setting
$TC_MOPC1	Int	CC monitoring parameters	userdata	REAL	0
...	Int	...	userdata	REAL	0
$TC_MOPC10	Int	CC monitoring parameters	userdata	REAL	0

5.2.6 Location-dependent offsets, fine (additive offsets)

$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 stands for D Location, whereby Location refers to where the cutting edge is used.

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)+ParameterNo.

Calculation of column: T number

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

5.2.7 Location-dependent offsets, coarse (setting-up offsets)

$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).

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)+ParameterNo.

Calculation of column:T number

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

5.3 Tool data

5.3.1 Overview

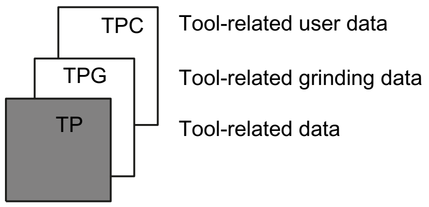


Image 5-5 Overview of tool data

5.3.2 Tool-related data

$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					
NCK identifier	**Type**	**Designation**	**OPI variables**	**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

Tool-related data, tool management					
NCK identifier	Type	Designation	OPI variables	Type	Default setting
$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 master tool Bit 13 reserved Bit 14 1:1 exchange Bit 15, manual tool	toolState	WORD	0=not enabled
$TC_TP9	INT	Tool monitoring method Value 0 no tool monitoring Bit 0 tool life Bit 1 workpiece count Bit 2 wear monitoring active Bit 3 wear monitoring, location-dependent offset fine active	toolMon	WORD	0
$TC_TP10	INT	Replacement change strategy	toolSearch	WORD	0
$TC_TP11	INT	Tool information: Allows tool groups to be divided into sub-groups. 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.			

Tool-related data, tool management					
NCK identifier	Type	Designation	OPI variables	Type	Default setting
$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	maxSpindVeloOf-Tool		
$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 moni-toring is performed	maxSpindAccOfTool		

$TC_TP1 and $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 to TP6

Size in terms of half locations:

Size 1 means that the tool exactly completely occupies its own magazine location. The maximum programmable size is 7.

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 still 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 language 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 still 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 language commands DELMRES and DELMOWNER).

$TC_TP8

The tool status is described with system variable $TC_TP8. The system variable is bit-coded. In other words, a particular 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 toolholder.

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

A tool cannot be loaded if its status is **bit 2=1** ("disabled"). The status is set automatically by the tool monitoring function, when the monitoring value of at least one cutting edge reaches the limit value. The status bit 2=4 of the tool on the toolholder can or will be ignored when generating the INIT blocks (see MD 20110 and 20112). The PLC also has the option to make NCK ignore the status during tool selection.

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

The status **bit 7=1** ("was in use") is set by the NCK if the tool is removed from a magazine location of the type spindle or toolholder.

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: All tools (new and old) involved in the tool change are given the status bit 5=1 by the tool selection. The status is reset again by the end acknowledgement for each tool command.

The following applies in particular:

The end acknowledgement of the PLC command 2 (programming the T address with $MC_TOOL_CHANGE_MODE=1), resets the status "W" of the old tool.

With the end acknowledgement 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 be used for a new programmed tool command if the tool status bit 5=! (being changed).

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.

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.

Bit 9 ignores 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" (DB21.DBx29.7).

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 NCK.
- When assigning a tool to a real magazine the tool status is reset by the NCK (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 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 NCK.
- Unloading a tool via an unloading location causes the NCK to reset the tool status.

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

Status bit 12 (master tool)

Bit 12 is set for tools that are to be permanently assigned to a magazine. This status is only set to provide information and has no effect on the NCK (e.g. does not disable a location). The user defines via the unload program whether the tool can be unloaded.

Bit	Value	Meaning
12	0	"Not a master tool"
	1	**"Master tool"**

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

NCK 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 NCK and result in incorrect machining.

$TC_TP9

If a monitoring type is activated for the tool with $TC_DP9, then 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 the 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 so-called "Technology Groups", to differentiate between tools with the same identifier and specifically release them for machining.
   Example 1:
   $P_USEKT=4
   i.e. the only tools to be taken into account are those with bit 2 in $TC_TP11 or
   Example 2:
   $P_USEKT=9
   i.e. 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 spare tool (and there only) the only tools to be taken into account are those with one of these bits set in system variable $TC_TP11.

5.3.3 Tool-related grinding data

$TC_TPGx[t]

Technology-specific grinding data

The default setting for grinding data is 0. Tools with **tool type 400 to 499** are always **grinding tools**, i.e. have these additional 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 calcu-lation	paramNrCCV	REAL

5.3.4 Tool-related user data

$TC_TPCx[t]

User-related tool data

An additional 10 tool-specific parameters can be set up per tool. Setting with MD18094 MM_CC_TDA_PARAM and enable with MD18080 MM_TOOL_MANAGEMENT_MASK (set bit 2)

x: = Parameter 1...10

t: = T number 1...32000

OPI block TU/TUD

Calculation of line: T number

Calculation of column: Parameter number

Tool-related OEM user data				
NCK identifier	**Type**	**Designation**	**OPI VAR**	**Type**
$TC_TPC1	Double		data	REAL
...	Double		data	REAL
$TC_TPC10	Double		data	REAL

Note

The data is displayed in the tool management. In addition, e.g. tool status information can also be stored here.

5.4 Magazine data

5.4.1 Overview of magazine data

Magazine data

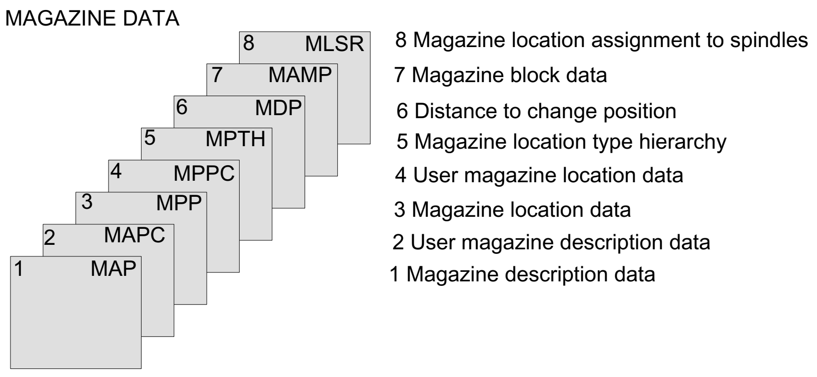

MAGAZINE DATA
8 MLSR
7 MAMP
6 MDP
5 MPTH
4 MPPC
3 MPP
2 MAPC
1 MAP
8 Magazine location assignment to spindles
7 Magazine block data
6 Distance to change position
5 Magazine location type hierarchy
4 User magazine location data
3 Magazine location data
2 User magazine description data
1 Magazine description data

Image 5-6 Overview of magazine data

5.4.2 Magazine description data

$TC_MAPx[n]

Magazine description data

This data identifies the real magazine

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					
NCK identifier	**Type**	**Designation**	**OPI variables**	**Type**	**Default setting**
		Magazine number	magNo	WORD	0
$TC_MAP2	String	Identifier of the magazine	magIdent	String	" "

Magazine description data, tool management					
NCK identifier	**Type**	**Designation**	**OPI variables**	**Type**	**Default setting**
$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_MAP6	INT	Number of lines (box magazines only)	magDim	WORD	1
$TC_MAP7	INT	Number of columns	-	-	
-	-	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_MAP8

The current magazine position $TC_MAP8 is refreshed by the NCK 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.

For buffer magazines, the default setting "0" always applies, i.e. the search is made forwards from the 1st magazine location.

5.4.3 Magazine user data

$TC_MAPCx[n]

Magazine user data

Up to 10 user data can be additionally created for each magazine. Setting in the MD18090 MM_NUM_CC_MAGAZINE_PARAM and enable with MD18080 MM_TOOL_MANAGEMENT_MASK (set bit 2) x: = Parameter 1...10

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_MAPC10			userData	DINT

5.4.4 Magazine location data

$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					
NCK identifier	**Typ e**	**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	BO OL	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					
NCK identifier	**Typ e**	**Designation**	**OPI variables**	**Type**	**Default setting**
$TC_MPP_SP	INT	For tool speed and acceleration monitor-ing Only relevant when $MC_TOOLHOLD-ER_MANAGEMENT > 0. Spindle no. which is linked to the tool holder.	placeData		
$TC_MPP66	INT	T No. of the tool, for which the identifica-tion "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 re-served (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, NCK 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 NCK 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 NCK.

$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 number 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.

5.4.5 Magazine location user data

$TC_MPPCx[n,m]

Magazine location user data

Up to 10 user data can be additionally created for each magazine. 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...10

n: = Magazine number 1...30

m:= Magazine location number 1...32000

OPI block TUP

Calculation of line: (m-1)*numMagLocParams_u+ParameterNo.

Calculation of column: Magazine number

OEM user magazine location data					
NCK identifier	**Type**	**Designation**	**OPI variables**	**Type**	**Default setting**
$TC_MPPC1	INT		userplaceData	DINT	0
...	INT		userplaceData	DINT	0
$TC_MPPC10	INT		userplaceData	DINT	0

5.4.6 Magazine location type hierarchy

$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 hierarchy, from 0...7

M: = Index within hierarchy n, location type 0...7

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					
NCK identifier	**Type**	**Designation**	**OPI varia-bles**	**Type**	**Default setting**
$TC_MPTH[n,m]	INT	Location type hierarchy n: Hierarchy 0-7 m: Location type 0-7	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, but a location type can 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.

5.4.7 Distance to change position

$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,..(2nd 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				
NCK 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 (loading 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.

Comment:

The value of the distance and the current magazine position is 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 NCK 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:

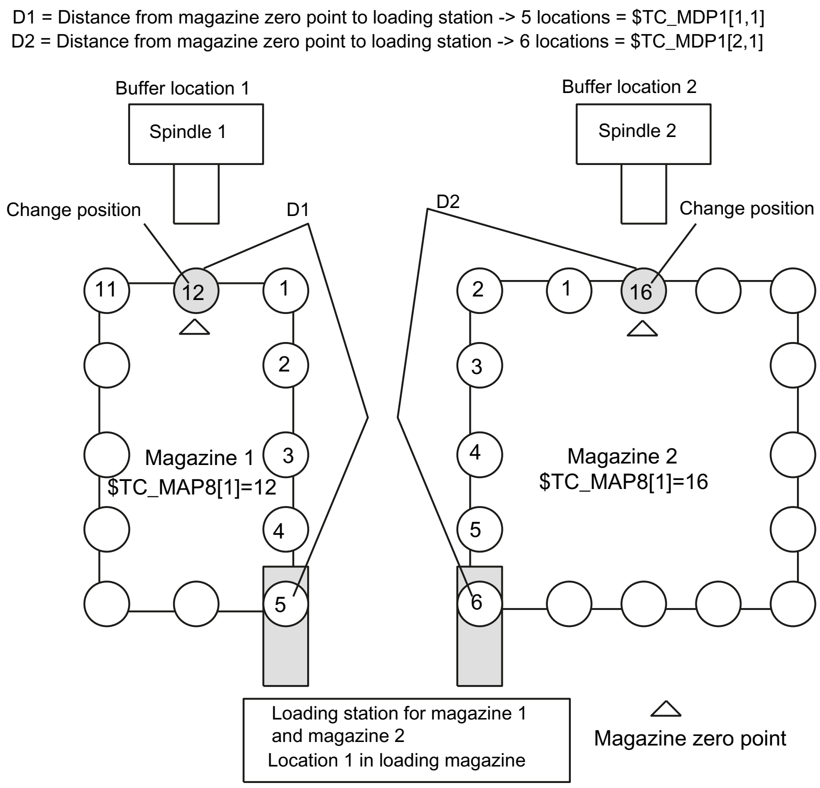

D1 = Distance from magazine zero point to loading station -> 5 locations = $TC_MDP1[1,1]
D2 = Distance from magazine zero point to loading station -> 6 locations = $TC_MDP1[2,1]
Buffer location 1
Spindle 1
Buffer location 2
Spindle 2
Change position
D1
D2
Change position
11
12
1
2
3
4
5
Magazine 1
$TC_MAP8[1]=12
2
1
16
3
4
5
6
Magazine 2
$TC_MAP8[1]=16
Loading station for magazine 1 and magazine 2
Location 1 in loading magazine
Magazine zero point

Image 5-7 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

5.4.8 Magazine blocks

$TC_MAMPx

Magazine block data

x: = Parameter 1, 2, 3

OPI block TMC

Calculation of line: n.a.

Calculation of column: n.a.

Magazine block data, magazine control block					
NCK 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	
$TC_MAMP2	INT	Type of tool search (bits 0...7) and type of empty location search (bits 8...15) 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, continue searching in the other associ-ated magazines Bit 1: Search for next replace-ment tool at the shortest possible distance from the current maga-zine position Bit 2: Select the "active" tool oth-erwise the replacement tool with the lowest number contained in $TC_TP10 Bit 3: Search for tool in the group with the **lowest actual value** for the monitored quantity Bit 4: Search for tool in the group with the **highest actual value** for the monitored quantity	magSearch		0

Magazine block data, magazine control block					
NCK identifier	**Type**	**Designation**	**OPI variables**	**Type**	**Default setting**
$TC_MAMP2	INT	Bit 5: Only consider those tools whose actual value is at least the factor $AC_MONMIN of the set-point away from the limiting value Bit 6: First search in the maga-zine currently being considered (this is only effective in conjunc-tion 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 dis-tance table Bit 7= 1 + bit 0=1 or bit 2=1, if no "activeTool" is found in the mag-azine, then - if available - the ac-tive tool is selected from another magazine linked with the tool holder	magSearch		0
$TC_MAMP2		Bit 8 Search forwards starting at first location number Bit 9 Search forwards starting at cur-rent magazine position Bit 10 Search backwards starting at last location number			

Magazine block data, magazine control block					
NCK identifier	Type	Designation	OPI variables	Type	Default setting
$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 (hierarchy) First searches the magazine with a search run across the location types of the hierarchy of the tool type. If a location type is not found, then a search run is performed across the magazine. Bit 14 = 1 (hierarchies) Search the location type of the tool in the magazine, if nothing is found, search run across the magazine. If a location is still not found, then a search run is repeated across all magazines with the next location type from the type hierarchy.			
		Bit 15 = 0 (type, location type hierarchy) Each location type can be included as a maximum in one location type hierarchy. Bit 15 = 1 (type, location type hierarchy) For the listed location types, 1, ... 8, location type hierarchies can be defined so that one **location type may be contained in several hierarchies**. The hierarchies for location type 1 are defined with $TC_MPTH[0,m], those for location type 8 with $TC_MPTH[7,m].			

Magazine block data, magazine control block					
NCK identifier	Type	Designation	OPI variables	Type	Default setting
$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	mode-WearGroup	WORD	0
$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 (language 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			

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

$TC_MAMP2 =		Meaning
7 + 0	'H81'	Otherwise, the same as value 1 - however, if an "active" tool is not found in the magazine, then - if available - the "active" tool is selected from another magazine linked with the tool holder. If no tool with the "active" state is available, then 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, then a search is made in the same magazine for any tool that can be used; if no tool is available, then 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, then - if available - the "active" tool is selected from another magazine linked with the tool holder. If no tool with the "active" state is available, then 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, then a search is made in the same magazine for a tool that can be used, if available, then a search is made for the replacement tool in another magazine linked with the tool holder with the lowest number in $TC_TP10.
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

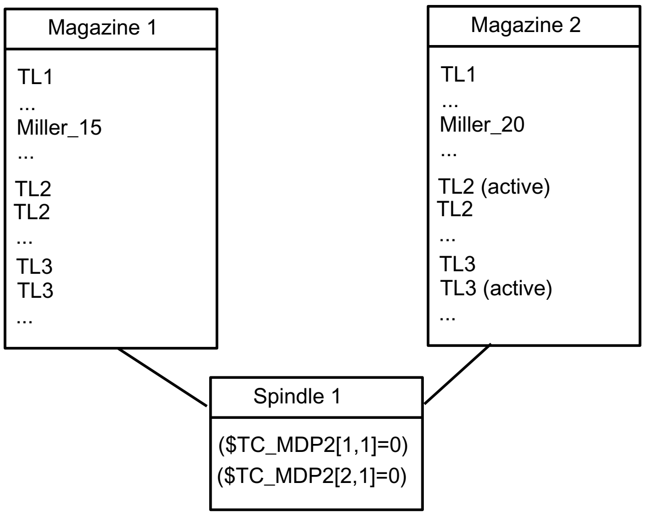
A search is made for the appropriate tool in the magazine according to the tool search strategy that has been set (defined using parameter $TC_MAMP2). The search is considered to have been successful if a tool was found according to the search strategy.

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="Miller_15"M06	There is only one "miller_15" tool, it was found in magazine_1 and was loaded
...	
T="Tool3" 06	MThe 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.

Example_2:

Bit 6 = 0
Bit 7 = 1

T="Miller_20"M06	There is only one "Miller_20" tool, it was found in Magazine_2 and was loaded
...	
T="Tool3" M 06	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.

Example_3:

Bit 6 = 1
Bit 7 = 1

T="Miller_20"M06	There is only one "Miller_20" tool, it was found in Magazine_2 and was loaded
...	
T="Tool3" M 06	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.

$TC_MAMP2

For software releases less than 2.5, the tool search is always made magazine for magazine, 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, then 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), then any tool that is capable of being used is found.

5.4.9 Assigning buffers to spindles

$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.

NCK identifier	Type	Designation	OPI variables	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.

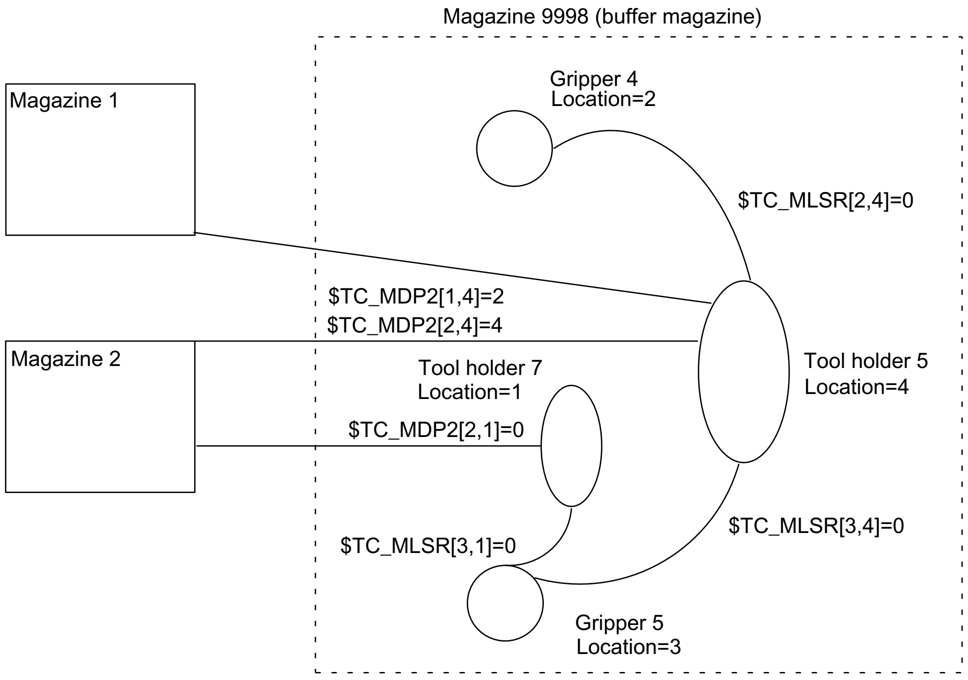


Image 5-8 Magazine distance to the buffer

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.

5.5 Adapter data

$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

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

5.6 Toolholder data

Overview

The orientation of the tool can be changed for a class of tool machines. The orientation once set is however subsequently fixed during operation and in particular cannot be changed during traversing. Therefore, a kinematic orientation transformation for this type of machine is neither necessary nor meaningful.

It is however necessary to consider the changes in the tool length components attributable to the change in the orientation. These calculations are assumed by the control.

The following must be available to calculate the change of tool length components:

- **Tool** data (geometry, wear ...)
- Tool holder data (data for the geometry of the tool holder with orientation capability).

A defined tool holder must be specified for the control for the function "orientable tool holder":

$TC_CARRx

x: = Parameter 1...33

The maximum number of tool holders can be defined in MD18088 $MN_MM_NUM_TOOL_CARRIER. The value is divided by the number of active TO units. The integer result indicates how many tool holders can be defined per TO unit. Values not set by the user are preset to 0.

OPI block TC

Calculation of line: Number of the tool holder

Calculation of column: n.a.

Tool holder data				
Name	**Type**	**Explanation/description**	**OPI VAR**	**Type**
$TC_CARR1	Double	No. of tool holder x component of offset vector L1	TcCarr1	REAL
$TC_CARR2	Double	No. of tool holder y component of offset vector L1	TcCarr2	REAL
$TC_CARR3	Double	No. of tool holder z component of offset vector L1	TcCarr3	REAL
$TC_CARR4	Double	No. of tool holder x component of offset vector L2	TcCarr4	REAL
$TC_CARR5	Double	No. of tool holder y component of offset vector L2	TcCarr5	REAL
$TC_CARR6	Double	No. of tool holder z component of offset vector L2	TcCarr6	REAL
$TC_CARR7	Double	No. of tool holder x component of rotary axis V1	TcCarr7	REAL

Tool holder data				
Name	**Type**	**Explanation/description**	**OPI VAR**	**Type**
$TC_CARR8	Double	No. of tool holder y component of rotary axis V1	TcCarr8	REAL
$TC_CARR9	Double	No. of tool holder z component of rotary axis V1	TcCarr9	REAL
$TC_CARR10	Double	No. of tool holder x component of rotary axis V2	TcCarr10	REAL
$TC_CARR11	Double	No. of tool holder y component of rotary axis V2	TcCarr11	REAL
$TC_CARR12	Double	No. of tool holder z component of rotary axis V2	TcCarr12	REAL
$TC_CARR13	Double	No. of tool holder Angle of rotation alpha1	TcCarr13	REAL
$TC_CARR14	Double	No. of tool holder Angle of rotation alpha2	TcCarr14	REAL
$TC_CARR15	Double	No. of tool holder x component of offset vector L3	TcCarr15	REAL
$TC_CARR16	Double	No. of tool holder y component of offset vector L3	TcCarr16	REAL
$TC_CARR17	Double	No. of tool holder z component of offset vector L3	TcCarr17	REAL
$TC_CARR18	Double	No. of tool holder x component of offset vector L4	TcCarr18	REAL
$TC_CARR19	Double	No. of tool holder y component of offset vector L4	TcCarr19	REAL
$TC_CARR20	Double	No. of tool holder z component of offset vector L4	TcCarr20	REAL
$TC_CARR21	Axis	No. of tool holder Axis name of the 1st rotary axis	TcCarr21	String
$TC_CARR22	Axis	No. of tool holder Axis name of 2nd rotary axis	TcCarr22	String
$TC_CARR23	Char	No. of tool holder Kinematic type	TcCarr23	String
$TC_CARR24	Double	No. of tool holder Offset of 1st rotary axis in degrees	TcCarr24	REAL
$TC_CARR25	Double	No. of tool holder Offset of 2nd rotary axis in degrees	TcCarr25	REAL
$TC_CARR26	Double	No. of tool holder Offset of Hirth joint in degrees for 1st rotary axis	TcCarr26	REAL
$TC_CARR27	Double	No. of tool holder Offset of Hirth joint in degrees for 2nd rotary axis	TcCarr27	REAL

Tool holder data				
Name	Type	Explanation/description	OPI VAR	Type
$TC_CARR28	Double	No. of tool holder Increment of Hirth joint in degrees for 1st rotary axis	TcCarr28	REAL
$TC_CARR29	Double	No. of tool holder Increment of Hirth joint in degrees for 2nd rotary axis	TcCarr29	REAL
$TC_CARR30	Double	No. of tool holder Minimum position of 1st rotary axis	TcCarr30	REAL
$TC_CARR31	Double	No. of tool holder Minimum position of 2nd rotary axis	TcCarr31	REAL
$TC_CARR32	Double	No. of tool holder Maximum position of 1st rotary axis	TcCarr32	REAL
$TC_CARR33	Double	No. of tool holder Maximum position of 2nd rotary axis	TcCarr33	REAL

Additional references:

/FB1/ Description of Functions, Basic Machine; Tool Offset (W1) and

/PGA/ Programming Guide Advanced

5.7 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: = Parameter 0, 1, 2

NCK 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

Interface DB72, DB73

The user parameters are output in DB72 and DB73 on the tool management interface. They are only valid when the status of the interface is active. The format is DINT.

Example

$P_VDITCP[0]=12; DB72.DBD(n+4) =12 or

$P_VDITCP[1]=33; DB72.DBD(n+8) =33 or

$P_VDITCP[2]=2000; DB72.DBD(n+12) =2000

T="Tool"

The variables must be set in the part program before the T call or M06 if these shall be transferred for a tool to the PLC as well.

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.

5.8 NC language commands

5.8.1 CHKDNO - Uniqueness check for D number

Overview

D number uniqueness is understood here (not for replacement tools) as being that the D numbers of all tools defined in the TO unit may occur exactly only once ⇒ the D numbers in the TO unit are unique and absolute. This is known in the tool-management function as the possibility of assigning "unique" D numbers only. The distinction is made on the basis of replacement tools that are generally present.

Status = CHKDNO (T1, T2, D)

Parameters used:

TRUE	Unique D numbers have been assigned for the checked area
FALSE	A D-number collision is the result or the parameterization is invalid

The parameters are optional.

CHKDNO (T1,T2)	All D numbers of the specified tools are checked.

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" (the programming T="location number" is referred back to T="identifier" internally). The program otherwise only contains the actual programming of the offset (the D number). Therefore, the D number for 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, then CHKDNO always returns the TRUE state, irrespective of the parameter settings.

5.8.2 CHKDM - Uniqueness check within a magazine

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

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

Result of check:

Value = TRUE: Checked D numbers are unique.

Value = FALSE: Check not OK.

Meaning of the parameters:

MagNo	Magazine number of magazine to be checked 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 the 3rd parameter are checked.
D No.	The D number against which the check will be made. 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.	Indicates which spindle numbers or tool holder numbers the magazines shall be checked for. Omission of the parameter means that the magazines are obtained for the check from the distance table of the spindle location for the master spindle or the master tool holder.

5.8.3 GETACTTD - Determining the T No. for a unique D No.

For an active tool management (e.g. measuring-cycle programs), this command serves to conclude the associated T number **of the tool active in the tool group** starting from a D number.

status = GETACTTD (T No., D No.)

D No.	D number for which the T number shall be searched. The D numbers are not checked for uniqueness. 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.	
T No.	T number found	
status	Result of search	
	0	T number found, T No. is assigned the value
	-1	No T number exists for the specified D number, T No. assigned the value 0.
	-2	D number is not unique; T no. 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, T no. 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; T no. contains the value of the first tool found with the desired D number.
	-5	Function could not be executed for other reasons.

5.8.4 GETDNO - Rename D numbers

The language command

d = GETDNO(t, ce)

allows the offset number d to be read for the cutting edge ce of the tool with the T number t. 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.

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.

5.8.5 SETDNO - Rename D numbers

The language command

state = SETDNO(t, ce, d) allows the offset number d of cutting edge ce of tool t to be set or changed. If t or ce are parameters which have no data record, state = FALSE is returned. Any parameters violating the syntax rules will generate an alarm.

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

5.8.6 DZERO - Invalidate D numbers

Marks all D numbers of the TO unit as invalid. This command is used for support during conversion or re-equipping.

Offset data records tagged with this command are no longer verified by the CHKDNO language command. The D numbers have to be set with SETDNO again in order to make these accessible again.

5.8.7 DELDL - Delete additive offsets

This command 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.

status = DELDL(t, d)

Explanation of the parameters:

DELDL(t, d)	All additive offsets of the cutting edge with D number d of tool t are deleted.	
DELDL(t)	All additive offsets of all cutting edges of tool t are deleted.	
DELDL	All additive offsets for all cutting edges of all tools of the TO units are deleted (of the TO unit of that channel where the command is programmed)	
status	Result of search	
	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 several cutting edges)

5.8.8 NEWT - Create a new tool

Overview

A new tool can bet set up in a number of ways by NC commands in NCK. Either by programming T No.=**NEWT**("Tool", duplo no.) or by programming a system variable **$TC_**...

Note that NEWT automatically generates a cutting edge with CE no. = 1, D no. = 1. If you want the tool to have a different CE no., you need to change this number after it has been generated.

The NEWT function allows a new tool to be created without specifying a T No. The function returns the automatically generated T No. with which the tool can subsequently be addressed. The 1st cutting edge is automatically created when a new tool is created. All offsets are set to 0 by default.

Return parameter = NEWT ("Tool", duplo no.)

If it is not possible to create a new tool for any reason, the NEWT(...) function generates an alarm.

Specification of a duplo number is optional. It is generated in the NCK if it is not specified. (duplo no.= old duplo no. +1)

Examples

Example 1:

Create a new tool with NEWT and the CE/D numbers = 2, 47

```
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
                              ;however, the cutting edge is to be named 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 for 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)

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

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

5.8.9 NEWMT - create new multitool

The language command is available with TMMG.

The language command creates a new multitool with the specified "name" and the MT "number of locations".

mtNo = NEWMT(name, location number)

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

The MT number is created automatically and returned as result value. If the "Several tools at magazine location" function is not activated, Alarm 6436 "Command cannot be programmed. Function is not activated." is output. If the name parameter is already in use, alarm 14020 "Illegal parameter value" is output.
If the number of locations parameter is outside the permissible range, alarm 14020 "Illegal parameter value" is output. If the command is interrupted with alarm, then for mtNo the value equal 0 is returned.

Example:

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:

```
def int mtNr
mtNr = NEWMT("C", 2)
```

The command is executed successfully, mtNr has the value 3. The multitool has been created for two locations. However, the locations were still not created. 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.

$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

$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.

5.8.10 DELT - Delete tool

A tool can be deleted with the DELT(...) function by specifying the tool identifier and duplo number. Only tools that have been unloaded can be deleted.

DELT("TL",DUPLO_NO)

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

Example:

DELT("DRILL", DUPLO_NO)

The function is for deleting tools in the part program.

5.8.11 DELMT - deleting a multitool

The language command is available with TMMG.

The DELMT command can delete a multitool.

DELMT (name)

Deletion is only possible if the multitool is not contained in a magazine (is unloaded) and none of the tools contained in the multitool is an active tool regarding 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, i.e. the tools are not deleted.

If a name is specified, for which a multitool has not been defined, then 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 to be loaded into a magazine - or into another multitool.

5.8.12 $TC_MTPN - Deletion of one or all multitools

The delete command is defined analogously to $TC_TP1[TNr] = 0:

$TC_MTPN[MTNr] = 0 ; Delete multitool with the number MTNr > 0

$TC_MTPN[0] = 0 ; Delete all multitools

For MTNr > 0, the multitool with the number MTNr is deleted. For MTNr = 0, all multitools are deleted. The same rules apply as defined for the DELMT command.

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.

5.8.13 GETT - Read T No.

The GETT function returns the associated T number on the basis of the tool identifier and its duplo number.

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

Result value = GETT("TOOL", DUPLO_NO);

Result value

>	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

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

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).

Example:

Determine the T number for drill with duplo number

```
R10=GETT("DRILL", DUPLO_NO)               ;The T number is in R10
$TC_TPx,[GETT("DRILL",DUPLO NO)]=value    ;Write tool-related data
```

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

5.8.14 SETPIECE - decrementing the workpiece counter

Overview

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 as a rule for programming 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.

Programming

SETPIECE(x,y)

x := 0 ... 32000	Value used when decrementing
y := 0...8	Spindle index, value 0 means index of main spindle (must not be programmed)

Example:

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.

```
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:

```
N500 T1
N600 M06
N700 D1                      ;With the offset selection, the tool that was loaded is
                             stored in the SETPIECE memory
N900 T2                      ;Preparing the next tool
                             ;Machining command
:
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
```

Note

SETPIECE works through the table "blindly". This means that every cutting edge entered in the table is recorded. Whereby, 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, then this tool is selected internally and an entry made in the Setpiece table.

5.8.15 GETSELT - Read the selected T No.

The function is available with TMBF, TMMO and TMMG.

This language command is used to access the offset data of the selected tool between the T programming and the cutting edge selection. It must establish synchronization with the main run earlier. The GETSELT language command returns the selected tool with regard to the individual tool holders or spindles from the NC program viewpoint.

GETSELT (Tno, th, ssl);

Tno = 0 the selected T number in int format		
	= 0	There is no 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 = spindle / tool holder number		
	= 0	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	= ""	Function as previously, i.e. the current state according to the NC program
	= "S"	Block search: Returns the tool selected before the block search or test mode

th and ssl must be specified optionally. If th is not specified, this acts as th=0.

If "ssl" is set (="S" for "searchrun"), during the block search or test mode, the GETSELT language command returns the T no. 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 language command returns the value independent of ssl after the first T programming for the specified tool holder.

After the program end / abort, the "getselt" T numbers of all tool holders are set to the value = 0. For the master 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.

If the GETSELT language command is programmed when the "TMFD" (flat D number) functionality is active, then alarm 6437 "%?C{channel %1: %} block %2 command '%3' cannot be programmed. Function '%4' is activated".

OPI variable

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 except in the "Flat D number" version.

Example 1:

There is no tool holder location in the magazine management for the programmed value of tool holder no. = 3

Example 1:

`N110 GETSELT ( tNo, 3 ) ; tNo = 0` is the result.

Example 2:

Tool preparation has not been programmed for the programmed tool holder with the number = 4, and no tool has been loaded:

`N110 GETSELT ( tNo, 4 ) ; tNo = 0` is the result.

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, then the T number found by GETSELT has the value = –1.

Example 3:

There is a tool with the name "disabled" and the T number = 5.

```
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:

The basic functionality is active (TMBF):

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.

```
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.

5.8.16 GETEXET - Reading of the loaded T number

The function is available with TMBF, TMMO and TMMG.

The command 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.

GETEXET(Tno, th, ssl)

Tno = the active T number in int format

= 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 = spindle / tool holder number

= 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 = "": Function as previously, i.e. the current state according to the NC program

= "S": Block search function: Returns the tool loaded before the block search or test mode

The specification of th and ssl is optional. If th is not specified, GETEXET refers to the current master spindle.

If "ssl" is set (="S" for "searchrun"), during the block search or test mode, the GETEXET language 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 language 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 master 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), then 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.

If the GETEXET language command is programmed when the "TMFD" (flat D number) functionality is active, then alarm 6437 "%?C{channel %1: %} block %2 command '%3' cannot be programmed. Function '%4' is activated".

OPI variable

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 except in the "Flat D number" version.

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 master tool holder.

At the start, there is neither a tool on the tool holder nor a tool active or programmed:

```
def int tNo1,
tNo2
T="Tool5"    ; tool selection for master too holder
getselt ( tNo2,  ;tNo2 = 5 preselected tool for master tool holder
2 )
getselt ( tNo1,  ;tNo1 = 0
1 )
```

```
getselt ( tNo2,  ;tNo2 = 0 no active tool for master tool holder
2 )
getexet ( tNo1,  ;tNo1 = 0
1 )

M6   ; tool change for master tool holder
getselt ( tNo2,  ;tNo2 = 5
2 )
getselt ( tNo2,  ;tNo2 = 5
2 )

T="Tool9"; tool selection for master tool holder
getselt ( tNo2,  ;tNo2 = 9
2 )
getselt ( tNo2,  ;tNo2 = 5
2 )

T1="Tool7"   ; tool selection for secondary tool holder
getselt ( tNo2,  ;tNo2 = 9 preselected tool for master tool holder
2 )
getselt ( tNo1,  ;tNo1 = 7 preselected tool for secondary tool holder
1 )

getselt ( tNo2,  ;tNo2 = 5 active tool for master tool holder
2 )
getexet ( tNo1,  ;tNo1 = 0 active tool for secondary tool holder
1 )

M1=6   ; tool change for secondary tool holder
getexet ( tNo1,  ;tNo1 = 7 active tool for secondary tool holder
1 )
```

Example 2:

The basic functionality is active (TMBF):

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.

```
def int tNo, tNo1
N70   T7 M6
N75   T8
N80   GETEXET(tNo)   ;tNo =7
N90   GETSELT(tNo1, ;tNo1=5
      ,"S")
N100  X1   G0        ;Target block
N110  GETEXET(tNo)   ;tNo =7
```

```
N120  GETEXET(tNo1, ;tNo1=5, because no new tool has been programmed
      ,"S")         ;after the target block
N130  M6
N140  GETEXET(tNo)  ;tNo=8
N150  GETEXET(tNo1, ;tNo1=8, because a new tool has been loaded
      ,"S")
```

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.

Example 4:

The basic functionality (TMBF) is active.

```
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.

def int tNo, tNo1
N70   T7 M6
N75   T8
N80   GETEXET(tNo)      ;tNo=7
N90   GETSELT(tNo1 ,,   ;tNo1=5
      "S")
N100  X1 G0             ;Target block
N110  GETEXET(tNo)      ;tNo=7
N120  GETEXET(tNo ,,    ;tNo1=5, because no new tool has been
      "S")              programmed
                        ;after the target block
N130  M6
N140  GETEXET(tNo)      ;tNo=8
N150  GETEXET(tNo ,,    ;tNo1=8, because a new tool has been loaded
      "S")
```

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.

```
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.

```
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

```
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,         ;Safe determination of the active Tno. tNo = 2
$PMTHSDC)
```

b) $MC_CUTTING_EDGE_DEFAULT = -2

```
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.

See also Section "Block search (SSL) in conjunction with active tool management (Page 109)".

5.8.17 $P_MTHSDC - master tool holder with regard to the D offset selection

This function is available with TMMG.

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

```
N10 SETMTH
    ( 1)
N20 T="Tool5" ;T no. = 5 is the only tool in the tool group
              ;tool change to the current master tool holder (tool
              holder 1)
N30 D5        ;Offset of the tool last changed on a master tool holder
              ;(T no. 5 in this case)
N10 SETMTH    ;Change of the master tool holder
0   ( 2)
N11 T="Tool7" ;T no. = 7 is the only tool in the tool group
0             ;tool change to the current master tool holder (tool
              holder 2)
N12 SETMTH    ;Change of the master tool holder
0   ( 1)
...           ;Further blocks without T, M6, D
N13 D5        ;Offset with D5 of the tool that was loaded last on to a
0             ;master tool holder (T=7 on tool holder 2
              ;in this case). Only now is a new tool specified
```

In order 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 master tool holder and must be performed as the last tool change. This may have to be ensured via ASUPs. This information cannot be determined with the previously available language commands and variables. The variable contains the information on which tool holder the last tool change was performed as this was master tool holder.

The system variable $P_MTHSDC (**M**aster **t**ool **h**older regarding **s**elected **D**correction) returns the number of the tool holder / spindle on which the last tool change was performed on a master 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).

In the above case, the variable $P_MTHSDC has the value 2, because on this tool holder the last tool change was performed on a master tool holder. With the aid of GETEXET(TNo,2) or if ($P_MTHSDC > 0) GETEXET(TNo,$P_MTHSDC), the tool can be determined which 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.

5.8.18 $P_TH_OF_D - master tool holder with regard to the current D offset

The function is available with TMBF, 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.

Settings:

$MC_CUTTING_EDGE_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

```
N00              ;No tool active → $P_TH_OF_D = 0
N10 SETMTH(1)
N20 T="Tool5"    ;T no. = 5 is the only tool in the tool group
                 ;tool change to the current master tool holder (tool
                 holder 1)
N30 D5           ;Offset is from the tool that is located on tool
                 holder 1
                 ;→ $P_TH_OF_D = 1
N100SETMTH(2)    ;Change of the master tool holder
N110T="Tool7"    ;T no. = 7 is the only tool in the tool group
                 ;tool change to the current master tool holder (tool
                 holder 2)
N120SETMTH(1)    ;Change of the master tool holder
...              ;Further blocks without T, M6, D
N130D5           ;Offset with D5 of the tool that was loaded last on
                 to a
                 ;master tool holder (T=7 on tool holder 2
                 ;in this case) → $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 been 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.

5.8.19 GETACTT - Read the active internal T No.

This function provides the possibility, from a tool group with the identifier "name", to obtain the T No. of the tool with the status "active" (a tool becomes "active" immediately before it is loaded into the tool holder) and "was in use" via parameter "TNo".

status=GETACTT(Tno,name)

The return parameter "status" indicates whether the call was successful or not:

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 are 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:

```
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
                                  ;The "Drill" tool is now assigned the status "was in
                                  use" as it was removed and the "active" status re-
                                  mains unchanged
```

Note

GETACCT cannot detect a tool which is positioned in the spindle for its first use.

The tool sequence in a group is not defined. This means GETACCT will read any random tool in the group where the status bits "active" and "was in use" are set.

5.8.20 SETMS - Spindle can be declared a master spindle

Available with TMBF, TMFD, TMMO, TMMG.

Command SETMS(n) declares the spindle specified in n to be the master spindle. A spindle can also be defined as the master via a machine data.

The programmed values from SETMS can remain active beyond program end/reset/Start.

When SETMS is programmed without a spindle name, the spindle programmed in the machine data used instead.

5.8.21 SETMTH - Set master toolholder number

Available with TMMG.

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 language command if MD is > 0.

Programming example:

T="Miller" M06	No address extension programmed → this refers to the master 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 expansion for the secondary tool holder was programmed. The tool change is performed at buffer location 2. The path is not corrected.
...	
SETMTH (2)	declares tool holder 2 to the **master tool holder**
T="Miller_2""M06	No address extension programmed → this refers to the master tool holder; i.e. tool holder 2. Tool change is performed at buffer location 2. The path is corrected with the tool offsets.
...	
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 specified in TOOL_MANAGEMENT_TOOL-HOLDER as the master tool holder
T="Miller_3" M06	No address extension programmed → this refers to the master tool holder; i.e. tool holder 1 (value of MD TOOL_MANAGEMENT_TOOL-HOLDER). Tool change is performed at buffer location 1. The path is corrected with the tool offsets.

Note

SETMTH does not change the active tool. The new master tool holder definition cannot be taken into account until the tool change is programmed.

The programmed values from SETMS can remain active beyond program end/RESET/START.

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.

```
T="Drill" M06 D2              ;Tool change change on master tool holder=2
SETMS(3)                      ;New master spindle=3
SETMTH(1)                     ;New master tool holder=1
T="Miller" M06 D1             ;Tool change on master tool holder=1
M17
```

After end of program or RESET,

Spindle no. = 3 is the master spindle,

tool holder no. = 1 is the master tool holder and a

tool = "Miller" with offset D1 determines the path correction.

After Power On, the settings for the machine data take effect:

Spindle no. = 1 is the master spindle,

tool holder no. = 2 is the master tool holder.

The tool offset is derived from the lowest D number of the tool that is located in the master tool holder; i.e.

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.

```
T="Drill" M06 D2            ;Tool change on master tool holder = 1
SETMS(3)                    ;New master spindle = master tool holder = 3
T="Miller" M06 D1           ;Tool change on master tool holder = 3
M17
```

After end of program or RESET,

Spindle no. = 1, the master spindle and a

tool = "Miller" with offset D1 (on spindle with no. = 3) determines the path correction.

After Power On, the settings for the machine data take effect:

Spindle no. = 1 is the master spindle / the master tool holder.

The tool offset is derived from the lowest D number of the tool that is located in the master tool holder; i.e.

T="Drill" with D1

(assuming that the tool has two D offsets D1, D2).

5.8.22 POSM - Positioning the magazine

Overview

This NC language command enables you to initiate a magazine positioning operation to a particular location in an internal magazine (e.g. spindle, tool holder, loading magazine), irrespective of how the location is assigned or the status of the tool it contains. The language command includes parts of the OPI PI service _N_TMPOSM.

The complete command is: **POSM (p, m, ip, im)**

Description of function

p: Location number which is to be positioned to.

m: Magazine number of the magazine to be moved.
The parameter is optional.
If it is not set, the location number refers to the magazine contained in the distance table as the first magazine for the specified internal location.

ip: Location number for the specified internal magazine (spindle location, loading magazine, etc.)

The parameter is optional.

If it is not specified, the positioning operation refers to the main spindle location or the main tool holder location.

im: 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.

The parameter is optional.

If it is not specified, then the command refers to the buffer magazine.

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).

Example of parameter settings

Specified configuration:

- Magazine (magazine number = 1),
- Spindle (buffer magazine = 9998, location 1),
- Loading magazine (loading magazine = 9999, location 2).

It should be moved from magazine 1, location number 4 to the spindle.

Command:

N100 POSM(4, 1, 1, 9998)

Command for traversing to loading magazine:

N100 POSM(4, 1, 1, 9999)

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).

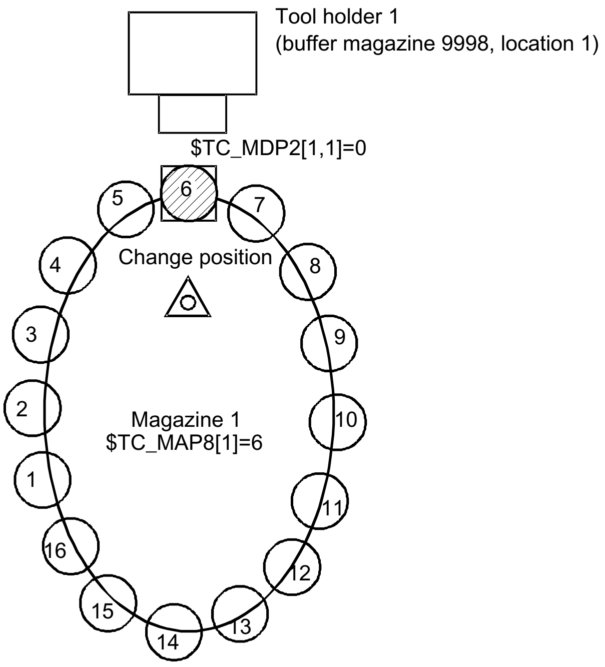


Image 5-9 Magazine positioning with a check of the positioning operation result

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 master spindle of the channel.

```
N100 POSM(12)           ;Moves location 12 to the change position, any non-programmed pa-
                        rameters 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.
```

Note

The language command POSM(...) is terminated without waiting for an acknowledgement from the PLC.

Multitool

The definition of the language command POSM remains unchanged. The magazine location to be positioned to is programmed independent of whether the location is empty, contains a tool or a multitool.

The PI service _N_TMPOSM offers various options to program 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, then the system positions to this magazine location. If the tool is in a multitool, which in turn is at a magazine location, then 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, then the system positions to this magazine location.

5.8.23 POSMT - positioning a multitool on a toolholder to the location number

Overview

This function is available for TMMG.

The POSMT positions a multitool, which 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)**

state	Success status of the command	
	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.
MTlocno	MT location number of the MT, which is located on the programmed tool holder.	
THno	Tool holder number on which the MT to be positioned is located. This parameter is optional. If it is not programmed, then the number of the master tool holder is automatically used.	

Note

Selecting an offset or deselecting a possibly active tool offset is not linked with the command.

The command initiates an implicit STOPRE.

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.

```
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 NCK, after the input, the PLC end acknowledgement is initiated:

- Activate the tool with tool number 34 and select its offset D1

Example 2

The same data applies as in example 1, with the difference that now there is no tool at location 5 of the multitool.

```
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 NCK, after the input, the PLC end acknowledgement is initiated:

- Activate the tool with tool number 0 (tool deselection) and deselect its offset D0.

The multitool remains on the tool holder - and neither a tool nor D offset are active.

Example 3

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 NCK with $TC_MTP8, then 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 NCK in the init block, the PLC must not change this.

PLC

To process the command, NCK command = 1 to the PLC is prepared, which is output via the tool management interface. The PLC must handle and acknowledge this command, in order that it can be correctly completed by the NCK.

PLC acknowledgement

In the NCK, the language command generates the command for the PLC - and 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 acknowledgement of the command is available from the PLC. This can be PLC status = 3 or 5. Only then can a new block be loaded for execution.

Status = 5, "The sequence has been completed. The MT is in position." With regard to the MT configuration, means "The sequence has been completed. The MT is in position."

Example of parameter settings

The following configuration applies:

Multitool	MT-No.	=5, MT locations 1, ...10,
		The multitool has location coding
		The tool with tool number = 4711 and tool name "4711" is at location 4
		The tool with tool number 815 and tool name "815" is at location 7
		There is no tool at location 8
Tool holder	Buffer magazine no. = 9998, location 1 is the master tool holder with no. = 3	
		The MT with MT No.=5 is on the tool holder
		Tool 4711 is active, active D No.=0 (no offset active)
	Buffer magazine no. = 9998, location 2 is tool holder with no. = 9	
		There is no tool and no MT at tool holder 9

At the start, the MT position is equal to 4 (tool with tool number=4711 from MT location 4 is the active tool), this means that MT location 4 is in the machining position.

def int state

POSMT(state, 7, 3) ; position MT of tool holder 3 to location 7 (with tool "815")

has the same contents as

POSMT(state, 7) ; position MT of tool holder 3 to location 7

The two commands are correctly programmed. The status value is state = 0.

The block remains active in the HL until the end acknowledgement from the PLC is available.

Tool "815" with tool number=815 at location 7 does not become active with the positioning

In order that tool "815" becomes active (and an offset D), then

- either a program must be started (and $MC_START_MODE_MASK appropriately set), or
- RESET executed ($MC_TOOL_MANAGEMENT_MASK, bit 14 = 1 and $MC_RESET_MODE_MASK appropriately set), or
- in the program after POSMT, the tool should be programmed to the positioned MT location with a tool change command (T="815" M06 D1).

With

POSMT(state, 8, 3) ; position MT of tool holder 3 to location 8,

the empty MT location is positioned to the machining position, if a D offset is still not active. The following init blocks are implicitly programmed with "T0 M06 D0" if the corresponding machine data has been appropriately set. The mechanical MT transport "MT from tool holder back into the magazine" is associated with this.

Programming of

POSMT(state, 3, 9) ; position MT of tool holder 9 to location 3,

Supplies the status value state=–5, as there is no MT and no tool holder at the programmed tool holder 9.

POSMT(state, 77, 3) ; position MT of tool holder 3 to location 77,

Supplies the status value state=-6, as MT location number=77 on tool holder 3 is not defined.

5.8.24 MVTOOL - Language command to move tool

Overview

This function is available for TMMG.

The function MVTOOL allows tools to be loaded and unloaded via NC programming only. It can also be used to transport a tool from one magazine location to another - regardless of where.

It is mandatory for a tool to be positioned at the source magazine location.

This language 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.

MVTOOL (state, magFrom, locFrom, magTo, locTo)

state	Success status of command	
	0	Execution was successful (PLC acknowledgement may still be pending)
	-1	Command cannot be used because TMMG is not active
	-2	Function is not carried out because of block search, program testing
	-3	Tool cannot be moved (because e.g. tool status "being changed" is set)
	-4	No tool located in source location

	-5	magFrom has invalid value
	-6	locFrom has invalid value
	-7	magTo has invalid value
	-8	locTo has invalid value
	-9	No distance relation defined (if exactly one magazine is an internal magazine)
	-10	No empty location found (if parameter locTo is not programmed)
	-11	Target location for tool occupied
	-12	The parameter locTo must be programmed, as magTo is an internal magazine

magFrom	Magazine number of magazine in which the tool to be moved is located
locFrom	Location number of location in which the tool to be moved is located
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 it is not programmed, the value from magFrom is used.
locTo	Target location number of location to which the tool is to be moved. This parameter is optional. If it 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 a value=0, then the status=-8 is set.

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 (acknowledgement 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 relationship that is required has not been defined, then the appropriate error code is returned via state.

Deselecting an active tool offset is not linked with the command.

PLC

In the NCK, the language 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 acknowledgement 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.

Example of parameter settings

The following configuration is present:

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 to this effect:

def int state

$TC_MPP6[9999, 1] =123; place tool with internal T no.=123 onto loading location

MVTOOL(state, 9999, 1, 5);search 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):

$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.

MVTOOL(state, 5, 7, 5)

or to location 3 in the same magazine

MVTOOL(state, 5, 7, 5, 3)

or to any location in magazine 11

MVTOOL(state, 5, 7, 11)

5.8.25 SETTIA - Deactivate tool from wear group

The SETTIA function cancels the "active" status for all active tools in the selected wear group. By parameterizing the language command, this can be either magazine-specific or wear group-specific.

SETTIA(STATUS, MNR, VNR,USEKT)

STATUS	Return parameters	
	0	Function was able to be executed correctly.
	-1	Function was not executed because there is no active wear group in the selected magazines.
	-2	Function was not executed because the programmed wear group number does not exist.
	-3	Function was not executed because the programmed magazine number does not exist.
	-4	Function was not executed because the function "wear group" is not enabled (MN_TOOL_MANAGEMENT_MASK).
	-5	Function was not executed.

All parameters are optional.

If SETTIA is not parameterized, the inactive setting refers to all loaded tools in the TO area which are in "active" state.

MNR	Magazine number	
	0	The inactive setting refers to all magazines regardless 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 is to be applied. Tools in this magazine which 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 relationship to a spindle or tool holder.
VNR	Wear group number	
	0	The inactive setting refers to all tools which 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 is to be applied.
	-1	Active wear group ($TC_MAP9).
USEKT	Tool subgroup	
	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.

A search strategy can be set in parameter $TC_MAMP3 for the tool to be activated by SETTIA.

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 function SETTIA.

Multitool

If SETTA is programmed for a multitool instead of for a magazine, alarm 6462 "[Channel %1:] block %2 command %' can only be programmed for magazines. '%4' means no magazine".

5.8.26 SETTA - Activate tool from wear group

Overview

With the function SETTA all not disabled tools from the desired wear group are activated. However, from a tool group, a max. of one tool referred 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, then the command can still be used. The call parameters must then be set accordingly.

If the selection criteria of the SETTA command match several tools within a tool group, then selection criterion $TC_MAMP3, bits 12-15 is applied.

Description of function:

SETTA (STATUS, MNR, VNR, USEKT)

STATUS	Return parameter which can consist of the following values:	
	0	Function was able to be executed correctly.
	1	Function was executed, but another tool is active in the group (e.g. an unloaded tool).
	-1	Function was not executed because there is no active wear group in the selec-ted magazines.
	-2	Function was not executed because the programmed wear group number does not exist.
	-3	Function was not executed because the programmed magazine number does not exist.
	-4	Function was not executed because the function wear group is not enabled (MN_TOOL_MANAGEMENT_MASK).
	-5	Function was not executed for other reasons.

All parameters are optional.

If SETTA is not parameterized, the active setting refers to all loaded tools that are ready for use in the TO area.

MNR	Magazine number	
	0	Or not specifying the parameter means that activation refers to all magazines - independent of whether a distance relationship to a spindle location exists.
	-1	All magazines with a distance relationship 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 (corresponds to calling SETTA several times with the specific magazine numbers that are involved). This means there can be e.g. 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, then this means that 2 tools from the tool group are allocated the active state.
If several magazines are addressed (MNR ≤ 0), then 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. Magazines ($TC_MPP4 Bit 1=1) are not taken into account.		
VNR	Wear group number	
	0	The active setting refers to the entire magazine.
	-1	Active wear group ($TC_MAP9). If there is no active wear group in any of the magazines, then status -1 is returned and the status of the tools is not changed.
	-2	Or if the parameter is not specified, then this means that activation applies to the whole magazine. Especially parameter VNR is not specified or set to -2, if the system does not operate with the wear group function.
USEKT	Tool subgroup	
	0	Or if the parameter is not specified, then all tools in the group are considered.
	-1	The currently programmed value of USEKT is used.

A search strategy can be set in parameter $TC_MAMP3 for the tool to be activated by SETTA.

Note

It is mandatory to set the wear group for the function SETTA.

If SETTA is programmed for a multitool instead of for a magazine, alarm 6462 "[Channel %1:] block %2 command %' can only be programmed for magazines. '%4' means no magazine".

5.8.27 RESETMON - Language command for setpoint activation

RESETMON (state, t, d, mon, resetStates)

Sets the actual value of the tool to the setpoint.

The command is used for tools as well as for multitools.

state	Return parameter which can consist of the following values:	
	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 NCK, i.e. the command has not been executed.
t	Internal T number	
	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 parameter). 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.	
	d > 0	The command applies specifically to the specified D number.
mon	Optional bit-coded parameter. 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 edge(s) are set to the setpoints.	
	mon > 0	Precisely the actual value of the specified monitoring type is handled. Possible values 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.
	mon < 0	Precisely the actual value in the monitoring type specified in "absolute value mon" is handled. There is no restriction by 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	Optional bit-coded parameters	
	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
	Bit 13	Not permitted
	Bit 14	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 analog PI service PI_TRESMO, this machine data is also effective.

Note

There is no explicit generation of alarms. Users can carry out the error handling themselves via the **state** parameter.

Multitool

The command is extended for a multitool:

RESETMON (state, MTno, d, mon, resetStates)

State specifies the status of the command execution:
0 = command 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

In this case, MTno is the multitool number with the following values:

MTno = T-no. = 0	Or tools are handled and therefore also all multitools.
MTno > 0	Precisely this multitool is handled.
MTno < 0	Not defined.

d is an optional parameter and designates the D number of the tool cutting edges in the multitool that should be reset.

Dno = 0	If the parameter is not specified, or if the tool is programmed with 0, then 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 refers to the tools in the multitool.

resetStates refers on one hand to states of the individual tool, which when reset to the setpoints must be additionally changed; however here, for this special programming, also the corresponding MT states, that are changed if at least the same state was changed in one of the tools loaded in the multitool. This includes:

Bit	Value	Meaning
1 =	0	Multitool status "enabled" remains unchanged
	1	Multitool status "enabled" is set
2 =	0	Multitool status "disabled" remains unchanged
	1	Multitool status "disabled" is deleted if permitted by the monitoring data and the 4th parameter is set accordingly.
10 =	0	Multitool status "to unload" remains unchanged
	1	Multitool status "to unload" is deleted

Note

If this optional parameter is not programmed, then the value of MD $MN_TOOL_RESETMON_MASK is valid as implicit value.

Analog 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", keeps 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.

		T	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 "Pre-warning limit" status if monitoring data permit this	X	-
Bit 7	Delete "was in use" status	X	-
Bit 10	Delete "to unload" status	x	x

Note

Bit 2 = 1, bit 16 = 1 of $TC_MTP8:

This state is then deleted in the tool if the monitoring values of the active monitoring of the cutting edges permit this. This state is then deleted in the multitool if the tool loaded last has deleted its "disabled" state.

Bit 2 = 1, bit 16 = 0 of $TC_MTP8:

This state is then deleted in the tool if the monitoring values of the active monitoring of the cutting edges permit this. This state is then deleted in the multitool if at least one loaded tool has deleted its "disabled" state with the programming of RESTMON.

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. Whereby, the state value of the tools whose time monitoring has been changed, is changed in accordance with specifications of the last parameter.

As 500 is the number of a multitool, the state of this multitool is also changed in accordance with specifications 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 cancelled (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.

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")

5.8.28 DELTC - Delete toolholder data record

The function "Tool holder orientation" must be active. The function can be additively superimposed over the functions TMBF, TMFD, TMMO and TMMG.

DELTC(n,m)

n	First number of the tool holder data area the values of which shall be set to zero. This parameter is optional. If it is not specified, all tool holder data records are set to zero starting at the lowest through to the highest number.
m	Last number of the tool holder data area the values of which shall be set to zero. This parameter is optional. If it is not specified, then the tool holder data record specified by n is set to zero. If m is greater than the highest number of a tool holder data record in this channel, then those data records up to the highest number are set to zero.

The tool holder data records are defined by the system variables $TC_CARRx. Only the command $TC_CARR1[0] was available up to now for setting all data records 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 contents of DELTC() are the same as $TC_CARR1[0]=0= set all data records to zero.

The parameters n, m have to be programmed with values larger than zero. Other values lead to an alarm.

Parameter n must be less than m. Programming otherwise leads to 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 records 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), then DELTC will also generate an alarm.

Example

In the TO unit there are 14 tool holder records defined with the numbers 1 to 14.

DELTC(5,8) ;sets the values of the data records 5, 6, 7, 8 to zero

DELTC(5,20) ;sets the values of the data records 5, 6, 7, ..., 14 to zero

DELTC(9) ;sets the values of the data record 9 to zero

DELTC() ;sets the values of the data records 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 may be max. 14

DELTC(20) ;error → alarm - n may be max. 14

5.8.29 TCA - Tool selection/tool change irrespective of tool status

Overview

This function is only available for TMMO and TMMG.

It is necessary for certain routines (e.g. measuring cycles) to load a specific tool onto the spindle / the tool holder for tool change regardless of its status (e.g. a disabled tool).

TCA("Tool name", duplo no., tool holder no.)

"Tool name"	Identifier of the tool to be loaded
Duplo no.	Duplo number of tool to be loaded. This parameter is optional. If it is not specified, then the tool with the lowest duplo number is loaded.
Tool holder no.	Tool holder or spindle on which the change is to take place. This parameter is optional. If it is not specified, the change refers to the currently set or programmed master spindle / master 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.)

TCA behaves like the T command in respect of 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. $MC_TOOL_CHANGE_MODE=0)**
   Configuration 1x turret, 1x tool holder
   There are 2 tools with the identifier "Finish cutter" and duplo numbers 1 and 2.
   TCA("Finish cutter", 1.1)
   The tool "Finish cutter" with duplo number 1 is loaded onto tool holder 1.
   With the machine configuration assumed above, the following programming would have the same result:
   TCA("Finish cutter")
   The duplo number is not specified, this means that the tool with the lowest duplo number is changed, i.e. duplo "1".
   The tool holder no. is not specified. Therefore, the change is effective for the current master tool holder, i.e. "1".
2. **Change with M06 ($MC_TOOL_CHANGE_MODE=1)**
   Configuration: 1x chain magazine, 2x spindles, spindle_1 is the master spindle. 4 tools are loaded, "MILLER_20MM", with duplo numbers 4, 5, 8 and 15.
   "MILLER_20MM", duplo "8" was disabled and must be measured. Measuring takes place on spindle 2.
   TCA("MILLER_20MM", 8.2)
   M2=6
   The tool "MILLER_20MM", duplo "8" is prepared for spindle "2" and changed.
   In this case, the following programming would lead to a different result:
   TCA("MILLER_20MM")
   M06
   Tool "MILLER_20MM" with duplo "4" (lowest duplo number) is prepared for spindle "1" (this is the master 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".

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.

Example:	$TC_VDITCP[2]=101	("101" - identifier that the PLC is not allowed to reject)
	TCA("Miller",1)	

Supplement to the definition if the programmed tool is a multitool

Using the TCA command, a specific tool with "tool name"/"duplo number" can be changed to the tool holder independent 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 removed 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 e.g. an MT name, then Alarm 6460 "[Channel%1:] block %2 command %3 can only be programmed for tools. %4 means no tool." is generated.

5.8.30 Replacement of the TCA command

With the TCA command, it is possible to select a tool by specifying the duplo number.

Substitution of the command is possible, a description of this can be found in Section "Function replacement (Page 103)".

Up until then, it is only possible to substitute using the procedures predefined using the general replacement function MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB.

In order to use the same replacement subprogram for the TCA and T replacement, the following replacement is inserted for the TCA command – analogous to the T function:

With MD15710 $MN_TCA_CYCLE_NAME, the replacement of the TCA command can be activated by specifying a program name. The parameters of the TCA command are transferred to the replacement subprogram in the following system variables:

$C_TS	Tool identifier (as in the existing T replacement)
$C_DUPLO	Duplo number
$C_THNO	Tool holder / spindle number

In addition, there are system variables, which can be used to query whether the substitution of the TCA command is active and which parameters have actually been programmed:

$C_TCA	TRUE : Replacement of the TCA command is active
$C_TS_PROG	TRUE: $C_TS contains the programmed tool identifier
$C_DUPLO_PROG	TRUE: $C_DUPLO contains the programmed duplo number
$C_THNO_PROG	TRUE: $C_THNO contains the programmed tool holder / spindle number

Behavior regarding other replacements:

In the TCA replacement subprogram, tool programming replacement is not carried out. All other replacements are carried out. Conversely, no TCA replacement is carried out in a replacement subprogram for tool programming.

If the TCA command is programmed when the tool management or tool monitoring is not active, then the part program processing is interrupted with Alarm 6431 "Tool management or tool monitoring not active" (this is the existing behavior). If TCA replacement is configured, then in this case, part program processing is also aborted with alarm 6431.

Note regarding MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB:

If the TCA command is renamed, e.g. to "_TCA", by appropriately configuring MD10712, then the TCA replacement is realized with the interpretation of the "_TCA" command.

Example with a common replacement cycle for T and TCA replacement:

Machine data:
$MN_T_NO_FCT_CYCLE_NAME = "T_SUB_PROG"

$MN_TCA_CYCLE_NAME = "T_SUB_PROG"

Replacement subprogram for T and TCA replacement:

```
N1000 PROC T_SUB_PROG DISPLOF SBLOF
;common part
...
;different tool selection:
N2000 IF ($C_TCA == 1)
N2010         ; TCA replacement active
N2020         IF ( $C_DUPLO_PROG == 1) AND ( $C_THNO_PROG == 1)
N2030             TCA( $C_TS, $C_DUPLO, $C_THNO )
N2040         ELSE
N2050             IF ( $C_DUPLO_PROG == 1) AND ( $C_THNO_PROG == 0)
N2060                 TCA( $C_TS, $C_DUPLO)
N2070             ELSE
N2080                 IF ( $C_DUPLO_PROG == 0) AND ( $C_THNO_PROG == 1)
N2090                     TCA( $C_TS, ,$C_THNO)
N2100                 ELSE
N2110                     TCA( $C_TS)
N2120                 ENDIF
N2130             ENDIF
N2140         ENDIF
N2150 ELSE
N2160         IF ( $C_T_PROG == 1) OR ( $C_TS_PROG == 1)
N2170             ; T–replacement active
N2180             IF ( $C_T_PROG == 1)
N2190                 T[ $C_TE ] = $C_T
N2200             ELSE
N2210                 T[ $C_TE ] = $C_TS
N2220             ENDIF
N2230         ENDIF
N2240 ENDIF
...
M17
```

5.8.31 TCI - Change tool from buffer into magazine

Overview

This function is available for TMMG.

The command TCI returns the tools from buffer locations back to 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 carried out in the same way as for a programmed tool change with T (see Section "Prepare a tool change (Page 72)").

Note

TCI cannot be programmed together with M06 in one NC block. Tool change preparation and execution are carried out 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, then the current master tool holder is automatically selected.

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 with the programmed tool holder or master tool holder by $TC_MLSR (alarm 6454)
- if no distance table is defined either for the buffer locNo or 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.

In order to successfully program TCI it is necessary that the specified location number locNo is 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. In order to use this language command in your own cycle programs, the location number can be identified using the system variable $P_MAGNREL, $P_MAGREL.

Example

The following magazine configuration is present:

- Magazine 1

In the buffer magazine with 5 locations, the following is defined:

- 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:

TCI(2): generates alarm 6450

TCI(5): changes 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.

The buffer number transferred in the DB72 has to be evaluated.

Multitool

The TCI command also changes multitools from internal magazines into the magazine.

5.8.32 GETFREELOC - Search for empty location

Overview

This function is available for TMMG.

For a given tool, search for an empty location in those magazines assigned to the specified loading location or the specified spindle / tool holder 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 when searching for an empty location by the PI service or for a programmed tool change.

Alarm 14020 is generated if less than three parameters are programmed.

Note

GETFREELOC only reserves the empty location that is found if the optional 6th parameter 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.

magNo	The parameter is not only an input parameter, but also a result parameter.	
	Input value	
	> 0	Magazine number of the magazine in which the search is to be made. In so doing, it should be noted that only magazine numbers are valid that can be reached from the programmed reference location (parameter refMag/refLog).
	0	The magazine in which the search is to be made is not specified. Search starts in the magazine, which corresponding to the selected search strategy, is the first.
	Result 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 loading 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 (irrespective of the value of this 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	The parameter is not only an input parameter, but also a result parameter.	

	Input value	
	> 0	Magazine location number of the location which should be checked for accepting the specified tool. If magNo=0 is programmed, then a value locNo > 0 is ignored.
	0	No specification of the magazine location. Search starts with the location which is the first according to the set search strategy.
	Result val-ue	
	> 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). "S" = buffer magazine "L" = loading 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	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", then the number of the location in the loading magazine is specified for which a search should 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 then the search for the **master tool holder** is carried out for refMag = "**S**". When refMag = "**L**", the search is carried out for **location number = 1 in the loading magazine**. When refMag = "-", the parameter is not taken into account.	

withReserv	**"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/M6). If the owner magazine address of this tool is = 0 and the tool is on a magazine location (i.e. 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, then this is replaced by the new magazine address, if the new address is different to the old address. If the location of the old owner magazine address is reserved, the reservation is cancelled. If a location is already reserved for the programmed tool, this reservation is cancelled before the new reservation is set. "-" = leave the reserved status of the found/checked empty location unchanged. The parameter is optional. If it is not programmed, then "-" 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 NCK 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.

An alarm is generated if the TMMG is not active.

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 relationships). 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 regarding loading location 2, 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 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 regarding loading location 2, 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 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 is defined with T no. = 4

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 master 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 NCK 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 number 2, a
; search is made for an empty location.
; The programming is illegal 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")
; 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. However, since the tool is not loaded
; the last parameter cannot be programmed with the value "S".

; The references magNo/locNo return the status values -8 / -8 = "Parameter 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.

5.8.33 DELMLRES - Delete location status "reserved for tool in the buffer"

Overview

This function is available for TMMG.

Cancel the location reservation, type resType of the location with the number magNo.

DELMLRES (magNo, locNo, resType)

state	Result of the command execution	
	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
magNo	Magazine number of the magazine that contains the location.	
locNo	Magazine location number of the location whose reservation should be deleted.	
resType	Type of reservation to be deleted. "**L**" Reservation for "tool to be loaded" ("L "= "Tool to be **L**oaded") "**S**" Reservation for "tool in the buffer" ("S" = tool in **S**pindle" or general tool in the buffer). The parameter is *optional*. If it is not programmed, then the reservation that is set is deleted - either "L" or "S". If no reservation is set, then the command has no effect.	

If the reservation of adjacent locations is also reserved for an oversized tool, then also these (sub) location reservations are cancelled.

Example 1

For the oversized tool located in gripper 1 (magazine No. 9998, location No. = 4), the location reservation, for tool in the buffer, is to be deleted.

```
def int tNo, magNo, locNo
tNo = $TC_MPP6[ 9998, 4 ]
magNo = $A_MYMN[ tNo ]          ; owner magazine
locNo = $A_MYLN[ tNo ]          ; owner magazine location, which
                                ; was reserved
                                ; when loading the tool
state = DELMLRES ( magNo, locNo, "s" )
```

```
                                ; the same meaning would be
"DELMLRES(magNo, locNo)"
if (state < 0 ) gotof error
```

If it should be ensured that the location is also reserved for the tool that is intended to be loaded on change, then 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.

```
def int tNo, magNo, locNo
tNo = 3    ; this tool should be loaded
```

GETFREELOC (magNo, locNo, tNo, "S", 2, "L") ; search for empty location with reservation
if ((magNo > 0) and (locNo > 0)
 and ("something is not as required"))
 state = DELMLRES (magNo, locNo, "L")
 if (state < 0) GOTOF ERROR
endif
MVTOOL(state, 9998, 2, magNo, locNo)

5.8.34 DELMLOWNER - Delete owner magazine location of the tool

Overview

This function is available for TMMG.

Delete the tool or the multitool in the buffer magazine - owner magazine location.

DELMLOWNER(t)

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
t	T number of the tool in which the owner location is deleted	

If the owner location is reserved for this tool, then 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.

Remark:

If the t number of the tool is not known, then it can be determined with one of the following commands:

```
def int tNo
tNo = GETT( name, duplo )
```

```
       ; if only the name and duplo number of the tool are known
tNo = $TC_MPP&[ magNo, locNo ]
       ; if only the magazine location is known, where the tool is
contained
tNo = $TC_MPP&&[ magNo, locNo ]
       ; if only the magazine location is known,
         ; whose reservation is to be deleted
```

Multitool

The command can also be programmed 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.

5.8.35 $P_USEKT - Tool change only with tools of subgroup

Overview

This function is only available for TMMO and 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 the bits 0 - 3 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". An alarm 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.

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.

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).

If the tool change is programmed with the function T="location", the $P_USEKT command can also be used. However, it is then necessary that the tool has set at least one bit of the programmed $P_USEKT value on the programmed magazine location in parameter $TC_TP11. Otherwise, the tool of another magazine location is selected. The following still applies for this setting: At NC start the value of $P_USEKT = 0 is pre-assigned, irrespective of $MC_RESET_MODE_MASK and $MC_USEKT_RESET_VALUE.

Note

The system variable $TC_TP11 was not evaluated in NCK up to now. The value is automatically assigned 0. A check should be made in existing data records whether the values included here are suitable.

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 working with the function T=location, $P_USEKT **cannot** be programmed.

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. Maximum four different tool subgroups can be implemented through the configuration of NCK; i.e. only bits 0, 1, 2, 3 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.

Example

The tool group "Miller_25" comprises four tools.

(The following applies: Tool_Change_Mode=1)

Miller_25	Duplo 1	T_No. 1	TP11=1
Miller_25	Duplo 2	T_No. 2	TP11=2
Miller_25	Duplo 3	T_No. 3	TP11=4
Miller_25	Duplo 4	T_No. 4	TP11=8

%MPF	
…	
T="Miller_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="Miller_25"	
M06	"Miller_25", duplo 2 loaded at change
…	
$P_USEKT=9	
…	
T="Miller_25"	
M06	"Miller_25", duplo 1 or duplo 4 loaded at change (depending on the selected search strategy)
…	
$P_USEKT=0	
…	
T="Miller_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=15	
…	
T="Miller_25"	
M06	Every tool in this group can be loaded, as all bits are set. The search strategy that has been set prevails.

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 on 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 replacement tool. Only tools that have one of the bits of the disabled tool in $TC_TP11 can be replacement tools.

5.8.36 TOOLGNT/TOOLGT - Tool groups

Comment

This function is only 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 language commands allow information to be obtained about the tools in the tool group.

TOOLGNT("identifier")

TOOLGT("identifier", i)

Result	Result value	
	> 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 NCK defines the sequence of the tools in the tool group and this changes in the course of the programmed tool change with tools from this group.

Example

The following tool is available "Drill_6mm"/Duplo_1, "Drill_6mm"/Duplo_2 and "Drill_6mm"/Duplo_3.

The number of tools of the group "Drill_6mm" is read-out first.

```
R1=TOOLGNT("Drill_6mm")        R1=3
```

The T number of this tool is then determined.

```
R11=TOOLGT("Drill_6mm,1)
R12=TOOLGT("Drill_6mm,2)
R13=TOOLGT("Drill_6mm,3)
```

5.8.37 $P_TMNOIS - is a number, T number, magazine number or MT number

This function is available with TMBF, TMFD, TMMO and TMMG.

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.

def int result

def int no=4711

result = $P_TMNOIS[no] ; is no a T no., an MT no., or a magazine no.?;

Result	Description
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. If the command is programmed in the TMFD function, then the result is also –3. If no = 0 then the result is –3

5.8.38 $P_TOOLEXIST - define the existence of a tool

$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, then the T no. that returns an existing tool is

$P_TOOLEXIST=true. $P_TMNOIS can be used as an alternative.

Example of $P_TMNOIS / $P_TOOLEXIST

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:

```
N1 DEF INT no=2 ; Number to be checked
```

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).

```
N5 if ( ( $P_TMNOIS[ no ] != 0 ) and
        ( $P_TMNOIS[ no ] != 1 ) )
N6 $TC_TP2[ no ] = "Tool_2" ; Create new tool
N7 endif
```

Example 2: A new MT is to be created (with a T number which is not identical to that of an already existing multitool, or an already existing tool, or an already existing magazine):

```
N5 if ( ( $P_TMNOIS[ no ] != 0 ) and
        ( $P_TMNOIS[ no ] != 1 ) and
        ( $P_TMNOIS[ no ] != 2 ) )
N6 $TC_MTPN[ no ] = 5 ; Create new MT with five locations
N7 endif
```

Example 3: An explicit number is to be checked as to whether it belongs to a defined tool:

```
N5 if ( $P_TOOLEXIST[ no ] == true )
N6 $TC_TP9[ no ] = 1 ; Defines the type of tool monitoring
N7 endif
```

5.8.39 $A_TOOLMN - Read magazine No. of tool

Comment: TOOLMN stands for = "**tool m**agazine **n**umber". The name component $A_TOOL was selected to show the association with the existing system variables.

Name	**$A_TOOLMN[t]**			
Meaning	Returns the magazine number of the tool with T no.=t. If the tool is 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			
Value range	1-32000			
Indices	Meaning			Value range
	The index specifies the T number			1-32000
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	x	-	x	-
Implicit preprocessing stop	x	-		

5.8.40 $P_MTOOLN / $P_MTOOLMT - determine the number of multitools /MT number

This function is available for TMMG.

The following system parameters can be used to obtain an overview of the multitools defined in the TO unit assigned to the channel.

result = $P_MTOOLN Number of multitools known to the channel.
Multi**TOOLN**umberOf

Result	Result value
> 0	Number of defined multitools
0	No multitool defined
-1	TMMG function not active
-2	Multitool function not active

Example

```
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 con-
                                   secutive R parameters
```

```
endfor
mtName = $P_MTP2[$P_MTOOLMT[i]] ;write the name of the last multitool after mtName
```

5.8.41 $P_MTOOLNT / $P_MTOOLT - number of tools in the multitool

This function is available for TMMG.

Note

Also refer to the analog tool-specific system parameters $P_TOOLNT, $P_TOOLT.

The following system parameters can be used to obtain an overview of the multitools defined in the TO unit assigned to the channel.

result = $P_MTOOLNT[mt]	Number of tools in the multitool mt = Multitool number **M**ulti**TOOLN**umber Of **T**ools

Result value

Result	Result value
> 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

result = $P_MTOOLT[mt , i]	T number of the i–th tool in the multitool with mt = Multitool number i = i–th tool in the multitool; i = 1, ..., $P_MTOOLNT **M**ulti**TOOLT**number

Result	Result value
> 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. Three tools with the T numbers 11, 22 and 33 are loaded to the locations 1, 2 and 5.

```
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 three tools and not over six MT locations
```

```
r[i] = $P_MTOOLT [500, i]          ;Write the T numbers of the loaded tools to successive R
                                   parameters
                                   ;r1=11, r1=22, r3=33
endfor
```

5.8.42 $A_TOOLMLN - Read magazine location No. of tool

Comment

TOOLMLN stands for = "**tool m**agazine **l**ocation **n**umber".

Name	**$A_TOOLMLN[t]**			
Meaning	Returns the magazine location number of the tool with T no.=t. If the tool is 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			
Value range	1-32000			
Indices	Meaning			Value range
	The index specifies the T number			1-32000
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	x	-	x	-
Implicit preprocessing stop	x	-		

Comment: 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.

5.8.43 $P_TOOLND - Read number of cutting edges for tool

Comment

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	
Value range	Standard: -1, 1 - 9 "Flat D number" function: -1, 1 - "Machine data value for the maximum number of D numbers"	

Name	$P_TOOLND[t]			
Indices	Meaning			Value range
	The index specifies the T number			1-32000
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	x	-	-	-
Implicit preprocessing stop	-	-		

Function "flat D numbers" (only without active tool management)

If the function "flat D numbers" is active, the behavior differs somewhat. With parameter t=1 the total number of offset data records of the TOA unit is returned. Other values for t return -1. If no offset data record is defined in the TOA unit, -1 is returned.

5.8.44 $A_MONIFACT - Factor for reading 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
		-

Name	**$A_MONIFACT**			
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous 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 the OPI block TS are thereby converted at the interface. In the NCK, the values are retained as before. These values are real-time values.

Reading the OPI:

The time values are divided by the current value of $A_MONIFACT and transferred.

Writing the OPI:

The time values output by the OPI are multiplied by the current value of $A_MONIFACT and stored in the NCK.

Example

The actual values are specified (units in real time, i.e. normalized to $A_MONIFACT = 1).

Setpoint 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.

$A_MONIFACT = 2 is programmed in the part program (clock runs faster). The actual tool life displayed on the monitor jumps and continues to run in real time. The setpoint tool life and prewarning limit displayed also jump as soon as **$A_MONIFACT = 2** takes effect.

Setpoint tool life: 5 minutes

Actual tool life: 1 Minute - in **half a minute** the prewarning limit is reached

Prewarning limit: 0.5 minutes

5.8.45 $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.	Actual value	Setpoint	Absolute
$TC_TP1	$TC_MOP2	$TC_MOP11	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 NCK 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

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] = 0.5

Q(T2) = $TC_MOP4[2,1] / $TC_MOP13[2,1] = 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.	Actual value	Setpoint	Absolute
$TC_TP1	$TC_MOP2	$TC_MOP11	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			
Value range	0-1			
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	X	X	X
Implicit preprocessing stop	-	X	-	

Boundary conditions

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.

5.8.46 $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			
Value range	1-32000			
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

5.8.47 $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			
Value range	1-32000			
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous 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.

5.8.48 $A_MYMTN / $A_MYMTLN - $A_TOOLMTN / $A_TOOLMTLN - tools in the multitool

In addition to the already defined parameters $A_MYMN / $A_MYMLN and $A_TOOLMN / $A_TOOLMLN parameters

$A_MYMTN / $A_MYMTLN - or with the same contents $A_TOOLMTN / $A_TOOLMTLN - specify in which MT and at which location in the MT the tool is located.

The following applies:

$A_MYMTN = $A_TOOLMTLN

and

$A_MYMTLN = $A_TOOLMTLN.

This therefore states that the tool remains permanently in the MT.

result = $A_MYMTN [t]	Tool Tno. t = 1, ..., 32000 MYMultiToolNumber
result = $A_TOOLMTN [t]	Tool Tno. t = 1, ..., 32000 TOOLisOnMultiToolNumber

Result	Result value
>0	The tool with the T number t is on the multitool with the result number
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.)

result = $A_MYMTLN [t]	Tool Tno. t = 1, ..., 32000 MYMultiToolLocationNumber
result = $A_TOOLMTLN [t]	Tool Tno. t = 1, ..., 32000 TOOLisOnMultiToolLocationNumber

Result	Result value
>0	The tool with the T number t is on the location with the result number within a multitool
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.)

5.8.49 $P_TOOLNT / $P_TOOLT - T numbers

This function is available for TMMO, TMMG, TMFD and TMBF.

Name	**$P_TOOLNT / $P_TOOLT[i]**			
Meaning	These system variables enables an overview of the tools defined in NCK. $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			
Value range	1-32000			
Indices	Meaning			Value range
	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 synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

The following applies specifically for TMFD:

$P_TOOLNT returns value 1 provided D offsets have been defined and returns value 0 when there are no D offsets.

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.

5.8.50 $P_TOOLD - D numbers

This function is available for TMMO, TMMG, TMFD and TMBF.

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		
Value range	1-32000		
Indices	Meaning		Value range

Name	$P_TOOLND / $P_TOOLD[t,i]			

	t = T number
	i = i-th T no.; i=1, ..., $P_TOOLND			

Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

5.8.51 $P_TOOLNDL - Number of defined DL offsets

This function is available for TMMO, TMMG, TMFD and TMBF. 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			
Value range	1-32000			
Indices	Meaning			Value range

	t = T number
	d = D number			

Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

5.8.52 $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 master 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 master 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			
Value range	1-32000			
Indices	Meaning			Value range
	s = 1, ..., MAXNUM_AXES_PER_CHAN s = 0 indicates the master tool holder			
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Example, see "$A_USEDT - Workpiece counting (Page 339)".

5.8.53 $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 master 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 master 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			
Value range	1-32000			
Indices	Meaning			Value range

	S = 1, ..., MAXNUM_AXES_PER_CHAN
	I = 1, ..., $A_USEDND			

Access	Read in part program	Write in part program	Read in synchronous 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 master 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.
```

5.8.54 $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 master 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 master 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	
Value range	1-32000	
Indices	Meaning	Value range

	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 synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

5.8.55 $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			
Value range	1-32000			
Indices	Meaning			Value range
	i = 1, ..., $P_MAGN			
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Example, see "Example of magazine configuration system variables (Page 345)".

5.8.56 $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			
Value range	1-32000			
Indices	Meaning			Value range
	i = 1, ..., $P_MAGNDIS			
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Example, see "Example of magazine configuration system variables (Page 345)".

5.8.57 $P_MAGNS / $P_MAGS - Toolholder

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			
Value range	1-32000			
Indices	Meaning			Value range
	n = 1, ..., max. tool holder number			
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Example, see "Example of magazine configuration system variables (Page 345)".

5.8.58 $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 no. / tool holder no. > 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			
Value range	1-32000			
Indices	Meaning			Value range

	m = 1, ..., $P_MAGNREL
	n = 1, ..., max. tool holder number			

Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Example, see "Example of magazine configuration system variables (Page 345)".

5.8.59 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.

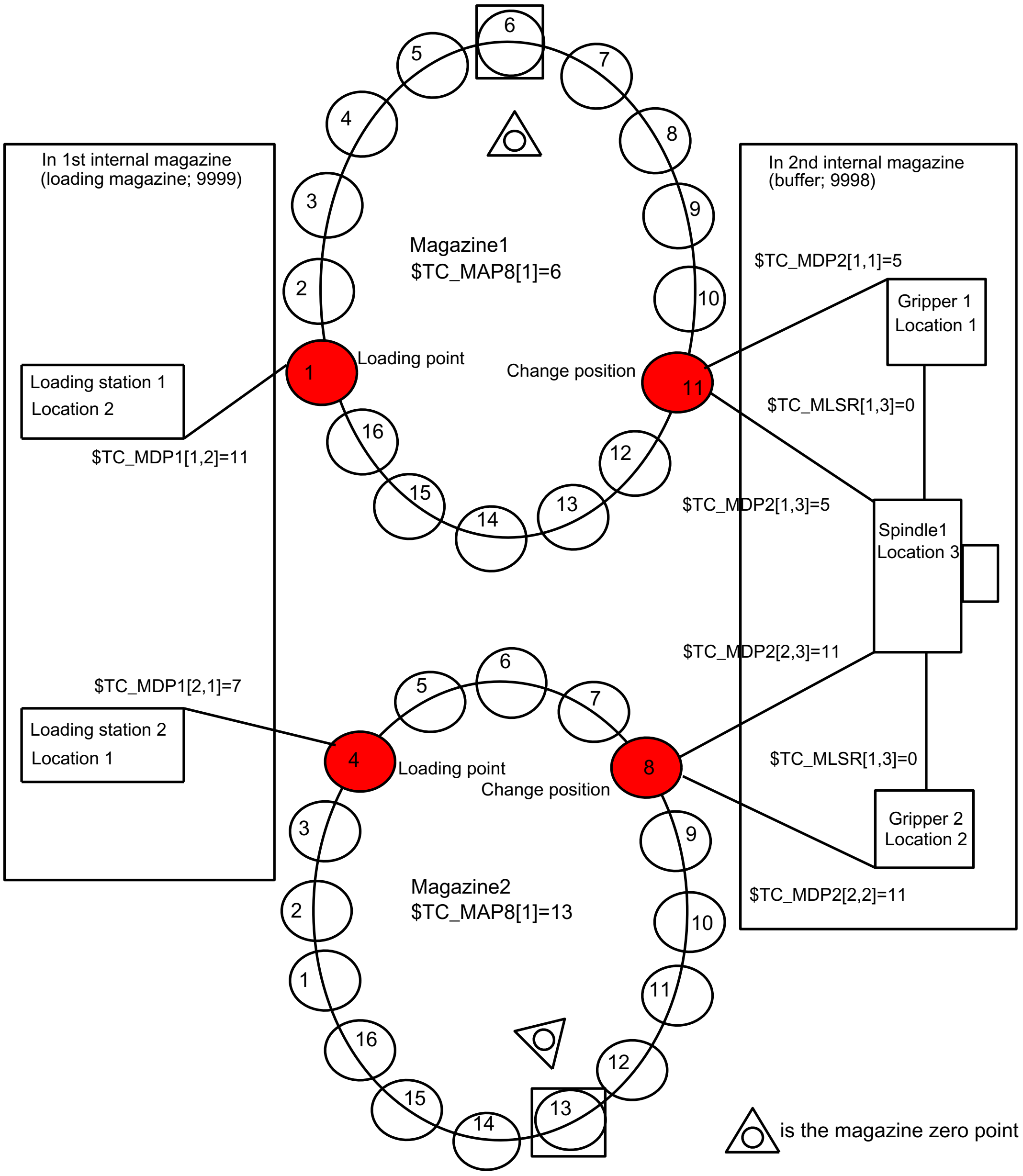


Image 5-10 Magazine configuration

N10 def int noOfMag=0, noOfLoc=0, noOfDist=0, noOfRel=0,

noOfSpindles=0, spindleNo=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 loading 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=        ;noOfSpindles contains value = 1
$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
```

5.8.60 $P_MAGNH / $P_MAGNHLT / $P_MAGHLT - Location type hierarchies

Overview

This function is available for TMMG.

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 n is outside of the defined range -1 TMMG is not active $P_MAGHLTn,m] m-th location type of hierarchy n > 0 Successful read access 0 m is outside of the defined range -1 TMMG is not active -2 Hierarchy n does not have any defined location types			
Data type	INT			
Value range	1-32000			
Indices	Meaning			Value range

	n = 1, ..., $P_MAGNH
	m = 1, ..., $P_MAGNHLT			

Access	Read in part pro-gram	Write in part pro-gram	Read in synchro-nous action	Write in synchro-nous action
	X	-	-	-
Implicit prepro-cessing stop	-	-	-	

Example

The following three hierarchies are defined:

Hierarchy 1: 5 < 4 < 3:

$TC_MPTH[0,0] = 5

$TC_MPTH[0,1] = 4

$TC_MPTH[0,2] = 3

Hierarchy 2: 7 < 8:

$TC_MPTH[1,0] = 7

$TC_MPTH[1,1] = 8

Hierarchy 3: 1 < 2 < 9 < 6:

$TC_MPTH[2,0] = 1

$TC_MPTH[2,1] = 2

$TC_MPTH[2,2] = 9

$TC_MPTH[2,3] = 6

We want to know *how many hierarchies* in total are defined and *how many magazine location types are contained in each hierarchy*.

```
N10 def int noOfH, noOfTypes[8], locTypeNo
N100 noOfH=$P_MAGNH                ;noOfH is assigned value = 3
N220 for i=1 to noOfH
N240 noOfTypes[i - 1]=$P_MAGNHLT[ i ]
                                   ;Set the values 3, 2, 4 in the
                                   array
N260 endfor
```

We also want to know *which magazine location types* are defined in the 2nd hierarchy

```
N220 for i=1 to noOfTypes[1]
N240 MSG ("Magazine no.="<<$P_MAGHLT[2, i])
                                   ;Display values 7, 8
N260 endfor
```

5.8.61 $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	
Value range	1-32000	
Indices	Meaning	Value range

Name	$P_MAGNA / $P_MAGA[i]			
	i = 1, ..., $P_MAGNA			
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

5.8.62 Additional language commands

Name	$P_TOOLNO			
Meaning	Active tool numbers T0 to T32000, with TMFD, T can have eight digits. Generally, the command should not be used when magazine management is active. When magazine management is active GETTEXET 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			
Value range	1-32000			
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_TOOLP			
Meaning	Tool number last programmed Command is available for TMBF, TMFD and TMMO. It is analogous to the TMMG-specific command GETSELT.			
Data type	Integer			
Value range	1-32000			
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_TOOL			
Meaning	Active tool cutting edge (Dx)			
Data type	Integer			
Value range				

Name	$P_TOOL			
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous 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			
Value range	0-6			
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-		

Name	$P_TOOLL[n]			
Meaning	Active tool total length; n = 1...3			
Data type	REAL			
Value range				
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_TOOLR			
Meaning	Active radius			
Data type	REAL			
Value range				
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_TC			
Meaning	Active tool holder			
Data type	Integer			
Value range				
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_TCANG[n]			
Meaning	Active angle of a tool holder axis; n = 1-2			
Data type	REAL			
Value range				
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_TCDIFF[n]			
Meaning	Difference between calculated and used angle of a tool holder axis for grid (Hirth tooth system) of the angle			
Data type	REAL			
Value range				
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$P_AD[n]			

Meaning	Active tool offset; n = 1...31
	n=1-25 $TC_DP1 to $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)			

Data type	DOUBLE			
Value range				
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous 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 to $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)			

Data type	DOUBLE			
Value range				
Indices	Meaning			Value range
	n: Parameter numbers 1 to 31			
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$AC_MSNUM			
Meaning	Master spindle, return value 0: No spindle configured 1...n: Number of master spindle			
Data type	Integer			
Value range				
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	X	-
Implicit preprocessing stop	X	-	-	

Name	$P_MSNUM			
Meaning	Master spindle 0: No spindle configured 1...n: Number of master spindle			
Data type	Integer			
Value range				
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-	-	

Name	$AC_MTHNUM			
Meaning	Master tool holder Value=0 no master tool holder defined Value>0 number of the master tool holder			
Data type	Integer			
Value range				
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	X	-
Implicit preprocessing stop	X	-	X	-

Name	$P_MTHNUM			
Meaning	Master tool holder Value=0 no master tool holder defined Value>0 number of the master tool holder			
Data type	Integer			
Value range				
Indices	Meaning			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	-	-	-
Implicit preprocessing stop	-	-		

Name	$TC_TP_MAX_VELO			
Meaning	Maximum tool speed			
Data type	REAL			
Value range	[0, DBL_MAX]			
Indices	Internal T number			Value range
Access	Read in part program	Write in part program	Read in synchronous 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			
Value range	[0, DBL_MAX]			
Indices	Internal T number			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action
	X	X	-	-
Implicit preprocessing stop	-	-	-	-

Name	$TC_MPP_SP_MAX_ACC			
Meaning	Only relevant, if $MC_TOOLHOLDER_MANAGEMENT>0: Spindle no. which is linked to the tool holder.			
Data type	Integer			
Value range	[0, MAXNUM_SPIND_PER_CHAN]			
Indices	Magazine no., magazine location no.			Value range
Access	Read in part program	Write in part program	Read in synchronous action	Write in synchronous action

Name	$TC_MPP_SP_MAX_ACC			
	X	X	-	-
Implicit prepro-cessing stop	-	-	-	-

Name	$P_MTHSDC			
Meaning	Master tool holder no. or master spindle no. is determined with reference to the active tool for the next D offset selection. > 0 Successful read access 0 No master tool holder or no master spindle available. The next D offset works with T0. -1 TMMG not available			
Data type	Integer			
Value range	[-1, MAXNUM_AXES_PER_CHAN]			
Indices	Magazine no., magazine location no.			Value range
Access	Read in part pro-gram	Write in part pro-gram	Read in synchron-ized action	Write in synchron-ized action
	X	-	-	-
Implicit prepro-cessing 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 because TMFD is active. If read as OPI variable, this applies for the state in the current main run block.			
Data type	Integer			
Value range	[-1, MAXNUM_AXES_PER_CHAN]			
Indices				Value range
Access	Read in part pro-gram	Write in part pro-gram	Read in synchron-ized action	Write in synchron-ized action
	X	-	-	-
Implicit prepro-cessing stop	-	-	-	-

Name	$P_MTHNUM_BEFORE_SEARCH			
Meaning	Master tool holder or master spindle 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 is active. -1: Function is not available because TMFD is active. 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 OPI variable, this applies for the state in the current main run block.			
Data type	Integer			
Value range	[-1, MAXNUM_AXES_PER_CHAN]			
Indices				Value range
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronized 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 because TMFD is active. 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 OPI variable, this applies for the state in the current main run block.			
Data type	Integer			
Value range	[-1, 32000]			
Indices				Value range
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronized 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 because TMFD is active. 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 OPI variable, this applies for the state in the current main run block.			
Data type	Integer			
Value range	[-1, 6]			
Indices				Value range
Access	Read in part program	Write in part program	Read in synchronized action	Write in synchronized action
	X	-	-	-
Implicit preprocessing stop	-	-	-	-

5.8.63 Variables for subroutine replacement technique

Tool management language 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
$C_TS	For T or TCA replacement, supplies the programmed tool identifier ($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)

Tool management language command	Functions
$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)

5.8.64 Variables for tool change in synchronized action

Tool management language 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
$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

Tool management language command	**Functions**
$AC_TC_CMDT	Trigger variable to NCK command output Set for one interpolation cycle when NCK outputs a new command.
$AC_TC_CMDC	Counter for NCK command output This variable is incremented by 1 at each NCK 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 NCK. 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	Read in synchronous action
	X	-	X	-
Implicit preprocessing stop	X	-		

5.9 Definitions when programming data

5.9.1 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.

Deleting data

When data is deleted the memory area is deleted with it 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 is enabled again, i.e. the grinding-specific data is lost.

Action	Program command	Description

Create a tool	Without tool management:
	$TC_DPx[y,z] = value;
		Create tool T if T does not yet exist!
		y = T number
		z = D number

	With active tool management: T_NO = NEWT("tool identifier", duplo number) or $TC_TP1[y] = duplo number; $TC_TP2[y] = "tool identifier"	y = T number

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

Action	Program command	Description
Set tool data	With active tool management: $TC_TPx[y] = value; or $TC_TPx,[GETT("DRILL",DUPLO NO)] = value or $TC_TPCx[y] = value; $TC_TGx[y] = value;	 y = T 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 = T number z = D number
Delete cutting edge data	Without tool management: $TC_DP1[0,0] = 0;	All tools of the channel are deleted, the memory is released.
	With tool management: $TC_TP1[0,0];	When deleting tools, the entries for the location data must also be corrected.
Delete tool data	Without tool management: $TC_DP1[y,0] = 0;	 y = T is deleted, memory is released.
	With tool management: $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.
Delete data for all tools	Without tool management: $TC_DP1[0,0] = 0;	 All tools of the channel are deleted and the memory is released.
	With tool management: $TC_TP1[0,0] = 0;	 When deleting tools, the entries for the location data must also be corrected.

5.9.2 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, ...

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).

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 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.
Create new magazine location	$TC_MPPx[y,z] = value;	Value <>0, y = location number not yet available. Before the data of the first location can be created, the associated magazine must be defined. When the first parameter of the first magazine location to be set up is written, then all magazine locations belonging to the magazine are set up using their default values in accordance with the requirements for 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 magazine 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;	

5.9.3 Tool change

Programming the tool selection

Tool selection can be divided into two 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:

1. Tx; Prepare tool change (select the tool)
2. 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 a T number as 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.

5.9.4 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 after 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 = "MILLER"	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"

5.9.5 Tool transfer from program test

With MD20110 $MC_RESET_MODE_MASK, **bit 3**, you can set whether the active tool and tool offset are to be taken

- (= 1) from the test program which was last terminated in test mode
  or
- (= 0) from the program which was last terminated before the test program 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.

5.10 Programming T=location number

This function is only available when tool management is active. This type of programming is not only suitable for turrets, but for all other types of magazine.

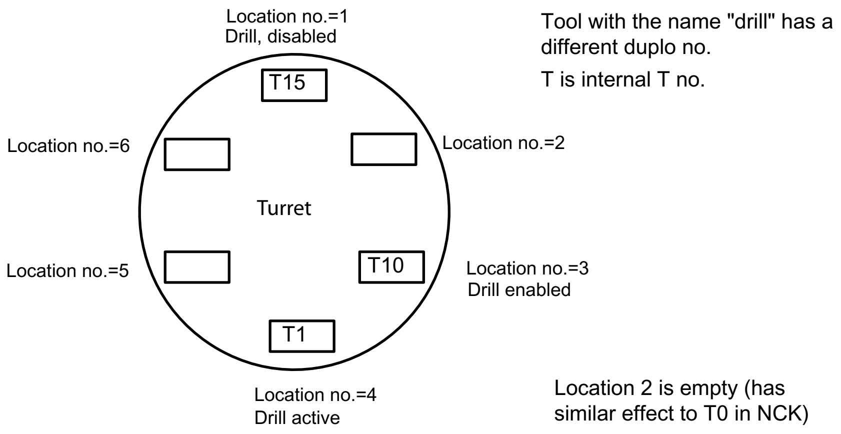


Image 5-11 Programming of T=location number

The programming type is set using machine data MD20310 $TC_TOOL_MANAGEMENT_MASK, bit 16=1:

- 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". The first magazine linked with the tool holder is accessed here. The identifier of the tool in this location is then determined ("Drill").

The subsequent procedure is as if T="Drill" had been programmed. Which of the three tools from the "Drills" group is determined as the first step of the tool change process.

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.
- When the strategy "Take the first tool with *active* status from the group" is applied, T1 is "loaded".

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.

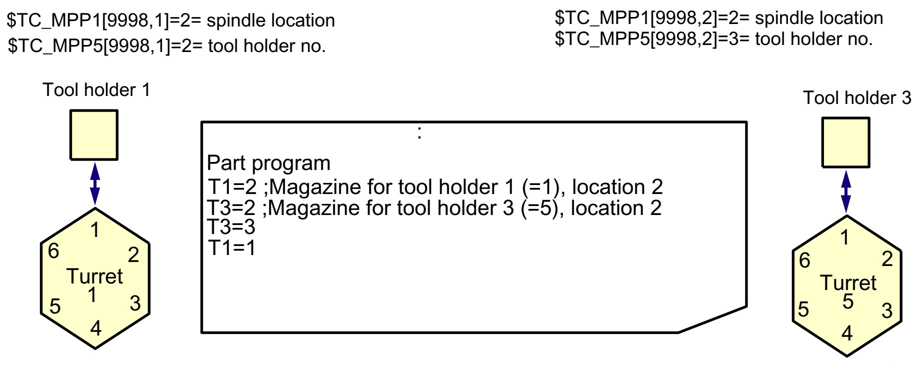
Note

With the T=location function, alternatively T= "Drill" can also be programmed.

T=1; tool

T="Drill"; tool with the identifier, drill

5.11 Call multiple turrets with "T=location number"


$TC_MPP1[9998,1]=2= spindle location
$TC_MPP5[9998,1]=2= tool holder no.
$TC_MPP1[9998,2]=2= spindle location
$TC_MPP5[9998,2]=3= tool holder no.
Tool holder 1
Tool holder 3
Turret 1
Turret 5
1
2
3
4
5
6

```
:
Part program
T1=2 ;Magazine for tool holder 1 (=1), location 2
T3=2 ;Magazine for tool holder 3 (=5), location 2
T3=3
T1=1
```

Image 5-12 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.

5.12 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_NR, 2] = 210	Write tool type for the 2nd cutting edge of the tool "T number"
Select tool	T="DRILL" or: T=GETT("DRILL", DUPLO_NO) or Tx	If there are several tools with this iden-tifier, then the T no. is returned for the first tool possible. Determines T number for "DRILL" with duplo number = DUPLO_NO and se-lects it. Call with T no. e.g. T1,T2,T3,....
Delete tool	$TC_TP1[T_NR,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

5.13 Overview of the remaining OPI blocks of tool management

5.13.1 Magazine data, directory

OPI block TMV

Calculation of line: Magazine number, if a field[ ] is present

Calculation of column: n.a.

NCK identifier	Designation	OPI variables	Type
None	Number of magazines	numActMags	WORD
	Number of the magazine	magVNo[ ]	WORD
	Identifier of the magazine	magVIdent[ ]	String

5.13.2 Tool data, directory

OPI block TV

Calculation of line: Sequential no. of the tools, if [ ] present

Calculation of column: n.a.

NCK 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

5.13.3 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	parDataToolI-dentTD	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
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

5.13.4 Working offsets

OPI block AEV

Calculation of line: Cutting edge number if [ ] available

Calculation of column: n.a.

NCK 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

5.13.5 PI services and language commands for tool management

Overview

FB4 (PI_SERV) or FB7 can be used to start program instance services (PI services) in the NCK 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 NCK by making a request via the PI service.

PI service	Functions	NC language command
MMCSEM	Semaphores for various PI services	
DELETO	Delete tool	DELT("Tool", Duplo)
DELECE	Delete a cutting edge	$TC_DP1[t,d]=0
CREATO	Generate tool	NEWT("Tool", Duplo)
CRTOCE	Create tool specifying 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

PI service	Functions	NC language command
TMRASS	Reset active status	
TMGETT	Confirm 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 NCK-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)
	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"	SETTIA(Stat,m,vnr)
CHKDM (not available in PLC)	Check uniqueness of D nos. in magazine; m=Magazine	Status=CHKDM(m)
	Set tool holder no. (h=holder no.)	SETMTH(h)
	Set master 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
	Language 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 the buffer"	DELMLRES
	Delete owner magazine location of the tool	DELMLOWNER

NC language commands

NCK states are read with the following language commands.

Functions	NC language command
Active tool no. T	$P_TOOLNO
Last programmed tool no. (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 Hirth gearing 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 D, n=1...31	$P_AD[n]
Active tool offsets; transformed; n=1-31	$P_ADT[n]
Active additive offset number	$P_DLNO
T number 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
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

Functions	NC language command
Source 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 master spindle	$AC_MSNUM
Returns the number of the master spindle	$P_MSNUM
Returns the number of the master tool holder	$AC_MTHNUM
Trigger variable to output command of the NCK (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 master 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]
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 load-ing 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 mag-azine 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 numbers 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 param-eter $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

Functions	NC language command
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 (location found is not reserved).
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 in an MT with the command.
TMGETT	T number of the tool with the specified tool identifier / duplo number or the 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

PI service	Meaning
DZERO	Set all D numbers invalid with the "Unique D number" function
CRCEDN	Create new tool cutting edge by specifying the D number
CREACE	Create tool cutting edge
DELECE	Delete a 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

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.

Code carrier connection - Tool Ident Connection (option) 6

6.1 Installation

6.1.1 System requirements

Tool Ident Connection is intended for the SINUMERIK 840D sl target system with SINUMERIK Operate.

Note

Tool Ident Connection is only available in conjunction with the SINUMERIK Operate operating software as of SW 4.5 SP2.

PLC interface "ParmaTM"

Assignment of the PLC interface for the Parameter area (see DB19).

Assignment of the PLC interface for the Parameter area (Page 431)

Functional scope of the PLC interface (K codes) (Page 439)

NCU

- As of NCU 7x0.3
- SINUMERIK Operate as of SW 4.5 SP2

PCU

- PCU 50.5 or higher
- SINUMERIK Operate as of SW 4.5 SP2

Tool management

An active magazine management is required in order to use the Tool Ident Connection option.

Expanding the DB19

If the PLC data block DB19 is not available via the PLC interface for "Tool Ident Connection", then this must be expanded by "Tool Ident Connection" before commissioning the system for the first time, see PLC interface in the DB19 (Page 425)

6.1.2 Installation of Tool Ident Connection

Note

We recommend that you backup your system before setting-up.

According to the system preconditions, if necessary, additional components must be first installed.

The storage medium for software applications for SINUMERIK 840D sl with SINUMERIK Operate is a CompactFlash card with a fixed directory structure. The "Tool Ident Connection" application is stored within this structure in the "addon" directory.

6.1.2.1 Activation of Tool Ident Connection

Configuration files

The following files are available to configure Tool Ident Connection:

Standard configuration files	Directory: CompactFlash card	Directory: PCU 50 WinXP	Directory: PCU 50 Win7
tdiidentcfg.xml	/card/siemens/sinumerik/hmi/ template\toolidentconnection	F:\hmisl\siemens\sinumerik\hmi \template\toolidentconnection	C:\ProgramFiles(x86)\Sie-mens\MotionControl\siemens \sinumerik\hmi\template\tooli-dentconnection
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.SlMcisTdiIdentConnec
tionService,

process:=SlHmiHost1, timeout:=5000, shutdowntime:=5000
```

Note

The `xxx` in `SVCxxx` must be replaced by the free number of a service. Only numbers as of 200 can be used.

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:

- CompactFlash card: /card/addon/sinumerik/hmi/cfg
- PCU 50 (WinXP): F:\hmisl\addon\sinumerik\hmi\cfg
- PCU 50 (Win7): C:/Programs(x86)\Siemens\MotionControl\/addon\sinumerik\hmi\cfg

The oem_sltmlistconfig.xml file must be renamed, depending on the technology. This is explained in the Commissioning Manual IM9, Section "Tool management", Section "Configuring the user interface".

A detailed description of the configuration options for the oem_sltmlistconfig.xml can be found in the Commissioning Manual IM9, Section "Tool management", Section "Configuring the code carrier connection".

The files wkonvert.txt, DefToolDat.txt and wkonvert.mcc are configuration files. They must be compiled after every change with the wkonvert wizard. The result of the compile action is a wkoonvert.mcx. This must be copied to the appropriate target directory for the settings to take effect.

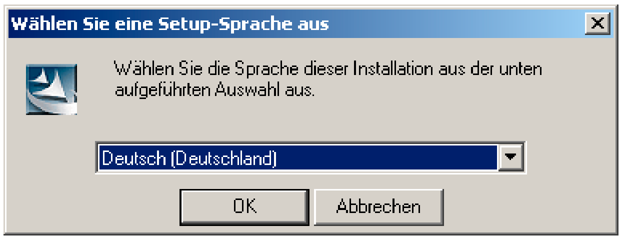
6.1.2.2 Installation of Wkonvert wizard

The Wkonvert wizard PC application is available when creating the required conversion rules for the Tool Ident Connection.

In order to be able to use the Wizard, the application must be installed on the configuring computer.

Installation procedure

To install the Wkonvert wizard, execute the setup.exe which is on the SINUMERIK CNC Software DVD.

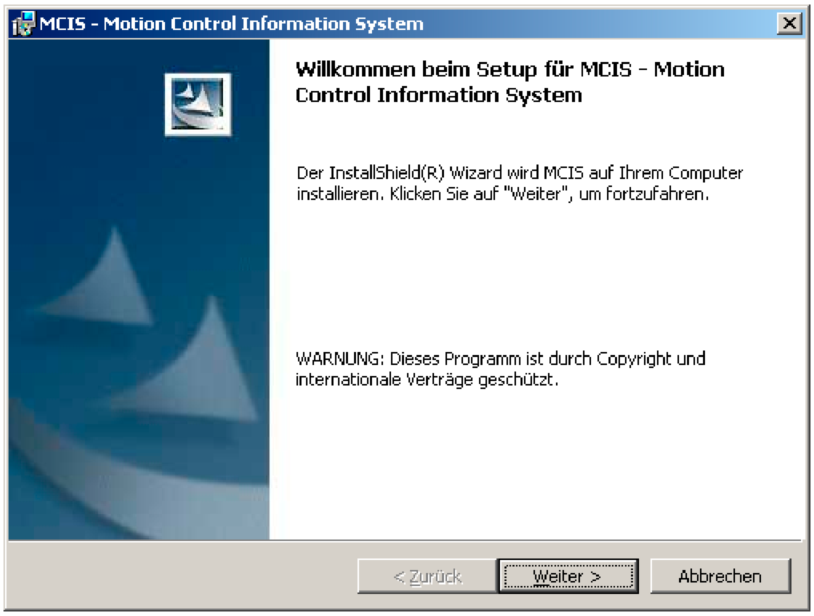

Wählen Sie eine Setup-Sprache aus

Wählen Sie die Sprache dieser Installation aus der unten aufgeführten Auswahl aus.

Deutsch (Deutschland)

Image 6-1 Language selection dialog

The individual installation steps follow after selecting the installation language.


MCIS - Motion Control Information System

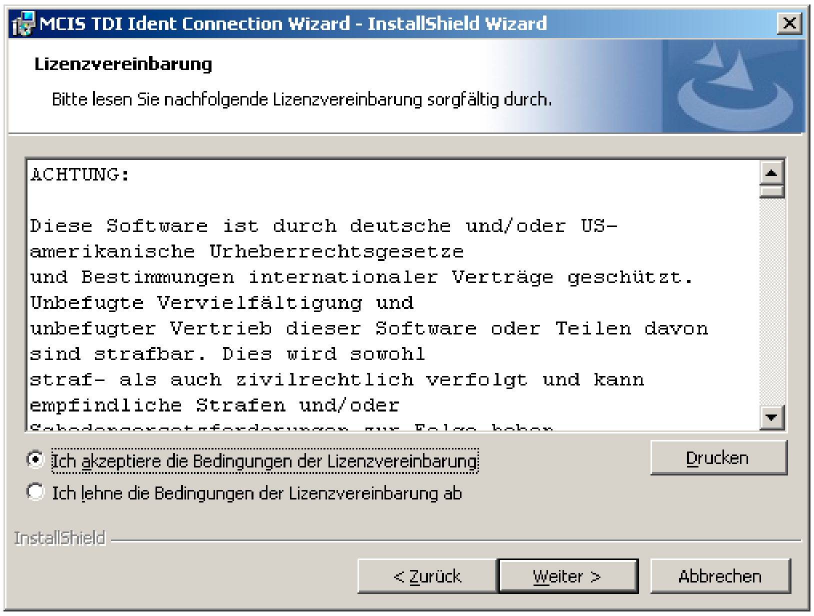
Willkommen beim Setup für MCIS - Motion Control Information System

Der InstallShield(R) Wizard wird MCIS auf Ihrem Computer installieren. Klicken Sie auf "Weiter", um fortzufahren.

WARNUNG: Dieses Programm ist durch Copyright und internationale Verträge geschützt.

Image 6-2 Preparing setup

After reading the license agreement, this must be accepted. If you are not in agreement with the license agreement, then setup is exited.


MCIS TDI Ident Connection Wizard - InstallShield Wizard

Lizenzvereinbarung

Bitte lesen Sie nachfolgende Lizenzvereinbarung sorgfältig durch.

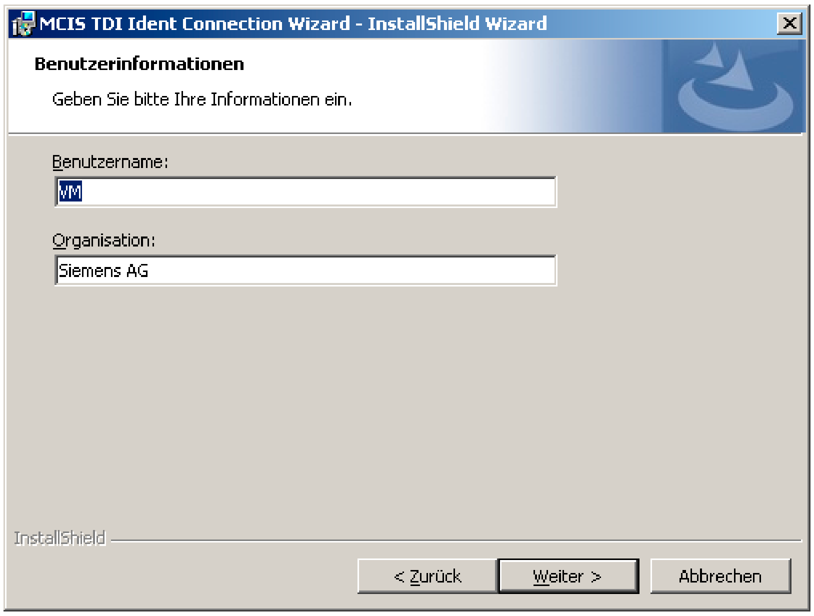
```
ACHTUNG:

Diese Software ist durch deutsche und/oder US-
amerikanische Urheberrechtsgesetze
und Bestimmungen internationaler Verträge geschützt.
Unbefugte Vervielfältigung und
unbefugter Vertrieb dieser Software oder Teilen davon
sind strafbar. Dies wird sowohl
straf- als auch zivilrechtlich verfolgt und kann
empfindliche Strafen und/oder
```

[✓ Ich akzeptiere die Bedingungen der Lizenzvereinbarung]
Ich lehne die Bedingungen der Lizenzvereinbarung ab

Image 6-3 License agreement dialog

After acknowledging the license agreement, you are prompted for a user name and organization.


MCIS TDI Ident Connection Wizard - InstallShield Wizard

Benutzerinformationen

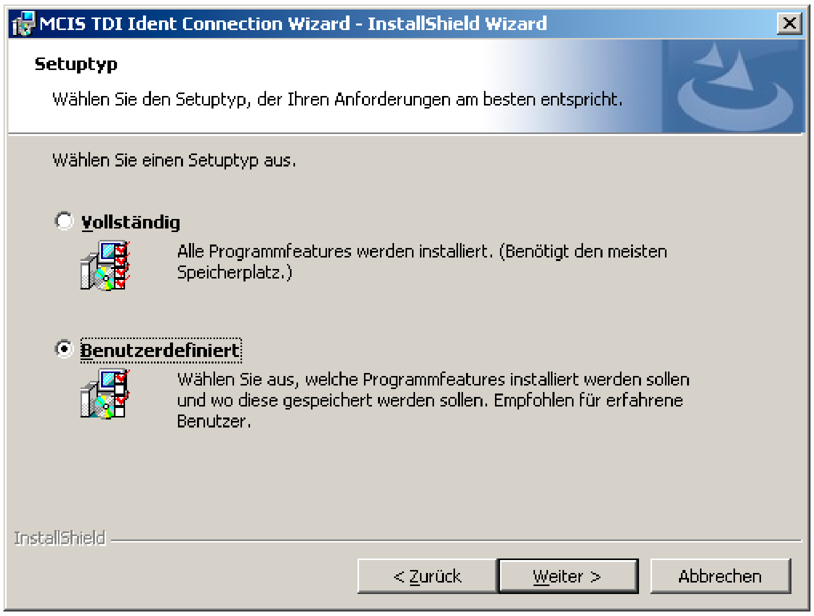
Geben Sie bitte Ihre Informationen ein.

Benutzername: VM

Organisation: Siemens AG

Image 6-4 User information dialog

You can then select either a complete or a user-defined installation.


MCIS TDI Ident Connection Wizard - InstallShield Wizard

Setuptyp

Wählen Sie den Setuptyp, der Ihren Anforderungen am besten entspricht.

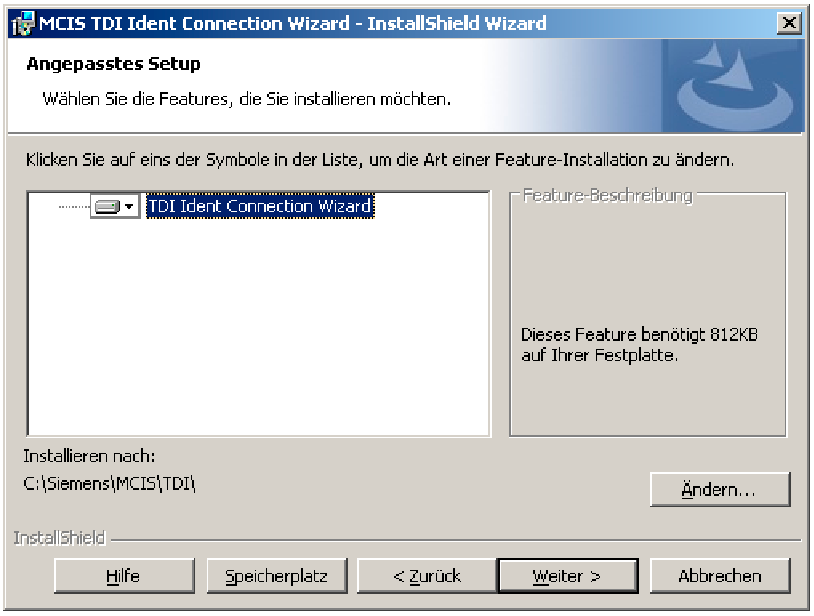
Wählen Sie einen Setuptyp aus.

○ **Vollständig**
Alle Programmfeatures werden installiert. (Benötigt den meisten Speicherplatz.)

● **Benutzerdefiniert**
Wählen Sie aus, welche Programmfeatures installiert werden sollen und wo diese gespeichert werden sollen. Empfohlen für erfahrene Benutzer.

Image 6-5 Setup type dialog

For a user-defined installation, you have the possibility of defining the target directory of the application.


MCIS TDI Ident Connection Wizard - InstallShield Wizard

Angepasstes Setup

Wählen Sie die Features, die Sie installieren möchten.

Klicken Sie auf eins der Symbole in der Liste, um die Art einer Feature-Installation zu ändern.

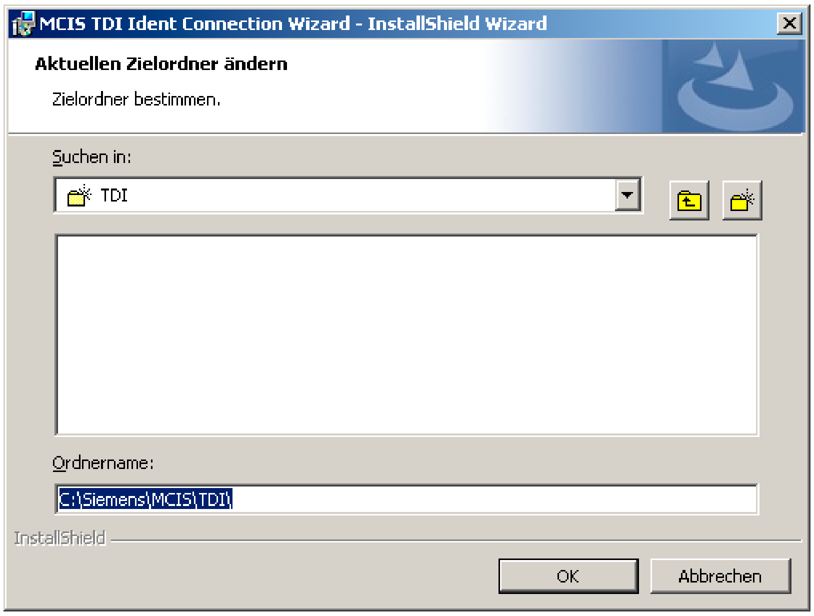
TDI Ident Connection Wizard

Feature-Beschreibung

Dieses Feature benötigt 812KB auf Ihrer Festplatte.

Installieren nach:
C:\Siemens\MCIS\TDI\

Image 6-6 Adapted setup dialog


MCIS TDI Ident Connection Wizard - InstallShield Wizard

Aktuellen Zielordner ändern

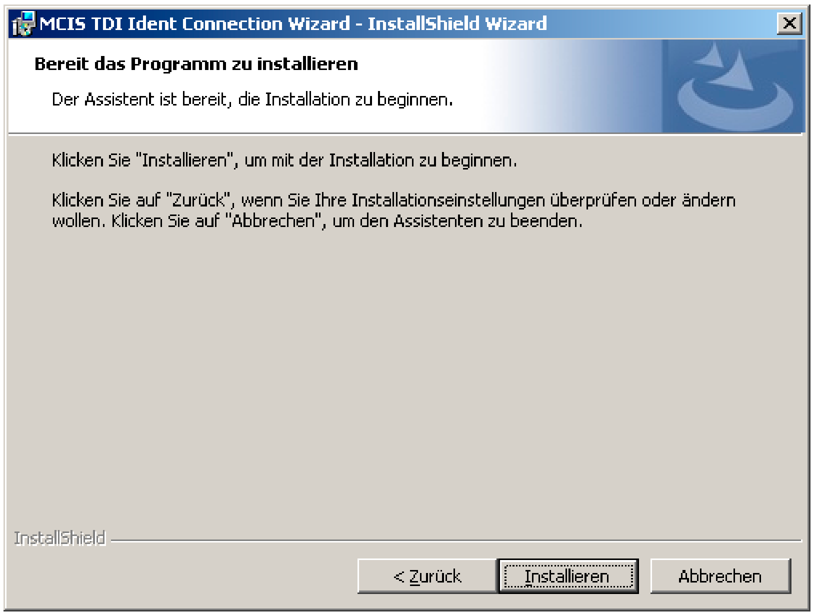
Zielordner bestimmen.

Suchen in: TDI

Ordnername: C:\Siemens\MCIS\TDI\

Image 6-7 Target directory selection dialog

If you have selected a complete installation, click "Next" or click "Next" in the component selection dialog to begin the installation.


MCIS TDI Ident Connection Wizard - InstallShield Wizard

Bereit das Programm zu installieren

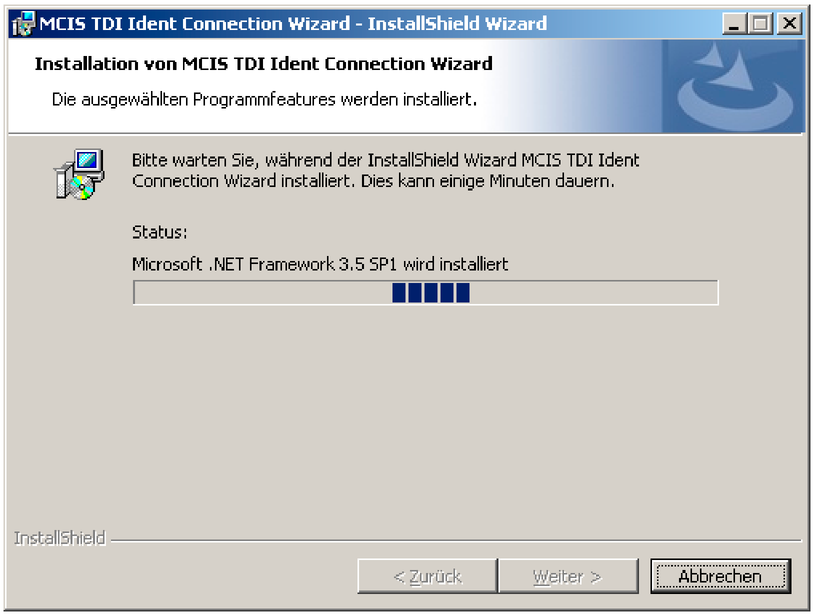
Der Assistent ist bereit, die Installation zu beginnen.

Klicken Sie "Installieren", um mit der Installation zu beginnen.

Klicken Sie auf "Zurück", wenn Sie Ihre Installationseinstellungen überprüfen oder ändern wollen. Klicken Sie auf "Abbrechen", um den Assistenten zu beenden.

Image 6-8 Installation dialog

Now click "Install" to start the installation.


MCIS TDI Ident Connection Wizard - InstallShield Wizard

Installation von MCIS TDI Ident Connection Wizard

Die ausgewählten Programmfeatures werden installiert.

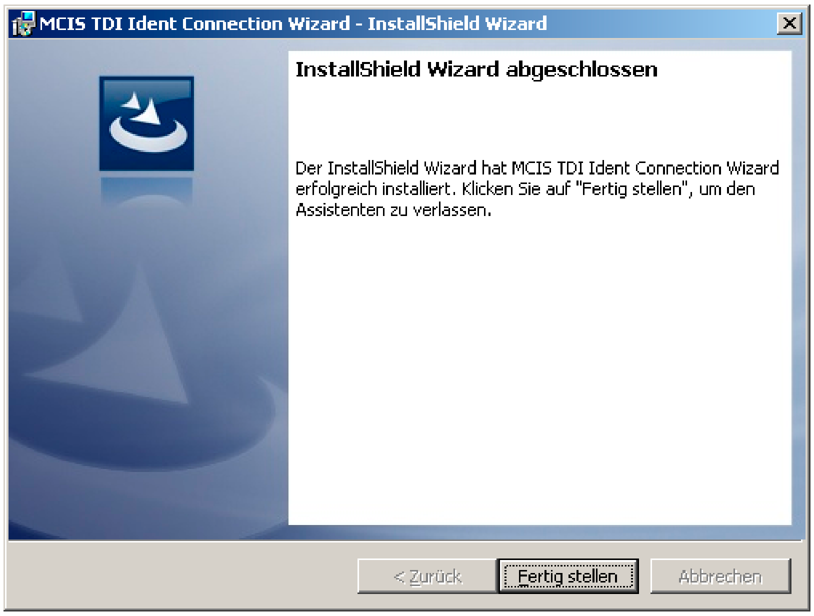
Bitte warten Sie, während der InstallShield Wizard MCIS TDI Ident Connection Wizard installiert. Dies kann einige Minuten dauern.

Status:

Microsoft .NET Framework 3.5 SP1 wird installiert

Image 6-9 Installation dialog

When required, setup installs the windows components ".NET Framework" and "Windows Installer 3.0".



InstallShield Wizard abgeschlossen

Der InstallShield Wizard hat MCIS TDI Ident Connection Wizard erfolgreich installiert. Klicken Sie auf "Fertig stellen", um den Assistenten zu verlassen.

Image 6-10 Completing the installation

When you have successfully run through the setup procedure, the Wkonvert wizard can be started via **Start > Programs > MCIS > TDI > TDI Ident Connection Wizard**.

6.2 Configuration

6.2.1 Configuration schematic

The following figure displays the configuration scheme of Tool Ident Connection:

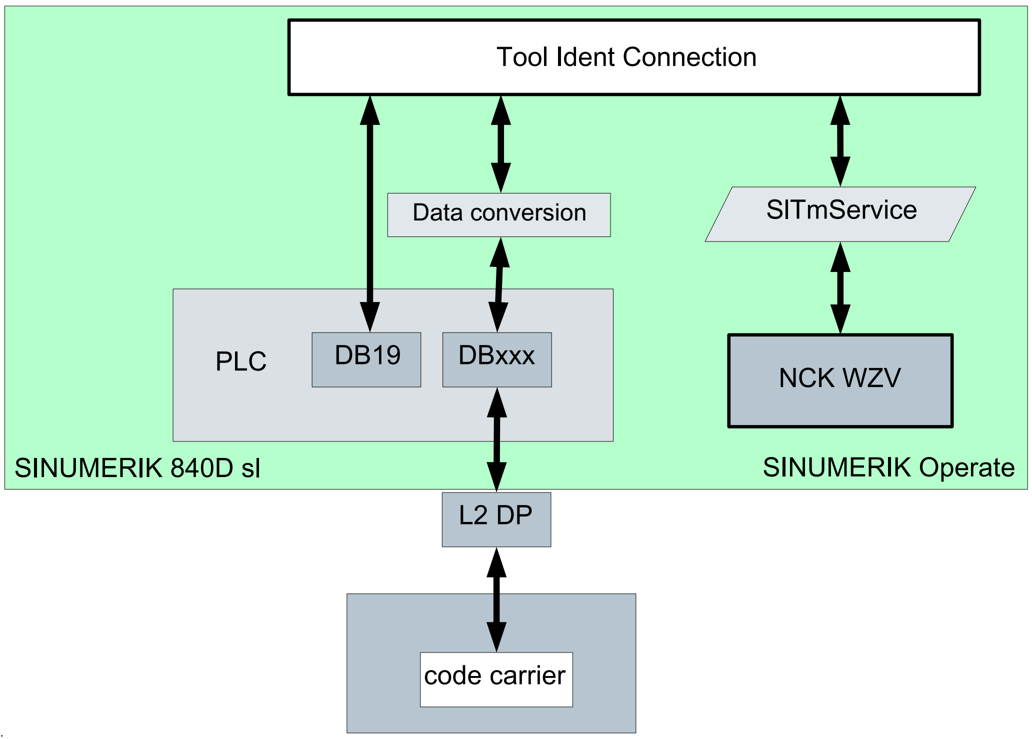


Image 6-11 Configuration principle of Tool Ident Connection

6.2.2 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 detailed information, please refer to:

References: SINUMERIK Operate Commissioning Manual (IM9), Section 12.2.16 "Configuring the code carrier connection"

6.2.3 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.SIMcisTdiIdentConnec
tionService,

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="UTP-8"?>
<!-- SIEMENS AG 2010 - GER I IA&DT WEST TSS 7C APC MT - ZN Köln - All rights reserved -->
<!-- Configuration-File for MCIS TDI Ident Connection sl -->
<!-- E-Mail @ kulik.michael@siemens.com -->
<!-- Do not manipulate this file -->
<Configuration>
      <Logging>
          <ErrorFile size="3" enable="0">/var/tmp/TDIIdentLog.log</ErrorFile>
          <TraceFile size="3" enable="0">/var/tmp/TDIIdentTrace.log</TraceFile>
<Logging>
      <GlobalParam>
          <InterfaceDB>19</InterfaceDB>
          <InfaceOffset>250</InfaceOffset>
          <DisableInface>false</DisableInface>
          <TimeOut>1000</TimeOut>
          <InputDB>19</InputDB>
          <InputOffset>140</InputOffset>
          <OutputDB>19</OutputDB>
          <OutputOffset>198</OutputOffset>
      </GlobalParam>
      <!-- Parametrierung des ParamTM-Interface -->
      <ParamTMInterface enable="1">
          <InterfaceDB>19</InterfaceDB>
          <InfaceOffset>256</InfaceOffset>
      <ParamTMInterface>
      <!-- Pfadangabe zur mcx-Datei und ToolSpec.xml -->
      <Logic>
      <!-- Aktivierung wenn TDI IdentConnection auf NCU -->
          <mcxPath>/card/addon/sinumerik/hmi/cfg/wkonvert.mcx</mcxPath>
          <specPath>/card/addon/sinumerik/hmi/cfg/ToolSpec.xml</specPath>
      <!-- Aktivierung wenn TDI IdentConnection auf PCU 50.5 -->
```

```
          <mcxPath>C://Program Files/Siemens/MotionControl/addon/sinumerik/
hmi/cfg/wkonvert.mcx</mcxPath>
         <specPath>C://Program Files/Siemens/MotionControl/addon/sinumerik/
hmi/cfg/wkonvert.mcx</specPath>
      <Logic>
      <MergeMode>
          <LoadPlaceNo>1</LoadPlaceNo>
          <UnloadPlaceNo>1</UnloadPlaceNo>
      </MergeMode>
      <!-- Parametrierung des Schreib-/Lesegerätes -->
      <!-- Type = 1-> PLC Type = 2 -> USB Type = 3 -> RJ45 -->
      <Device Type="1">
          <Units>1</Units>
          <Data Unit="1">
                <DataDB>300</DataDB>
                <Offset>0</Offset>
                <MaxData>1138</MaxData>
                <EOT>0*2F2F</EOT>
          </Data>
      </Device>
</Configuration>
```

Description

Note

Depending on whether Tool Ident Connection is activated on an NCU or on a PCU50.5, the required program lines must be activated in the <Logic> area.

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 sub-divided into various nodes which contain different entries. The sections and their entries are described in the following table.

Section	Parameter	Significance
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

Section	Parameter	Significance
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 occur, for example, if the operator does not acknowledge a tool being loaded within the parameterized time window.

The default loading and unloading location number of the loading magazine can be changed using the entries of "LoadPlaceNo" and "UnloadPlaceNo" using the "MergeMode" parameter.

The "Device" node has the "Type" attribute; the type of the 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 PROFIBUS

6.2.4 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 significance "1" for a PLC read/write device. The following parameters must be supplied with valid values below the node:

Parameter	Data type	Range of values
Units	int	1-4
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>2</Units>
        <Data Unit="1">
              <DataDB>300</DataDB>
              <Offset>0</Offset>
              <MaxData>140</MaxData>
              <EOT>0*2F2F</EOT>
        </Data>
        <Data Unit="2">
              <DataDB>300</DataDB>
              <Offset>141</Offset>
              <MaxData>40</MaxData>
              <EOT>0*2F2F</EOT>
       </Data>
</Device>
```

6.2.5 Data conversion

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.

The following figure displays the scheme for the configuration of the conversion rules for Tool Ident Connection:

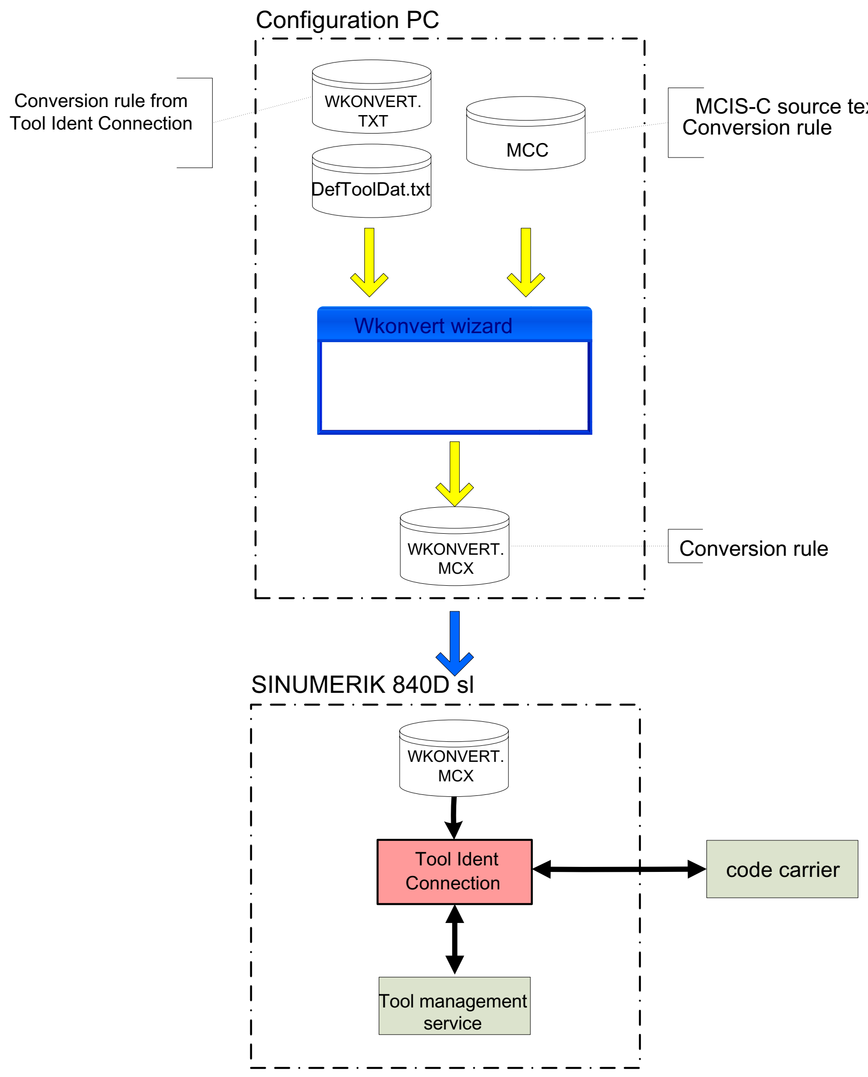


Image 6-12 Configuring the conversion rules

Generating conversion rules

As part of the commissioning of Tool Ident Connection, a conversion file must be created with the Wkonvert wizard application on a configuration PC. The binary file wkonvert.mcx is created. It contains the conversion rules 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 Section Structure of the description file wkonvert.txt (Page 398).

  The structure of the DefToolDat.txt file is described in Section Tool data initialization file DefToolDat.txt (Page 406).

- Conversion rule in the form of source text in the MCIS-C script language.
  Benefits: Extended conversion functionality.

  Note

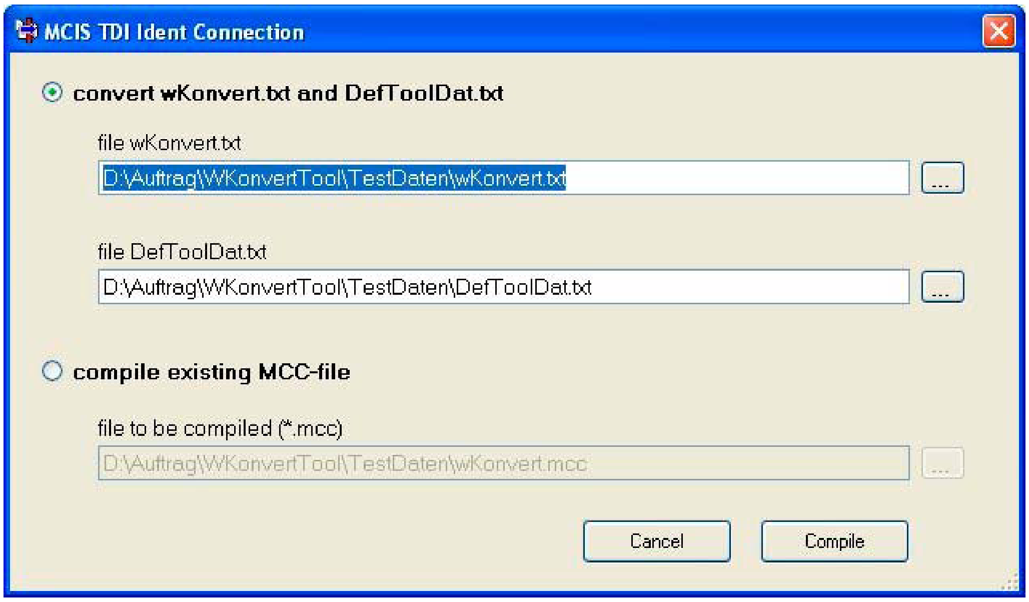
  If you want Siemens to create a customer-specific conversion rule for Tool Ident Connection, please contact your Siemens sales partner.

6.2.5.1 Using the Wkonvert wizard

After installation, the Wkonvert wizard can be started via

Start > Programs > MCIS > TDI > TDI Wkonvert Wizard.

After executing the application, the basic screen form for this application is displayed:


MCIS TDI Ident Connection

[✓ convert wKonvert.txt and DefToolDat.txt]

file wKonvert.txt: D:\Auftrag\WKonvertTool\TestDaten\wKonvert.txt

file DefToolDat.txt: D:\Auftrag\WKonvertTool\TestDaten\DefToolDat.txt

compile existing MCC-file

file to be compiled (*.mcc): D:\Auftrag\WKonvertTool\TestDaten\wKonvert.mcc

Image 6-13 Wkonvert wizard screen form

One of the following alternatives can be selected in the basic screen form of the application:

- "convert wKonvert.txt and DefToolDat.txt"
  Create conversion rule from Tool Ident Connection (files: wkonvert.txt and DefToolDat.txt).
- "compile existing MCC-file"
  Conversion rule is generated in the form of source text in the MCIS-C script language.

The target directory to output the result file wkonvert.mcx corresponds to the directory in which the wkonvert.txt file is located.

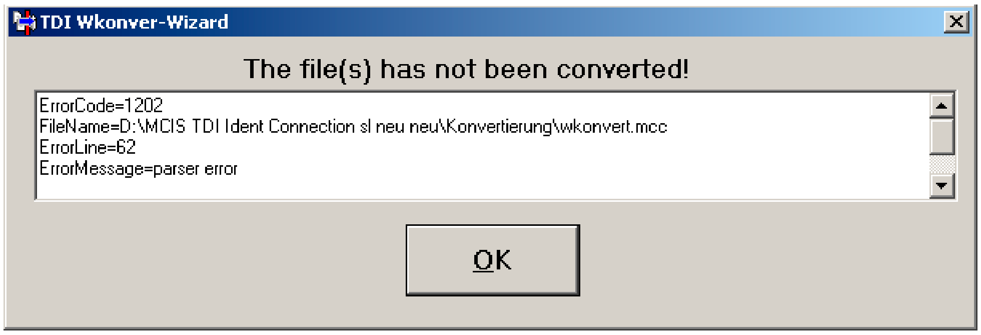
Note

After creating the conversion rule, the mcx file must be copied to the "/card/addon/sinumerik/appl/" directory. The operating software must be rebooted in order that the new conversion rule becomes effective.

6.2.5.2 Messages of the Wkonvert wizard

Messages of the Wkonvert wizard application that can occur while interpreting the wkonvert.txt and DefToolDat.txt or MCIS-C scripts, are described in this section.

All of the messages are displayed in the final dialog.



The file(s) has not been converted!

```
ErrorCode=1202
FileName=D:\MCIS TDI Ident Connection sl neu neu\Konvertierung\wkonvert.mcc
ErrorLine=62
ErrorMessage=parser error
```

Image 6-14 Final dialog

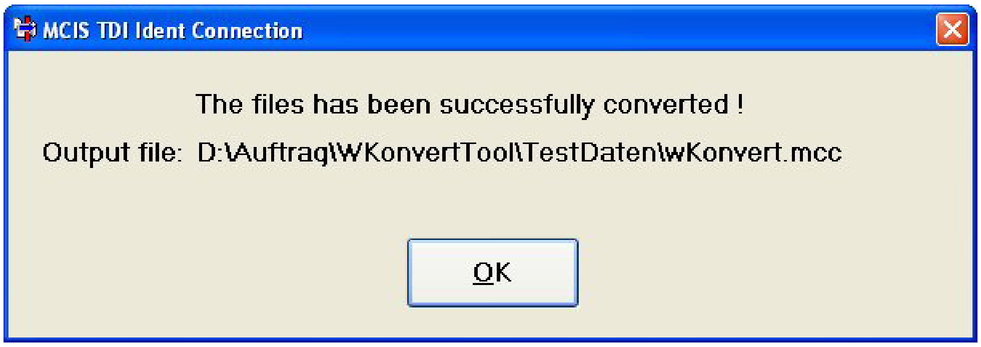
For error messages, the line as well as the source file, in which the error occurred, as well as additional information on the error profile are displayed in a text box that can be scrolled. To resolve the error, the error that has occurred must be removed in the source file.

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.

ErrorCode	ErrorMessage	Meaning
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.

After a successful compilation of the source file(s), the following message is output:



The files has been successfully converted !

Output file: D:\Auftrag\WKonvertTool\TestDaten\wKonvert.mcc

Image 6-15 MessageBox

6.2.6 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.

6.2.6.1 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 6-1 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 to the left in half locations	$TC_TP3
T5	Integer	Tool size to the right in half locations	$TC_TP4
T6	Integer	Tool size upwards in half locations	$TC_TP5
T7	Integer	Tool size downwards 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 6-2 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
C13	Double	Angle 2	$TC_DP11
		Wear tool length compensation	
C14	Double	Length 1	$TC_DP12
C15	Double	Length 2	$TC_DP13

Dialog variable	Data type	Designation	Assignment $TC...
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	Pre-warning limit for tool life in minutes	$TC_MOP1
C31 *)	Integer	Unit quantity	$TC_MOP4
C32 *)	Integer	Prewarning limit unit quantity	$TC_MOP3
C33	Double	Setpoint tool life in minutes	$TC_MOP11
C34 *)	Integer	Setpoint unit quantity	$TC_MOP13
C35	Double	Pre-warning 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	Pre-warning limit for tool life in minutes	$TC_MOP1
C40 *)	Double	Unit quantity	$TC_MOP4
C41 *)	Double	Prewarning limit unit quantity	$TC_MOP3
C42 *)	Double	Setpoint unit quantity	$TC_MOP13
A1 - A10	Double	User tool data	$TC_TPCx
U1 - U10	Double	User cutting edge data	$TC_DPCx
S1 - S10	Double	User monitoring data	$TC_MPOCx

*) 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 to +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 are supported (see `Item`/`BItem`).

Table 6-3 Code carrier data formats

Data format	Explanation	Range of values
ASCII	ASCII character set	-
INT	16-bit integer (Intel format)	-32768 to +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	$\pm1.5\times10^{-45}$ to $\pm 3.4\times10^{38}$
DOUBLE	32-bit floating-point number (Intel format): Accuracy, 15-16 digits	$\pm5.0\times10^{-324}$ to $\pm1.7\times10^{308}$
FPX2	Bit integer (SINUMERIK): Least significant byte at the higher order address (inverse to the Intel format)	-32768 to +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.

6.2.6.2 Examples for the description files wkonvert.txt

After installation, the name of the file is "wkonvert.txt".

Table 6-4 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	’ Qty. 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				
BItem12	4	BCD	C14	' Length 1

Code carrier variable	Length (bytes)	Data format	Dialog variable	Comment
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 6-5 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	’ Qty. 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
’ Cutting edge data Block1 * Item6				

Code carrier variable	Length (bytes)	Data format	Dialog variable	Comment
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				

6.2.6.3 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 NCK 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
```

6.2.7 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. 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. >, <).

6.2.7.1 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], whereby 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 help 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.

6.2.7.2 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 "Programming (Page 223)". 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_TPC10	OEM user data 10	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
TC_DP12 [EdgeNo]	Wear - length 1	double
TC_DP13 [EdgeNo]	Wear - length 2	double
TC_DP14 [EdgeNo]	Wear - length 3	double

Identifier	Designation	MCIC-C data type
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	long
Cutting edge supplementary information		
TC_DPC1 [EdgeNo]	OEM cutting edge user data 1	double
...	...	...
TC_DPC10 [Edge-No]	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 [Edge-No]	Reference tool life	double
TC_MOP13 [Edge-No]	Unit quantity reference	long
TC_MOP5 [EdgeNo]	Wear prewarning limit	double
TC_MOP6 [EdgeNo]	Wear actual value	double
TC_MOP15 [Edge-No]	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		

Identifier	Designation	MCIC-C data type
EDGE_NUM	Number of cutting edges (read only)	long
EDGE_NO [Edge-Idx]	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.	long

6.2.7.3 Auxiliary functions for numerical coding

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 "precison" 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.

PlaceTypeToString () / PlaceTypeFromString ()

The location type of a tool (TC_TP7) is managed in SINUMERIK Operate exclusively as a numerical value. In HMI Advanced, a string value is managed for the location type. The assignment between the numerical value and the string value is realized in the database WZACCESS.MDB.

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.

Table 6-6 Example, ToolSpec.xml

```
<!--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>
```

It is possible to access ToolSpec.xml within the conversion rule using the auxiliary functions `PlaceTypeToString()` and `PlaceTypeFromString()` .

```
 PlaceTypeToString ( NumType )
 PlaceTypeFromString ( StrType )
```

Example:

```
PlaceTypeToString ( 1 );            □ "normal"
PlaceTypeToString ( 2 );            □ "large"
PlaceTypeToString ( 3 );            □ "small"


PlaceTypeFromString ("normal" );    □ 1
PlaceTypeFromString ("large" );     □ 2
PlaceTypeFromString ("small" );     □ 3
```

6.2.7.4 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 HMI-Advanced. 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
```

6.2.7.5 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	

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	Range of values
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	up to 64KB	Dynamic up to 64KB

Operators and special characters

The MCIS-C script language supports the following operators and special characters:

!	*	-=	<=	[	}
!=	*=	/	=	]	~
%	+	/=	==	^	
%=	++	:	>	^=	
&	+=	;	>=	{	
&&	,	<	>>	|	
&=	-	<<	>>=	|=	
(	--	<<=	?	||	
)	.				

Special features of the "string" data type

Contrary to the programming language C, MCIS-C has a dynamic string-data type. The data type is comparable with the string data type of the Java or C# programming languages. A "string" type variable can include a character string of up to 64KB. Strings can be linked using the ± operator. Assigning binary data types to a string variable means that the binary value is converted into a string.

Example:

```
string s;

s = "Hello"; // s= "Hello"
s = s + " "; // s= "Hello "
s += "World!"; // s= "Hello World!"
```

6.2.8 Examples

Overview

The following source code examples are based on the MCIS-C script language to create conversion examples for Tool Ident Connection. These examples are included on the data carrier of the Wkonvert wizard.

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
    }
}
```

6.3 PLC interface in the DB19

6.3.1 Structure

Tool Ident Connection has a parameterizable PLC interface with which defined services/ functions can be triggered or executed. Data for the services/functions is saved in a defined data area of the PLC interface; the area for the return value of the services/functions is also provided there.

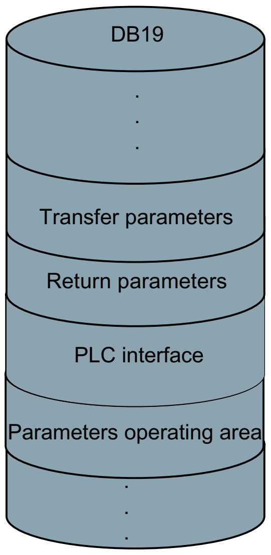


Image 6-16 Structure of the PLC interface

6.3.2 Function call of the PLC interface

The interface occupies six bytes in the PLC, has a fixed structure and the following assignment (the table refers to the default parameterization of the interface in DB19 with offset 250):

Table 6-7 Structure of the PLC interface

PLC data	Identifier	Type	Range of values	Meaning
DB19.DBX250.0	Request	BOOL	True,False	Requesting the task
DB19.DBB251	K_Code	BYTE	0…127	Command code
DB19.DBX252.0	Done	BOOL	True,False	Task successfully completed
DB19.DBX252.1	Error	BOOL	True,False	Task ended with error
DB19.DBX252.2	Active	BOOL	True,False	Task is being processed
DB19.DBX252.7	Ident_Ready	BOOL	True,False	Status for power up
DB19.DBB253	K_CodeError	BYTE	0 ... 127	K-Code in case of an error
DB19.DBW254	Error_Code	INT	-3276 ... 32767	Error code

Note

The PLC interface can only function and accept tasks after the status bit Ident_Ready has been set to "1".

If the request bit has already been set, it is reset and error code 0xFFFF is written into the Error_Code word in the PLC.

6.3.3 Assignment of the PLC for transfer parameters

Various transfer parameters are required, depending on the command code, to execute certain functions. The required transfer parameters are also stored in the PLC; this data area occupies 56 bytes and has the following fixed structure (the table refers to the default parameterization of the interface in DB19 with offset 140):

Table 6-8 Assignment of the PLC for transfer parameters

PLC data	Identifier	Type	Range of values	Meaning
DB19.DBW140	Subtype	INT	-1…32000	Subtype of the tool
DB19.DBW142	Duplo	INT	-1…32000	Duplo number of the tool
DB19.DBB144	Ident	STRING	32 characters	Tool identifier
DB19.DBW178	T number	INT	-1…32000	T number of the tool
DB19.DBW180	Magazine location	INT	-1…32000	Magazine location number of the tool
DB19.DBW182	Magazine location type	INT	-1…32000	Tool size
DB19.DBB184	ToolSize_Upper	BYTE	1…127	Tool size upwards in half locations
DB19.DBB185	ToolSize_Down	BYTE	1…127	Tool size downwards in half locations
DB19.DBB186	ToolSize_Left	BYTE	1…127	Tool size to the left in half locations
DB19.DBB187	ToolSize_Right	BYTE	1…127	Tool size to the right in half locations
DB19.DBB188	Channel	BYTE	1…127	Channel number of the tool
DB19.DBB189	Loading Place	BYTE	1…127	Number of the loading/unloading location
DB19.DBW190	Magazines	INT	0…32000	Magazine number
DB19.DBB192	Unit	BYTE	1…127	Code carrier number
DB19.DBB193	NCU_Index	BYTE	1…127	Name of the NCU
DB19.DBW194	FileIndex	INT	0…32000	File index (Import/Export)
DB19.DBX196.0	Active Enable	BOOL	True, False	Hide status bit "Tool active"
DB19.DBX196.1	Override	BOOL	True, False	Overwrite file
DB19.DBX196.2	LoadTool	BOOL	True, False	Load tool after being created
DB19.DBX196.3	DeleteTool	BOOL	True, False	Delete tool after unloading

Note

The "NCU_Index" parameter is currently not evaluated by Tool Ident Connection.

6.3.4 Assignment of the PLC for return values

To execute certain functions, various return parameters are supplied, depending on the command code. They are also stored in the PLC; this data area occupies 50 bytes and has the following fixed structure (the table refers to the default parameterization of the interface in DB19 with offset 198):

Table 6-9 Assignment of the PLC for return parameters

PLC data	Identifier	Type	Range of values	Meaning
DBW 198	Subtype	INT	-1…32000	Subtype of the tool
DBW 200	Duplo	INT	-1…32000	Duplo number of the tool
DBB 202	Ident	STRING	32 characters	Tool identifier
DBW 236	T number	INT	-1…32000	T number of the tool
DBW 238	ToolState	INT	-1…32000	Tool status
DBW 240	Magazine location type	INT	-1…32000	Tool size
DBB 242	ToolSize_Upper	BYTE	1…127	Tool size upwards in half locations
DBB 243	ToolSize_Down	BYTE	1…127	Tool size downwards in half locations
DBB 244	ToolSize_Left	BYTE	1…127	Tool size to the left in half locations
DBB 245	ToolSize_Right	BYTE	1…127	Tool size to the right in half locations
DBW 246	Magazines	INT	0…32000	Magazine number of the tool
DBW 248	Magazine Place	INT	0…32000	Magazine location number of the tool

6.3.5 Executing functions

6.3.5.1 General sequence when executing functions

The following chronological sequence is defined (positive execution) for executing functions for the PLC interface:

PLC interface user:

1. Transfer parameters required are written in the PLC (DB19 – offset 140)
2. Command code is defined (DB19.DBB251)
3. Request bit is set (DB19.DBX.250.0)

Tool Ident Connection:

1. Reads out the command code (DB19.DBB251)
2. Checks whether the value read-out is valid
3. Sets the active bit (DB19.DBB251.2)
4. Request is executed
5. Writes the value "0" to the error code word (DB19.DBW253)
6. Sets the done bit (DB19.DBX251.0)

PLC interface user:

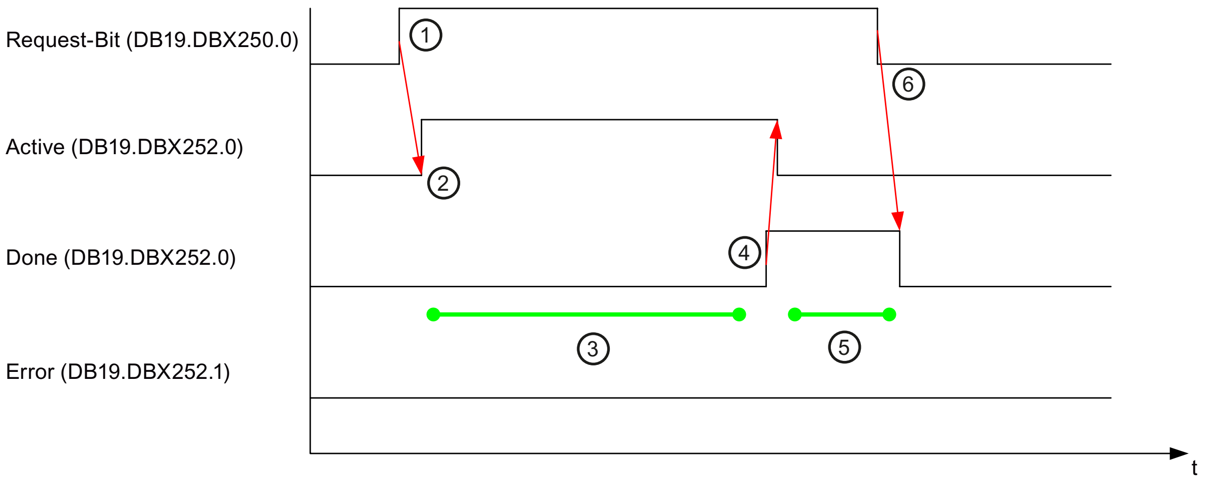
1. Evaluates done bit or error bit and possibly the error code
2. Resets the request bit (DB19.DBX.250.0)

Tool Ident Connection:

1. Resets the done bit and error bit back to "0" (DB19.DBX251.0, DB19.DBX251.1)
2. Deletes the error code word (DB19.DBW253)

6.3.5.2 Pulse diagram for positive execution of a function

The following pulse diagram clearly shows the chronological sequence when executing a function via the PLC interface:



① **Function initiated by the user**
- Before initiating, the user writes the required transfer parameters and the K code into the PLC
- When triggered, Tool Ident Connection evaluates the K code

② **Feedback from Tool Ident Connection**
- The active bit is set if the K code is valid

③ **Processing period**
- Tool Ident Connection reads the required transfer parameters and performs the desired operation / PI service
- Depending on the function, return parameters are written to the PLC

④ **Feedback, processing**
- After successful execution of the function, Tool Ident Connection writes "0" to the error code word and sets the done bit
- Resets the active bit

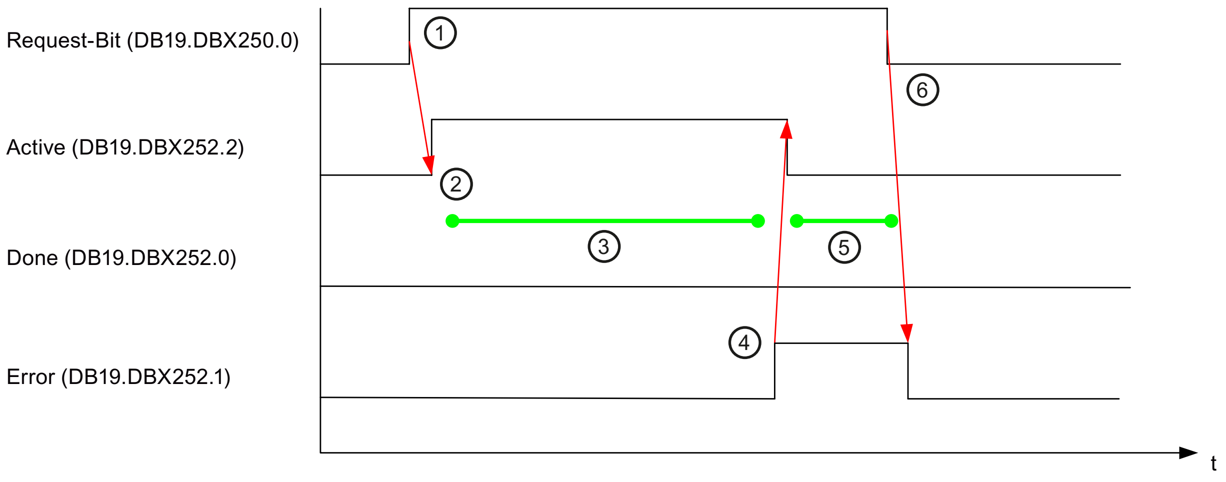
⑤ **User responds to the falling edge of the active bit**
- The user evaluates the done bit or error bit and possibly the error code word, reads possible return parameters and resets the request bit

⑥ **Tool Ident Connection reacts to the falling edge of the request bit**
- Sets the done bit and error bit to "0"
- Error code word is set to "0"

Image 6-17 Pulse diagram for positive execution

6.3.5.3 Pulse diagram when negative execution of a function

The following pulse diagram clearly shows the chronological sequence when executing a function via the PLC interface with negative output:



① **Function initiated by the user**
- Before initiating, the user writes the required transfer parameters and the K code into the PLC
- When triggered, Tool Ident Connection evaluates the K code

② **Feedback from Tool Ident Connection**
- The active bit is set if the K code is valid

③ **Processing period**
- Tool Ident Connection reads the required transfer parameters and performs the desired operation / PI service
- Depending on the function, return parameters are written to the PLC

④ **Feedback, processing**
- After unsuccessful execution of the function, Tool Ident Connection writes the returned error code to the error code word and sets the error bit
- Resets the active bit

⑤ **User responds to the falling edge of the active bit**
- The user evaluates the error bit and error code word and resets the request bit

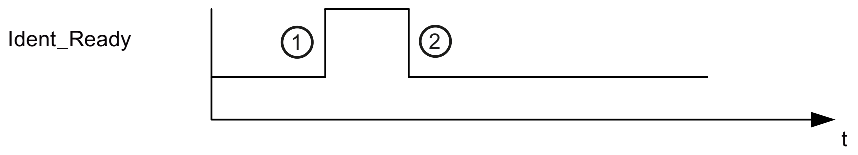
⑥ **Tool Ident Connection reacts to the falling edge of the request bit**
- Sets the done bit and error bit to "0"
- Error code word is set to "0"

Image 6-18 Pulse diagram for negative execution

6.3.5.4 Pulse diagram "Ident_Ready" signal

The "Ident_Ready" signal serves as the ready signal of Tool Ident Connection. The PLC user sets the parameter "Ident_Ready" (DB19.DBX252.7) in the PLC interface to the value "1".

Tool Ident Connection reacts to the rising edge of the "Ident_Ready" signal and resets the significance of the parameter to the value "0".



① **Function initiated by the user**

- The PLC user sets the PLC data "Ident_Ready" to a value of "1"

② **Feedback from Tool Ident Connection**

- Tool Ident Connection reacts to the rising edge and resets the significance of the PLC data to "0".

Image 6-19 Pulse diagram of the status signal Ident_Ready

6.3.6 PLC interface for the Parameter operating area

6.3.6.1 Assignment of the PLC interface for the Parameter area

ParamTM PLC interface

The "ParamTM" PLC interface is used in conjunction with the following functions:

- New tool from code carrier
- Unload tool/delete from code carrier

Configuration

In addition, this interface must be configured in the "tdidentcfg.xml" file in the "ParamTMInterface" section.

Table 6-10 Structure of the "ParamTM-Interface" PLC interface

PLC address	Identifier	Type	Range of values	Significance
DB19.DBX256.0	Request	BOOL	True, False	Requests the task
DB19.DBB257	Reserve	BYTE		
DB19.DBX258.0	Done	BOOL	True, False	Task successfully completed
DB19.DBX258.1	Error	BOOL	True, False	Task ended with error
DB19.DBX258.2	Active	BOOL	True, False	Task is being processed
DB19.DBX258.3	Reserve	BOOL		
DB19.DBX258.4	Reserve	BOOL		
DB19.DBX258.5	Reserve	BOOL		
DB19.DBX258.6	Reserve	BOOL		
DB19.DBX252.7	Reserve	BOOL		
DB19.DBX259.0	Read	BOOL	True, False	Read tool data from CT
DB19.DBX259.1	Write	BOOL	True, False	Write tool data to CT

PLC address	Identifier	Type	Range of values	Significance
DB19.DBX259.2	Cancel	BOOL	True, False	Operator has canceled read/write operation
DB19.DBX259.3	Write	BOOL	True, False	Selective writing of CT data
DB19.DBX259.4	Reserve	BOOL		
DB19.DBX259.5	Reserve	BOOL		
DB19.DBX259.6	Reserve	BOOL		
DB19.DBX259.7	Reserve	BOOL		
DB19.DBW260	Error_Code	INT	32768 … 32767	PLC error code
Transfer parameters				
DB19.DBW262	PMagazine	INT	32768 … 32767	Magazine number
DB19.DBW264	PPlace	INT	32768 … 32767	Magazine location number
DB19.DBW266	PTnumber	INT	32768 … 32767	T number

6.3.6.2 Load the tool

"New tool from code carrier" function

The "New tool from code carrier" function is executed on the SINUMERIK Operate in the "Parameter" area. As a consequence, the "Read" function is executed in the PLC interface.

The following pulse diagrams are shown in the following:

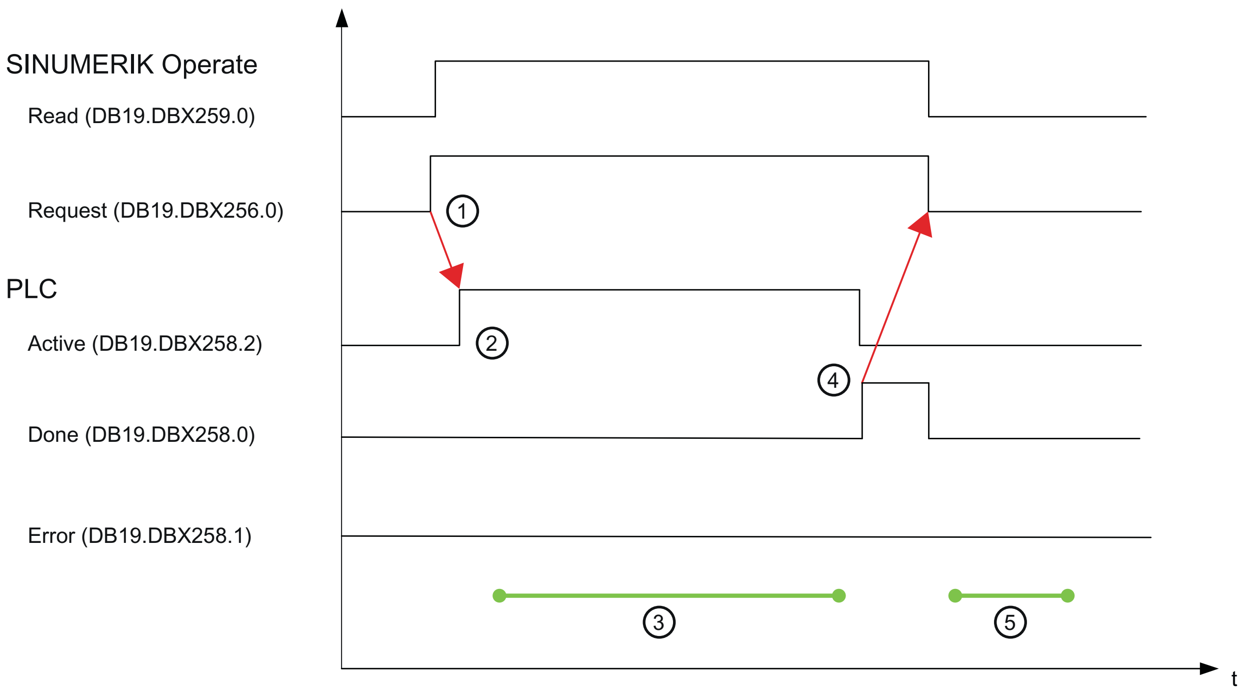
Sequence (Page 433)

Sequence when canceled by the PLC (Page 434)

6.3.6.3 Sequence

Sequence

The following pulse diagram clearly shows an error-free sequence of the "New tool from code carrier" function.



① **Function initiated by the user**
With the "New tool from code carrier" function, in the PLC interface, the signals are set to Read = 1" and Request = 1.

② **Feedback from the PLC**
The PLC then sets the signal to Active = 1".

③ **Machining time**
The PLC reads the tool data from the code carrier, and saves it in the configured data block.

④ **Feedback, machining time**
- If all of the tool data is in the data block, the PLC sets the following signals "Active = 0" and "Done = 1".
- SINUMERIK Operate then resets the "Read" and "Request" signals.
- The PLC also resets the "Done" signal.

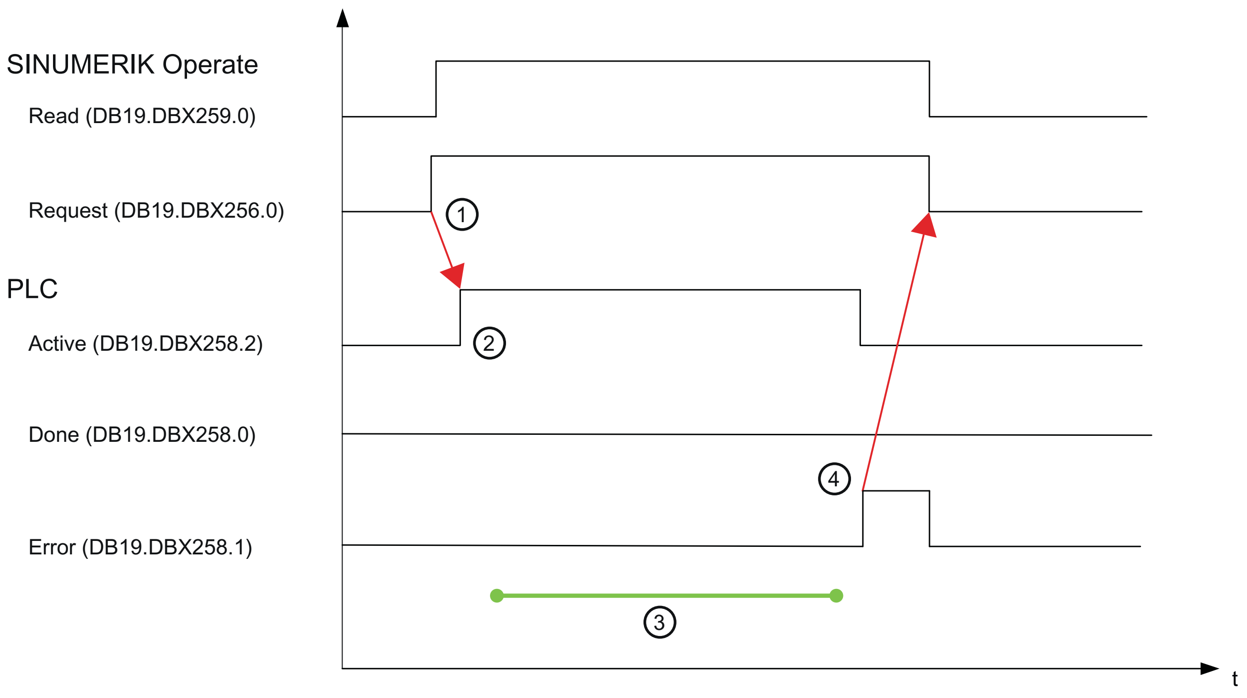
⑤ **Machining time**
The HMI reads the tool data from the data block and the tool is loaded.

Image 6-20 Pulse diagram for positive execution

6.3.6.4 Sequence when canceled by the PLC

Sequence

The following pulse diagram shows an incorrect sequence of the "New tool from code carrier" function.



① **Function initiated by the user**
In the PLC interface, the signals "Read = 1" and "Request = 1" are set.

② **Feedback from the PLC**
The PLC sets the signal "Active = 1".

③ **Machining time**
The PLC reads the tool data from the code carrier, and saves it in the configured data block.

④ **Feedback, machining time**
If the code carrier was not read, then the following happens in the PLC interface:

- The error cause is written into the "Error_Code" element.
- The PLC sets the signals "Active = 0" and "Error = 1".
- SINUMERIK Operate then resets the "Read" and "Request" signals.
- The PLC also resets the "Error" and the "Error_Code" signal.

Image 6-21 Pulse diagram when canceled by the PLC

6.3.6.5 Unloading a tool

"Unload tool / delete on code carrier" function

The "Unload tool / delete on code carrier" function is executed on the SINUMERIK Operate in the "Parameter" area. As a consequence, the "Read" function is executed in the PLC interface, and then the "Write" function is executed.

This sequence ensures that the content on the code carrier, which is not written to with the current tool data as result of the conversion rule, is retained.

Please refer to the following sections for the sequence of the "Read" function.

The sequence of the "Write" function corresponds to the "Read" sequence, with the difference that the "DB19.DBX259.1" signal is used.

Sequence (Page 433)

Sequence when canceled by the PLC (Page 434)

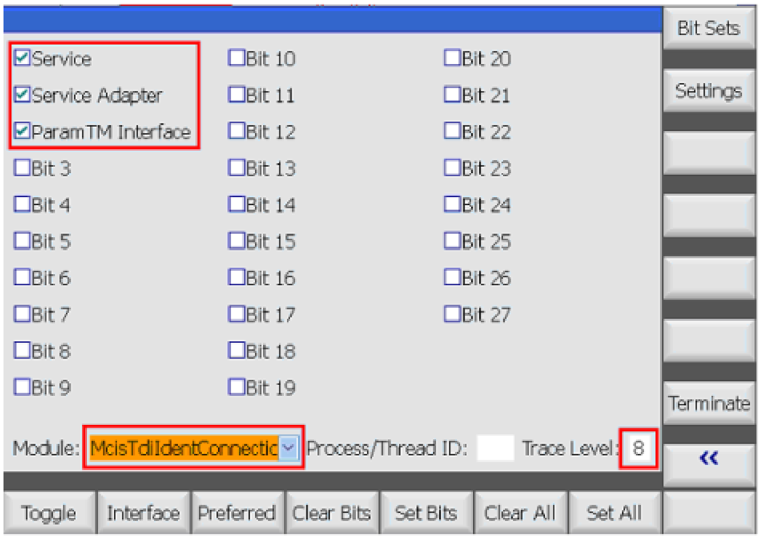
6.3.6.6 Diagnostics

Settings

The following settings must be made in order to set the diagnostics of the described sequences.

Procedure

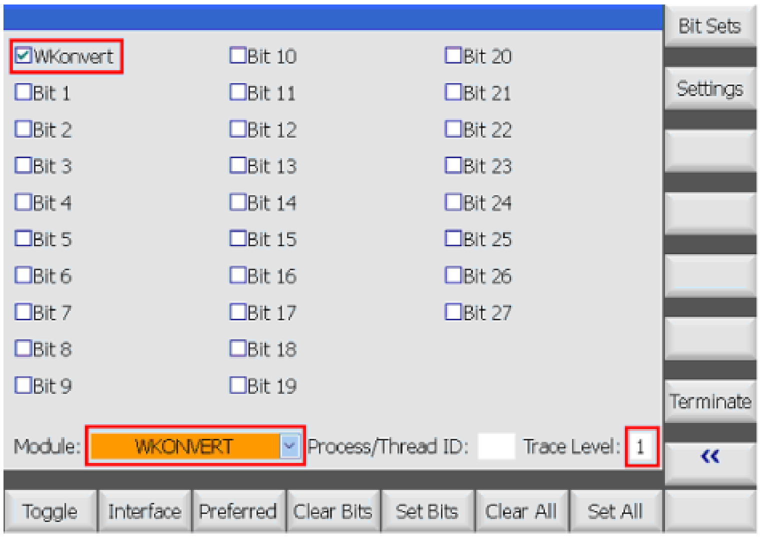
1. Select the "Startup" operating area.
2. Press the "SINUMERIK Operate", "Diagnostics", and "Extended" softkeys.
3. Make the following settings:



☑ Service	☐ Bit 10	☐ Bit 20
☑ Service Adapter	☐ Bit 11	☐ Bit 21
☑ ParamTM Interface	☐ Bit 12	☐ Bit 22
☐ Bit 3	☐ Bit 13	☐ Bit 23
☐ Bit 4	☐ Bit 14	☐ Bit 24
☐ Bit 5	☐ Bit 15	☐ Bit 25
☐ Bit 6	☐ Bit 16	☐ Bit 26
☐ Bit 7	☐ Bit 17	☐ Bit 27
☐ Bit 8	☐ Bit 18	
☐ Bit 9	☐ Bit 19	

Module: McisTdiIdentConnectic Process/Thread ID: Trace Level: 8

Image 6-22 Activate trace



☑ WKonvert	☐ Bit 10	☐ Bit 20
☐ Bit 1	☐ Bit 11	☐ Bit 21
☐ Bit 2	☐ Bit 12	☐ Bit 22
☐ Bit 3	☐ Bit 13	☐ Bit 23
☐ Bit 4	☐ Bit 14	☐ Bit 24
☐ Bit 5	☐ Bit 15	☐ Bit 25
☐ Bit 6	☐ Bit 16	☐ Bit 26
☐ Bit 7	☐ Bit 17	☐ Bit 27
☐ Bit 8	☐ Bit 18	
☐ Bit 9	☐ Bit 19	

Module: WKONVERT Process/Thread ID: Trace Level: 1

Image 6-23 Activate trace

Note

Diagnostics are only available when the manufacturer's password has been entered.

Within a conversion rule, using the print() command, the user can generate an output in the diagnostics, for example: print("Radius", Tool.TC_DP6[EdgeNo], " EdgeNo:=", EdgeNo, "\r\n");

6.3.6.7 Restrictions

The following restrictions apply:

- The "Active (DB19.DBX258.2)" signal is optional. SINUMERIK Operate does not evaluate the signal.
- The "Cancel (DB19.DBX259.2)" signal is not used. A function cannot be canceled by SINUMERIK Operate.
- SINUMERIK Operate does not use the following transfer parameters:
  - PMagazine
  - PPlace
  - PTnumber
- Several units can be configured in the "tdiidentcfg.xml" file. However, in the data block only the "Unit = 1" is used.

6.3.6.8 Error messages

Error messages that are transferred in the PLC are displayed on the user interface in the form of an error code, for example, Error_Code (DB19.DBW260)
Further, the following error codes can occur:

Error code	Description
-602	The user initiates a new function, while an active function has still not been completed.
-603	The PLC interface is not ready. The "Done" signal is present, and prevents a new function from being initiated.

6.3.7 PLC data types

The used PLC data types and how they are interpreted by "Tool Ident Connection" is described in the following table.

Table 6-11 PLC data used

Data type	Range of values	Note
BOOL	TRUE, FALSE	Is used if individual bits should be set in the PLC. (TRUE: Bit = 1, FALSE: Bit = 0)
BYTE	-128...127	PLC byte

Data type	Range of values	Note
INT	-32768...32767	PLC word, whose first byte contains the high byte and the second byte contains the low byte. Example: PLC word 140 comprises the two PLC bytes 140 and 141 together and should be allocated the value 120. Byte 140: 0 byte 141: 120
String	xx character	Strings occupy xx+2 bytes in the PLC as the first two bytes contain the maximum length and the actual length of the string. The actual string therefore does not start until the 3rd byte. The individual characters are saved in the corresponding bytes as ASCII code. PLC strings do not have to end with a zero because the actual length is always specified in the 2nd byte of the string. Example: "Hello" should be saved in a string, which starts at PLC byte 144 and may only have a maximum length of 32 characters. This results in the following assignment of the bytes: Byte 144 = 32 max. length of the string Byte 145 = 5 actual length of the string Byte 146 = 72 H Byte 147 = 97 a Byte 148 = 108 l Byte 149 = 108 l Byte 150 = 111 o

6.4 Command codes

6.4.1 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
1	Unloading from the code carrier
4	Unloading to the tool cabinet
5	Loading from the production control system
7	Loading from the tool list
8	Unloading a tool from the tool list
9	Empty location search
10	Delete tool in the NCK
14	Update code carrier
15	Read data from code carrier

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.

6.4.2 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 NCK (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

Parameter	Description
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.

6.4.3 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 NCK. It is also loaded if "LoadTool" has been set. The number of the loading location (place) 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 6-12 PLC parameters for "loading from code carrier"

PLC data		Identifier	Type	Range of values	Meaning
Transfer parameters					
DBW	180	Magazine location	INT	-1…32000	Magazine location number of the tool
DBB	188	Channel	BYTE	1…127	TO area

PLC data		Identifier	Type	Range of values	Meaning
DBB	189	Loading Place	BYTE	1…127	Number of the loading/unloading location
DBW	190	Magazines	INT	0…32000	Magazine number
DBB	192	Unit	BYTE	1…127	Code carrier number
DBX	196.2	LoadTool	BOOL	True, False	Load tool after being created
Return parameters					
DBW	198	Subtype	INT	-1…32000	Subtype of the tool
DBW	200	Duplo	INT	-1…32000	Duplo number of the tool
DBB	202	Ident	STRING	32 characters	Tool identifier
DBW	236	T number	INT	-1…32000	T number of the tool
DBW	238	ToolState	INT	-1…32000	Tool status
DBW	240	Magazine location type	INT	-1…32000	Tool size
DBB	242	ToolSize_Upper	BYTE	1…127	Tool size upwards in half locations
DBB	243	ToolSize_Down	BYTE	1…127	Tool size downwards in half locations
DBB	244	ToolSize_Left	BYTE	1…127	Tool size to the left in half locations
DBB	245	ToolSize_Right	BYTE	1…127	Tool size to the right in half locations
DBW	246	Magazines	INT	0…32000	Magazine number of the tool
DBW	248	Magazine place	INT	0…32000	Magazine location number of the tool

6.4.4 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 NCK after the unload operation.

Table 6-13 PLC parameters for "unload to the code carrier"

PLC data		Identifier	Type	Range of values	Meaning
Transfer parameters					
DBW	142	Duplo	INT	0…32000	Duplo number
DBB	144	Ident	String	32 characters	Tool identifier
DBW	178	TNumber	INT	0…32000	T number
DBW	180	MagazinePlace	INT	0…32000	Magazine location
DBB	188	Channel	BYTE	1…127	TO area

PLC data		Identifier	Type	Range of values	Meaning
DBB	189	Loading Place	BYTE	1…127	Number of the loading/unloading location
DBW	190	Magazines	INT	0…32000	Magazine number
DBB	192	Unit	BYTE	1…127	Code carrier number
DBX	196.3	DeleteTool	BOOL	True, False	Delete tool after unloading
Return parameters					
None					

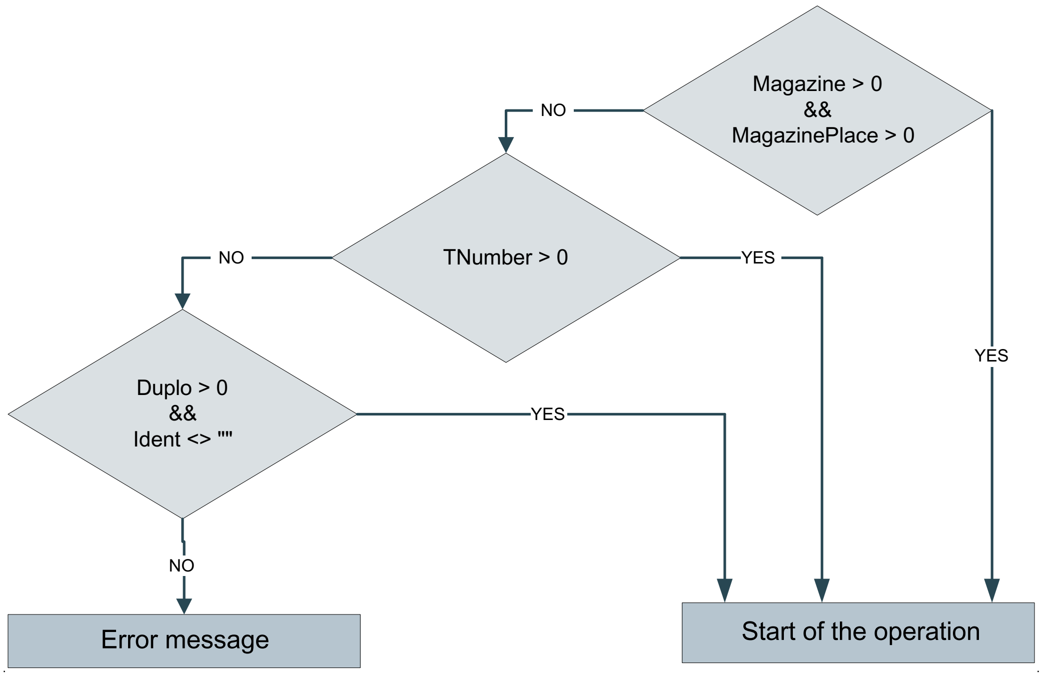


Image 6-24 Flowchart to identify tools

6.4.5 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 6-14 PLC parameters for "Load tool - tool list"

PLC data		Identifier	Type	Range of values	Meaning
Transfer parameters					
DBW	142	Duplo	INT	1…32000	Duplo number
DBB	144	Ident	String	32 characters	Tool identifier
DBW	178	TNumber	INT	0…32000	T number
DBW	180	MagazinePlace	INT	0…32000	Magazine location number
DBB	188	Channel	BYTE	1…127	TO area
DBB	189	Loading Place	BYTE	1…127	Number of the loading/unloading location
DBW	190	Magazines	INT	0…32000	Magazine number
Return parameters					
None					

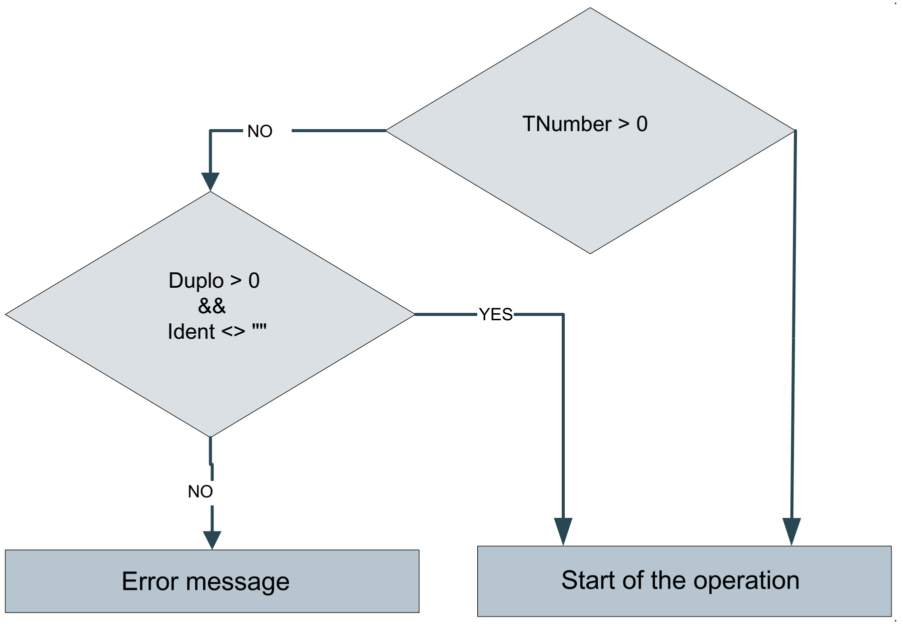


Image 6-25 Flowchart to identify tools

6.4.6 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 NCK after the unload operation.

Table 6-15 PLC parameters for "Unload tool - tool list"

PLC data		Identifier	Type	Range of values	Meaning
Transfer parameters					
DBW	142	Duplo	INT	0…32000	Duplo number
DBB	144	Ident	String	32 characters	Tool identifier
DBW	178	TNumber	INT	0…32000	T number
DBB	188	Channel	BYTE	1…127	TO area
DBB	189	LoadingPlace	BYTE	0…127	Number of loading point
DBW	190	Magazines	INT	0…32000	Magazine number
DBW	180	MagazinPlace	INT	0...32000	Magazine location
DBX	196.3	DeleteTool	BOOL	True, False	Delete tool after unloading
Return parameters					
None					

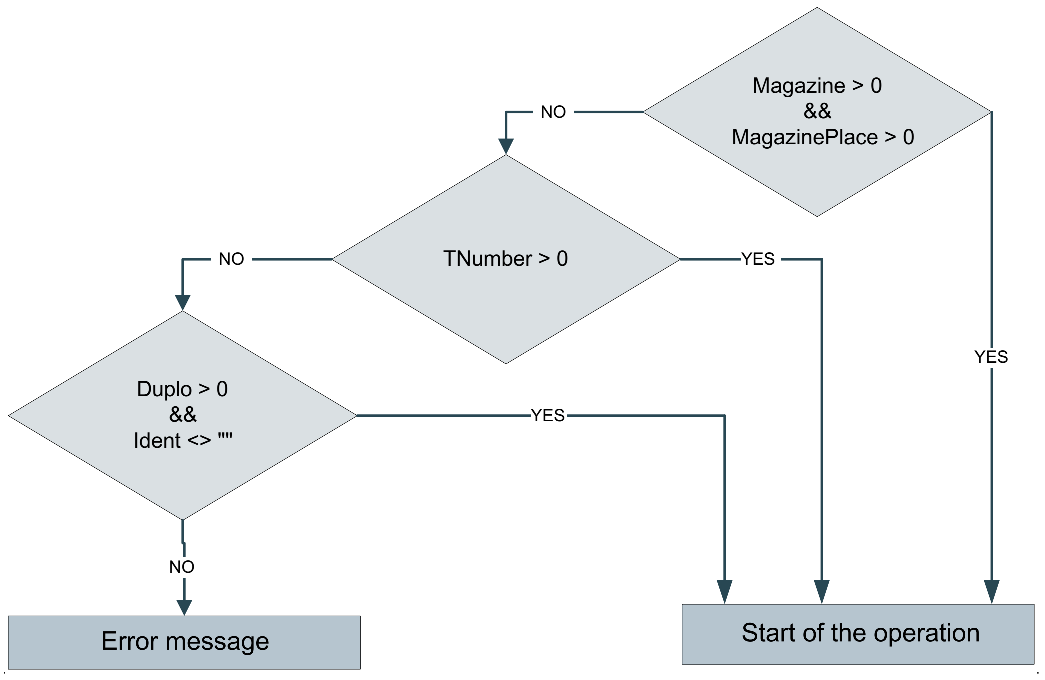


Image 6-26 Flowchart to identify tools

6.4.7 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 6-16 PLC parameters for "empty location search"

PLC data		Identifier	Type	Range of values	Meaning
Transfer parameters					
DBW	142	Duplo	INT	0…32000	Duplo number
DBB	144	Ident	String	32 characters	Tool identifier
DBW	178	TNumber	INT	0…32000	T number
DBB	188	Channel	BYTE	1…127	TO area
DBB	189	LoadingPlace	BYTE	0…127	Number of loading point
DBW	190	Magazines	INT	0…32000	Magazine number
DBW	180	MagazinPlace	INT	0...32000	Magazine location
Return parameters					
DBW	198	Subtype	INT	-1…32000	Subtype of the tool
DBW	200	Duplo	INT	-1…32000	Duplo number of the tool
DBB	202	Ident	STRING	32 characters	Tool identifier
DBW	236	T number	INT	-1…32000	T number of the tool
DBW	238	ToolState	INT	-1…32000	Tool status
DBW	240	Magazine location type	INT	-1…32000	Tool size
DBB	242	ToolSize_Upper	BYTE	1…127	Tool size upwards in half locations
DBB	243	ToolSize_Down	BYTE	1…127	Tool size downwards in half locations
DBB	244	ToolSize_Left	BYTE	1…127	Tool size to the left in half locations
DBB	245	ToolSize_Right	BYTE	1…127	Tool size to the right in half locations
DBW	246	Magazines	INT	0…32000	Magazine number of the tool
DBW	248	MagazinePlace	INT	0…32000	Magazine location number of the tool

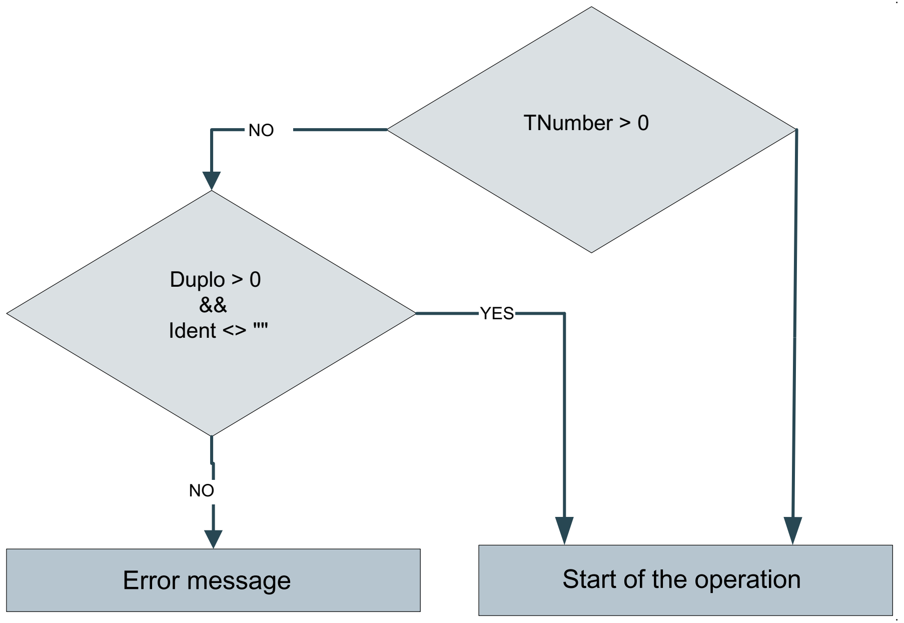


Image 6-27 Flowchart to identify tools

6.4.8 Deleting a tool in the NCK (K code = 10)

Deletes the tool specified with "TNumber" or with "Ident" and "Duplo" in the NCK 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 NCK.

If "TNumber = 0" is set, an attempt is made to delete the tool with the specified "Ident" and "Duplo" in the NCK.

Table 6-17 PLC parameters for "Delete tool in the NCK"

PLC data		Identifier	Type	Range of values	Meaning
Transfer parameters					
DBW	142	Duplo	INT	0…32000	Duplo number
DBB	144	Ident	String	32 characters	Tool identifier
DBW	178	TNumber	INT	0…32000	T number
DBB	188	Channel	BYTE	1…127	TO area
Return parameters					
None					

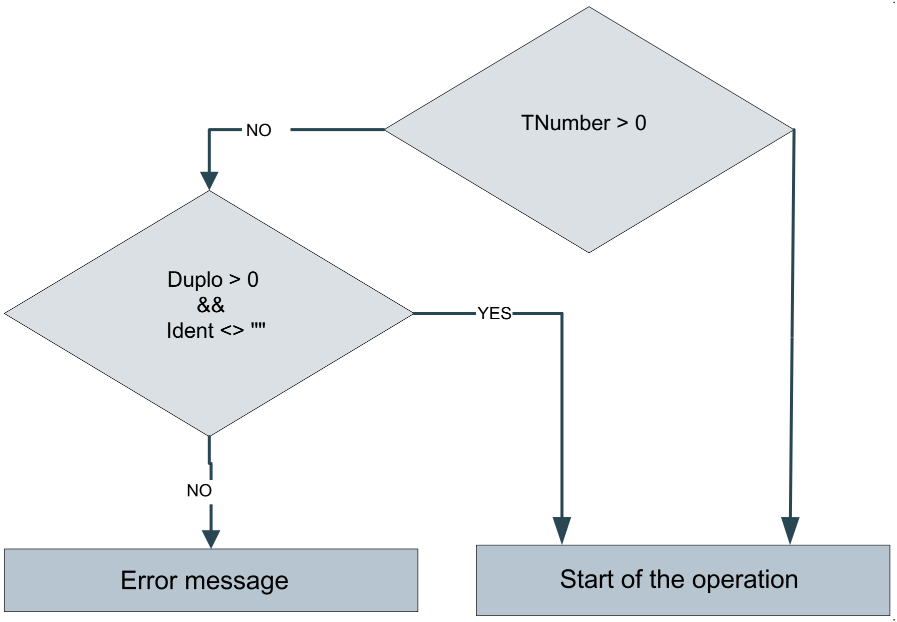


Image 6-28 Flowchart to identify tools

6.4.9 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 6-18 PLC parameters for "Update code carrier"

PLC data		Identifier	Type	Range of values	Meaning
Transfer parameters					
DBW	142	Duplo	INT	0…32000	Duplo number
DBB	144	Ident	String	32 characters	Tool identifier
DBW	178	TNumber	INT	0…32000	T number
DBB	188	Channel	BYTE	1…127	TO area
DBW	190	Magazines	INT	0…32000	Magazine number
DBW	180	MagazinPlace	INT	0...32000	Magazine location
DBB	192	Unit	BYTE	1...127	Number of the code carrier
Return parameters					
None					

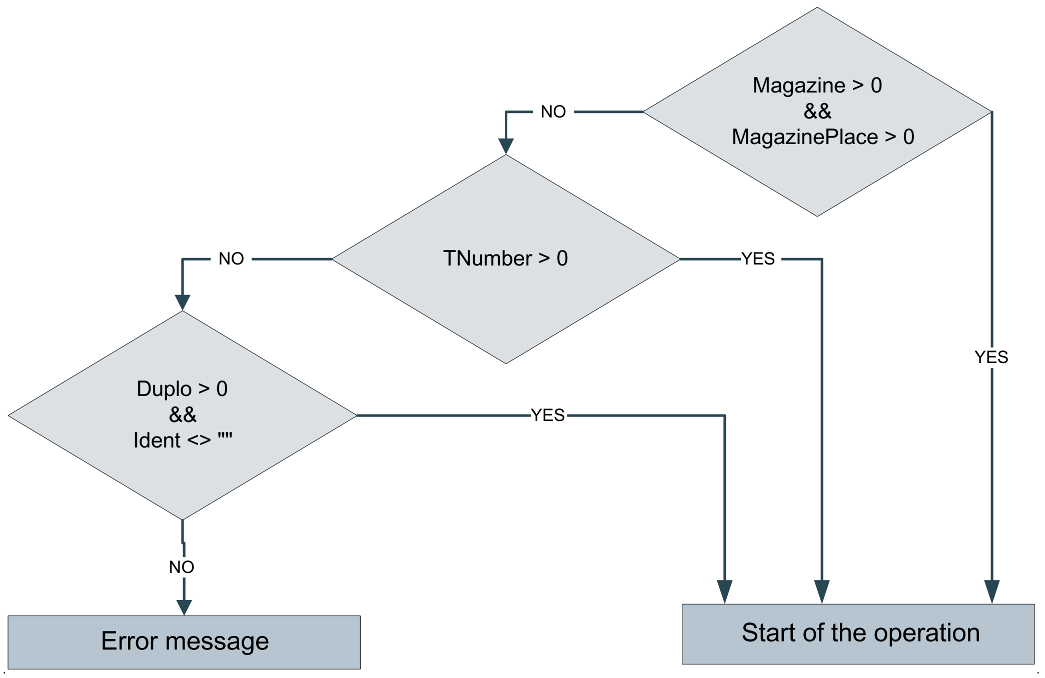


Image 6-29 Flowchart to identify tools

6.4.10 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 6-19 PLC parameters for "Reading data from the code carrier"

PLC data		Identifier	Type	Range of values	Meaning
Transfer parameters					
DBB	192	Unit	BYTE	1…127	Number of the read head
Return parameters					
DBW	198	Subtype	INT	-1…32000	Subtype of the tool
DBW	200	Duplo	INT	-1…32000	Duplo number of the tool
DBB	202	Ident	STRING	32 characters	Tool identifier
DBW	238	ToolState	INT	-1…32000	Tool status
DBW	240	Magazine location type	INT	-1…32000	Tool size
DBB	242	ToolSize_Upper	BYTE	1…127	Tool size upwards in half locations
DBB	243	ToolSize_Down	BYTE	1…127	Tool size downwards in half loca-tions

PLC data		Identifier	Type	Range of values	Meaning
DBB	244	ToolSize_Left	BYTE	1…127	Tool size to the left in half locations
DBB	245	ToolSize_Right	BYTE	1…127	Tool size to the right in half locations

6.4.11 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 6-20 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	Load from the tool cabinet
04	Unload into the tool cabinet
05	Load from production host computer
06	Unload to the production host computer
07	Unload all tools in the tool cabinet
08	Load tool
09	Unload tool
0A	Empty location search
0B	Delete tool in the NCK
0C	Delete tool in the database
0D	Export tool data
0E	Import tool data
0F	Update code carrier
10	Read data from code carrier
11	Save tool data in the NCK
...	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

XX	YY

The contents of XX = 00 and YY = 00 mean that no error has occurred.

XX = 01: General group error

Table 6-21 Low byte (YY) of the function error code

	Cause of the error
01	Error when reading parameter from DB19
02	Error when calling the internal service
03	Timeout when executing the service
04	Still no data determined
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
1F	ActiveEnable
20	Override
21	LoadTool
22	DeleteTool
23	Length of Ident
...	Reserved

XX = 02: Code carrier error group

Table 6-22 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 carrier not recognized (reading/writing, first page)
69	Data carrier not recognized (reading/writing, following page)
6D	Data carrier not recognized (writing, first page)
6E	Data carrier not recognized (reading before writing)
6F	Data carrier 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 carrier
8D	Data carrier access error
8E	Data in the data carrier incomplete

XX = 05: NCK error group

Table 6-23 Low byte (YY) of NCK error code

	Cause of the error
00	Connection not established to the requested NCU
05	Reading the NCK 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

	Cause of the error
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 NCK
47	The tool specified with the T number was not found in the NCK
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 NCK
4B	Number of found tools was not able to be read in the NCK
50	The tool specified with ident and duplo was not created in the NCK
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 NCK
5B	The tool cutting edge with the specified number has not been deleted in the NCK
5C	The specified cutting edge does not exist in the NCK
5D	The specified tool cutting edge number is not valid.
5E	The specified cutting edge is already in the NCK
5F	Tool state was not able to be read in the NCK
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 NCK
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 NCK
87	The specified tool is not loaded into the NCK
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 NCK
96	Magazine not in the loading position or not enabled for unloading
9B	The tool unload operation was exited with error
A0	The tool was not able to be deleted in the NCK
A5	The tool is active in the NCK! The tool was not able to be deleted in the NCK
D2	The specified tool is loaded into the NCK! The tool was not deleted in the DB

XX = 0B: Tool conversion error group

Table 6-24 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 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.
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

	Cause of the 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, basis 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 'DecodeToolData' 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

6.5 Examples

6.5.1 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 NCK 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 DB19. 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 6-25 Transfer parameters for "Loading from code carrier"

PLC data		Identifier	Value
DBW	180	MagazinePlace	0
DBB	188	Channel	1
DBB	189	LoadingPlace	1
DBW	190	Magazines	1
DBB	192	Unit	1
DBB	193	NCU_Index	0
DBX	196.2	LoadTool	TRUE

Table 6-26 Return parameters from Tool Ident Connection for "Loading from the code carrier"

PLC data		Identifier	Value
DBW	198	Subtype	120
DBW	200	Duplo	1
DBB	202	Ident	Tool_Test13
DBW	236	TNumber	289
DBW	238	ToolState	14
DBW	240	MagazinePlaceType	1
DBB	242	ToolSize_Upper	1
DBB	243	ToolSize_Down	1
DBB	244	ToolSize_Left	1
DBB	245	ToolSize_Right	1
DBW	246	Magazines	1
DBW	248	MagazinePlace	1

The sequence of the individual operations can be clearly seen in the following flowchart:

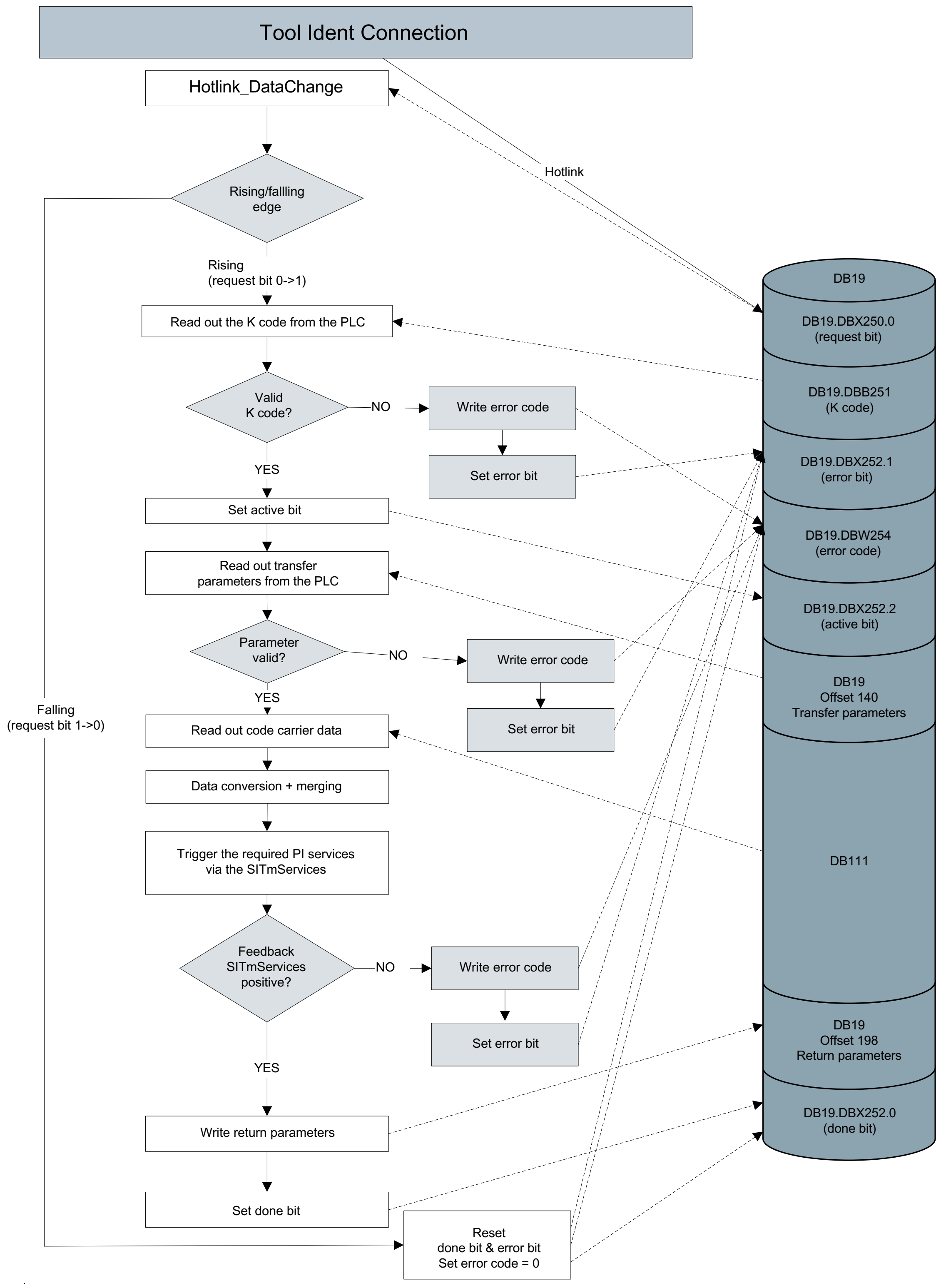
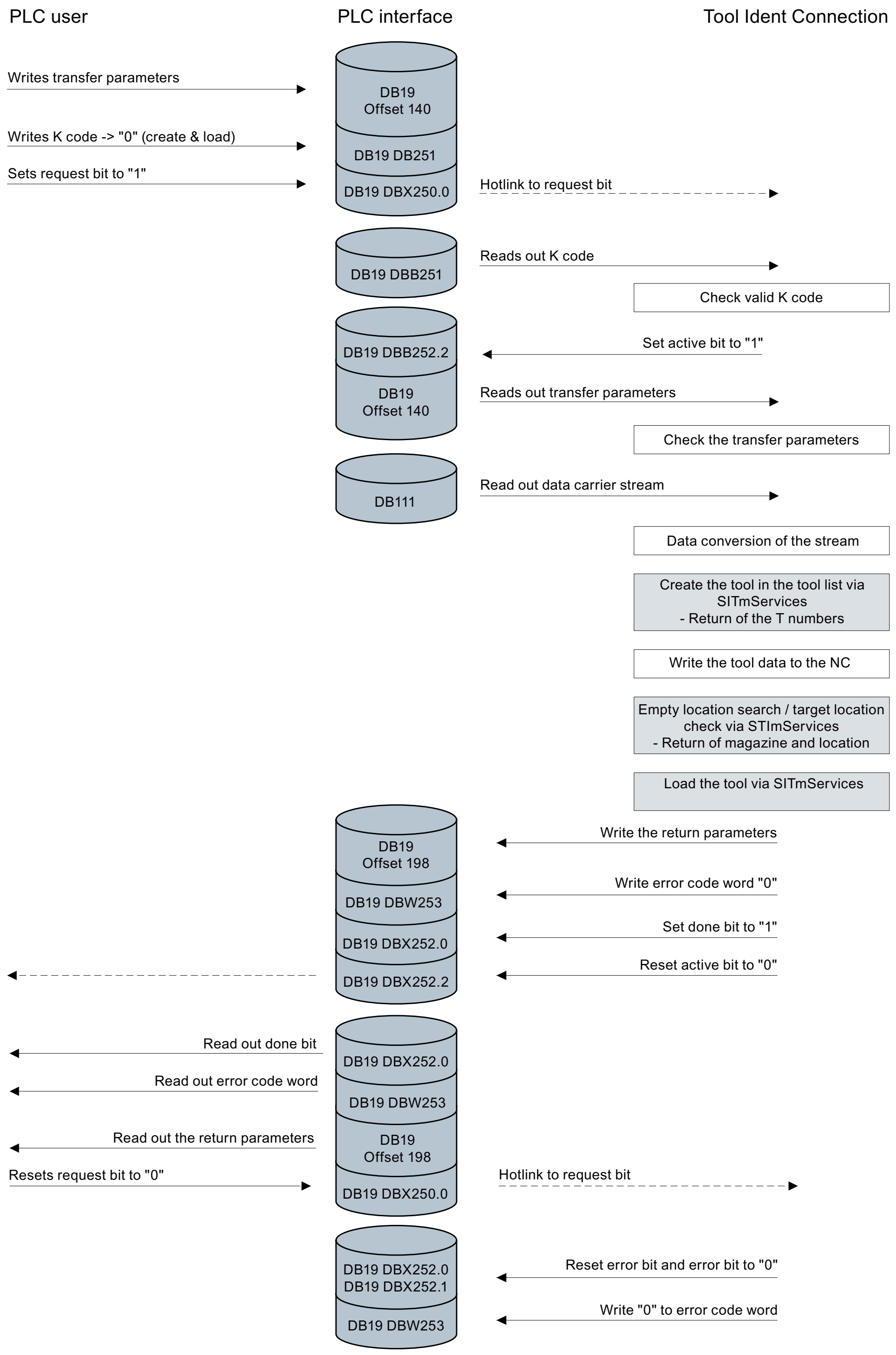

Tool Ident Connection
Hotlink_DataChange
Hotlink
Rising/fallling edge
Rising (request bit 0->1)
Falling (request bit 1->0)
Read out the K code from the PLC
Valid K code?
NO
Write error code
Set error bit
YES
Set active bit
Read out transfer parameters from the PLC
Parameter valid?
NO
Write error code
Set error bit
YES
Read out code carrier data
Data conversion + merging
Trigger the required PI services via the SITmServices
Feedback SITmServices positive?
NO
Write error code
Set error bit
YES
Write return parameters
Set done bit
Reset done bit & error bit Set error code = 0
DB19
DB19.DBX250.0 (request bit)
DB19.DBB251 (K code)
DB19.DBX252.1 (error bit)
DB19.DBW254 (error code)
DB19.DBX252.2 (active bit)
DB19 Offset 140 Transfer parameters
DB111
DB19 Offset 198 Return parameters
DB19.DBX252.0 (done bit)

Image 6-30 Flowchart: Loading from the code carrier via the PLC interface


PLC user
PLC interface
Tool Ident Connection
Writes transfer parameters
DB19 Offset 140
Writes K code -> "0" (create & load)
DB19 DB251
Sets request bit to "1"
DB19 DBX250.0
Hotlink to request bit
DB19 DBB251
Reads out K code
Check valid K code
DB19 DBB252.2
Set active bit to "1"
DB19 Offset 140
Reads out transfer parameters
Check the transfer parameters
DB111
Read out data carrier stream
Data conversion of the stream
Create the tool in the tool list via SITmServices - Return of the T numbers
Write the tool data to the NC
Empty location search / target location check via STImServices - Return of magazine and location
Load the tool via SITmServices
DB19 Offset 198
Write the return parameters
DB19 DBW253
Write error code word "0"
DB19 DBX252.0
Set done bit to "1"
DB19 DBX252.2
Reset active bit to "0"
Read out done bit
DB19 DBX252.0
Read out error code word
DB19 DBW253
Read out the return parameters
DB19 Offset 198
Resets request bit to "0"
DB19 DBX250.0
Hotlink to request bit
DB19 DBX252.0
DB19 DBX252.1
Reset error bit and error bit to "0"
DB19 DBW253
Write "0" to error code word

6.5.2 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. The tool is deleted in the NCK after being unloaded. 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). Starting the service and the procedure are equivalent to the procedure in Loading the code carrier via PLC interface (Page 455).

Table 6-27 Transfer parameters for "Unloading from code carrier"

PLC data		Identifier	Value
DBW	142	Duplo	1
DBB	144	Ident	Tool_Test13
DBW	180	MagazinePlace	1
DBB	188	Channel	1
DBB	189	LoadingPlace	2
DBW	190	Magazines	1
DBB	192	Unit	1
DBB	193	NCU_Index	0
DBX	196.3	DeleteTool	TRUE

The sequence of the individual operations can be clearly seen in the following flowchart:

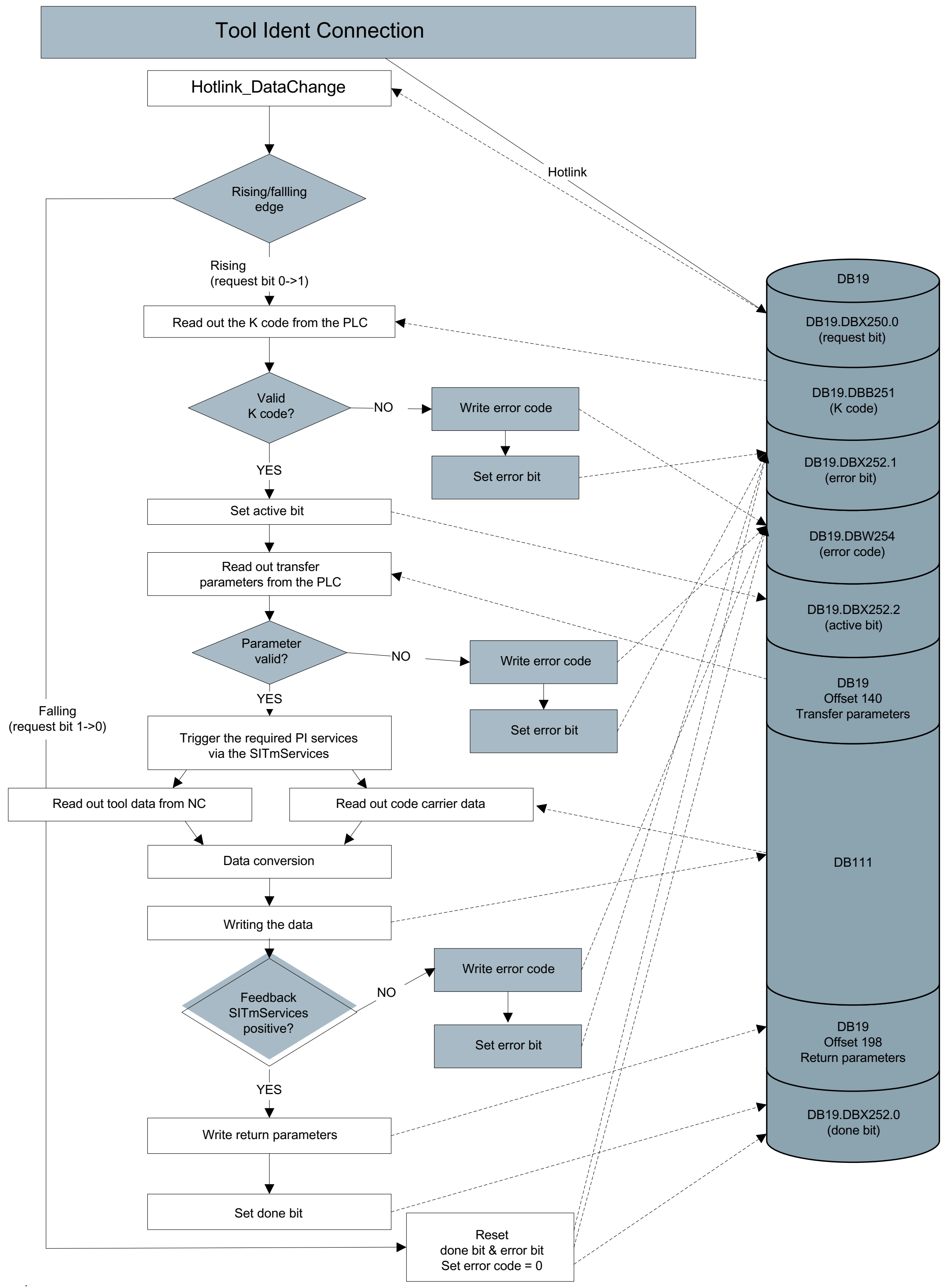

Tool Ident Connection
Hotlink_DataChange
Hotlink
Rising/fallling edge
Rising (request bit 0->1)
Read out the K code from the PLC
Valid K code?
NO
Write error code
Set error bit
YES
Set active bit
Read out transfer parameters from the PLC
Parameter valid?
NO
Write error code
Set error bit
YES
Falling (request bit 1->0)
Trigger the required PI services via the SITmServices
Read out tool data from NC
Read out code carrier data
Data conversion
Writing the data
Feedback SITmServices positive?
NO
Write error code
Set error bit
YES
Write return parameters
Set done bit
Reset done bit & error bit Set error code = 0
DB19
DB19.DBX250.0 (request bit)
DB19.DBB251 (K code)
DB19.DBX252.1 (error bit)
DB19.DBW254 (error code)
DB19.DBX252.2 (active bit)
DB19 Offset 140 Transfer parameters
DB111
DB19 Offset 198 Return parameters
DB19.DBX252.0 (done bit)

Image 6-31 Flowchart: Unloading from the code carrier via the PLC interface

Machine data 7

7.1 NC-specific machine data

15710	**TCA_CYCLE_NAME**	
MD number	Name of subprogram for TCA replacement	
Default setting: -	Min. input limit: -	Max. input limit: -
Change becomes effective after: POWER ON	Protection level: 7/2	Unit: -
Data type: STRING		
Meaning:	Program name for the replacement program when calling the TCA command. If the TCA command is programmed in a part program block, then at the end of the block, the subprogram defined in $MN_TCA_CYCLE_NAME is called. The programmed tool can be queried in the replacement program using the system variables $C_TS_PROG / $C_TS, the duplo number using $C_DUPLO_PROG / $C_DUPLO and the tool holder / spindle number using $C_THNO_PROG / $C_THNO. In the replacement program, the system variable $C_TCA supplies the value TRUE. If $MN_TCA_CYCLE_NAME contains an empty string, then the replacement is deactivated (default setting).	
Corresponding to...		
Additional references:		

17500	**MAXNUM_REPLACEMENT_TOOLS**	
MD number	Maximum number of replacement tools	
Default setting: 0	Min. input limit: 0	Max. input limit: 32
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Only meaningful if tool management function or tool monitoring function are active 0: The number of replacement tools is not monitored. 1: There can be exactly one replacement tool for each identifier. This data does not affect memory requirements, but merely has a monitoring function.	
Corresponding to...	MD18080 MM_TOOL_MANAGEMENT_MASK MD20310 TOOL_MANAGEMENT_MASK	
Additional references:	Description of functions: Memory configuration (S7)	

17504	**MAX_TOOLS_PER_MULTITOOL**	
MD number	Multitool function. Number of tool locations per multitool	
Default setting: 6	Min. input limit: 2	Max. input limit: 64
Change becomes effective after: POWER ON	Protection level: 1/1	Unit: -
Data type: DWORD		
Meaning:	"Several tools at a magazine location" function (multitool). Maximum number of locations or tools per multitool.	

17504	MAX_TOOLS_PER_MULTITOOL
Corresponding to...	
Additional references:	

17510	$MN_TOOL_UNLOAD_MASK	
MD number	Behavior of tool data at unloading	
Default setting: 0	Min. input limit: 0	Max. input limit: 0xF
Changes effective after Power On	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	When the tool is unloaded, some tool data can be assigned fixed values: Bit 0 = 0: Tool status "active" remains unchanged Bit 0 = 1: Tool status "active" is deleted ($TC_TP8, bit 0) Bit 1 = 0: Tool status "was in use" remains unchanged Bit 1 = 1: Tool status "was in use" is deleted ($TC_TP8, bit 7) Bit 2 = 0: Tool parameter $TC_TP10 remains unchanged Bit 2 = 1: Tool parameter $TC_TP10 is set to value 0. This means the tool replacement strategy is reset. Bit 3 = 0: Tool parameter $TC_TP11 remains unchanged Bit 3 = 1: Tool parameter $TC_TP11 is set to value 0. This means the assignment to the tool subgroup is canceled. If a multitool is unloaded, then also the tools contained in it corresponding to MD $MN_TOOL_UNLOAD_MASK are unloaded from the magazine location and therefore the tools states are also changed according to the specifications of the MD. When unloading the MT, the tools remain in the MT. Data defined in the MD, which is also defined for the MT - that is previously the MT state $TC_MTP8, bit 7 "was in use" - is also changed for the unloaded MT.	
Corresponding to...		
Additional references:		

17515	$MN_TOOL_RESETMON_MASK	
MD number	Behavior of tool data at RESETMON	
Default setting: 0x14	Min. input limit: 0	Max. input limit: 0x69F
Changes effective after Power On	Protection level: 2/7	Unit: -
Data type: DWORD		

17515	$MN_TOOL_RESETMON_MASK
Meaning:	The RESETMON command specifies in the 5th parameter which tool status is to be reset. If the 5th parameter is omitted, it is replaced with the value from this MD. This value is always used with PI service "_N_TRESMON". The bits are assigned like the bits in tool status $TC_TP8[x]. Bit 0 = 0: Tool status "active" remains unchanged Bit 0 = 1: Tool status "active" is deleted Bit 1 = 0: Tool status "enabled" remains unchanged Bit 1 = 1: Tool status "enabled" is set Bit 2 = 0: Tool status "disabled" remains unchanged Bit 2 = 1: Tool status "disabled" is deleted if permitted by the monitoring data and the 4th parameter is set accordingly. Bit 3 = 0: Tool status "measured" remains unchanged. Bit 3 = 1: Tool status "measured" is set. Bit 4 = 0: Tool status "prewarning limit" remains unchanged Bit 4 = 1: Tool status "prewarning limit" is deleted if permitted by the monitoring data and the 4th parameter is set. Bit 5: Not permitted (tool status "tool being changed") Bit 6: Not permitted (tool status "tool is fixed-location-coded") Bit 7 = 0: Tool status "was in use" remains unchanged Bit 7 = 1: Tool status "was in use" is deleted Bit 8 = 0 not permitted (tool status "being transported back") Bit 9: Not permitted Bit 10 = 0: Tool status "to unload" remains unchanged Bit 10 = 1: Tool status "to unload" is deleted Bit 11: Not permitted (tool status "to load") Bit 12 = 0: Not permitted (tool status "master tool") Bit 13: Not permitted Bit 14: 1:1 exchange (not permitted) The default setting corresponds to behavior up to now. The bits that are not permitted are filtered and hidden by the limit screen.
Corresponding to...	
Additional references:	

17520	$MN_TOOL_DEFAULT_DATA_MASK	
MD number	Create new tool: Data default setting	
Default setting: 0	Min. input limit: 0	Max. input limit: 0x1F
Changes effective after Power On	Protection level: 2/7	Unit: -
Data type: DWORD		

17520	$MN_TOOL_DEFAULT_DATA_MASK
Meaning:	When a tool is redefined, some tool data can be assigned fixed default values. This way simple applications do not need to process data which does not necessarily need to be assigned individual values. Bit 0 = 0: Default value of tool status ($TC_TP8), Bit1=0="not enabled" Bit 0 = 1: Default value of tool status ($TC_TP8), Bit1=1="enabled" Bit 1 = 0: Default value of tool status ($TC_TP8), bit 6=0="not fixed-location-coded" Bit 1 = 1: Default value of tool status ($TC_TP8), bit 6=1="fixed-location-coded" Bit 2 = 0: The tool is only included in the tool group with the explicit write command for the tool name. Only then can it be loaded at change using the appropriate programming. Bit 2 = 1 The tool is automatically included in the tool group when redefined. (Now the tool change can be performed with the default name ("t"=t no.). The "tool name" ($TC_TP2) can be hidden to the user. (This only makes sense if replacement tools are not being used for machining or if the tool name is not explicitly written. Then, data inconsistency problems could occur.) Bit 3 = 0: Only with TMMG: Default value of location type ($TC_TP7)=9999=not defined Bit 3 = 1: Significance, only with TMMG: Default value of location type ($TC_TP7)=1 and associated default values of magazine location type (TC_MPP2)=1. All magazine locations can now accept all tools. Bit 4 = 0: Only significant with TMMG + active adjacent location consideration: When setting/ resetting the magazine location status "disabled", the magazine location status "overlapping permitted" remains unchanged. Bit 4 = 1: Only significant with TMMG + active adjacent location consideration: When setting/ resetting the magazine location status "disabled", the magazine location status "overlapping permitted" is also automatically set/reset. If a multitool is generated, then the data defined by MD $MN_TOOL_DEFAULT_DATA_MASK and for the MT are set to the required values.
Corresponding to...	
Additional references:	

17530	$MN_TOOL_DATA_CHANGE_COUNTER	
MD number	Identifying tool data change for HMI	
Default setting: 0	Min. input limit: 0	Max. input limit: 0xF
Changes effective after Power On	Protection level: 2/7	Unit: -
Data type: DWORD		

17530	$MN_TOOL_DATA_CHANGE_COUNTER
Meaning:	HMI display support. This data allows explicit inclusion/exclusion of individual data in the OPI variables (block C/S) toolCounter, toolCounterC, toolCounterM to be taken into account. Bit 0 = 0: Value changes to tool status ($TC_TP8) are not taken into account in toolCounterC Bit 0 = 1: Value changes to the tool status ($TC_TP8) are taken into account in toolCounterC Bit 1 = 0: Value changes to remaining tool count ($TC_MOP4) are not taken into account in toolCounterC Bit 1 = 1: Value changes to the remaining tool count ($TC_MOP4) are taken into account in toolCounterC. Bit 2 = 0: Value changes to the tool data is not taken into account in the tool data update service Bit 2 = 1: Value changes to the tool data is taken into account in the tool data update service Bit 3 = 0: Value changes to the magazine data is not taken into account in the tool data update service Bit 3 = 1: Value changes to the magazine data is taken into account in the tool data update service "Value changes to tool status" and "Value changes to remaining tool count" are relative to the value changes which are caused by internal processes in the NC, as well as to value changes caused by writing the respective system variables.
Corresponding to...	
Additional references:	

17540	TOOLTYPES_ALLOWED	
MD number	Permitted tool types	
Default setting: 0x3FF	Min. input limit: 0	Max. input limit: 0x3FF
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Definition of tool types (see $TC_DP1) permitted in the NCK for tool offset selection. This means tools of any tool type can be loaded to the NCK, however, only the tool types specified here can be defined in the tool determining the offset. A bit value = 1 means that the specified tool type range is permitted for the offset selection. A bit value = 0 means that the specified tool type range is not permitted for the offset selection. When an attempt is made to select an offset for a cutting edge of this type, the selection is rejected and an offset-capable alarm is issued. Value = 0, 9999 for the tool type means "not defined". In general, it is not possible to select tool offsets with this value for the tool type. Bit 0 = 0x1: Tool types 1 to 99 permitted Bit 1 = 0x2: Tool types 100 to 199 permitted (milling tools) Bit 2 = 0x4: Tool types 200 to 299 permitted (drilling tools) Bit 3 = 0x8: Tool types 300 to 399 permitted Bit 4 = 0x10: Tool types 400 to 499 permitted (grinding tools) Bit 5 = 0x20: Tool types 500 to 599 permitted (turning tools) Bit 6 = 0x40: Tool types 600 to 699 permitted Bit 7 = 0x80: Tool types 700 to 799 permitted Bit 8 = 0x100: Tool types 800 to 899 permitted Bit 9 = 0x200: Tool types 900 to 999 permitted	
Corresponding to...	MD18100 $MN_NUM_CUTTING_EDGES_IN_TOA	
Additional references:		

18074	**MM_TOOL_MANAGEMENT_TRACE_SZ**	
MD number	Maximum size of the tool management diagnostics ring buffer	
Default setting: 25, 25	Min. input limit: 4	Max. input limit: 500
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Number of entries in the diagnostics ring buffer of the tool management. Index 0 = buffer size of the IPO trace. Index 1 = buffer size of the prep trace. There is a dedicated IPO trace buffer in each channel, and there is only a prep trace buffer in channel 1. The memories are only allocated if bit 0 (0x0001) for a warm restart is at ON - and more precisely in both MD18080: MM_TOOL_MANAGEMENT_MASK and MD20310: TOOL_MANAGEMENT_MASK for each channel. Trace data is written to the buffer if bit 13 (0x2000) is at ON in MD20310: TOOL_MANAGEMENT_MASK for each channel.	
Corresponding to...		
Additional references:		

18075	**MM_NUM_TOOLHOLDERS**	
MD number	Max. number of tool holders per TOA	
Default setting: 16	Min. input limit: 1	Max. input limit:
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		

18075	MM_NUM_TOOLHOLDERS
Meaning:	Maximum number of definable tool holders per TO area. The maximum value is 20. The number 20 is derived from the maximum number of axes per channel in the 840D sl. The address extension e of the commands Te=T, Me=6 (*) is the number of the tool holder. t=T number / tool name - depending on the function, which is activated in the NCK. (*) if the following applies: $MC_TOOL_CHANGE_MODE=1 and $MC_TOOL_CHANGE_M_CODE=6 In milling machines, the tool holder is generally not a spindle, see also $MC_SPIND_DEF_MASTER_SPIND. In turning machines, the tool holder is generally not a spindle axis, see also $MC_TOOL_MANAGEMENT_TOOLHOLDER. Then the following should apply $MN_MM_NUM_TOOLHOLDERS greater or equal to $MC_SPIND_DEF_MASTER_SPIND/$MC_TOOL_MANAGEMENT_TOOLHOLDER. If bit 0 = 1 in $MN_MM_TOOL_MANAGEMENT_MASK and $MC_TOOL_MANAGEMENT_MASK is set (= magazine management (TMMG)) is only applicable for sensible values $MN_MM_NUM_TOOLHOLDERS less than or equal to $MN_MM_NUM_LOCS_WITH_DISTANCE. Then, a maximum of $MN_MM_NUM_TOOLHOLDERS buffer locations, type spindle ($TC_MPP1[9998,x]=2) can be defined. Example: TMMG not active Assuming $MC_SPIND_DEF_MASTER_SPIND=3, $MN_MM_NUM_TOOLHOLDERS = 3. Then T1=t, T2=t, T3=t, T=t can be programmed. Example: TMMG active, milling machine with Me=6 as tool change command Assuming $MN_MM_NUM_TOOLHOLDERS=14, $MN_MM_NUM_LOCS_WITH_DISTANCE=20, 10 channels are active, all channels have TMMG active and have the same tool and machine data (=one TO area for all channels). $MC_SPIND_DEF_MASTER_SPIND=1,....,10 for the channels. Then, in the magazine buffer, up to 14 locations, type "tool holder"/"spindle" can be defined. In addition, 6 other grippers or similar can be defined. These locations (a maximum of 20) can be linked to magazines. T1=t,....T14=t and Tt or M1=6,....M14=6 and M06 can be programmed in the channels. The PLC version used can limit the maximum number of tool holders.
Corresponding to...	
Additional references:	

18076	MM_NUM_LOCS_WITH_DISTANCE	
MD number	Max. number of magazine locations per TOA with distance connection	
Default setting: 32	Min. input limit: 1	Max. input limit:
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		

18076	MM_NUM_LOCS_WITH_DISTANCE
Meaning:	The machine data makes sense if the magazine management function TMMG is active - see $MN_MM_TOOL_MANAGEMENT_MASK, $MC_TOOL_MANAGEMENT_MASK; in each case bit 0 = 1 Maximum number of magazine locations (spindles, loading locations, etc.) per TOA, that can have a distance connection to a magazine defined using $TC_MDPx[n,m]. Example: TMMG is active: $MN_MM_NUM_LOCS_WITH_DISTANCE is = 5 and $MN_MM_NUM_DIST_REL_PER_MAGLOC = 2. Two TOA units are defined each with three tool holders / spindles, two loading points. Further, there are two grippers defined in each TO unit. This means that in total there are 14 locations defined in the buffer magazine / loading magazine for which distances and assignments should be defined. TO unit 1 has 4 magazines defined, TO unit 2 has 6 magazines defined. With the set value of $MN_MM_NUM_LOCS_WITH_DISTANCE = 5, each tool holder and each loading point can be linked; (see $TC_MDP1 and $TC_MDP2) and up to two grippers can be additionally assigned to each tool holder ($MN_MM_NUM_DIST_REL_PER_MAGLOC = 2); (see $TC_MLSR). As a consequence, a tool holder / a spindle location can have two tables - a distance table to magazines and an assignment table to grippers and similar locations.
Corresponding to...	
Additional references:	

18077	MM_NUM_DIST_REL_PER_MAGLOC	
MD number	Max. number of magazines in the distance table of a magazine location	
Default setting:	Min. input limit: 0	Max. input limit:
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	The machine data is only active, if the magazine management function, TMMG is active - see $MN_MM_TOOL_MANAGEMENT_MASK, $MC_TOOL_MANAGEMENT_MASK. Two quantities are defined with the data: 1.) Max. number of magazines in the distance table of a magazine location (spindle, loading location, etc.) 2.) Maximum number of locations (gripper, ...) in the connection table of a spindle / tool holder. Example: $MN_MM_NUM_DIST_REL_PER_MAGLOC = 3. Two TOA units are defined each with two tool holders / spindles and one loading point each. Further, four grippers are defined in each TO unit. TO unit 1 has 4 magazines defined, TO unit 2 has 6 magazines defined. Then, each tool holder can define up to three distances to the magazines (see $TC_MDP2) and in addition, can define up to three relationships to grippers ($TC_MLSR).	
Corresponding to...		
Additional references:		

18078	MM_MAX_NUM_OF_HIERARCHIES	
MD number	Maximum number of definable hierarchies for magazine location types	
Default setting: 8	Min. input limit: 0	Max. input limit:
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -

18078	MM_MAX_NUM_OF_HIERARCHIES	
Data type: DWORD		
Meaning:	The machine data is only active, if the magazine management function, TMMG is active - see $MN_MM_TOOL_MANAGEMENT_MASK, $MC_TOOL_MANAGEMENT_MASK. Maximum number of definable hierarchies for magazine location types The permissible value of index n of system parameter $TC_MPTH[n,m] is from 0 to "$MN_MM_MAX_NUM_OF_HIERARCHIES - 1". (The maximum of index m can be specified using machine data $MN_MM_MAX_HIERA-CHY_ENTRIES.) A value = 0 means that the "Magazine location type hierarchy" function is not available.	
Corresponding to...		
Additional references:		

18079	MM_MAX_HIERARCHY_ENTRIES	
MD number	Maximum permissible number of entries in a magazine-location-type hierarchy	
Default setting: 8	Min. input limit: 1	Max. input limit:
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	The machine data is only active, if the magazine management function, TMMG is active - see $MN_MM_TOOL_MANAGEMENT_MASK, $MC_TOOL_MANAGEMENT_MASK - and if $MN_MM_MAX_NUM_OF_HIERARCHIES is greater than zero. Maximum number of entries in a magazine location type hierarchy. The permissible value of index m of system parameter $TC_MPTH[n,m] is from 0 to "$MN_MM_MAX_NUM_OF_HIERARCHIES - 1". (The maximum of index n can be specified using machine data $MN_MM_MAX_NUM_OF_HI-ERARCHIES.) A value = 0 means that the "Magazine location type hierarchy" function is not available.	
Corresponding to...		
Additional references:		

18080	MM_TOOL_MANAGEMENT_MASK	
MD number	Memory for the tool management is reserved step-by-step	
Default setting: 0x0	Min. input limit: 0	Max. input limit: 0xFFFF
Change becomes effective after: POWER ON	Protection level: 1/7	Unit: -
Data type: DWORD		

18080	MM_TOOL_MANAGEMENT_MASK
Meaning:	Activation of the tool management memory with "0" means: The set tool management data does not occupy any memory, tool management is not available. Bit 0=1: Memory for data specific to tool management is available; the MDs for reserving memory must be set accordingly (MD18086 MM_NUM_MAGAZINE_LOCATION, MD18084 MM_NUM_MAGAZINE) Bit 1=1: Memory is available for monitoring data (TMMO) Bit 2=1: Memory is available for user data (CC data) Bit 3=1: Memory is available for considering the adjacent location Bit 4=1: Memory and function release for PI service _N_TSEARC = "Complex search for tools in magazine" is available. Bit 5=1: Wear monitoring active Bit 6=1: Wear group available Bit 7=1: Reserve memory for adapter of magazine locations Bit 8=1: Memory for machining and/or setting up offsets Bit 9=1: Tools in a turret no longer vacate their turret location during a tool change. Only relevant for HMI adv. Bit 10 = 1: The multitool function is available. The configuration can be changed by other machine data. Bit 10=0: The multitool function is not available. The function version set by other machine data is not active. The coded type of memory reservation enables economic use of memory management for the functionality provided. Example: Standard memory reservation for tool management: MM_TOOL_MANAGEMENT_MASK = 3 (Bit 0 + 1=1) means tool management and tool monitoring data is made available MM_TOOL_MANAGEMENT_MASK = 1 means tool management without tool monitoring function data MM_TOOL_MANAGEMENT_MASK = 2 means tool management without tool monitoring function data
Special cases, errors,...	

18082	MM_NUM_TOOL	
MD number	Number of tools the NCK can manage	
Default setting: 30	Min. input limit: 0	Max. input limit: 1500
Changes effective after POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	The NC can manage as a maximum the number of tools entered in the MD. A tool has at least one cutting edge. A buffered (backed-up) user memory is used. The maximum number of tools possible correspond to the number of cutting edges. The MD should also be set even if no tool management is being used. If this machine data is altered the buffered data is lost. A maximum of 1500 tools are possible per TO unit and in the entire NCK.	

18082	MM_NUM_TOOL
Special cases, errors,...:	
Corresponding to...	MD18100 MM_NUM_CUTTING_EDGES_IN_TOA
Additional references:	Description of functions: Memory Configuration (S7), Tool Offset (W1)

18083	MM_NUM_MULTITOOL	
MD number	Multitool function. Number of multitools that the NCK can manage	
Default setting: 15	Min. input limit: 0	Max. input limit: 1500
Changes effective after POWER ON	Protection level: 1/1	Unit: -
Data type: DWORD		
Meaning:	"Several tools at a magazine location (multitool)" function. Number of multitools (multiple tools) that the NCK can manage.	
Special cases, errors,...:		
Corresponding to...		
Additional references:		

18084	MM_NUM_MAGAZINE	
MD number	Number of magazines the NCK can manage	
Default setting: 4	Min. input limit: 0	Max. input limit: 64
Changes effective after POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Tool management (TMG or TMMG) only if MD tool management and option tool management are set: Number of magazines that the NCK can manage (active and background magazines). The buffered memory for the magazines is reserved with this machine data. Important: One load magazine and one buffer magazine are set up in the tool management for each TOA unit. These magazines must be taken into account. Value = 0: Tool management cannot be activated because no data can be created.	
Special cases, errors,...:		
Corresponding to...	MD18080 MM_TOOL_MANAGEMENT_MASK MD20310 TOOL_MANAGEMENT_MASK	
Additional references:	Description of functions: Memory configuration (S7)	

18085	MM_NUM_MULTITOOL_LOCATIONS	
MD number	Multitool function. Number of multitool locations that the NCK can manage	
Default setting: 30	Min. input limit: 0	Max. input limit: 1500
Changes effective after POWER ON	Protection level: 1/1	Unit: -
Data type: DWORD		
Meaning:	"Several tools at a magazine location (multitool)" function. Number of multitool locations that the NCK can manage.	
Special cases, errors,...:		
Corresponding to...		
Additional references:		

18086	MM_NUM_MAGAZINE_LOCATION	
MD number	Number of magazine locations the NCK can manage	
Default setting: 30	Min. input limit: 0	Max. input limit: 1500
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	TMMG - only if MD for tool management and tool management option are set: Number of magazine locations that the NCK can manage. The buffered memory for the magazine locations is reserved with this MD. Important: The number of all buffer locations and loading points must be taken into account here. Value = 0: Tool management cannot be activated because no data can be created.	
Special cases, errors,...:		
Corresponding to...	MD18080 MM_TOOL_MANAGEMENT_MASK MD20310 TOOL_MANAGEMENT_MASK	
Additional references:	Description of functions: Memory configuration (S7)	

18088	MM_NUM_TOOL_CARRIER	
MD number	Maximum number of defined tool holders	
Default setting: 0	Min. input limit: 0	Max. input limit: 99999999
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Maximum number of definable tool holders for tools that can be oriented in the TO area. The value is divided by the number of active TO units. The integer result indicates how many tool holders can be defined per TO unit. The data to define a tool holder is set using the system variables $TC_CARR1, ... $TC_CARR14. The data is located in the buffered memory. Example: 2 channels are active, with one channel per TO unit (=default). 3 holders must be defined in channel 1 and one holder in channel 2. The value to be set is 6 because 6/2 = 3. This means a max. of 3 holder definitions in each TO unit.	
Special cases, errors,...:		
Corresponding to...		
Additional references:	Description of functions: Tool Offsets (S7)	

18090	MM_NUM_CC_MAGAZINE_PARAM	
MD number	Number of OEM magazine data	
Default setting: 0	Min. input limit: 0	Max. input limit: 64
Changes effective after POWER ON	Protection level: 2/2	Unit: -
Data type: DWORD		
Meaning:	Number of magazine parameters (of the integer type) that are made available to the user or the Compile Cycle. If this machine data is set, the amount of buffered memory required increases by sizeof(int) * max. number of magazines.	

18090	MM_NUM_CC_MAGAZINE_PARAM
Special cases, errors,...:	
Corresponding to...	MD18080 MM_TOOL_MANAGEMENT_MASK MD18084 MM_NUM_MAGAZINE
Additional references:	

18091	MM_TYPE_CC_MAGAZINE_PARAM[n]	
MD number	OEM magazine data type	
Default setting: 3	Min. input limit: 1	Max. input limit: 6
Change becomes effective after: POWER ON	Protection level: 2/2	Unit: -
Data type: DWORD		
Meaning:	Only the standard default values may be applied. Used to assign individual types to the parameters. The array index n can assume values between 0 and the setting in machine data MD18090: MM_NUM_CC_MAGAZINE_PARAM. The possible values of the MD = 1, 2, 3, 4, 5 and 6 denote the NC command types BOOL, CHAR, INT, REAL, STRING and AXIS. The type FRAME cannot be defined here. Type STRING must not be longer than 31 characters. Example: MD18090: MM_NUM_CC_MAGAZINE_PARAM=1 MD18091: MM_TYPE_CC_MAGAZINE_PARAM=5 = "UserMagazine" can be programmed for parameter $TC_MAPC1. The buffered work memory is used. Changing the value can, but does not necessarily, result in reconfiguration of the buffered memory.	
Corresponding to...	MD18090 MM_NUM_CC_MAGAZINE_PARAM MD18084 MM_NUM_MAGAZINE	
Additional references:		

18092	MM_NUM_CC_MAGLOC_PARAM	
MD number	Number of OEM magazine location data	
Default setting: 0	Min. input limit: 0	Max. input limit: 64
Changes effective after POWER ON	Protection level: 2/2	Unit: -
Data type: DWORD		
Meaning:	Number of magazine-location data parameters (of the integer type) that are made available to the user or the Compile Cycle. If this machine data is set, the amount of buffered memory required increases by sizeof(int) * max. number of magazine locations.	
Special cases, errors,...:		
Corresponding to...	MD18080 MM_TOOL_MANAGEMENT_MASK MD18086 MM_NUM_MAGAZINE_LOCATION	
Additional references:		

18093	MM_TYPE_CC_MAGLOG_PARAM[n]	
MD number	OEM magazine location data type	
Default setting: 3	Min. input limit: 1	Max. input limit: 6
Change becomes effective after: POWER ON	Protection level: 2/2	Unit: -
Data type: DWORD		
Meaning:	Only the standard default values may be applied. Used to assign individual types to the parameters. The array index n can assume values between 0 and the setting in machine data MD18090: MM_NUM_CC_MAGAZINE_PARAM. The possible values of the MD = 1, 2, 3, 4 and 6 denote the NC command types 1 BOOL 2 CHAR 3 INT 4 REAL and 6 AXIS Here, the STRING type is not explicitly possible. The value 5 is treated just like 2. The type FRAME cannot be defined here. Example: MD18090: MM_NUM_CC_MAGAZINE_PARAM=1 MD18091: MM_TYPE_CC_MAGAZINE_PARAM=2 = "UserMagazineLocation" can be programmed for parameter $TC_MPPC1. The buffered work memory is used. Changing the value can, but does not necessarily, result in reconfiguration of the buffered memory.	
Corresponding to...	MD18092 MM_NUM_CC_MAGLOG_PARAM	
Additional references:		

18094	MM_NUM_CC_TDA_PARAM	
MD number	Number of OEM tool data	
Default setting: 0	Min. input limit: 0	Max. input limit: 64
Changes effective after POWER ON	Protection level: 2/2	Unit: -
Data type: DWORD		
Meaning:	Number of tool-specific data that can be created for each tool (of type integer) and are available to the user or compile cycle. If this machine data is set, the amount of buffered memory required increases by sizeof(double) * max. number of tools.	
Special cases, errors,...:		
Corresponding to...	MD18080 MM_TOOL_MANAGEMENT_MASK MD18082 MM_NUM_TOOL	
Additional references:		

18095	MM_TYPE_CC_TDA_PARAM[n]	
MD number	OEM tool data type	
Default setting: 4	Min. input limit: 1	Max. input limit: 6
Change becomes effective after: POWER ON	Protection level: 2/2	Unit: -

18095	MM_TYPE_CC_TDA_PARAM[n]	
Data type: DWORD		
Meaning:	Only the standard default values may be applied. Used to assign individual types to the parameters. The array index n can assume values between 0 and the setting in machine data MD18094: MM_NUM_CC_TDA_PARAM. The possible values of the MD = 1, 2, 3, 4, 5 and 6 denote the NC command types. 1 BOOL 2 CHAR 3 INT 4 REAL 5 STRING and 6 AXIS The type FRAME cannot be defined here. Type STRING must not be longer than 31 charac-ters. Example: MD18094: MM_NUM_CC_TDA_PARAM=1 MD18095: MM_TYPE_CC_TDA_PARAM=5 = "UserCuttingEdge" can be programmed for parameter $TC_TPC1. The buffered work memory is used. Changing the value can, but does not necessarily, result in reconfiguration of the buffered memory.	
Corresponding to...	MD18094 MM_NUM_CC_TDA_PARAM MD18082 MM_NUM_TOOL	
Additional references:		

18096	MM_NUM_CC_TOA_PARAM	
MD number	Number of data per tool cutting edge for compile cycles	
Default setting: 0	Min. input limit: 0	Max. input limit: 64
Change becomes effective after: POWER ON	Protection level: 2/2	Unit: -
Data type: DWORD		
Meaning:	Number of TOA data that can be created for each tool (of type Double) and are available to the user or compile cycle. If this machine data is set, the amount of buffered memory required increases by sizeof(dou-ble) * max. number of cutting edges.	
Special cases, errors,...:		
Corresponding to...	MD18080 MM_TOOL_MANAGEMENT_MASK MD18100 MM_NUM_CUTTING_EDGES_IN_TOA	
Additional references:		

18097	MM_TYPE_CC_TOA_PARAM[n]	
MD number	OEM data type per cutting edge	
Default setting: 4	Min. input limit: 1	Max. input limit: 6
Change becomes effective after: POWER ON	Protection level: 2/2	Unit: -
Data type: DWORD		

18097	MM_TYPE_CC_TOA_PARAM[n]
Meaning:	Only the standard default values may be applied. Used to assign individual types to the parameters. The array index n can assume values between 0 and the setting in machine data MD18096: MM_NUM_CC_TOA_PARAM. The possible values of the MD = 1, 2, 3, 4 and 6 denote the NC command types 1 BOOL 2 CHAR 3 INT 4 REAL 6 AXIS The type STRING cannot be used explicitly here, value 5 is treated like value 2. The type FRAME cannot be defined here. Example: MD18096: MM_NUM_CC_TOA_PARAM=1 MD18097: MM_TYPE_CC_TOA_PARAM=5 = "UserCuttingEdge" can be programmed for parameter $TC_DPC1. The buffered work memory is used. Changing the value can, but does not necessarily, result in reconfiguration of the buffered memory.
Corresponding to...	MD18096 MM_NUM_CC_TOA_PARAM MD18100 MM_NUM_CUTTING_EDGES_IN_TOA
Additional references:	

18098	MM_NUM_CC_MON_PARAM	
MD number	Number of monitoring data per tool for compile cycles	
Default setting: 0	Min. input limit: 0	Max. input limit: 64
Change becomes effective after: POWER ON	Protection level: 2/2	Unit: -
Data type: DWORD		
Meaning:	Number of monitoring data that is created for each tool (of type integer) and is available to the user or compile cycle. If this machine data is set, the amount of buffered memory required increases by sizeof(int) * max. number of cutting edges.	
Special cases, errors,...:		
Corresponding to...	MD18080 MM_TOOL_MANAGEMENT_MASK MD18100 MM_NUM_CUTTING_EDGES_IN_TOA	
Additional references:		

18099	MM_TYPE_CC_MON_PARAM[n]	
MD number	OEM monitor data type	
Default setting: 3	Min. input limit: 1	Max. input limit: 6
Change becomes effective after: POWER ON	Protection level: 2/2	Unit: -
Data type: DWORD		

18099	MM_TYPE_CC_MON_PARAM[n]
Meaning:	Only the standard default values may be applied. Used to assign individual types to the parameters. The array index n can assume values between 0 and the setting in machine data MD18098: MM_NUM_CC_MON_PARAM. The possible values of the MD = 1, 2, 3, 4 and 6 denote the NC command types 1 BOOL 2 CHAR 3 INT 4 REAL and 6 AXIS The type FRAME cannot be defined here. (Type STRING is not explicitly possible here; the value 5 is treated just like 2). Example: MD18098: MM_NUM_CC_MON_PARAM=1 MD18099: MM_TYPE_CC_MON_PARAM=2 = "UserCuttingEdge" can be programmed for parameter $TC_MOPC1. The buffered work memory is used. Changing the value can, but does not necessarily, result in reconfiguration of the buffered memory.
Corresponding to...	MD18100 MM_NUM_CUTTING_EDGES_IN_TOA MD18098 MM_NUM_CC_MON_PARAM
Additional references:	

18100	MM_NUM_CUTTING_EDGES_IN_TOA	
MD number	Tool cutting edges per TO area	
Default setting: 30	Min. input limit: 0	Max. input limit: 3000
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Defines the number of tool cutting edges in a TO area. For each tool edge approx. 250 bytes per TOA block of the battery-backed memory are reserved with this machine data irrespective of the tool type. Tools with type 400-499 cutting edges (=grinding tools) also occupy the location of a cutting edge. Example: Define 10 grinding tools with one cutting edge each. Then, as a minimum, the following must apply: MM_NUM_TOOL = 10 MM_NUM_CUTTING_EDGES_IN_TOA = 20 See also MM_NUM_TOOL A buffered (backed-up) user memory is used.	
Special cases, errors,...:	If this machine data is altered the buffered data is lost!	
Corresponding to...		
Additional references:	Description of functions: Memory configuration (S7)	

18102	MM_TYPE_OF_CUTTING_EDGE	
MD number	Type of D number programming	
Default setting: 0	Min. input limit: 0	Max. input limit: 1
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	The "flat D number management" is activated with the MD. The individual values can determine the type of D programming - direct or - indirect programming. The default value is 0. This setting means that the NCK manages the T and D numbers. A value > 0 is only accepted by the NCK if bit 0 is not set in MD $MN_MM_TOOL_MANAGEMENT_MASK, i.e. the tool management function must not be simultaneously active. 0: No "flat D number management" active 1: D numbers are directly and absolutely programmed Notice: SINUMERIK Operate does not support the function "Flat D numbers".	
Special cases, errors,...:		
Corresponding to...		
Additional references:	Description of functions: Tool Offset (W1)	

18104	MM_NUM_TOOL_ADAPTER	
MD number	Tool adapter in TO area	
Default setting: -1	Min. input limit: -1	Max. input limit: 600
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		

18104	MM_NUM_TOOL_ADAPTER
Meaning:	Number of tool adapters in the TO area. This function can only be used if magazine locations are available in the NCK. The tool management function must be active. In order to activate the setting, bit 7 (=0x80) must be set in MD $MN_MM_TOOL_MANAGEMENT _MASK. Adapter data records and the cutting-edge-specific basis / adapter data records mutually exclude one another, i.e. when adapter data is defined, then parameters $TC_DP21, $TC_DP22, $TC_DP23 or their values in NCK are available. -1: Every magazine location is automatically assigned an adapter. This means that internally there are just as many adapters foreseen as are foreseen by the magazine locations set in machine data $MN_MM_NUM_MAGAZINE_LOCATION. -2 Every magazine location and each multitool location is automatically assigned an adapter. This means that internally there are just as many adapters foreseen as are foreseen by the magazine locations set in machine data $MN_MM_NUM_MAGAZINE_LOCATION and multitool locations set in $MN_MM_NUM_MULTITOOL_LOCATIONS. 0: No adapter-data definition possible. Cutting-edge-specific parameters $TC_DP21, $TC_DP22 and $TC_DP23 are available in cases where adapters are utilized outside the active TM. > 0: Number of adapter data records. By this, adapters can be defined independently of magazine locations. An additional step following definition of the data assigns the adapters to magazine locations.
Corresponding to...	MD18080 MM_TOOL_MANAGEMENT_MASK MD20310 TOOL_MANAGEMENT_MASK MD18084 MM_NUM_MAGAZINE MD18086 MM_NUM_MAGAZINE_LOCATION
Additional references:	

18105	MM_MAX_CUTTING_EDGE_NO	
MD number	Maximum value of D number	
Default setting: 9	Min. input limit: 1	Max. input limit: 32000
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		

18105	MM_MAX_CUTTING_EDGE_NO
Meaning:	Maximum value of D number. The maximum number of D numbers per cutting edge is not affected by this. The monitoring of the D number assignment associated with this value is only effective for new definitions of D numbers. This means the existing data records are not checked later - if the MD is changed. Advisable setting: $MN_MM_MAX_CUTTING_EDGE_NO equal to $MN_MM_MAX_CUTTING_EDGE_PER_TOOL. If $MN_MM_MAX_CUTTING_EDGE_NO > $MN_MM_MAX_CUTTING_EDGE_PER_TOOL is set, you should familiarize yourself with the difference between offset number D and the tool edge number CE. See also language commands CHKDNO, CHKDM, GETDNO, SETDNO, DZERO. The MD is not evaluated with the function "flat D number" and accordingly is not meaningful there. The MD can change the memory requirements: A change from "less than equal to" to "greater than" - or vice versa - in the values of both the above mentioned MDs can influence the demand for non-buffered memory.
Corresponding to...	MD18106 MM_MAX_CUTTING_EDGE_PER_TOOL
Additional references:	Description of functions: Tool Offset (W1)

18106	MM_MAX_CUTTING_EDGE_PER_TOOL	
MD number	Maximum number of D numbers per tool	
Default setting: 9	Min. input limit: 1	Max. input limit: 12
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Maximum number of cutting edges (D offset) per tool (per T number) This allows greater security with the data definition. A value of 1 can be set if only tools with one cutting edge are to be used. This will avoid the problem of assigning more than one cutting edge to the tool when data is defined. Logically the same value is set for MM_MAX_CUTTING_EDGE_NO as for MM_MAX_CUTTING_EDGE_PER_TOOL. If MM_MAX_CUTTING_EDGE_NO is set greater than MM_MAX_CUTTING_EDGE_PER_TOOL, you should familiarize yourself with the difference between offset number D and the tool edge number CE. See also language commands CHKDNO, CHKDM, GETDNO, SETDNO, DZERO. The MD is not evaluated with the function "flat D number" and accordingly is not meaningful there. The MD can change the memory requirements: A change from "less than equal to" to "greater than" - or vice versa - in the values of both the above mentioned MDs can influence the demand for non-buffered memory.	
Corresponding to...	MD18105 MM_MAX_CUTTING_EDGE_NO	
Additional references:	Description of functions: Tool Offset (W1)	

18108	MM_NUM_SUMCORR	
MD number	Additive offsets in the TO area	
Default setting: -1	Min. input limit: -1	Max. input limit: 9000
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -

18108	MM_NUM_SUMCORR
Data type: DWORD	
Meaning:	Total number of additive offsets in NCK. A value of -1 means that the number of additive offsets equals the number of cutting edges * number of additive offsets per cutting edge. A value > 0 and < "Number of cutting edges * number of additive offsets per cutting edge" means that per cutting edge, a maximum "Number of additive offsets per cutting edge" additive offsets can - but does not need to - be defined, i.e. this provides an option of using the buffered memory sparingly. Only the cutting edges defined for the explicit data have a additive offset data record. Buffered memory is reserved. The memory requirements for additive offset are doubled if "Setup offset" is also configured and active; see MD $MN_MM__KIND_OF_SUMCORR.
Corresponding to...	MD18100 MM_NUM_CUTTING_EDGE_IN_TOA MD18110 MM_MAX_SUMCORR_PER_CUTTEDGE
Additional references:	Description of functions: Tool Offset (W1)

18110	MM_MAX_SUMCORR_PER_CUTTEDGE	
MD number	Maximum number of additive offsets per cutting edge	
Default setting: 1	Min. input limit: 1	Max. input limit: 6
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Maximum number of additive offsets per cutting edge The following applies for MM_NUM_SUMCORR > 0: This data does not define the memory, but is used for monitoring purposes only. The following applies to MM_NUM_SUMCORR = -1: This data defines the memory.	
Corresponding to...	MD18100 MM_NUM_CUTTING_EDGES_IN_TOA MD18108 MM_NUM_SUMCORR	
Additional references:	Description of functions: Memory configuration (S7)	

18112	MM_KIND_OF_SUMCORR	
MD number	Properties of additive offsets in the TO area	
Default setting: 0	Min. input limit: 0	Max. input limit: 0x1F
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		

18112	MM_KIND_OF_SUMCORR
Meaning:	Properties of additive offsets in NCK. Bit 0=0: "Additive offsets fine" are saved when the tool data is backed up. Bit 0=1: "Additive offsets fine" are saved when the tool data is not backed up. Bit 1=0: Set-up offsets are saved when the tool data is backed up. Bit 1=1: Set-up offsets are not saved when the tool data is backed up. Bit 2=0: If the function tool management (TMMG) or tool monitoring (TMMO) are used, then setting the tool status to "active" has no effect on the "additive offsets fine"/setup offsets already in use. Bit 2=1 When the tool status is set to "active", the existing additive offsets are set to the value 0. This does not influence the setup offsets. Bit 3=0: If the functions "TMG" = "Adapter" are in use: "Additive offsets fine"/setup offsets are transformed. Bit 3=1: "Additive offsets fine"/setup offsets are not transformed. Bit 4=0: No setting up offset data records. Bit 4=1: Setting up offset data records are additionally created. The additive offset is thus the sum of the setting up offset + additive offset fine. Changing the states of bits 0, 1, 2 and 3 does not alter the memory configuration. Changing the status of bit 4 causes the buffered memory to be reconfigured with the next POWER ON.
Corresponding to...	MD18100 MM_NUM_CUTTING_EDGES_IN_TOA MD18108 MM_NUM_SUMCORR MD18110 MM_MAX_SUMCORR_PER_CUTTEDGE MD18080 MM_TOOL_MANAGEMENT_MASK MD20310 MC_TOOL_MANAGEMENT_MASK MD18086 MM_NUM_MAGAZINE_LOCATION MD18104 MM_NUM_TOOL_ADAPTER
Additional references:	Description of functions: Tool Offset (W1)

18192	MM_NUM_CC_MULTITOOL_PARAM	
MD number	Number of multitool-specific parameters $TC_MTPCn per multitool	
Default setting: 0	Min. input limit: 0	Max. input limit: 64
Change becomes effective after: POWER ON	Protection level: 1/1	Unit: -
Data type: DWORD		
Meaning:	Number of multitool-specific parameters $TC_MTPCn, that can be created for each multitool and that are available to users or the compile cycle.	
Corresponding to...		
Additional references:		

18193	MM_TYPE_CC_MULTITOOL_PARAM	
MD number	OEM multitool data type	
Default setting: 3, 3, 3, 3	Min. input limit: 1	Max. input limit: 10
Change becomes effective after: POWER ON	Protection level: 1/1	Unit: -
Data type: DWORD		

18193	**MM_TYPE_CC_MULTITOOL_PARAM**
Meaning:	User or OEM data in the tool management. Type of the multitool-specific Siemens user data $TC_MTPCn configured using MD18192 $MN_MM_NUM_CC_MULTITOOL_PARAM. Every parameter can be allocated its own type. The following types are permissible: Type value of the machine data (see types of the NC language) BOOL 1 CHAR 2 INT 3 REAL 4 STRING 5 (Permits identifiers of up to 31 characters) AXIS 6 FRAME Not defined
Corresponding to...	MD18192 MM_NUM_CC_MULTITOOL_PARAM MD18083 MM_NUM_MULTITOOL MD18085 MM_NUM_MULTITOOL_LOCATIONS
Additional references:	

18194	**MM_NUM_CC_MTLOC_PARAM**	
MD number	Number of OEM multitool location data parameters $TC_MTPPCn per multitool location	
Default setting: 3, 3, 3, 3	Min. input limit: 1	Max. input limit: 64
Change becomes effective after: POWER ON	Protection level: 1/1	Unit: -
Data type: DWORD		
Meaning:	Number of multitool location-specific parameters $TC_MTPPCn that can be created for each multitool location and that are available to users or the compile cycle.	
Corresponding to...		
Additional references:		

18195	**MM_NUM_CC_MTLOC_PARAM**	
MD number	OEM multitool location data type	
Default setting: 3, 3, 3, 3	Min. input limit: 1	Max. input limit: 10
Change becomes effective after: POWER ON	Protection level: 1/1	Unit: -
Data type: DWORD		

18195	**MM_NUM_CC_MTLOC_PARAM**
Meaning:	User or OEM data in the tool management. Type of the multitool location-specific Siemens user data $TC_MTPCn configured using MD18194 $MN_MM_NUM_CC_MTLOC_PARAM. Every parameter can be allocated its own type. The following types are permissible: Type value of the machine data (see types of the NC language) BOOL 1 CHAR 2 INT 3 REAL 4 STRING 5 (Permits identifiers of up to 31 characters) AXIS 6 FRAME Not defined
Corresponding to...	MD18192 MM_NUM_CC_MULTITOOL_PARAM MD18083 MM_NUM_MULTITOOL MD18085 MM_NUM_MULTITOOL_LOCATIONS
Additional references:	

7.2 Channel-specific machine data

20090	SPIND_DEF_MASTER_SPIND[<channel>]	
MD number	Initial setting of master spindle in channel	
Default setting: 1, 1, …	Min. input limit: 1	Max. input limit: 20
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: BYTE		
Meaning:	Definition of master spindle in channel. The number of the spindle is set. Example: 1 corresponds to spindle S1. When S is programmed, the current master spindle is automatically addressed. The SETMS(n) command can be programmed to declare the spindle number as the master spindle. SETMS declares the spindle defined in the MD to be the master spindle again.	
Corresponding to...		
Additional references:	Description of functions: Spindles (S1)	

20096	T_M_ADDRESS_EXT_IS_SPINO[<channel>]	
MD number	Meaning of the address extension with T, M tool change	
Default setting: FALSE	Min. input limit: -	Max. input limit: -
Change becomes effective after: POWER ON	Protection level: 2/7	Unit:
Data type: Boolean		
Meaning:	The MD is only of significance when the "Tool management" / "Flat D numbers" are inactive. FALSE The contents of the address extension of the NC addresses T and M "change-command number" are not evaluated by the NCK. The PLC determines the meaning of the programmed extension TRUE The address extension of NC addresses T and M "tool change command number" - tool change command number" = TOOL_CHANGE_M_CODE with 6 as default value - are interpreted as spindle number. NCK handles the extension essentially the same as the active functions "tool management" or "flat D number management". This means that the programmed D number always refers to the T number of programmed main-spindle numbers.	
Corresponding to...	MD20090 SPIND_DEF_MASTER_SPIND MD22550 TOOL_CHANGE_MODE MD22560 TOOL_CHANGE_M_CODE	
Additional references:		

20110	RESET_MODE_MASK[<channel>]	
MD number	Determination of basic control settings after Reset/TP End	
Default setting: 0x0	Min. input limit: 0	Max. input limit: 0x7FFFF
Change becomes effective after: RESET	Protection level: 2/7	Unit: HEX
Data type: DWORD		

20110	RESET_MODE_MASK[<channel>]
Meaning:	Definition of the basic PLC setting after booting and reset/end of part program with respect to the G codes (especially current level and settable work offset), tool length compensation and transformation by setting the following bits. Bit 0: Reset mode Bit 1: Suppress auxiliary function output for tool selection Bit 2: Selection of the reset response after Power On; e.g. of tool offset Bit 3: Only of significance without active tool management: Selection of the reset response at the end of the test mode for active tool offsets. The bit is only of significance if bits 0 and 6 are set. It defines to what "current setting for the active tool length compensation" refers; - the program that was active at the end of test mode - the program that was active before switching-on test mode Bit 4: Reserved! Setting is now made via $MC_GCODE_RESET_MODE[.] Bit 5: Reserved! Setting is now made via $MC_GCODE_RESET_MODE[.] Bit 6: Reset behavior, "active tool length compensation" Bit 7: Reset behavior, "active kinematic transformation" Bit 8: Reset behavior, "coupled-motion axes" Bit 9: Reset behavior, "tangential tracking" Bit 10: Reset behavior, "synchronous spindle" Bit 11: Reset behavior, "revolutional feed rate" Bit 12: Reset behavior, "geo axis exchange" Bit 13: Reset behavior, "master value coupling" Bit 14: Reset behavior, "basic frame" Bits 4 to 11 are evaluated only if bit 0=1. Bit 15: Function for electronic gears, not relevant for tool management. Bit 16=0: After end of program/reset, the number given by the MD SPIND_DEF_MAS-TER_SPIND is the number of the master spindle Bit 16=1: The programmed value of SETMS is retained after end of program/reset Bit 17=0: After end of program / reset, the tool holder number provided in MD TOOL_MAN-AGEMENT_TOOLHOLDER is the number of the master tool holder Bit 17=1: The programmed value of SETMTH is retained after end of program / reset Bit 18: Reset behavior, "reference axis for G96/G961/G962" The bits 4 to 11, 16 and 17 are only evaluated for bit 0=1. Bit value=0 is set so that the behavior applicable up to now is retained with bit 0=1. (The effect of bit 0=0 was and is that the values programmed for SETMTH/SETMS are retained at end of program.) Bit 20=1: Reset behavior, "$P_USEKT"
Corresponding to...	MD20120 TOOL_RESET_VALUE MD20130 CUTTING_EDGE_RESET_VALUE MD20150 GCODE_RESET_VALUES MD20152 GCODE_RESET_MODE MD20140 TRAFO_RESET_VALUE MD20112 START_MODE_MASK MD20121 TOOL_PRESEL_RESET_VALUE MD20118 GEOAX_CHANGE_RESET
Additional references:	Description of functions: Coordinate Systems (K2)

20112	START_MODE_MASK[<channel>]	
MD number	Definition of the initial state of the control after part program start	
Default setting: 0x400	Min. input limit: 0	Max. input limit: 0x7FFFF
Change becomes effective after: RESET	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Definition of the initial state of the control for part program start with respect to the G codes (especially current level and adjustable work offset), tool length compensation, transformation and axis coupling setting the following bits (only the bits in bold are relevant for tool management): Bit 0: Not assigned: $MC_START_MODE_MASK is evaluated every time the part program is started Bit 1: Suppress auxiliary function output for tool selection Bit 4: Starting behavior, G code "current plane" Bit 5: Starting behavior, G code "settable work offset" Bit 6: Starting behavior, "active tool length compensation" Bit 7: Starting behavior, "active kinematic transformation" Bit 8: Starting behavior, "coupled-motion axes" Bit 9: Starting behavior, "tangential tracking" Bit 10: Starting behavior, "synchronous spindle" Bit 11: Reserved Bit 12: Starting behavior, "geo axis exchange" Bit 13: Starting behavior, "master value coupling" Bit 14: Starting behavior, "basic frame" Bit 15: Function for electronic gears, not relevant for tool management. Bit 16=0: The current value SETMS is retained (depends on the settings in RESET_MODE_MASK) Bit 16=1: At program start, the spindle defined in MD $MC_SPIND_DEF_MASTER_SPIND is the master spindle. Bit 17=0: The current value SETMTH is retained (depends on the settings in RESET_MODE_MASK) Bit 17=1: At program start, the number defined in MD $MC_Tool_Management_Toolholder is the number of the master tool holder Bit 18=1: Reference axis for G96/G961/G962 Bit value=0 is set so that the behavior applicable up to now is retained. Bit 2: Reserved	
Corresponding to...	MD20120 TOOL_RESET_VALUE MD20130 CUTTING_EDGE_RESET_VALUE MD20150 GCODE_RESET_VALUES MD20152 GCODE_RESET_MODE MD20140 TRAFO_RESET_VALUE MD20110 RESET_MODE_MASK MD20121 TOOL_PRESEL_RESET_VALUE MD20118 GEOAX_CHANGE_RESET	
Additional references:	Description of functions: Coordinate Systems (K2)	

20120	**TOOL_RESET_VALUE[<channel>] (only without tool management)**	
MD number	Tool length compensation at power-up (Reset/TP end)	
Default setting: 0	Min. input limit: 0	Max. input limit: 32000
Change becomes effective after: RESET	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Definition of the tool where the tool length compensation is selected at power up and for reset or TP end depending on MD20110 RESET_MODE_MASK and for TP depending on MD 20112: START_MODE_MASK.	
Corresponding to...	MD20110 RESET_MODE_MASK MD20112 START_MODE_MASK	
Additional references:	Description of functions: Coordinate Systems (K2)	

20121	**TOOL_PRESEL_RESET_VALUE[<channel>] (only without tool management)**	
MD number	Preselect tool on Reset	
Default setting: 0	Min. input limit: 0	Max. input limit: 32000
Change becomes effective after: RESET	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Definition of the preselected tool with MD 20310=1. A tool is preselected after power-up and at reset or end of part program depending on MD20110 and on start of part program depending on MD20112. This data is only valid without tool management.	
Corresponding to...	MD20110 RESET_MODE_MASK MD20112 START_MODE_MASK	
Additional references:	Description of functions: Coordinate Systems (K2)	

20122	**TOOL_RESET_NAME[<channel>]**	
MD number	Active tool at reset/start with tool management	
Default setting: -	Min. input limit: -	Max. input limit: -
Change becomes effective after: RESET	Protection level: 2/7	Unit: -
Data type: STRING		
Meaning:	This data is valid only if the TM function is active. Definition of the tool holder with which the tool length compensation is selected during power up and for a reset or an end of the part program depending on MD20110 RE-SET_MODE_MASK and for a part program start depending on MD20112 START_MODE_MASK.	
Corresponding to...	MD20110 RESET_MODE_MASK MD20112 START_MODE_MASK MD20124 TOOL_MANAGEMENT_TOOLHOLDER MD20130 CUTTING_EDGE_RESET_VALUE	
Additional references:		

20123	**USEKT_RESET_VALUE[<channel>]**	
MD number	Preselected value of $P_USEKT at RESET	
Default setting: 0x0,...	Min. input limit: 0	Max. input limit: 0xF
Change becomes effective after: RESET	Protection level: 2/7	Unit:
Data type: DWORD		
Meaning:	This data is valid only if the TM function is active. The system variable $P_USEKT is preassigned with the value of this MD: - after reset or part program end: depending on $MC_RESET_MODE_MASK	
Corresponding to...	MD20110 RESET_MODE_MASK	
Additional references:		

20124	**TOOL_MANAGEMENT_TOOLHOLDER[<channel>]**	
MD number	Tool holder number	
Default setting: 0,0,0,...	Min. input limit: 0	Max. input limit: 16
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	This MD is only of significance when tool management is active. Tool management must know on which tool holder the tool is to be loaded. The data is only evaluated if the value is greater than zero. Then, $TC_MPP5 numbers are no longer seen as "spindle numbers", but as tool holder number. The automatic address extension of T and of M06 is then the value for this MD and no longer the value of $MC_SPIND_DEF_MASTER_SPIND. The MD is used to define the master tool holder number to which a tool preparation or a tool change refers. When determining the tool on the tool holder for the setting "keep old offset" of MD $MC_RESET_MODE_MASK, then reference is also made to this value. If a machine has several tool holders, but no specific master spindle, then the MD serves as a default value to determine the tool holder to which the tool should be loaded at a tool change (reset, start, T="identifier", M06). When defining the magazine locations of internal magazines, locations, type "spindle" - $TC_MPP1=2=spindle-location - are assigned a "location type index" ($TC_MPP5). This allocates a specific tool holder to the location. The tool holder with number n can be declared the master tool holder using the language command SETMTH(n). This means that the offsets of a tool that is loaded to a buffer location, type "SPINDLE" and with the value $TC_MPP5=n, correct the tool path. Tool management change to "SPINDLE" locations with $TC_MPP5 not equal to the number of the master tool holder have no effect on the path. The command SETMTH is used to declare the tool holder defined in the MD as the master tool holder again.	
Corresponding to...	MD20110 RESET_MODE_MASK MD20112 START_MODE_MASK MD20122 TOOL_RESET_NAME MD20130 CUTTING_EDGE_RESET_VALUE	
Additional references:	Description of functions: Coordinate Systems (K2)	

20126	TOOL_CARRIER_RESET_VALUE[<channel>]	
MD number	Operative tool holder on Reset	
Default setting: 0	Min. input limit: 0.0	Max. input limit: -
Change becomes effective after: Reset	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Definition of the tool holder with which the tool length compensation is selected during power up and for a reset or an end of the part program as a dependency on the machine data $MC_RESET_MODE_MASK and for a main program start as a dependency on the machine data $MC_START_MODE_MASK. This data is valid without tool management.	
Corresponding to...	MD20110 RESET_MODE_MASK MD20112 START_MODE_MASK	
Additional references:	Description of functions: Tool Offset (W1)	

20128	COLLECT_TOOL_CHANGE[<channel>]	
MD number	Tool change commands to the PLC after block search	
Default setting: 1	Min. input limit: -	Max. input limit: -
Change becomes effective after: immediately	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	This MD is only of significance when magazine management is active. ($MN_MM_TOOL_MANAGEMENT_MASK, $MC_TOOL_MANAGEMENT_MASK) It determines whether, after a block search with calculation of the tool change command, tool preparation commands (general tool change commands) are output to the PLC or not. 1: Tool change commands and tool preparation commands are collected and output to the PLC when the program starts after the search target has been reached 0: All tool/magazine-specific commands, which were collected in the block search, are not output to the PLC with the subsequent program start! This means that also programmed POSM, TCI, TCA commands are also not output. Comment 1: Without active magazine management the tool change M code is not collected if it is not assigned to any auxiliary function group. With active magazine management, the machine data value corresponds to =0 Comment 2: The value =0 is, e.g. practical, if, after reaching the search target, the collected tool change commands are output to the PLC in an ASUP program using the commands GETSELT, GE-TEXTET.	
Corresponding to...	MD22560 TOOL_CHANGE_M_CODE	
Additional references:		

20130	CUTTING_EDGE_RESET_VALUE[<channel>]	
MD number	Tool cutting edge length compensation at power-up (Reset/TP end)	
Default setting: 0, 0, 0, ...	Min. input limit: 0	Max. input limit: 32000
Change becomes effective after: RESET	Protection level: 2/7	Unit: -
Data type: DWORD		

20130	CUTTING_EDGE_RESET_VALUE[<channel>]
Meaning:	Definition of the tool cutting edge with which the tool length compensation is selected during power up and for a reset or an end of the part program depending on MD20110 $MC_RE-SET_MODE_MASK and for a part program start depending on MD20112 $MC_START_MODE_MASK. When tool management is active and bits 0 and 6 set in $MC_RESET_MODE_MASK, the last offset of the tool which was active on power-off - generally the tool in the spindle - is operative after power up.
Corresponding to...	MD20110 RESET_MODE_MASK MD20112 START_MODE_MASK
Additional references:	Description of functions: Coordinate Systems (K2)

20132	SUMCORR_RESET_VALUE[<channel>]	
MD number	Additive offset effective at reset	
Default setting: 0, 0, ...	Min. input limit: 0	Max. input limit: 6
Change becomes effective after: RESET	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Definition of the additive offset with which the additive offset is selected during power up and for a reset or an end of the part program as a dependency on the machine data $MC_RE-SET_MODE_MASK and for a part program start as a dependency on the machine data $MC_START_MODE_MASK. Machine data 18110 $MN_MAX_SUMCORR_PER_CUTTEDGE determines the maximum meaningful value which can be entered.	
Corresponding to...	MD20110 RESET_MODE_MASK MD20112 START_MODE_MASK	
Additional references:		

20140	TRAFO_RESET_VALUE[<channel>]	
MD number	Transformation data record at power up (Reset/TP end)	
Default setting: 0	Min. input limit: 0	Max. input limit: 20
Change becomes effective after: RESET	Protection level: 2/7	Unit: -
Data type: BYTE		
Meaning:	Definition of the transformation data record selected during power up and for a reset or an end of the part program depending on MD20110 $MC_RESET_MODE_MASK and for a part program start depending on MD20112 $MC_START_MODE_MASK.	
Corresponding to...	MD20110 RESET_MODE_MASK MD20112 START_MODE_MASK	
Additional references:	Description of functions: Axes, coordinate systems,... (K2)	

20150	GCODE_RESET_VALUES[<channel>][n]	
MD number	Initial setting of G group	
Default setting:{2, 0, 0, 1, 0...}	Min. input limit:-	Max. input limit: -
Change becomes effective after: RESET	Protection level: 2/7	Unit: -
Data type: BYTE		

20150	GCODE_RESET_VALUES[<channel>][n]
Meaning:	Specification of the G code of every G group that takes effect at: • Run-up, reset and end of part program depending on MD20110 $MC_RESET_MODE_MASK • Start of part program depending on MD20112 $MC_START_MODE_MASK GCODE_RESET_VALUES[<channel>][<G group index>] = <group-specific number of the G command> with: • <G group index> = (number of the G group) - 1 • <group-specific number of the G command>: See References: Programming Manual, Fundamentals; Section "Tables" > "G commands"

Machine data	Group	Default value for 840D sl
GCODE_RESET_VALUES[0]	1	2 (G01)
GCODE_RESET_VALUES[1]	2	0 (inactive)
GCODE_RESET_VALUES[2]	3	2 (inactive)
GCODE_RESET_VALUES[3]	4	2 (STARTFIFO)
GCODE_RESET_VALUES[4]	5	0 (inactive)
GCODE_RESET_VALUES[5]	6	1 (G17)
GCODE_RESET_VALUES[6]	7	1 (G40)
GCODE_RESET_VALUES[7]	8	1 (G500)
GCODE_RESET_VALUES[8]	9	0 (inactive)
GCODE_RESET_VALUES[9]	10	1 (G60)
GCODE_RESET_VALUES[10]	11	0 (inactive)
GCODE_RESET_VALUES[11]	12	1 (G601)
GCODE_RESET_VALUES[12]	13	2 (G71)
GCODE_RESET_VALUES[13]	14	1 (G90)
GCODE_RESET_VALUES[14]	15	2 (G94)
GCODE_RESET_VALUES[15]	16	1 (CFC)
GCODE_RESET_VALUES[16]	17	1 (NORM)
GCODE_RESET_VALUES[17]	18	1 (G450)
GCODE_RESET_VALUES[18]	19	1 (BNAT)
GCODE_RESET_VALUES[19]	20	1 (ENAT)
GCODE_RESET_VALUES[20]	21	1 (BRISK)
GCODE_RESET_VALUES[21]	22	1 (CUT2D)
GCODE_RESET_VALUES[22]	23	1 (CDOF)
GCODE_RESET_VALUES[23]	24	1 (FFWOF)
GCODE_RESET_VALUES[24]	25	1 (ORIWKS)
GCODE_RESET_VALUES[25]	26	2 (RMI)
GCODE_RESET_VALUES[26]	27	1 (ORIC)
GCODE_RESET_VALUES[27]	28	1 (WALIMON)
GCODE_RESET_VALUES[28]	29	1 (DIAMOF)
GCODE_RESET_VALUES[29]	30	1 (COMPOF)
GCODE_RESET_VALUES[30]	31	1 (inactive)
GCODE_RESET_VALUES[31]	32	1 (inactive)
GCODE_RESET_VALUES[32]	33	1 (FTCOF)
GCODE_RESET_VALUES[33]	34	1 (OSOF)
GCODE_RESET_VALUES[34]	35	1 (SPOF)
GCODE_RESET_VALUES[35]	36	1 (PDELAYON)
GCODE_RESET_VALUES[36]	37	1 (FNORM)
GCODE_RESET_VALUES[37]	38	1 (SPIF1)
GCODE_RESET_VALUES[38]	39	1 (CCPRECOF)
GCODE_RESET_VALUES[39]	40	1 (CUTCONOF)
GCODE_RESET_VALUES[40]	41	1 (LFOF)
GCODE_RESET_VALUES[41]	42	1 (TCOABS)
GCODE_RESET_VALUES[42]	43	1 (G140)
GCODE_RESET_VALUES[43]	44	1 (G340)

20150	GCODE_RESET_VALUES[<channel>][n]		
Meaning:	GCODE_RESET_VALUES[44]	45	1 (SPATH)
	GCODE_RESET_VALUES[45]	46	1 (LFTXT)
	GCODE_RESET_VALUES[46]	47	1 (G290 Sinumerik mode)
	GCODE_RESET_VALUES[47]	48	3 (G460)
	GCODE_RESET_VALUES[48]	49	1 (CP)
	GCODE_RESET_VALUES[49]	50	1 (ORIEULER)
	GCODE_RESET_VALUES[50]	51	1 (ORIVECT)
	GCODE_RESET_VALUES[51]	52	1 (PAROTOF)
	GCODE_RESET_VALUES[52]	53	1 (TOROTOF)
	GCODE_RESET_VALUES[53]	54	1 (ORIROTA)
	GCODE_RESET_VALUES[54]	55	1 (RTLION)
	GCODE_RESET_VALUES[55]	56	1 (TOWSTD)
	GCODE_RESET_VALUES[56]	57	1 (FENDNORM)
	GCODE_RESET_VALUES[57]	58	1 (RELIEVEON)
	GCODE_RESET_VALUES[58]	59	1 (DYNNORM)
	GCODE_RESET_VALUES[59]	60	1 (WALCS0)
	GCODE_RESET_VALUES[60]	61	1 (ORISOF)
	GCODE_RESET_VALUES[61]	62	1 (not defined)
	GCODE_RESET_VALUES[69]	70	1 (not defined)
Corresponding to...	MD20110 RESET_MODE_MASK MD20112 START_MODE_MASK		
Additional references:	(K1, G2) The complete list of all G groups with the G functions contained therein can be found in: **References:** Programming Manual, Fundamentals		

20152	GCODE_RESET_MODE[n]	
MD number	Reset behavior of G groups	
Default setting: 0	Min. input limit: -	Max. input limit: -
Change becomes effective after: RESET	Protection level: 2/7	Unit: -
Data type: BYTE		
Meaning:	This MD is only evaluated if bit 0 is set in $MC_RESET_MODE_MASK. This MD is used to define for each entry in MD $MN_GCODE_RESET_VALUES (i.e. for each G group), whether a setting according to the $MC_GCODE_RESET_VALUES is undertaken again for a reset/part program end (MD=0) or whether the current setting is retained (MD=1). Example: The normal position of the 6th G group (current level) is read from the machine data $MC_GCODE_RESET_VALUES for each reset / part program end here: $MC_GCODE_RESET_VALUES[5]=1; reset value of 6th G group is M17 $MC_GCODE_RESET_MODE[5]=0; initial state for the 6th G group after ;reset/part program end according to ;$MC_GCODE_RESET_VALUES[5] However, if the current setting of the 6th G group (current level) is to be retained beyond reset / part program end, then the following setting is obtained: $MC_GCODE_RESET_VALUES[5]=1; reset value of 6th G group is M17 $MC_GCODE_RESET_MODE[5]=1; current setting for the 6th G group ; is also kept after reset/part program end	

20152	GCODE_RESET_MODE[n]
Corresponding to...	MD20110 RESET_MODE_MASK MD20112 START_MODE_MASK
Additional references:	Description of functions: Axes, coordinate systems,... (K2)

20270	CUTTING_EDGE_DEFAULT	
MD number	Initial state of tool cutting edge without programming	
Default setting: 1	Min. input limit: -2	Max. input limit: 32000
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	Default tool cutting edge after tool change If no cutting edge is programmed after a tool change, then the edge number preset in CUTTING_EDGE_DEFAULT is applied. Value = 0: No cutting edge is initially active after a tool change. Cutting-edge selection only takes place at D programming. Value = 1: No. of the cutting edge Value = -1: Cutting edge number of old tool also applies to new tool. Value = -2: Cutting edge (offset) of the old tool remains active until D is programmed. Example: MD: CUTTING_EDGE_DEFAULT = 1; after a tool change, without programming a cutting edge, the first cutting edge is active	
Corresponding to...		
Additional references:	Description of functions: Tool Offset (W1)	

20272	SUMCORR_DEFAULT	
MD number	Initial state of additive offset without programming	
Default setting: 0	Min. input limit: -1	Max. input limit: 6
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		
Meaning:	The number of the additive offset for the cutting edge that is active when a new cutting-edge offset is activated without a programmed DL value. The MD18110 $MN_MM_MAX_SUMCORR_PERCUTTEDGE determines the maximum meaningful value which can be entered. Value: Meaning > 0: Number of additive offset = 0: No additive offset active with D programming = -1: The additive offset number for the previously programmed D is used.	
Corresponding to...	MD20270 CUTTING_EDGE_DEFAULT	
Additional references:	Description of functions: Tool Offset (W1)	

Note

The output of the DL number is controlled by the MD AUXFU_DL_SYNC_TYPE.

20310	TOOL_MANAGEMENT_MASK	
MD number	Channel-specific activation of tool management functions	
Default setting: 0x0,...	Min. input limit: 0	Max. input limit: 0xFFFFFF
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: HEX
Data type: DWORD		
Meaning:	MD = 0: Tool management inactive Bit 0=1: Tool management active The tool management functions are enabled for the current channel. Bit 1=1: TM monitoring function active Functions for monitoring tools (tool life and workpiece count) are enabled. Bit 2=1: OEM functions active The memory can be used for the user data (see also M 18090 to18098) Bit 3=1: Adjacent location consideration active Bit 0 to bit 3 must be set just like MD18080 MM_TOOL_MANAGEMENT_MASK.	
Meaning:	Bit 4=1: The PLC has the option to request another T preparation with modified parameters. Acknowledgement status "2", "7" and "103" is enabled with this bit. This causes the tool selection to be repeated in the NCK	
	Bits 5 to 8 Bit 5 and bit 7 refer to the main spindle Bit 6 and bit 8 refer to the secondary spindles Bit 5 = 1: The command output is considered completed if the internal transport acknowledgement + the transport acknowledgement are present, i.e. if the command was received by the basic PLC program. Bit 19=1 additionally allows the block change to be prevented (main run) until the required acknowledgements are received. Bit 7 = 1: The command output is not considered completed until the end acknowledgement from the PLC is received, i.e. the command was acknowledged by the PLC user program with status "1". Bit 19=1 additionally allows the block change to be prevented (main run) until the required acknowledgements are received. Bit 5 and bit 7 (alternatively bit 6 and bit 8) are mutually exclusive! Only the following combinations are permitted: Bit 5: …0…1…0 Bit 7: …0…0…1 With the default setting, i.e. bit 5 to 8 = 0, synchronization is performed in the block in which a cutting edge was first selected. Setting these bits delays block processing.	
Meaning:	Bit 9: Reserved for test purposes (no command output to the PLC) can also be used by the machine manufacturer in the test phase, as long as the PLC program does not yet change tools	
Meaning:	Bit 10=1: M06 is delayed until the preparation is taken over by the PLC user program. The change command is only output when the preparation acknowledgement is received. This can be, for example status "1" or "105". Bit 10=0: The change command is output without delay, immediately after the preparation command.	

20310	TOOL_MANAGEMENT_MASK
Meaning:	Bit 11=1: The tool preparation command (PLC command numbers = 2, 4, 5) is carried out even if the same tool preparation command has taken place! (Commands 4, 5 contain the tool preparation) Example: (tool change takes place with M06 (PLC command number=3): T="Tool1"; Tool preparation M06; tool change T="Tool2"; 1st tool preparation after M6 (for the same tool holder) ; is always output to PLC T="Tool2"; 2nd tool preparation, is only output as command to PLC if bit 11 = 1 ; This tool preparation counts as the first one if the status of the tool has changed since the previous tool preparation such that it can no longer be used. A possible reason for this would be, e.g. asynchronous unloading of a tool. This tool preparation them attempts to select a replacement tool. Bit 11=0: The preparation command can be output only once for each tool.
Meaning:	Bit 12=1: The tool preparation command (PLC command numbers = 2, 4, 5) is carried out even if the tool is already positioned in the spindle /tool holder. T="Tool1"; Tool preparation M06; tool change T="Tool1"; Tool is already placed on the tool holder ; 1st tool preparation after M06 (for the same tool holder) ; is only output to PLC if bit 12 = 1 ; A tool that cannot be used (e.g. disabled due to tool monitoring) on the tool holder does not count as if placed on the tool holder. This tool preparation then attempts to select a replacement tool. T="Tool2"; 2nd tool preparation - the following rules applies for bit 11 for output Bit 12=0: The preparation command is not executed if the tool is already positioned in the spindle.
	Bit 13=1: Trace For reset, the command output of the NCK and the acknowledgements of the PLC are saved in a trace file (TCTRAxx.mpf, xx=channel no.). Corresponds with $MN_MM_TOOL_MANAGEMENT_TRACE_SZ..
	Bit 14=1: Reset mode Tool and offset selection according to the settings in MD $MC_RESET_MODE_MASK and $MC_START_MODE_MASK. Bit 14=0: No reset mode

20310	TOOL_MANAGEMENT_MASK
Meaning:	Bit 15=1: The tool is not returned if several preparation commands have been issued (Tx->Tx). Bit 15=0: Tool is returned from possibly defined buffers. Bit 16=1: T=location number is active Bit 16=0: T="Tool name" Bit 17=1: Tool life monitoring is controlled by the PLC (DB21.DBX1.3). Bit 18=1: Activation of monitoring "last tool of tool group" Bit 18 extends the search for a suitable tool, especially if there are many disabled replacement tools. Bit 18=0: No monitoring for "Last tool in tool group" Bit 19=1: The synchronizations defined by bits 5...8 are relative to the main run block, i.e. there is no block change until the required acknowledgements are received Bit 19 in conjunction with bits 5, 6, 7, 8 set delays the block processing. Bit 19=0: The synchronizations defined by bits 5...8 are relative to the tool management command output, i.e. there is no block change delay
Meaning:	Bit 20=0: The commands generated on PLC signal "program testing active" are not output to the PLC. NCK acknowledges the commands automatically. Magazine and tool data is not changed. Bit 20=1: The commands generated for PLC signal "program testing active" are output to the PLC. Depending on the type of acknowledgement, tool/magazine data can be changed in the NCK. If the acknowledgement parameters for the "target magazine" are set to the same values as the "source magazine", the tool is not transported and thus no data modified in the NCK.
	Bit 21=0: Default setting: Ignore the tool status "W" at tool selection Bit 21=1: Tools in status "W" cannot be selected by another tool change, tool preparation command.
	Bit 22=1: "Tool subgroups" function $TC_TP11[x] is the grouping or selection parameter
	Bit 23=0: Default setting The tool management optimally and reliably selects the tool in the main run, i.e. the interpreter may have to wait for the end of the tool selection for offset selection. Bit 23=1: For basic applications The interpreter selects the tool itself, i.e. no synchronization is required with the main run for offset selection. (If the tool becomes no longer useable after selection, but before loading, an uncorrectable alarm may result.)
	Bit 24=0: Default setting If the PLC commands 8 and 9 (asynchronous transfer) want to move a tool to a location that is reserved for another tool, this is rejected and an alarm is issued. Bit 24=1: If the PLC commands 8 and 9 are to move a tool to a location that is reserved for another tool with "Reserved for tool from buffer" (bit values="H4"), this is possible. This location reservation is then removed before the movement is executed ("Reserved for new tool to be loaded" (bit value="H8") remains effective).
Corresponding to...	MD18080 MM_TOOL_MANAGEMENT_MASK MD20320 TOOL_TIME_MONITOR_MASK MD20122 MC_TOOL_RESET_NAME MD20110 MC_RESET_MODE_MASK MD20124 MC_TOOL_MANAGEMENT_TOOLHOLDER MD22560 TOOL_CHANGE_M_CODE
Additional references:	

20320	TOOL_TIME_MONITOR_MASK	
MD number	Time monitoring for tool in the tool holder	
Default setting: 0x0	Min. input limit: -	Max. input limit: -
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: HEX
Data type: DWORD		
Meaning:	Activation of tool time monitoring for tool holder or spindle 1....x. As soon as the path axes are moved (not for G00, always for G63), the tool time monitoring data for the tool which is located in the selected tool holder – which is at the same time the master tool holder – is updated. Bit 0...x-1: Monitoring of active tool in spindle 1...x	
Corresponding to...		
Additional references:	Description of functions: Memory configuration (S7)	

22550	TOOL_CHANGE_MODE	
MD number	New tool offset for M function	
Default setting: 0	Min. input limit: 0	Max. input limit: 1
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: BYTE		
Meaning:	A tool is selected in the program with the T function. The setting in this machine data determines whether the new tool is loaded immediately on execution of the T function: MD: TOOL_CHANGE_MODE = 0 The new tool data becomes effective directly when T or D is programmed. This setting is used mainly for turning machines with tool turret. If there is no D programmed in the block with T, the tool offset which is defined in $MC_CUTTING_EDGE_DEFAULT becomes effective. MD: TOOL_CHANGE_MODE = 1 The new tool is prepared for changing with the T function. This setting is used mainly on milling machines with a tool magazine, in order to bring the new tool into the tool change position without interrupting the machining process. With the M function set in MD 22560: TOOL_CHANGE_MODE, the old tool is removed from the spindle and the new tool is loaded into the spindle. According to DIN 66025, this tool change must be programmed with the M function M06.	
Corresponding to...	MD22560 TOOL_CHANGE_M_CODE	
Additional references:	Description of functions: Tool Offset (W1)	

22560	TOOL_CHANGE_M_CODE	
MD number	M function for tool change	
Default setting: 6	Min. input limit: 0	Max. input limit: 99999999
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		

22560	TOOL_CHANGE_M_CODE
Meaning:	If the T function is used only to prepare a new tool for a tool change (this setting is used mainly on milling machines with a tool magazine, in order to bring the new tool into the tool change position without interrupting the machining process), the tool change must be initiated with an additional M function. The M function entered in TOOL_CHANGE_M_CODE initiates the tool change (remove old tool from the spindle and load the new tool in the spindle). This tool change is required to be programmed with M function M06, in accordance with DIN66025.
Corresponding to...	MD22550 TOOL_CHANGE_MODE
Additional references:	Functional description for tool offset (W1)

22562	TOOL_CHANGE_ERROR_MODE	
MD number	Behavior when errors occur at tool change	
Default setting: 0x0	Min. input limit: 0	Max. input limit: 0xFF
Change becomes effective after: POWER ON	Protection level: 2/7	Unit: -
Data type: DWORD		

22562	TOOL_CHANGE_ERROR_MODE
Meaning:	Behavior in the case that faults/problems occur for a programmed tool change. Bit 0=0: Standard behavior: Stop on faulty NC block Bit 0=1: If the error occurs in the block containing the tool change preparation command, the alarm activated by the preparation command (T) is delayed until the program run reaches the point at which the associated tool change command (M06) is interpreted. Only then is the alarm output that is triggered by the preparation command. The operator can make corrections in this block. When execution of the program is continued, the faulty NC block is interpreted again and the preparatory command is internally executed again automatically. The value = 1 is only of significance if the setting MD22550 TOOL_CHANGE_MODE = 1 is used. Bit 1 is only meaningful if tool management is active. Bit 1=0: Standard behavior: During the tool-change preparation only those tools are recognized whose data is assigned to a magazine. Bit 1=1: Manual tools can be loaded at change. A tool will also be loaded at change if its data is registered in the NCK, but not assigned to a magazine. In this case, the tool data is automatically assigned to the programmed tool holder. The user is prompted to place tools into the tool holder or remove tools from it. Bit 2 Qualifying the offset programming Bit 2=0: Active D no. > 0 and active D no.=0 results in offset 0 Active D no. > 0 and active D no.=0 result in additive offset 0 Bit 2=1: Active D no. > 0 and active D no.=0 generates an alarm message Active D no. > 0 and active D no.=0 generates an alarm message Bits 3 and 4 are only meaningful if tool management is active. Function: Control of behavior of Init block generation at program start if disabled tool is positioned in the spindle and needs to be activated. For further details, see: MD 20112: START_MODE_MASK, MD 20110: RESET_MODE_MASK At RESET the behavior of "keep disabled tool on spindle active" is not affected by this. Bit 3=0: Default: If the tool on the spindle is disabled: Generate a tool-change command that requests a replacement tool. If there is no replacement, then an alarm is produced. Bit 3=1: The disabled status of the spindle tool is ignored. The tool is active. The following part program should be formulated such that no parts are machined with the disabled tool. Bit 4=0: Default: An attempt is made to activate the spindle tool or the replacement tool Bit 4=1: If the tool on the spindle is disabled, T0 is programmed in the start init. block. The following statements are made for the combination of bit 3 and bit 4: 0 / 0: Behavior as before, automatic change at NC start if the disabled tool is in the spindle 1 / 0: Is not changed automatically 0 / 1: A T0 is generated automatically for a disabled tool in the spindle at NC Start 1 / 1: No statement

22562	TOOL_CHANGE_ERROR_MODE
	Bit 5: Reserved Bit 6=0: Default: with T0 or D0 then precisely only T0 or D0 are programmed. In other words, the MD $MC_CUTTING_EDGE_DEFAULT, $MC_SUMCORR_DEFAULT define the value of D, DL when T0 is programmed. For example, $MC_CUTTING_EDGE_DEFAULT=1, $MC_SUMCORR_DEFAULT=2, $MC_TOOL_CHANGE_MODE=0 (tool change with T programming) N10 T0; T no. 0 has active number D1 and DL=2 which results in an offset of zero. If, in addition, bit 2 is also set: Programming of a) T0; to deselect tool b) D0; for offset deselection generates an alarm, if a) At least one of MD $MC_CUTTING_EDGE_DEFAULT, $MC_SUMCORR_DEFAULT is not equal to zero (T0 D0 DL=0 is the correct programming). b) The MD $MC_SUMCORR_DEFAULT is not equal to zero (D0 DL=0 is the correct programming). Bit 6=1: Controls the NCK behavior with programming of (x, y, z all greater than zero), if at least one of MD $MC_CUTTING_EDGE_DEFAULT, $MC_SUMCORR_DEFAULT is not equal to zero. a) Tx Dy -> T0 with T0 - D0 or D0 DL=0 is automatically programmed in the NCK; i.e. values not equal to zero for MD $MC_CUTTING_EDGE_DEFAULT, $MC_SUMCORR_DEFAULT are processed as if the value were equal to zero. b) Tx Dy >T0 Dy, or T0 DL =z, or T0 Dy DL=z, or T0 D0 DL=z explicitly programmed values of D, DL are not influenced. c) Dy DL=z -> D0 with D0 - DL=0 is automatically programmed in the NCK; i.e. values not equal to zero for MD $MC_SUMCORR_DEFAULT are processed as if the value were equal to zero. d) Dy DL=z -> D0 DL=z explicitly programmed values of DL are not influenced. If in addition bit 2 is also set: only T0/D0 have to be programmed for tool/offset deselection and therefore no alarm is output. The statements regarding $MC_SUMCORR_DEFAULT or DL are only valid if the additive offset function is active (see $MN_MM_TOOL_MANAGEMENT_MASK, bit 8).

22562	TOOL_CHANGE_ERROR_MODE
	Bit 7=0: Programming Tx checks whether a tool with the T number x is known in the TO unit of the channel. If it is not, processing stops in this block and alarm 17190 is issued. Bit 7=1: Only if tool basic functionality is active ($MC_TOOL_MANAGEMENT_MASK, bit 0.1=0) and ($MN_MM_TYPE_OF_CUTTING_EDGE=0): If Tx is programmed, any unknown Tx is first ignored and the alarm for the preparation command (Tx) is ignored until D selection is interpreted in the program execution. Only then is alarm 17191 output that was triggered by the preparation command. This means that in this block, the operator could use the D selection to make corrections. When execution of the program is continued, the faulty NC block is interpreted again and the preparatory command is internally executed again automatically. (It is interesting for cutting edge default=0 or =-2 or D0 programming, otherwise when changing the tool, the D from cutting edge default is deselected.) This variant can be required if you want to program "Tool number=Location" (turret as tool holder) without tool management. The turret can only be positioned at a location for which there is no tool defined (yet). If bit 0=1 is set, this bit is irrelevant. Bit 8=0: A tool that is on a disabled magazine location is not taken into account for the tool selection, i.e. it cannot be selected (default setting). Bit 8=1: Also a tool that is on a disabled magazine location is taken into account for the tool selection (corresponds to earlier behavior).
Additional references:	Description of functions: Tool Offset (W1)

28085	MM_LINK_TOA_UNIT	
MD number	Allocation of a TO unit to a channel	
Default setting: 1, 2, 3, 4, 5, ...	Min. input limit: 1	Max. input limit: 10
Change becomes effective after: POWER ON	Protection level: 2/7	Unit:
Data type: DWORD		
Meaning:	The area T0 includes all tool, magazine, ... data blocks, which the NCK knows. The maximum number of units in the TO area match the maximum number of channels. If MM_LINK_TOA_UNIT = default, then each channel is individually assigned a TO unit. The channel is assigned to the TO unit i with MM_LINK_TOA_UNIT = i. It is thus possible to assign one TO unit to several channels. Notice The upper limit value does not imply that the value is always meaningful or without conflict. If one channel (the first) is active and the other not on a system with a total of 2 channels, the MD on channel 1 can be formally set to a value of 2. However, the NCK cannot work with this setting. This setting would mean that channel 1 possesses no data blocks for tool offsets since a channel with Id=2 does not exist. The NCK detects this conflict during power-on or a warm restart and reacts by independently changing the (incorrect) setting to the default setting for the MD.	
Corresponding to...		
Additional references:	Description of functions: Memory configuration (S7)	

7.3 Machine data for function replacement

10715	M_NO_FCT_CYCLE		
MD number	M function to be replaced by a subprogram		
Default setting: -1	Min. input limit: -	Max. input limit: -	
Change valid after POWER ON	Protection level: 2/4	Unit: -	
Data type: DWORD			
Meaning:	M number with which the subprogram is called. The name of the subprogram is stored in $MN_M_NO_FCT_CYCLE_NAME[n]. If the M function defined using $MN_M_NO_FCT_CYCLE[n] is programmed in a part program block, then the subprogram defined in M_NO_FCT_CYCLE_NAME[n] is started at the end of block. If the M function is re-programmed in the subprogram, then there is no longer any replacement by a subprogram call. $MN_M_NO_FCT_CYCLE[n] acts both in the Siemens mode G290 as well as in the external language mode G291. The subprograms configured with $MN_M_NO_FCT_CYCLE_NAME[n] and $MN_T_NO_FCT_CYCLE_NAME may not be effective in one block (part program line) at the same time, i.e. max. one M/T function replacement can be effective per block. Neither an M98 nor a modal subprogram call may be programmed in the block with the M function replacement. A subprogram return jump or end of part program is also not permitted. Alarm 14016 is generated in the case of a conflict. Constraints: M functions with a fixed meaning and configurable M functions are checked for competing settings. A conflict is signaled with an alarm. The following M functions are checked: - M0 to M5, - M17, M30, - M40 to M45, - M function for selecting spindle/axis mode according to $MC_SPIND_RIGID_TAPPING_M_NR (default M70) - M functions for nibbling/punching as configured in $MC_NIBBLE_PUNCH_CODE if activated by $MC_PUNCHNIB_ACTIVATION. - for applied external language ($MN_MM_EXTERN_LANGUAGE) M19, M96-M99. Exception: The M functions defined by $MC_TOOL_CHANGE_M_CODE for the tool change.		
Corresponding to...			
Additional references:	ISO dialects for SINUMERIK (FBFA)		

10716	M_NO_FCT_CYCLE_NAME		
MD number	Subprograms for M function replacement		
Default setting: -	Min. input limit: -	Max. input limit: -	
Changes effective after Power On	Protection level: 2/4	Unit: -	
Data type: STRING			

10716	M_NO_FCT_CYCLE_NAME
Meaning:	The cycle name is stored in the machine data. This cycle is called if the M function was programmed from machine datum $MN_M_NO_FCT_CYCLE. If the M function is programmed in a motion block, then the cycle is executed after the motion. $MN_M_NO_FCT_CYCLE acts both in the Siemens mode G290 as well as in the external language mode G291. If a T number is programmed in the calling block, the programmed T number can be queried in the cycle under the variable $P_TOOL. M and T function replacement may not be effective in one block at the same time, i.e. max. one M/T function replacement can be effective per block. Neither an M98 nor a modal sub-program call may be programmed in the block with the M function replacement. A subprogram return jump or end of part program is also not permitted. Alarm 14016 is generated in the case of a conflict.
Corresponding to...	MD10715 M_NO_FCT_CYCLE MD10717 T_NO_FCT_CYCLE_NAME
Additional references:	ISO dialects for SINUMERIK (FBFA)

10717	T_NO_FCT_CYCLE_NAME	
MD number	Name of tool change cycle for T function replacement	
Default setting: -	Min. input limit: -	Max. input limit: -
Changes effective after Power On	Protection level: 2/4	Unit: -
Data type: STRING		
Meaning:	Cycle name for tool change routine during call via T function. If a T function is programmed in a part program block, then the subprogram defined in t_NO_FCT_CYCLE_NAME is called at the end of block. The programmed T number can be queried in the cycle via the system variables $C_T/ $C_T_PROG as decimal value and via $C_TS / $C_TS / $C_TS_PROG as string (only in tool management). $MN_T_NO_FCT_CYCLE_NAME acts both in the Siemens mode G290 as well as in the ex-ternal language mode G291. $MN_M_NO_FCT_CYCLE_NAME and $MN_T_NO_FCT_CYCLE_NAME may not be effec-tive in one block at the same time, i.e. max. one M/T function replacement can be effective per block. Neither an M98 nor a modal subprogram call may be programmed in the block with the T function replacement. A subprogram return jump or end of part program is also not permitted. Alarm 14016 is generated in the case of a conflict.	
Corresponding to...	MD10715 M_NO_FCT_CYCLE MD10716 M_NO_FCT_CYCLE_NAME	
Additional references:	ISO dialects for SINUMERIK (FBFA)	

10718	M_NO_FCT_CYCLE_PAR	
MD number	M function replacement with parameters	
Default setting: -1	Min. input limit: -	Max. input limit: -
Changes effective after Power On	Protection level: 7/2	Unit: -
Data type: DWORD		

10718	M_NO_FCT_CYCLE_PAR
Meaning:	If a M function replacement was configured with $MN_M_NO_FCT_CYCLE[n] / $MN_M_NO_FCT_CYCLE_NAME[n], then $MN_M_NO_FCT_CYCLE_PAR can be used for specifying parameter transfer for one of these M functions per system variable as is the case for the T function replacement. The parameters stored in the system variables always refer to the part program line in which the M function to be replaced is programmed. The following system variables are available: $C_ME: Address extension of the substituted M function $C_T_PROG: TRUE if address T has been programmed $C_T: Value of address T (integer) $C_TE: Address extension of address T $C_TS_PROG: TRUE if address TS has been programmed $C_D_PROG: TRUE if address D has been programmed $C_D: Value of address D $C_DL_PROG: TRUE if address DL has been programmed $C_DL: Value of address DL
Corresponding to...	
Additional references:	ISO dialects for SINUMERIK (FBFA)

10719	T_NO_FCT_CYCLE_MODE	
MD number	Parameter assignment for T function replacement	
Default setting: 0	Min. input limit: 0	Max. input limit: 7
Changes effective after Power On	Protection level: 7/2	Unit: -
Data type: DWORD		
Meaning:	Processing of the substitution program is parameterized in this MD for the tool selection/tool offset selection. Bit 0 = 0: The D or DL number is transferred to the substitution program (default value) Bit 0 = 1: The D or DL number is not transferred to the substitution program if the following conditions are fulfilled: $MC_TOOL_CHANGE_MODE = 1 programming of D/DL with T or M function with which the tool change cycle is called, in one part program line Bit 1 = 0: Execution of substitution subprogram at end of block (default value) Bit 1 = 1: Processing of substitution subprogram at start of block Bit 2 = 0: Processing of substitution subprogram according to setting for bit 1 Bit 2 = 1: Execution of substitution subprogram at start of block and end of block	
Corresponding to...		
Additional references:	ISO dialects for SINUMERIK (FBFA)	

11717	D_NO_FCT_CYCLE_NAME	
MD number	Subprogram name for D function replacement	
Default setting: -	Min. input limit: -	Max. input limit: -

11717	D_NO_FCT_CYCLE_NAME	
Changes effective after Power On	Protection level: 7/2	Unit: -
Data type: STRING		
Meaning:	Cycle name for D function replacement routine. If a D function is programmed in a part program block, the subprogram defined with $MN_D_NO_FCT_CYCLE_NAME is called in accordance with the machine data $MN_T_NO_FCT_CYCLE_NAME, $MN_T_NO_FCT_CYCLE_MODE and $MN_M_NO_FCT_CYCLE_PAR. The programmed D number can be queried in the cycle via the system variables $C_D / $C_D_PROG. $MN_D_NO_FCT_CYCLE_NAME can only run in Siemens mode (G290). A maximum of one M/T/D function replacement can be effective for each part program line. A modal subprogram call may not be programmed in the block with the D function replacement. It is also not permissible to program a subprogram return jump or end of part program. Alarm 14016 is generated in the case of a conflict.	
Corresponding to...		
Additional references:	ISO dialects for SINUMERIK (FBFA)	

7.4 Machine data for the Siemens user data

The numbers of the Siemens machine data are listed in the following. This data is defined by Siemens and must not be used by users. No detailed description of them is given for this reason.

18200

18201

18202

18203

18204

18205

18206

18207

18208

18209

Note

A detailed description of machine data 18091, 18093, 18095, 18097 and 18099 has been provided, but these MD may be used only if they are set to their respective defaults.

Multitool machine data

18196

18197

18198

18199

Signal description of the PLC interface 8

8.1 Overview of data blocks

General

The table below shows an overview of the data blocks used for data management.

DB71	For loading/unloading points
DB1071	Extended data for multitools
DB72	For spindle as change position
DB1072	Extended data for multitools
DB73	For turret as change position
DB1073	Extended data for multitools
DB74	Internal data block of basic program for tool management

1. There is a separate interface area in DB71 for each loading point configured in the magazine configuration. The interface area for loading point 1 generally has the task of loading into the spindle.
   Relocation- and positioning tasks are always handled via loading point_1 (spindle loading point).
2. DB72 contains a separate interface area for every spindle defined in the tool management function.
3. A separate interface area is provided in DB73 for each turret of the magazine configuration. The turret numbers are numbered without any gaps from the lowest up to the highest magazine number.
4. Data for multitools is available in data blocks DB1071, DB1072 and DB1073 (analog to DB71, DB72 and DB73 loading and unloading, spindle change positions, turret).

All interfaces are designed for receiving tool-management commands (load, tool change, ...). Basic program blocks FC6 (multitool), FC7 and FC 8 are used to communicate the current positions of tools.

One of the interfaces is updated by NCK via the basic program in accordance with a command (e.g. by operating the function "Load" or by a part-program function like "Tool change").

Note

The PLC must also be adapted if data of magazines, buffers or loading/unloading positions is changed in the commissioning branch.

Either automatically after the next Power On by changing DB4 or with HMI-Advanced.

With the next restart, the basic PLC program automatically clears DB71 ... DB73, DB1071 ... 1073 and DB74 and creates the blocks again.

8.2 Interface for loading/unloading magazine

DB71 Data block	**Signals of loading/unloading points NCK -> PLC interface**							
Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	Interfaces							
DBB0	INT 8	INT 7	INT 6	INT 5	INT 4	INT 3	INT 2	INT 1
DBB1	INT 16	INT 15	INT 14	INT 13	INT 12	INT 11	INT 10	INT 9
DBB2	Ackn INT 8	Ackn INT 7	Ackn INT 6	Ackn INT 5	Ackn INT 4	Ackn INT 3	Ackn INT 2	Ackn INT 1
DBB3	Ackn INT 16	Ackn INT 15	Ackn INT 14	Ackn INT 13	Ackn INT 12	Ackn INT 11	Ackn INT 10	Ackn INT 9
DBBn+0	Res.		Positioning a multitool	Positioning from the NC program	Positioning a multitool	Relocating	Unloading	Loading
DBBn+1	Data in the extended range (DB 1071)	Res.	Res.	Res.	Res.	Res.	Res.	Acknowledgement status = 3
DBBn+2	Assigned channel (8bit-Int)							
DBBn+3	Tool management number (8bit-Int)							
DB n+4	Reserved							
DBDn+8	Reserved							
DBDn+12	Reserved							
DBWn+16	Identifier for loading/unloading point (Int), (fixed value 9999)							
DBWn+18	Location no. of loading/unloading point (Int)							
DBWn+20	Magazine no. (Source) for unloading/relocating/positioning (Int)							
DBWn+22	Location no. (Source) for unloading/relocating/positioning (Int)							
DBWn+24	Magazine no. (Target) for unloading/relocating/positioning (Int)							
DBWn+26	Location no. (Target) for unloading/relocating/positioning (Int)							
DBWn+28 HMI to PLC								Loading/unloading without moving magazine
DBWn+29	Spare							

Starting addresses of loading/unloading points:

Loading/unloading point 1: n = 4

Loading/unloading point 2: n = 34

Loading/unloading point 3: n = 64

Loading/unloading point 4: n = 94

Example of calculating address DBWn+24 (target magazine no.)

n=(m-1)*len+4 m = location no. of loading station/point

len = 30 (length of loading point)
m = 2; len = 30 n= (2-1)*30+4 --> n = 34
DBW (34+24) = DBW 58

Address for magazine no. target of 2nd loading point is DBW58.

Loading point 1 is intended for loading/unloading in all spindles. Loading point is also used to relocate/position tools in any location (e.g. buffer location).

DB71.DBX0.0 - 1.7	**Active status of interface 1- 16**
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	The active interface has a valid data record. A task bit has been set in DBB (n+0). There are 16 interfaces. Address "n" must always be calculated for the active interface.
Signal state 0	Operation for this interface has ended. Is reset by FC8 or FC6.

DB71.DBX2.0 - 3.7	**"auto" acknowledgement of interface 1 - 16**
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	When setting (edge) the bit, the pending command is acknowledged with status_1. Unless it is a relocation operation from a real magazine to a buffer. Then acknowledgement is with status_6. This reserves the source location. For example U DB71.DBX0.0 U DB71.DBX4.1 (unloading) S DB71.DBX2.0
Signal state 0	Operation for this interface has ended. Is reset by the basic program.

DB71.DBX(n+0).0	**Command: Loading**
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	Loading operation for a tool is initiated. The magazine location into which the tool is to be loaded is defined in DBW(n+26). The loading point in question is the location number of the loading point. It is also located in DBW(n+18)
Corresponding to...	DB71DBX(n+16) and (n+18) and (n+26)

DB71.DBX(n+0).1	**Command: Unloading**
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	Unloading operation is initiated for a tool; the magazine location from which it is to be unloaded is in DBW (n+20) and DBW (n+22). The number of the unloading point is in DBW(n+18).
Corresponding to...	DB71DBX(n+16) and (n+18) or (n+20) and (n+22)

DB71.DBX(n+0).2	Command: Relocating
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	Relocate operation for a tool is initiated. From magazine/location (n +20, n+22=source) to magazine/location (n+24, n+26=target)

Note

The bits in DBB(n+0) (loading, unloading,....) are not updated by the basic program until a new task exists for this interface. They are current only if the corresponding interface bit in DBB0 is set to "1". If required, the user can reset bits DBB(n+0).

DB71.DBX(n+0).3	Command: Positioning to the loading point
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	A magazine location is to be positioned at the loading point (magazine no. 9999). The magazine location that is to be moved to the loading point is in DB71.DBW(n+20) and (n+22). The loading point is in DB71.DBWn+18.

DB71.DBX(n+0).4	Command: Task comes from the NC program
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	The signal is set if the task is received from the part program or via a cycle. 1. Positioning task with the language command POSM 2. A relocation task or tool transport using the language command MVTOOL 3. Positioning a multitool using the language command POSMT

DB71.DBX(n+0).5	Command: Positioning a multitool
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	Positioning a multitool has been initiated. This can be realized using a language command, PI service or the HMI user interface.

DB71.DBX(n+1).0	"auto" ackn negative
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	The bit is only evaluated in conjunction with "auto" ackn. If it is set, then the auto acknowledge is negative, i.e. with status_3.

DB71.DBX(n+1).7	Command: Data in the extended range
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	The signal is set if a multitool is loaded, unloaded or positioned. The multitool data is then available in DB 1071.

DB71.DBB(n+2)	Assigned channel
Edge evaluation	Signal(s) updated: Conditional
Meaning	Number of the channel for which the active interface is valid.

DB71.DBB(n+3)	Tool management no.
Edge evaluation	Signal(s) updated: Conditional
Meaning	Associated tool management number; corresponds to the number of the TO unit within a TO area.

DB71.DBW(n+16)	Identifier for loading/unloading point (fixed value 9999)
Edge evaluation	Signal(s) updated: Conditional
Meaning	The identifier for the loading/unloading point is fixed to the value 9999.

DB71.DBW(n+18)	Location no. of the loading/unloading point
Edge evaluation	Signal(s) updated: Conditional
Meaning	The location no. of the loading/unloading point is displayed.

DB71.DBW(n+20)	Magazine no. (Source) for unloading/relocating/positioning
Edge evaluation	Signal(s) updated: Conditional
Meaning	Unloading: Magazine from which the tool is to be unloaded Relocating: Magazine from which the tool comes Positioning: Magazine that is to be positioned
Corresponding to...	DBW(n+22)

DB71.DBW(n+22)	Location no. (Source) for unloading/relocating/positioning
Edge evaluation	Signal(s) updated: Conditional
Meaning	Unloading: Location from which the tool is to be unloaded Relocating: Location from where the tool comes Positioning: Location that should be positioned
Corresponding to...	DBW(n+20)

DB71.DBW(n+24)	**Magazine no. (Target) for unloading/relocating/positioning**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Loading: Magazine into which the tool is to be loaded Relocating: Magazine into which the tool comes Positioning: Magazine to be positioned to Tool remains in the original magazine Only meaningful for interface 1. If values other than 0 are entered here, the data defines the magazine or location for positioning (language command POSM).
Corresponding to...	DBW(n+26)

DB71.DBW(n+26)	**Location no. (Target) for unloading/relocating/positioning**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Loading: Location into which the tool is to be loaded Relocating: Location into which the tool comes Positioning: Location to be positioned to Tool remains at the original location Only meaningful for interface 1. If values other than 0 are entered here, the data define the magazine or location for positioning (language command POSM).
Corresponding to...	DBW(n+24)

DB71.DBX(n+28) Bit0	**Loading/unloading without moving magazine**
Edge evaluation	Signal(s) updated: Conditional
Meaning	HMI / Jobshop sets/deletes this signal when requested by the operator. If the bit is active, there must be no traversing motion of the magazine, only a mechanical unlocking/locking of the location. The loading/unloading command must be acknowledged after the action. For a positioning and relocating request, this signal is not valid for a traversing motion.

8.3 Interface for spindle as change position

DB72 data block	**Spindle as change position Interface NCK -> PLC**							
Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	Interfaces							
DBB0	INT 8	INT 7	INT 6	INT 5	INT 4	INT 3	INT 2	INT 1
DBB1	INT 16	INT 15	INT 14	INT 13	INT 12	INT 11	INT 10	INT 9
DBB2	Ackn INT 8	Ackn INT 7	Ackn INT 6	Ackn INT 5	Ackn INT 4	Ackn INT 3	Ackn INT 2	Ackn INT 1
DBB3	Ackn INT 16	Ackn INT 15	Ackn INT 14	Ackn INT 13	Ackn INT 12	Ackn INT 11	Ackn INT 10	Ackn INT 9
DBB(n+0)	Tool remains in spindle	Remove manual tool	Load manual tool at change	OldTool in BL no. (n +42)	T0	Prepare change	Change tool (initiated by M06)	Obligatory change
DBB(n+1)	Data in the extended range (DB 1072)	Res.	Res.	Res.	Res.	Res.	Res.	Acknowledgement status = 3
DBB(n+2)	Assigned channel (8bit-Int)							
DBB(n+3)	Tool management number (8bit-Int)							
DBD(n+4)	$P_VDITCP[0] User-definable parameter 0 (DWord)							
DBD(n+8)	$P_VDITCP[1] User-definable parameter 1 (DWord)							
DBD(n+12)	$P_VDITCP[2] User-definable parameter 2 (DWord)							
DBW(n+16)	Buffer identifier (Int), (fixed value 9998) corresponds to "Target position for new tool"							
DBW(n+18)	Relative location (target) in buffer magazine (Int)							
DBW(n+20)	Magazine no. (Source) for new tool (Int)							
DBW(n+22)	Location no. (Source) for new tool (Int)							
DBW(n+24)	Magazine no. (Target) for old tool (Int)							
DBW(n+26)	Location no. (Target) for old tool (Int)							
DBW(n+28)	New tool: Location type (Int)							
DBW(n+30)	New tool: Size on left (Int)							
DBW(n+32)	New tool: Size on right (Int)							
DBW(n+34)	New tool: Size at top (Int)							
DBW(n+36)	New tool: Size at bottom (Int)							
DBB(n+38)	Tool status for new tool							
	Manual tool	1:1 exchange		Master tool	Tool to be loaded	Tool to be unloaded	Disabled, but ignore	Identifier for tools in the buffer
DBB(n+39)	Tool status for new tool							
	Tool was in use	Tool, fixed-location-coded	Tool is being changed	Prewarning limit reached	Measure tool	Tool disabled	Enable tool	Active tool

DB72 data block	Spindle as change position Interface NCK -> PLC
DBW(n+40)	New tool: Internal T number of NCK (Int)
DBW(n+42)	If DBX (n+0.4) = 1, then the buffer location of the old tool must be entered here
DBW(n+44)	Original magazine of new tool
DBW(n+46)	Original location of the new tool

Start address of the spindle:

Spindle 1: n = 4

Spindle 2: n = 52

Spindle 3: n = 100

n = (m-1)*len + 4

m = location number of the change position

len = 48

Note

If only M06 is programmed, only free parameters, channel, tool management number and the bit for "Perform change" are updated.

DB72.DBX0.0 - 1.7	Active status of interface 1- 16
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	Associated interface has a valid data record, a tool change request of tool preparation has been initiated.
Signal state 0	Operation for this interface has ended. Is reset by FC 8/FC 6.

DB72.DBX2.0-3.7	"auto" acknowledgement of interface 1-16
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	When setting (edge) the bit, the pending command is acknowl-edged with status_1.
Signal state 0	Operation for this interface has ended. Is reset by the basic program.

DB72.DBX(n+0).0	Command code: Obligatory change
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	The new tool is fixed-location-coded.

DB72.DBX(n+0).1	Command code: Perform change with M06
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	M06 command was programmed for tool change, the tool change can now take place

DB72.DBX(n+0).2	Command code: Prepare change
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	Prepare new tool for change. If necessary, move location for old tool to spindle.

DB72.DBX(n+0).3	Command code: T0
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	Indicates that T0 has been programmed (no-load traversing of spin- dle).

DB72.DBX(n+0).4	Command code: Old tool in buffer
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	The buffer number of the tool to be changed is written in DB72.DBW (n+42)

DB72.DBX(n+0).5	Command code: Load manual tool
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	A manual tool is to be loaded. The HMI displays the tool which is to be loaded.

DB72.DBX(n+0).6	Command code: Unload manual tool
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	The tool is to be changed via manual operation.

DB72.DBX(n+0).7	Command code: Tool remains in spindle
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	The bit is set at change from spindle to spindle. Initiated, e.g. by reset and start mode or block search.

Note

The bit in DBB (n+0).2 (prepare change) is not reset by the system with a change command. The bits in DBB(n+0) ... are current only if the corresponding interface bit in DBB0 is set to "1". However, the user can reset the bits when required.
If DBX(n+0).1 and DBX(n+0).2 are present at the same time, it means that T and M06 were programmed in one block.

DB72.DBX(n+1).0	"auto" ackn negative
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	The bit is only evaluated in conjunction with "auto" ackn. If it is set, then the auto acknowledge is negative, i.e. with status_3.

DB72.DBX(n+1).7	Command: Data in the extended range
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	The signal is set if a tool is prepared in a multitool or is loaded into a multitool. DB 72 contains the data of the multitool, the data of the selected tool is available in DB 1072. For T0 the signal is not set.

DB72.DBB(n+2)	Assigned channel
Edge evaluation	Signal(s) updated: Conditional
Meaning	Number of channel for which active interface applies.

DB72.DBB(n+3)	Tool management no.
Edge evaluation	Signal(s) updated: Conditional
Meaning	Associated tool management no. (TO area)

DB72.DBD(n+4)	User-definable parameter 0 (DInt)
Edge evaluation	Signal(s) updated: Conditional
Meaning	If you need to send a value to the PLC via the part program, the transfer can be programmed with $P_VDITCP[0];. Values are transferred at T call.

DB72.DBD(n+8)	User-definable parameter 1 (DInt)
Edge evaluation	Signal(s) updated: Conditional
Meaning	If you need to send a value to the PLC via the part program, the transfer can be programmed with $P_VDITCP[1];.

DB72.DBD(n+12)	User-definable parameter 2 (DInt)
Edge evaluation	Signal(s) updated: Conditional
Meaning	If you need to send a value to the PLC via the part program, the transfer can be programmed with $P_VDITCP[2]

DB72.DBW(n+16)	Buffer magazine no. (fixed value 9998) Target position for new tool
Edge evaluation	Signal(s) updated: Conditional
Meaning	Magazine no. 9998 (buffer magazine); target magazine for new tool.

DB72.DBW(n+18)	Location in buffer magazine (spindle)
Edge evaluation	Signal(s) updated: Conditional
Meaning	Location no. of buffer magazine to which the new tool must be loaded. This is normally the spindle. The location number defined for this particular buffer during commissioning is output.

DB72.DBW(n+20)	Magazine no. (source) for new tool to be loaded
Edge evaluation	Signal(s) updated: Conditional
Meaning	No. of magazine from which the new spindle tool comes.
Corresponding to ...	DBW(n+22)

DB72.DBW(n+22)	Location no. (source) for new tool
Edge evaluation	Signal(s) updated: Conditional
Meaning	Location no. of the magazine from which the new spindle tool comes.
Corresponding to ...	DBW(n+20)

DB7.2DBW(n+24)	Magazine no. (target) for old tool to be removed
Edge evaluation	Signal(s) updated: Conditional
Meaning	Number of magazine in which the tool to be removed at change will be placed.
Corresponding to ...	DBW(n+26)

DB72.DBW(n+26)	Location no. (target) for new tool
Edge evaluation	Signal(s) updated: Conditional
Meaning	Magazine location for tool that is unloaded at change.

DB72.DBW(n+28)	New tool: Location type
Edge evaluation	Signal(s) updated: Conditional
Meaning	The location type of the new spindle tool is entered here.
Corresponding to ...	Tool size: Left, right, top, bottom.

DB72.DBW(n+30)	New tool: Size left
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size **left** in half locations for the new spindle tool.

DB72.DBW(n+32)	New tool: Size right
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size **right** in half locations for the new spin-dle tool.

DB72.DBW(n+34)	New tool: Size top
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size **top** in half locations for the new spindle tool.

DB72.DBW(n+36)	New tool: Size bottom
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size **bottom** in half locations for the new spindle tool.

DB72.DBW(n+38)	Tool status for new tool
Edge evaluation	Signal(s) updated: Conditional
Meaning	Bit 0: = Active tool Bit 1: Enable tool Bit 2: Tool disabled Bit 3: Measure tool Bit 4: Prewarning limit reached Bit 5: Tool is being changed Bit 6: Tool is fixed-location-coded Bit 7: Tool was in use Bit 8: Tool in the buffer Bit 9: Ignore disabled Bit 10: To be unloaded Bit 11: To be loaded Bit 12: Master tool Bit 13: Reserved Bit 14: 1:1 exchange Bit 15: Manual tool

DB72.DBW(n+40)	New tool: Internal T number of NCK
Edge evaluation	Signal(s) updated: Conditional
Meaning	Display of internal T number of NCK for the new spindle tool.
Signal state 0	.

DB72.DBW(n+42)	Buffer location of old tool
Edge evaluation	Signal(s) updated: Conditional
Meaning	If DB72.(n+0.4) = 1, the buffer location of the old tool must be entered here. This can be any buffer (also a gripper).

DB72.DBW(n+44)	Original magazine of new tool
Edge evaluation	Signal(s) updated: Conditional
Meaning	Owner magazine of the new tool Corresponds to NCK variables $A_MYMN[T-No] If the new tool is located in the magazine, then this value is identical with DB72.DBW(n+20). If the new tool is located in the buffer (e.g. gripper), then the magazine no. is entered here from where the tool originally came.

DB72.DBW(n+46)	Original location of new tool
Edge evaluation	Signal(s) updated: Conditional
Meaning	Owner location of the new tool Corresponds to NCK variable $A_MYMLN[T-No]
Signal state 0	.

8.4 Interface for turrets as change position

DB73 data block	Turret as change position Interface NCK -> PLC							
Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	Interfaces							
DBB0	INT 8	INT 7	INT 6	INT 5	INT 4	INT 3	INT 2	INT 1
DBB1	INT 16	INT 15	INT 14	INT 13	INT 12	INT 11	INT 10	INT 9
DBB2	Ackn INT 8	Ackn INT 7	Ackn INT 6	Ackn INT 5	Ackn INT 4	Ackn INT 3	Ackn INT 2	Ackn INT 1
DBB3	Ackn INT 16	Ackn INT 15	Ackn INT 14	Ackn INT 13	Ackn INT 12	Ackn INT 11	Ackn INT 10	Ackn INT 9
DBB(n+0)	Res.	Remove manual tool	Res.	Res.	T0	Res.	Perform change, initiation: T no.	Obligatory change
DBB(n+1)	Data in the extended range (DB1073)	Res.	Res.	Res.	Res.	Res.	Res.	Acknowledgement status = 3
DBB(n+2)	Assigned channel (8bit-Int)							
DBB(n+3)	Tool management number (8bit-Int)							
DBD(n+4)	$P_VDITCP[0] User-definable parameter 0 (DWord)							
DBD(n+8)	$P_VDITCP[1] User-definable parameter 1 (DWord)							
DBW(n+12)	$P_VDITCP[2] User-definable parameter 2 (DWord)							
DBW(n+16)	Reserved							
DBW(n+18)	Reserved							
DBW(n+20)	Magazine no. of turret (Int)							
DBW(n+22)	Location no. of new tool (Int)							
DBW(n+24)	Reserved							
DBW(n+26)	Location no. of old tool (Int)							
DBW(n+28)	New tool: Location type (Int)							
DBW(n+30)	New tool: Size on left (Int)							
DBW(n+32)	New tool: Size on right (Int)							
DBW(n+34)	New tool: Size at top (Int)							
DBW(n+36)	New tool: Size at bottom (Int)							
DBB(n+38)	Tool status for new tool							
	Manual tool	1:1 exchange		Master tool	Tool to be loaded	Tool to be unloaded	Disabled, but ignore	Identifier for tools in the buffer
DBB(n+39)	Tool status for new tool							
	Tool was in use	Tool, fixed-location-coded	Tool being changed	Prewarning limit reached	Measure tool	Tool disabled	Enable tool	Active tool

DB73 data block	Turret as change position Interface NCK -> PLC
DBW(n+40)	New tool: Internal T number of NCK (Int)
DBW(n+42)	Original location of new tool in this circular magazine

Start address of the turret:

Turret, 1: n = 4

Turret 2: n = 48

Turret 3: n = 92

n = (m-1)*len + 4

m = location no. of the change position

len = 44

Example for change position 3: n = (3-1)*n44 + 4 = 2*44 + 4= 88 + 4 = 92

DB73.DBX0.0 - 1.7	Active status of interface 1- 16
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	Associated interface has a valid data record.
Signal state 0	Operation for this interface has ended. Is reset by FC 7.

DB73.DBX2.0 - 3.7	"auto" acknowledgement of interface 1-16
Edge evaluation	Signal(s) updated: Conditional
Meaning	When setting (edge) the bit, the pending command is acknowledged with status_1.
Signal state 0	Operation for this interface has ended. Is reset by the basic program.

DB73.DBX (n+0).0	Command code: Obligatory change
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	
Signal state 0	
Corresponding to ...	Position of participating tools

DB73.DBX(n+0).1	Command code: Perform change
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	Execute tool change

DB73.DBX(n+0).3	T0
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	Indicates that T0 has been programmed

DB73.DBX(n+1).7	Command: Data in the extended range
Edge evaluation	Signal(s) updated: Conditional
Signal state 1	The signal is set if a tool is loaded into a multitool. DB 73 contains the data of the multitool, the data of the selected tool is available in DB 1073. For T0 the signal is not set.

DB73.DBX(n+1).30	"auto" ackn negative
Edge evaluation	Signal(s) updated: Conditional
Meaning	The bit is only evaluated in conjunction with "auto" ackn. If it is set, then the auto acknowledge is negative, i.e. with status_3.

DB73.DBB(n+2)	Assigned channel
Edge evaluation	Signal(s) updated: Conditional
Meaning	Number of channel from which the T word was programmed.

DB73.DBB(n+3)	Tool management number
Edge evaluation	Signal(s) updated: Conditional
Meaning	Associated tool management number (TO area) of channel.

Note

The bits in DBB(n+0) (obligatory change, execute change, ...) are not reset by the system. They are current only if the corresponding interface bit in DBB0 is set to "1". However, the bits can be reset by the user if necessary.

DB73.DBD(n+4)	User-definable parameter 0 (DInt)
Edge evaluation	Signal(s) updated: Conditional
Meaning	If you need to send a value to the PLC via the part program, the transfer can be programmed with $P_VDITCP[0] = value. Parameters 0-2 are transferred with the T command.

DB73.DBD(n+8)	User-definable parameter 1 (DInt)
Edge evaluation	Signal(s) updated: Conditional
Meaning	If you need to send a value to the PLC via the part program, the transfer can be programmed with $P_VDITCP[1] = value.

DB73.DBD(n+12)	User-definable parameter 2 (DInt)
Edge evaluation	Signal(s) updated: Conditional
Meaning	If you need to send a value to the PLC via the part program, the transfer can be programmed with $P_VDITCP[2] = value.

DB73.DBW(n+16)	Reserved
Edge evaluation	Signal(s) updated: Conditional
Meaning	

DB73.DBW(n+18)	Reserved
Edge evaluation	Signal(s) updated: Conditional
Meaning	

DB73.DBW(n+20)	Magazine no. of new tool
Edge evaluation	Signal(s) updated: Conditional
Meaning	Magazine no. of the new tool, which should be processed.
Corresponding to ...	DBW(n+22)

DB73.DBW(n+22)	Location no. of new tool to be loaded at change
Edge evaluation	Signal(s) updated: Conditional
Meaning	Location no. of the new tool, which should be processed.
Corresponding to ...	DBW(n+20)

DB73.DBW(n+24)	Reserved
Edge evaluation	
Meaning	

DB73.DBW(n+26)	Location no. of old tool to be unloaded
Edge evaluation	Signal(s) updated: Conditional
Meaning	Location no. of the old tool (used up to now for machining).
Signal state 0	.

DB73.DBW(n+28)	New tool: Location type
Edge evaluation	Signal(s) updated: Conditional
Meaning	The location type of the new tool is entered here.
Corresponding to ...	Tool size: Left, right, top, bottom

DB73.DBW(n+30)	New tool: Size left
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size **left** in half locations for the new tool.

DB73.DBW(n+32)	**New tool: Size right**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size **right** in half locations for the new tool

DB73.DBW(n+34)	**New tool: Size top**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size **top** in half locations for the new tool

DB73.DBW(n+36)	**New tool: Size bottom**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size **bottom** in half locations for the new tool

DB73.DBW(n+38)	**Tool status for new tool**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Bit 0: Active tool Bit 1: Enable tool Bit 2: Disabled Bit 3: Measure tool Bit 4: Prewarning limit reached Bit 5: Tool being changed Bit 6: Tool is fixed-location-coded Bit 7: Tool was in use Bit 8: Tool in the buffer Bit 9: Ignore disabled Bit 10: To be unloaded Bit 11: To be loaded Bit 12: Master tool Bit 13: Reserved Bi 14: 1:1 exchange Bit 15: Manual tool

DB73.DBW(n+40)	**New tool: Internal T no. of NCK**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Display of internal T no. of NCK for the new tool. Tool management variables can be read/written via FB2/FB3 using this T no.

DB73.DBW(n+42)	**Original location of new tool in this circular magazine**
Edge evaluation	Signal(s) updated: Conditional
Meaning	

8.5 Interface for loading/unloading magazine (multitool) (DB 1071)

DB1071 data block	**Magazine, loading/unloading (multitool) NCK -> PLC interface**							
Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
DBW(n+0)	Distance coding							
DBW(n+2)	Multitool location number							
DBD(n+4)	Multitool distance							
DBW(n+8)	Multitool number							
DBW(n+10)	Multitool location number							
DBW(n+12)	Tool holder							
DBW(n+14)	Reserved							
DBW(n+16)	Reserved							
DBW(n+18)	Reserved							

Starting address of loading/unloading points:

Loading/unloading point 1: n = 0

Loading/unloading point 2: n = 20

Loading/unloading point 3: n = 40

Loading/unloading point 4: n = 60

DB1071.DBW(n+0)	**Distance coding**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Type of distance coding of the multitool (corresponds to $TC_MTP_KD 1 = location number 2 = distance 3 = angle
Signal state 0	

DB1071.DBW(n+2)	**Multitool location number**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Number of locations of the multitool
Signal state 0	

DB1071.DBD(n+4)	**Multitool location distance**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Distance of the MT location to be positioned from the reference location (real value). In accordance with the distance coding
Signal state 0	

DB1071.DBW(n+8)	Multitool number
Edge evaluation	Signal(s) updated: Conditional
Meaning	Internal T number of the multitool
Signal state 0	

DB1071.DBW(n+10)	Multitool location number
Edge evaluation	Signal(s) updated: Conditional
Meaning	Location number within the multitool (which the system positions to)
Signal state 0	

DB1071.DBW(n+12)	Tool holder
Edge evaluation	Signal(s) updated: Conditional
Meaning	Spindle or tool holder number
Signal state 0	

8.6 Interface for spindle (multitool) (DB 1072)

DB1072 data block	Spindle (multitool) Interface NCK -> PLC							
Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
DBW(n+0)	Distance coding							
DBW(n+2)	Multitool location number							
DBD(n+4)	Multitool distance							
DBW(n+8)	Multitool number (new tool)							
DBW(n+10)	Multitool location number (new tool)							
DBW(n+12)	Multitool number (old tool)							
DBW(n+14)	Multitool location number (old tool)							
DBW(n+16)	Location type							
DBW(n+18)	Tool size, left							
DBW(n+20)	Tool size, right							
DBW(n+22)	Tool size, top							
DBW(n+24)	Tools size, bottom							
DBW(n+26)	Tool status							
DBW(n+28)	T number of the tool							
DBW(n+30)	Tool holder							
DBW(n+32)	Original magazine of new tool							
DBW(n+34)	Original location of new tool							
DBW(n+36) to DBW(n+48)	Reserved							

Start addresses of the spindles:

Spindle 1: n = 0

Spindle 2: n = 50

Spindle 3: n = 100

DB1072.DBW(n+0)	Distance coding
Edge evaluation	Signal(s) updated: Conditional
Meaning	Type of distance coding of the multitool (corresponds to $TC_MTP_KD 1 = location number 2 = distance 3 = angle
Signal state 0	

DB1072.DBW(n+2)	Multitool location number
Edge evaluation	Signal(s) updated: Conditional
Meaning	Number of locations of the multitool
Signal state 0	

DB1072.DBD(n+4)	Multitool location distance
Edge evaluation	Signal(s) updated: Conditional
Meaning	Distance of the MT location to be positioned from the reference location (real value). In accordance with the distance coding
Signal state 0	

DB1072.DBW(n+8)	Multitool number (new tool)
Edge evaluation	Signal(s) updated: Conditional
Meaning	Internal T number of the (new) multitool
Signal state 0	

DB1072.DBW(n+10)	Multitool location number (new tool)
Edge evaluation	Signal(s) updated: Conditional
Meaning	Location number within the multitool (where the new tool is located
Signal state 0	

DB1072.DBW(n+12)	Multitool number (old tool)
Edge evaluation	Signal(s) updated: Conditional
Meaning	Internal T number of the (old) multitool. The T number is entered here if the preparation or the change to a tool is realized within the same multitool (which, due to a previous change, is located on the tool holder). It is identical with DB 1072.DBW(n+8)
Signal state 0	

DB1072.DBW(n+14)	Multitool location number (old tool)
Edge evaluation	Signal(s) updated: Conditional
Meaning	Location number within the multitool (where the old tool is located). The location number - where the old tool is located - is entered here if the preparation or the change to a tool is realized within the same multitool (which, due to a previous change, is located on the tool holder).
Signal state 0	

DB1072.DBW(n+16)	New tool: Location type
Edge evaluation	Signal(s) updated: Conditional
Meaning	Location type of the new tool (of the programmed tool in the multi-tool).
Corresponding to ...	Tool size: Left, right, top, bottom

DB1072.DBW(n+18)	**New tool: Size left**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size to the left in half locations of the new tool (of the programmed tool in the multitool)
Signal state 0	

DB1072.DBW(n+20)	**New tool: Size right**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size to the right in half locations of the new tool (of the programmed tool in the multitool)
Signal state 0	

DB1072.DBW(n+22)	**New tool: Size top**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size to the top in half locations of the new tool (of the programmed tool in the multitool)
Signal state 0	

DB1072.DBW(n+24)	**New tool: Size bottom**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size to the bottom in half locations of the new tool (of the programmed tool in the multitool)
Signal state 0	

DB1072.DBW(n+26)	Tool status for new tool:
Edge evaluation	Signal(s) updated: Conditional
Meaning	Tool status of the programmed tool in the multitool Corresponds to parameter $TC_TP8[T_No] Bit 0: = Active tool Bit 1: Enable tool Bit 2: Tool disabled Bit 3: Measure tool Bit 4: Prewarning limit reached Bit 5: Tool is being changed Bit 6: Tool is fixed-location-coded Bit 7: Tool was in use Bit 8: Tool in the buffer Bit 9: Ignore disabled Bit 10: To be unloaded Bit 11: To be loaded Bit 12: Master tool Bit 13: Reserved Bit 14: 1:1 exchange Bit 15: Manual tool
Signal state 0	

DB1072.DBW(n+28)	New tool: Internal T number of NCK
Edge evaluation	Signal(s) updated: Conditional
Meaning	Display of internal T number of NCK for the new spindle tool (of the programmed tool in the multitool)
Signal state 0	

DB1072.DBW(n+30)	Tool holder
Edge evaluation	Signal(s) updated: Conditional
Meaning	Spindle or tool holder number to which the change refers (the multitool that is to be changed)
Signal state 0	

DB1072.DBW(n+32)	**Original magazine of new tool**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Owner magazine of the new tool Corresponds to NCK variables $A_MYMN[T-No] If the new tool is located in the magazine, then this value is identical with DB72.DBW(n+20). If the new tool is in the buffer (e.g. gripper), then the magazine no. is entered here from where the tool originally came. The new tool has the same owner location as the multitool onto which it is loaded.
Signal state 0	

DB1072.DBW(n+34)	**Original location of new tool**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Owner location of the new tool Corresponds to NCK variable $A_MYMLNT[T-No] If the new tool is located in the magazine, then this value is identical with DB72.DBW(n+20). If the new tool is in the buffer (e.g. gripper), then the magazine no. is entered here from where the tool originally came. The new tool has the same owner location as the multitool onto which it is loaded.
Signal state 0	

8.7 Interface for revolver (multitool) (DB 1073)

DB1073 data block	**Turret (multitool) Interface NCK -> PLC**							
Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
DBW(n+0)	Distance coding							
DBW(n+2)	Multitool location number							
DBD(n+4)	Multitool distance							
DBW(n+8)	Multitool number (new tool)							
DBW(n+10)	Multitool location number (new tool)							
DBW(n+12)	Multitool number (old tool)							
DBW(n+14)	Multitool location number (old tool)							
DBW(n+16)	Location type							
DBW(n+18)	Tool size, left							
DBW(n+20)	Tool size, right							
DBW(n+22)	Tool size, top							
DBW(n+24)	Tools size, bottom							
DBW(n+26)	Tool status							
DBW(n+28)	T number of the tool							
DBW(n+30)	Tool holder							
DBW(n+32)	Original magazine of new tool							
DBW(n+34)	Original location of new tool							
DBW(n+36) to DBW(n+48)	Reserved							

Start addresses of the turrets:

Spindle 1: n = 0

Spindle 2: n = 50

Spindle 3: n = 100

DB1073.DBW(n+0)	**Distance coding**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Type of distance coding of the multitool (corresponds to $TC_MTP_KD 1 = location number 2 = distance 3 = angle
Signal state 0	

DB1073.DBW(n+2)	Multitool location number
Edge evaluation	Signal(s) updated: Conditional
Meaning	Number of locations of the multitool
Signal state 0	

DB1073.DBD(n+4)	Multitool location distance
Edge evaluation	Signal(s) updated: Conditional
Meaning	Distance of the MT location to be positioned from the reference location (real value). In accordance with the distance coding
Signal state 0	

DB1073.DBW(n+8)	Multitool number (new tool)
Edge evaluation	Signal(s) updated: Conditional
Meaning	Internal T number of the (new) multitool
Signal state 0	

DB1073.DBW(n+10)	Multitool location number (new tool)
Edge evaluation	Signal(s) updated: Conditional
Meaning	Location number within the multitool (where the new tool is located
Signal state 0	

DB1073.DBW(n+12)	Multitool number (old tool)
Edge evaluation	Signal(s) updated: Conditional
Meaning	Internal T number of the (old) multitool. The T number is entered here if the preparation or the change to a tool is realized within the same multitool (which, due to a previous change, is located on the tool holder). It is identical with DB 1072.DBW(n+8)
Signal state 0	

DB1073.DBW(n+14)	Multitool location number (old tool)
Edge evaluation	Signal(s) updated: Conditional
Meaning	Location number within the multitool (where the old tool is located). The location number - where the old tool is located - is entered here if the preparation or the change to a tool is realized within the same multitool (which, due to a previous change, is located on the tool holder).
Signal state 0	

DB1073.DBW(n+16)	New tool: Location type
Edge evaluation	Signal(s) updated: Conditional
Meaning	Location type of the new tool (of the programmed tool in the multi-tool).
Corresponding to ...	Tool size: Left, right, top, bottom

DB1073.DBW(n+18)	New tool: Size left
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size to the left in half locations of the new tool (of the programmed tool in the multitool)
Signal state 0	

DB1073.DBW(n+20)	New tool: Size right
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size to the right in half locations of the new tool (of the programmed tool in the multitool)
Signal state 0	

DB1073.DBW(n+22)	New tool: Size top
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size to the top in half locations of the new tool (of the programmed tool in the multitool)
Signal state 0	

DB1073.DBW(n+24)	New tool: Size bottom
Edge evaluation	Signal(s) updated: Conditional
Meaning	Specification of the tool size to the bottom in half locations of the new tool (of the programmed tool in the multitool)
Signal state 0	

DB1073.DBW(n+26)	Tool status for new tool:
Edge evaluation	Signal(s) updated: Conditional
Meaning	Tool status of the programmed tool in the multitool Corresponds to parameter $TC_TP8[T_No] Bit 0: = Active tool Bit 1: Enable tool Bit 2: Tool disabled Bit 3: Measure tool Bit 4: Prewarning limit reached Bit 5: Tool is being changed Bit 6: Tool is fixed-location-coded Bit 7: Tool was in use Bit 8: Tool in the buffer Bit 9: Ignore disabled Bit 10: To be unloaded Bit 11: To be loaded Bit 12: Master tool Bit 13: Reserved Bit 14: 1:1 exchange Bit 15: Manual tool
Signal state 0	

DB1073.DBW(n+28)	New tool: Internal T number of NCK
Edge evaluation	Signal(s) updated: Conditional
Meaning	Display of internal T number of NCK for the new spindle tool (of the programmed tool in the multitool)
Signal state 0	

DB1073.DBW(n+30)	Tool holder
Edge evaluation	Signal(s) updated: Conditional
Meaning	Spindle or tool holder number to which the change refers (the multitool that is to be changed)
Signal state 0	

DB1073.DBW(n+32)	**Original magazine of new tool**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Owner magazine of the new tool Corresponds to NCK variables $A_MYMN[T-No] If the new tool is located in the magazine, then this value is identical with DB72.DBW(n+20). If the new tool is in the buffer (e.g. gripper), then the magazine no. is entered here from where the tool originally came. The new tool has the same owner location as the multitool onto which it is loaded.
Signal state 0	

DB1073.DBW(n+34)	**Original location of new tool**
Edge evaluation	Signal(s) updated: Conditional
Meaning	Owner location of the new tool Corresponds to NCK variable $A_MYMLNT[T-No] If the new tool is located in the magazine, then this value is identical with DB72.DBW(n+20). If the new tool is in the buffer (e.g. gripper), then the magazine no. is entered here from where the tool originally came. The new tool has the same owner location as the multitool onto which it is loaded.
Signal state 0	

8.8 Interface NC channels

Signals are also contained in the channel data blocks for tool management functions. The data relevant for tool management is in bold formatting.

DB21-30 data block	**Signals to/from NC channel Interface PLC -> NCK**							
Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
DBB1	Activate program test	PLC action completed	CLC override	CLC stop	**Time monitoring active**	Synchronous action off	Enable protective area	Activate referencing
DBB29	**Tool disable not effective**	**Deactivate wear monitoring**	**Deactivate workpiece counter**	Activate PTP traversal	Activate fixed feed 4	Activate fixed feed 3	Activate fixed feed 2	Activate fixed feed 1
Cyclic signals from NC channel								
DBB317	**Tool missing**	PTP traversal active						External language mode active
Change signals TM functions								
DBB344					**Last replacement tool of the tool group**	**Transition to new replacement tool**	**Tool prewarning limit reached**	**Tool prewarning limit reached**
Transferred tool management functions								
DBD348	**T number for tool management prewarning limit (DInt)**							
DBD352	T **number for tool limit value (DInt)**							
DBD356	**T number of new replacement tool (DInt)**							
DBD360	**T number of last replacement tool (DInt)**							

DB21.DBX1.3	The user can start and stop tool life monitoring time using PLC signal "Time monitor active". The effectiveness of this control is set via MD20310 TOOL_MANAGEMENT_MASK, bit 17.

DB21.DBX29.5	Switches workpiece count monitoring on/off.

DB21.DBX29.6	Switches wear monitoring on/off.

DB21.DBX29.7	VDI signal "Tool disable ineffective" (bit value=1) means that the NCK does not process the tool status "Disabled" during tool search. The signal is not effective, if the tool selection is realized using the Init blocks (Reset- and Start_Mode_Mask). If, for instance, it is to be possible to select a disabled spindle tool, then this must be set in machine data $MC_TOOL_CHANGE_ERROR_MODE.

DB21.DBX317.7	Display in PLC that the programmed tool is missing. (Tool is either not available or cannot be used.)

DB21.DBX 344.0 - 344.3	**Modification signals of the tool management functions**
Edge evaluation	Signal(s) updated: Conditional
Meaning	A T number for tool prewarning limit, limit value, new replacement tool, last replacement tool has been output with a value at the interface at the beginning of an OB1 cycle together with the associated modification signal. In this case, the change signal indicates that the appropriate value is valid.

8.9 Interface magazine configuration

DB4	Interface HMI -> PLC	
Address	**Meaning**	**Data type**
DBB64	Number of magazines including buffer magazines and loading magazines	BYTE
<	Beginning of repeat loop; number of repeats from DB4.DBB64	
DBW65 (70, 75, ...)	Magazine number	INT
DBB67 (...)	Magazine type	BYTE
DBW68 (...)	Number of locations	INT
>	Return loop end	
Address = (contents DBB64*5) +65	Number of spindles	BYTE

9 Alarms

9.1 Overview

Alarm No.	Brief description
6402	Tool change not possible, magazine number does not exist.
6403	Tool change not possible, specified magazine location does not exist.
6404	Tool change not possible because the tool is not available or cannot be used.
6405	Command has invalid PLC acknowledgement parameter.
6406	PLC acknowledgement missing.
6407	Tool is to be set down at a location that does not meet the requirements for loading.
6410	One cutting edge of the monitored tool has reached a prewarning limit.
6411	One cutting edge of the monitored tool has reached a prewarning limit.
6412	One cutting edge of the monitored tool has reached a monitoring limit.
6413	One cutting edge of the monitored tool has reached a monitoring limit.
6421	No free location for tool in the magazine.
6422	Tool motion is not possible because the magazine has not been defined.
6423	Tool motion is not possible because no location in the magazine.
6424	Tool motion is not possible because tool not available or cannot be used.
6425	Tool motion is not possible because tool cannot be placed at the specified location in the magazine
6430	Workpiece counter: Overflow in table of monitored cutting edges.
6431	Function not permitted because TOOLMAN / TOOLMAN monitoring not activated.
6432	Function cannot be executed because there is no tool in the spindle.
6433	System variable not available for active tool management.
6436	The command cannot be programmed
6438	Inconsistent data change not permitted
6441	Writing of $P_USEKT not allowed.
6450	Tool change not possible because magazine-location number is not valid.
6451	No buffer magazine defined.
6452	Tool holder number / spindle number not defined.
6453	No relationship defined between tool holder number / spindle number and buffer magazine.
6454	Neither spindle nor buffer location has a distance relationship.
6460	Command can only be programmed for tools
6462	Command can only be programmed for magazines
6464	Command cannot be programmed for the actual multitool distance coding
6924	Tool data changed for program test.
17001	No more memory for tool magazine data.
17020	Illegal array index 1
17160	No tool selected.
17180	Illegal D number.
17181	D no. not known.

Alarm No.	Brief description
17182	Illegal additive offset number.
17188	The D number specified in the channel TO unit is not unique.
17189	D number is not unique.
17191	Unknown tool identifier.
17192	No further replacement tools possible.
17193	The active tool is no longer in the tool holder
17194	No suitable tool found.
17200	Tool cannot be deleted
17202	Magazine data cannot be deleted.
17212	Manual tool must be changed.
17214	Remove manual tool from tool holder.
17215	Remove manual tool from buffer
17216	Manual tools must be changed.
17218	Tool cannot become a manual tool
17220	Tool does not exist.
17224	It is not possible on this system to select tool offsets for tools of the specified tool type.
17230	Duplo no. already assigned.
17240	Invalid tool definition.
17242	Manual tool cannot be set, as the function is not active
17250	Invalid magazine definition.
17255	Magazine location hierarchies have been deleted
17260	Invalid magazine location definition.
17262	Incorrect tool-adapter assignment
20150	PLC terminates the interrupted command.
20160	PLC can terminate only incorrectly aborted commands.
22066	Tool motion not possible because specified tool is not in magazine.
22067	Tool change not possible because no tool ready for use in the tool group.
22068	No tool ready for use in the tool group.
22069	No tool ready for use in the tool group.
22070	Change tool into magazine. Repeat data backup.
22071	Tool has the status "active" in an "inactive" wear group.
400601	Incorrect configuration of loading points
400602	Incorrect spindle configuration
400603	Incorrect turret configuration
400604	Set change with M06 in machine data.
410141	Number of loading points too high
410142	Number of tool holders too high
410143	Number of turrets too high
410151	Magazine data for tool management missing in PLC.

With machine dates MD11410 SUPPRESS_ALARM_MASK und MD11415 SUPPRESS_ALARM_MASK_2 it can be defined bit by bit which alarms should be suppressed.

Bit	Alarm number
2	16924
4	17189
5	22071
7	22070
8	6411, 6413
9	6410, 6412

9.2 Alarm description

Alarm No.	
6402	Channel %1 tool change not possible because magazine no. %2 not available
Explanation	%1 = channel ID, %2 = magazine number The desired tool change is not possible. The magazine with the specified number is not available.
Reaction	Alarm display Interface signals are set NC start disable NC stop for alarm
Remedy	Check whether the magazine data is correctly defined. Check whether the magazine is connected to the required spindle via a distance relationship The user PLC program may have supplied incorrect data to the NCK.
Continue program	Cancel the alarm with the RESET button and start the part program again.

Alarm No.	
6403	Channel %1 tool change not possible because magazine no. %2 in magazine %3 not available
Explanation	%1 = channel ID, %2 = magazine number, %3 = magazine location number The desired tool change is not possible. The specified magazine location is not available in the specified magazine.
Reaction	Alarm display Interface signals are set NC start disable NC stop for alarm
Remedy	Check whether the magazine data is correctly defined. The user PLC program may have supplied incorrect data to the NCK.
Continue program	Cancel the alarm with the RESET button and start the part program again.

Alarm No.	
6404	Channel %1 tool change not possible. Tool %2 not available or cannot be used
Explanation	%1 = channel ID, %2 = string (identifier) The desired tool change is not possible. The specified tool does not exist or cannot be used.
Reaction	Alarm display Interface signals are set NC start disable NC stop for alarm
Remedy	Check whether the part program is written correctly. Check whether the magazine data is correctly defined. Check whether there is a replacement tool which can be used for the specified tool.
Continue program	Cancel the alarm with the RESET button and start the part program again.

Alarm No.	
6405	Channel %1 command %2 has an invalid PLC acknowledgement parameter %3 identification %4
Explanation	%1 = channel ID, %2 = command no. %3 = PLC acknowledgement parameter, %4 = error identification The specified command has been answered by the PLC with an invalid acknowledgement in the actual combination. The following assignments are defined for command number: 1 Move tool, load or unload magazine 2 Prepare tool change 3 Execute tool change 4 Prepare tool change and execute with T command 5 Prepare tool change and execute with M command 7 Terminate interrupted tool command 8 Check tool motion with reservation 9 Check tool motion 0 Transport acknowledgement Parameters 2 and 3 specify the PLC command and the status number of the acknowledgement. Example: Parameter 4 of the alarm message is =10. It is not defined for asynchronous tool motion, to reserve a buffer location. In the example, the NCK ignores the parameter. Additional possible reasons for the alarm: The tool change defined by the command cannot be executed. The magazine location specified in the invalid parameter does not exist in the magazine. The error code (%4) defines the alarm in more detail: 0 Not defined 1 Status now not permitted, or non-defined status received from PLC 2 Source and/or target magazine no. / location no. not known 3 Not defined 4 Target magazine no. and/or target location no. for tool move command not final destination 5 Not defined 6 Source and/or target magazine no. / location no. not known for tool change 7 PLC command with inconsistent data: Either magazine addresses inconsistent in VDI or NCK command not the same as the PLC acknowledgement, or both 8 PLC command with inconsistent data: When lifting a tool, asynchronous tool to be rejected was unloaded. NCK cannot execute a new selection. 9 PLC command with inconsistent data: The command acknowledgements want to move a tool to a location where another tool is already located. 10 The asynchronous tool motion with reservation is only defined for movement from a magazine to a buffer.
Reaction	Alarm display Interface signals are set NC start disable NC stop for alarm
Remedy	Notify authorized personnel / service Faulty PLC communication: Correct the PLC program
Continue program	Cancel the alarm with the RESET button and start the part program again.

Alarm No.	
6406	Channel %1 PLC acknowledgement for command %2 is missing
Explanation	%1 = channel ID, %2 = command no. There is still no acknowledgement from the PLC for the tool change. The NCK cannot continue processing until it receives this acknowledgement for the specified command number. Possible command no. values are described under alarm 6405
Reaction	Alarm display Interface signals are set NC start disable
Remedy	Notify authorized personnel / service Faulty PLC communication: Correct the PLC program. It is possible to release NCK from the wait condition with the PLC command 7. This aborts the waiting command.
Continue program	Cancel the alarm with the RESET button and start the part program again.

Alarm No.	
6407	Channel %1 The tool %2 cannot be placed in magazine %3 on location %4. Invalid magazine definition!
Explanation	%1 = channel ID, %2 = string (identifier), %3 = magazine number, %4 = magazine location number A tool change task or a verification task was issued to place the tool in a location whose definition does not satisfy the prerequisites for filling. The following causes for the error are possible: Location is disabled or not free Tool type does not match the location type Tool possibly too large, adjacent locations are not free
Reaction	Alarm display Interface signals are set NC start disable NC stop for alarm
Remedy	Check whether the magazine data is correctly defined (especially the location type) Check whether the tool data is correctly defined (especially the location type)
Continue program	Cancel the alarm with the RESET button and start the part program again.

Alarm No.	
6410	TO unit %1 tool %2 with duplo no. %3 has reached tool prewarning limit with D=%4
Explanation	%1 = TO unit, %2 = tool identifier (name), %3 = duplo number, %4 = D number Indicates that the specified D offset of the time, workpiece quantity or wear-monitored tool has reached its prewarning limit. If possible, specify the D number - if not, then the 4th parameter is assigned the value 0. If the "additive offset" function is being used, then instead of the wear monitoring, an additive offset monitoring can also be active. The specific type of tool monitoring is a property of the tool (see $TC_TP9). The duplo no. specification has no additional significance if replacement tools are not being used. The alarm is triggered via the OPI interface (HMI, PLC). The channel context is not defined. The TO unit is therefore specified (see $MC_MM_LINK_TOA_UNIT).
Reaction	Alarm display Interface signals are set
Remedy	For information only. The user must decide what to do.
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6411	Channel %1 Tool %2 Prewarning limit reached with D=%4
Explanation	%1 = channel number, %2 = tool identifier (Name), %4 = D number Indicates that the specified D offset of the time, workpiece quantity or wear-monitored tool has reached its prewarning limit. If possible, specify the D number - if not, then the 4th parameter is assigned the value 0. If the "additive offset" function is being used, then instead of the wear monitoring, an additive offset monitoring can also be active. The specific type of tool monitoring is a property of the tool (see $TC_TP9). The duplo no. specification has no additional significance if replacement tools are not being used. Limit is detected in the context of the channel. The alarm is caused while the NC program is being executed.
Reaction	Alarm display Interface signals are set
Remedy	For information only. The user must decide what to do.
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6412	TO unit %1 tool %2 with duplo no.%3 has reached tool monitoring limit with D=%4
Explanation	%1 = TO unit, %2 = tool identifier (name), %3 = duplo number, %4 = D number Indicates that the specified D offset of the time, workpiece quantity or wear-monitored tool has reached its prewarning limit. If possible, specify the D number - if not, then the 4th parameter is assigned the value 0. If the "additive offset" function is being used, then instead of the wear monitoring, an additive offset monitoring can also be active. The specific type of tool monitoring is a property of the tool (see $TC_TP9). The duplo no. specification has no additional significance if replacement tools are not being used. The alarm is triggered via the OPI interface (HMI, PLC). The channel context is not defined. The TO unit is therefore specified (see $MC_MM_LINK_TOA_UNIT).

Alarm No.	
Reaction	Alarm display Interface signals are set
Remedy	For information only. The user must decide what to do.
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6413	Channel %1 Tool %2 has reached its monitoring limit with D=%4
Explanation	%1 = channel number, %2 = tool identifier (Name), %4 = D number Indicates that the specified D offset of the time, workpiece quantity or wear-monitored tool has reached its prewarning limit. If possible, specify the D number - if not, then the 4th parameter is assigned the value 0. If the "additive offset" function is being used, then instead of the wear monitoring, an additive offset monitoring can also be active. The specific type of tool monitoring is a property of the tool (see $TC_TP9). The duplo no. specification has no additional significance if replacement tools are not being used. The alarm originates during NC program execution.
Reaction	Alarm display Interface signals are set
Remedy	For information only. The user must decide what to do.
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6421	Channel %1 Tool motion not possible. There is no empty location for tool %2 duplo no. %3 in magazine %4.
Explanation	%1 = channel ID, %2 = string (identifier), %3 = duplo number, %4 = magazine number The requested tool motion command - initiated from the HMI or PLC - is not possible. The tool cannot be moved into the specified tool magazine. There is no appropriate location available for this tool.
Reaction	Alarm display Interface signals are set NC start disable
Remedy	Check whether the magazine data is defined correctly (e.g. the magazine must not be disabled). Check whether the tool data is correctly defined (e.g. the tool location type must match the permitted location types in the magazine). Check whether the magazine simply no longer has any space to accept an additional tool due to operator actions. Check whether a location type hierarchy is defined and whether it, for example, does not allow insertion of a type "A" tool in a free location with type "B".
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6422	Channel %1 Tool motion not possible because magazine no. %2 not available!
Explanation	%1 = channel ID, %2 = magazine number The requested tool motion command - initiated from the HMI or PLC - is not possible. The magazine with the specified number is not available.
Reaction	Alarm display Interface signals are set NC start disable
Remedy	Check whether the magazine data is correctly defined. If the PLC issued the command for motion: check whether the PLC program is correct. If the HMI issued the command for motion: check whether the HMI command was assigned correct parameters.
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6423	Channel %1 Tool motion not possible because magazine location no. %2 in magazine %3 not available!
Explanation	%1 = channel ID, %2 = magazine location number, %3 = magazine number The requested tool motion command - initiated from the HMI or PLC - is not possible. The specified magazine location is not contained in the specified magazine.
Reaction	Alarm display Interface signals are set NC start disable
Remedy	Check whether the magazine data is correctly defined.
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6424	Channel %1 Tool motion not possible. Tool %2 not available or cannot be used
Explanation	%1 = channel ID, %2 = string (identifier) The requested tool motion command - initiated from the HMI or PLC - is not possible. The state of the specified tool does not permit the tool to be moved. The specified tool is not defined or is not permitted for the command.
Reaction	Alarm display Interface signals are set NC start disable
Remedy	Check whether the tool state "being changed" is set. If yes, then initially the corresponding tool change command from the PLC must be terminated. The tool should then be able to be moved. Check whether the magazine data is correctly defined. Check whether the move command has been correctly parameterized. Check whether the tool is already loaded (if the alarm occurs when the tool is loaded).
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6425	Channel %1 The tool %2 cannot be placed in magazine %3 on location %4. Invalid magazine definition
Explanation	%1 = channel ID, %2 = string (identifier), %3 = magazine number, %4 = magazine location number The requested tool motion command - initiated from the HMI or PLC - is not possible. Using a movement task, the tool is to be placed down at a location whose definition does not fulfill the preconditions for loading a tool. The following causes for the error are possible: Location is disabled or not free Tool type does not match the location type. Tool possibly too large, adjacent locations are not free. If loading/unloading - the loading/unloading location must be of type "loading point". If loading/unloading - is the relevant magazine linked with the loading/unloading location? See $TC_MDP1, $TC_MDP2
Reaction	Alarm display Interface signals are set NC start disable
Remedy	Check whether the magazine data is correctly defined. Check whether the magazine simply no longer has any space to accept an additional tool due to operator actions. Check whether a location hierarchy is defined and whether it, for example, does not allow insertion of a type "A" tool in a free location with type "B". Check whether the relevant magazine is linked with the loading/unloading location or has a defined dis- tance. Check whether the loading/unloading location is of type "loading point". See also $TC_MPP1
Continue pro- gram	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6430	Workpiece counter: Overflow in table of monitored cutting edges
Explanation	No more cutting edges can be entered in the workpiece counter table. As many cutting edges can be noted for the workpiece counter as a whole as the total number of cutting edges that are possible in the NCK. This means that if every cutting edge of each tool is precisely used for one workpiece, then the limit is reached. If several workpieces are machined simultaneously at several tool holders / spindles, then across all workpieces, 18100 MM_NUM_CUTTING_EDGES_IN_TOA cutting edges can be noted for the workpiece counter. If the alarm is present, then this means that the cutting edges that will now be used are no longer monitored from a workpiece number perspective; and more precisely, until the table is emptied again, e.g. using the NC language command SETPIECE or the corresponding task from the HMI, PLC (PI service).
Reaction	Alarm display Interface signals are set NC start disable

Alarm No.	
Remedy	Have you forgotten to decrement the workpiece counter? Then program SETPIECE in the part program or correctly insert the corresponding command in the PLC program. If the part program or the PLC program is correct, then more memory should be set for tool cutting edges using machine data $MM_NUM_CUTTING_EDGES_IN_TOA (this is only possible for those with the appropriate access authorization).
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6431	Function not allowed. Tool management/tool management monitoring not activated
Explanation	Occurs when a data management function is called which is not available because tool management is deactivated or tool monitoring is not available. For example, the language commands GETT, SETPIECE, GETSELT, NEWT, DELT.
Reaction	Alarm display Interface signals are set Interpreter stop NC start disable
Remedy	Please inform the authorized personnel / service department. Verify how the NC control shall be configured. Is tool management or tool monitoring required, but not activated? Is a part program used that has been designed for NC control with tool management / tool monitoring? Either run a part program on the suitable NC control or modify the part program. Activate tool management / tool monitoring by setting the corresponding machine data. See $MN_MM_TOOL_MANAGEMENT_MASK, $MC_TOOL_MANAGEMENT_MASK. Check whether the required option is set accordingly.
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6432	Function cannot be executed. There is no tool on the spindle
Explanation	An attempt was made to perform an operation that requires a tool to be located on the spindle. This can be the workpiece count monitoring function, for example.
Reaction	Alarm display Interface signals are set
Remedy	Select another function, select another tool holder / spindle or locate a tool on the tool holder / spindle.
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6433	Channel %1 block %2 variable %3 not available with tool management
Explanation	%1 = channel number, %2 = block number, Label, %3 = source symbol The system variable specified in %3 is not available with active tool management. The function GETSELT should be used with $P_TOOLP.
Reaction	Alarm display Interface signals are set NC start disable
Remedy	Change program
Continue program	Clear the alarm with the delete key.

Alarm No.	
6436	Channel %1 block %2 command %3 cannot be programmed. Function %4 is not activated.
Explanation	%1 = channel number, %2 = block number, %3 = command, %4 = function The command cannot be programmed due to a missing function enable or activation.
Reaction	Alarm display Interface signals are set NC start disable
Remedy	Correct the NC program
Continue program	Clear the alarm with NC start or RESET key and continue the program.

Alarm No.	
6438	Channel %1 block %2 inconsistent data modification is not permitted
Explanation	%1 = channel number, %2 = block number For example, in a defined multitool, after creating the multitool location it is not permissible to change the distance coding $TC_MTP_KD.
Reaction	Alarm display Interface signals are set NC start disable
Remedy	Correct the NC program
Continue program	Clear the alarm with NC start or RESET key and continue the program.

Alarm No.	
6441	Writing of $P_USEKT not allowed.
Explanation	An attempt was made to write the value from $P_USEKT. This is not possible because the programming T="location number" with automatic setting of $P_USEKT is active.

Alarm No.	
Reaction	Alarm display Interface signals are set Interpreter stop NC start disable in this channel
Remedy	Verify how the NC control should be configured (bit 16 and bit 22 in $MC_TOOL_MANAGEMENT_MASK) Is a part program used that is configured for an NC control without T="location number" with automatic setting of $P_USEKT? It is not possible to start this program on the NC control with T="location number" with automatic setting of $P_USEKT. Either run a part program on the suitable NC control or modify the part program.
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6442	Channel %1 Function cannot be executed. No tool at the required magazine/location %2
Explanation	%1 = channel number, %2 magazine/location It is probable that the PLC logic is incorrect. Tool change with reject tool is configured. Preparation command present. Selected tool is unloaded from its location (e.g. from the PLC). PLC acknowledges preparation command with "repeat tool selection" (e.g. status=7). NCK does not find the tool in the magazine location specified in the PLC command. Or: Illegal operator intervention in a tool selection that is presently being executed (unloading the tool to be selected) has taken place. This is the reason that the PLC acknowledgement was unsuccessful.
Reaction	Alarm display Interface signals are set
Remedy	PLC programmers must observe the following: Ensure that the tool is not removed from the specified magazine location (e.g. PLC program incorrect). Do not withdraw (=unload) the programmed tool change before a command has been finally acknowledged. However, it is possible to change the location of the tool to be loaded. The alarm supplements alarm 6405, if it contains the ID 8.
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
6450	Channel %1 Tool change not possible. Invalid magazine location number %2 in the buffer magazine
Explanation	%1 = channel number, %2 magazine location number The desired tool change is not possible. The specified magazine location is a tool holder / spindle or is empty. Using the language command TCI, only the buffer numbers may be programmed which are not tool holders / spindles; i.e. the location number of a gripper is permitted.
Reaction	Alarm display Interface signals are set
Remedy	Check whether the magazine data ($TC_MPP1) is correctly defined. Check whether the programmed command that was the cause has been correctly parameterized - e.g. TCI.
Continue program	Clear the alarm with NC start or RESET key and continue the program.

Alarm No.	
6451	Channel %1 Tool change not possible. No buffer magazine defined.
Explanation	%1 Channel number The desired tool change is not possible. No buffer has been defined.
Reaction	Alarm display Interface signals are set
Remedy	Check whether the magazine data is correctly defined.
Continue program	Clear the alarm with NC start or RESET key and continue the program.

Alarm No.	
6452	Channel %1 block %2 tool change not possible. The tool holder no. / spindle no. = %3 is not defined.
Explanation	%1 = channel number, %2 tool holder no. / spindle no. The desired tool change is not possible. The tool holder number / spindle number is not defined.
Reaction	Alarm display Interface signals are set
Remedy	Check whether the tool holder no. / spindle no. and magazine data is correctly defined. (See also the system variables $TC_MPP1, $TC_MPP5 of the buffer magazine).
Continue program	Clear the alarm with NC start or RESET key and continue the program.

Alarm No.	
6453	Channel %1 Tool change not possible. No assignment between tool holder / spindle no. = %2 and buffer location %3
Explanation	%1 = channel number, %2 tool holder no. / spindle no. %3 buffer location The desired tool change is not possible. No relationship has been defined between the tool holder / spindle number and the buffer location LocNo.
Reaction	Alarm display Interface signals are set
Remedy	Check whether the magazine data ($TC_MLSR) is correctly defined. Check whether the program command causing the error (e.g. TCI) is programmed correctly.
Continue program	Clear the alarm with NC start or RESET key and continue the program.

Alarm No.	
6454	Channel %1 Tool change not possible. There is no distance relationship available.
Explanation	%1 Channel number The desired tool change is not possible. Neither spindle nor buffer location has a distance relationship.
Reaction	Alarm display Interface signals are set

Alarm No.	
Remedy	Check whether the magazine data ($TC_MDP2) is correctly defined. Check whether the program command causing the error (e.g. TCI) is programmed correctly.
Continue program	Clear the alarm with NC start or RESET key and continue the program.

Alarm No.	
6455	Channel %1 block %2 tool change not possible. Magazine location no. %3 not available in magazine %4
Explanation	%1 channel number %2 block number %3 magazine location number %4 magazine The desired tool change is not possible. The specified magazine location is not available in the specified magazine.
Reaction	Alarm display Interface signals are set
Remedy	Check whether the magazine data ($TC_MAP6 and $TC_MAP7 of the buffer magazine) is correctly defined. Check whether the program command causing the error (e.g. TCI) is programmed correctly.
Continue program	Clear the alarm with NC start or RESET key and continue the program.

Alarm No.	
6460	Channel %1 block %2 command %3 can only be programmed for tools. %4 means no tool.
Explanation	%1 channel number %2 block number %3 command parameter The specified command can only be programmed for tools. The command parameter is not a T number or not a tool name. If a multitool was programmed: The command cannot be programmed for multitools.
Reaction	Alarm display Interface signals are set
Remedy	Correct the NC program
Continue program	Clear the alarm with NC start or RESET key and continue the program.

Alarm No.	
6462	Channel %1 block %2 command %3 can only be programmed for magazines. %4 means no tool.
Explanation	%1 channel number %2 block number %3 command parameter The specified command can only be programmed for magazines. The command parameter is not a magazine number or not a magazine name. If a multitool was programmed: The command cannot be programmed for multitools.
Reaction	Alarm display Interface signals are set
Remedy	Correct the NC program
Continue program	Clear the alarm with NC start or RESET key and continue the program.

Alarm No.	
6464	Channel %1 block %2 command %3 cannot be programmed for the actual multitool distance coding %4.
Explanation	%1 channel number %2 block number %3 command parameter $TC_MTPPL can only be programmed, if $TC_MTP_KD has the value 2. $TC_MTPPA can only be programmed, if $TC_MTP_KD has the value 3.
Reaction	Alarm display Interface signals are set
Remedy	Correct the NC program
Continue program	Clear the alarm with NC start or RESET key and continue the program.

Alarm No.	
6924	Channel %1 caution: Program test changes tool management data
Explanation	%1 = channel number Tool data is changed during program testing. You cannot automatically correct the tool data again on termination of program test. This alarm prompts you to create a backup of the tool data which must be loaded again when you have finished testing the program.
Reaction	Alarm display
Remedy	Please inform the authorized personnel / service department. Save tool data on HMI and load again after "ProgtestOff".
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
17001	Channel %1 block %2 no more memory for tool / magazine data
Explanation	%1 = channel number, %2= block number, label The number of following tool/magazine data variables in the NC is specified by machine data: - number of tools + number of grinding data blocks: MD18082 $MN_MM_NUM_TOOL - number of cutting edges: MD18100 $MN_MM_NUM_CUTTING_EDGES_IN_TOA Tools, grinding data blocks, cutting edges can be used independently of the tool management. The memory for the following data is only available if the corresponding bit has been set in MD18080 $MN_MM_TOOL_MANAGEMENT_MASK. - Number of monitoring data blocks: MD18100 $MN_MM_NUM_CUTTING_EDGES_IN_TOA - Number of magazines: MD18084 $MN_MM_NUM_MAGAZINE - Number of magazine locations: MD18086 $MN_MM_NUM_MAGAZINE_LOCATION The following variable is determined by the software configuration: Number of magazine distance data blocks: P2 permits 32 such distance data blocks. Definition: - "Grinding data blocks": Grinding data can be defined for a tool of type 400 to 499. A data block of this type occupies as much additional memory as that provided for a cutting edge. - "Monitoring data blocks": Each cutting edge of a tool can be supplemented with monitoring data. If the alarm occurs when writing from one of the parameters $TC_MDP1/$TC_MDP2/$TC_MLSR, check whether the machine data MD18077 $MN_MM_NUM_DIST_REL_PER_MAGLOC / MD18076 $MN_MM_NUM_LOCS_WITH_DISTANCE has been set correctly. MD18077 $MN_MM_NUM_DIST_REL_PER_MAGLOC defines the number of different Index1 statements that may be made for an Index2 value. MD18076 $MN_MM_NUM_LOCS_WITH_DISTANCE defines the number of different buffer locations that may be named in Index2. If a multitool is to be created - or its locations - then the alarm indicates that either more multitools are to be created than is permitted in MD18083 $MN_MM_NUM_MULTITOOL - or if the alarm is generated when creating the multitool locations, then the alarm indicates that more multitool locations are to be created than permitted via MD18085 $MN_MM_NUM_MULTITOOL_LOCATIONS.
Reaction	Alarm display Interface signals are set Correction block with reorganization
Remedy	Please inform the authorized personnel / service department. Change machine data Change NC program, i.e. reduce the number of rejected variables
Continue program	Clear the alarm with the RESET key. Restart the part program.

Alarm No.	
17020	Channel %1 block %2 illegal array index 1
Explanation	%1 = channel number, %2= block number General: A read or write access was programmed to a field variable with invalid 1st field index. The valid field indices must be within the defined field size and the absolute limits (0 - 32 766). [DPA]: PROFIBUS I/O: [DPA]: When reading/writing data, an invalid slot / I/O range index was used. [DPA]: Cause: [DPA]: 1.: Slot / I/O range index >= max. available number of slots / I/O ranges. [DPA]: 2.: Slot / I/O range index references a slot / I/O range that is not configured. [DPA]: 3.: Slot / I/O range index references a slot / I/O range that is not released for system variables. [DPA]: The following specifically applies: If the alarm occurs when writing from one of the parameters $TC_MDP1/$TC_MDP2/$TC_MLSR, [DPA]: then it must be checked as to whether MD18077 $MN_MM_NUM_DIST_REL_PER_MAGLOC has been set correctly. [DPA]: MD18077 $MN_MM_NUM_DIST_REL_PER_MAGLOC defines the number of different Index1 statements that may be made for an Index2 value. If an MT number is programmed, then the value can collide with an already defined T number or an already defined magazine number.
Reaction	Alarm display Interface signals are set NC start disable
Remedy	Correct the data relating to the field elements for the access instruction corresponding to the defined size. When using an SPL in Safety Integrated, additional restrictions can apply to the field index via the option data.
Continue program	Clear the alarm with NC start or RESET key and continue the program.

Alarm No.	
17050	Channel %1 block %2 impermissible value
Explanation	%1 = channel number, %2= block number A value has been programmed that exceeds the value range or a limit value of a variable or a machine data item. For example - In a string variable (e.g. GUD or LUD), a string needs to be written that exceeds the agreed string length in the variable definition. - If an illegal value is to be written to a tool or magazine management variable (e.g. illegal cutting edge number in $TC_DPCE[x,y] or illegal magazine location number in $TC_MDP2[x,y]). - An illegal value is to be written to $P_USEKT or $A_DPB_OUT[x,y]. - An illegal value is to be written to a machine data (e.g. MD10010 $MN_AS-SIGN_CHAN_TO_MODE_GROUP[0] = 0). - When accessing an individual frame element, a frame component other than TRANS, ROT, SCALE or MIRROR was addressed or the CSCALE function was assigned a negative scale factor. A multitool number has been programmed that collides with a previously defined T number or a previously defined magazine number. When programming DELMLOWNER: The command cannot be programmed with the T number of a tool that is part of a multitool.
Reaction	Alarm display Interface signals are set NC start disable
Remedy	Only address frame components with the keywords intended for the purpose; program a scale factor within the limits 0.000 01 to 999.999 99.
Continue program	Clear the alarm with the RESET key. Restart the part program.

Alarm No.	
17160	Channel %1 block %2 no tool selected
Explanation	%1 = channel number, %2= block number, label An attempt has been made to access the current tool offset data via the system variables: $P_AD[n]: Contents of parameter (n: 1 - 25) $P_TOOL: Active D number (cutting edge number) $P_TOOLL[n]: Active tool length (n: 1 - 3) $P_TOOLR: Active tool radius although no tool was previously selected.
Reaction	Alarm display Interface signals are set Interpreter stop NC start disable

Alarm No.	
Remedy	Program or activate a tool offset in the NC program before using the system variables. Example: N100 G.. ... T5 D1 ...LF The channel-specific machine data: 22550: TOOL_CHANGE_MODE new tool offset for M function 22560: TOOL_CHANGE_M_MODE M function for tool change is set to define whether activating a tool offset in the block is carried out with the T word or whether the new offset values are only computed with the M word for the tool change.
Continue program	Clear the alarm with the RESET key. Restart the part program.

Alarm No.	
17180	Channel %1 block %2 illegal D number
Explanation	%1 = channel number %2 = block number, label In the displayed block, access is made to a D number (tool edge number) that is not defined and therefore not available.
Reaction	Alarm display Interface signals are set Interpreter stop NC start disable
Remedy	Check tool call in the NC part program: Correct cutting edge number programmed? If a cutting edge number is not specified, then the D number D1, set using machine data $MC_CUTTING_EDGE_DEFAULT is automatically active. Tool parameters defined (tool type, length, …)? The dimensions of the tool cutting edge must have been previously entered in the NCK either from the operator panel or using a tool data file. Description of the system variables $TC_DPx[t,d] as to how they are contained in a tool data file, x ...offset parameter number P, t ... associated tool number T, d ...tool offset number D.
Continue program	Clear the alarm with the RESET key. Restart the part program.

Alarm No.	
17181	Channel %1 block %2 T no.= %3, D No.= %4 does not exist
Explanation	%1 = channel number, %2 = block number, label, %3 = T number, %4 = D number A D number has been programmed that the NCK does not recognize. As standard, the D number refers to the given T number. If the "flat D number" function is active, T = 1 is output.
Reaction	Alarm display Interface signals are set Correction block with reorganization

Alarm No.	
Remedy	In case of a programming error, eliminate the error with a correction block and continue the program run. If the data record is missing, then load the NCK with a data record for the specified T/D values (via HMI, with overstore) and continue program.
Continue program	Cancel the alarm with NC START and continue processing.

Alarm No.	
17182	Channel %1 block %2 illegal additive offset number
Explanation	%1 = channel number, %2= block number, label An attempt was made to access a non-defined additive offset of the current tool edge.
Reaction	Alarm display Interface signals are set Correction block with reorganization
Remedy	Access the additive offset memory with $TC_SCP, $TC_ECP, check the additive offset selection DLx or tool selection Ty or offset selection Dz.
Continue program	Cancel the alarm with NC START and continue processing.

Alarm No.	
17188	Channel %1 D number %2 defined in tool T No. %3 and %4
Explanation	%1 = channel number, %2 = D number, %3 = T number for first tool, %4 = T number for second tool The specified D number %2 in the TO unit of channel %1 is not unique. The specified T numbers %3 and %4 each have an offset with number %2. If tool management is active, the following also applies: The specified T numbers belong to tool groups with different identifiers.
Reaction	Alarm display Interface signals are set
Remedy	Ensure uniqueness of the D numbering within the TO units. Do not use the command that caused the problem if uniqueness is not needed in the following - CHKDNO.
Continue program	The alarm is for information purposes. You can suppress the alarm output by setting bit 4 in MD $MN_SUPPRESS_ALARM_MASK.

Alarm No.	
17189	Channel %1 D number %2 of tools defined on magazine/location %3 and %4
Explanation	%1 = channel ID, %2 = D number, %3 = magazine no./ magazine location no. – "/" as separator, %4 = magazine no. / magazine location no. – "/" as separator The specified D number %2 in the TO unit of channel %1 is not unique. The tools in the specified magazine locations %3 and %4 each have an offset with the number %2. If tool management is active, the following also applies: The specified T numbers belong to tool groups with different identifiers.
Reaction	Alarm display Set interface signals

Alarm No.	
Remedy	Ensure the uniqueness of the D numbering within the TO unit, e.g. by renaming the D numbers If the uniqueness is not required, do not use the command that caused the problem. The alarm is for information purposes. It can be suppressed by setting bit 4 of MD 11410 SUP-PRESS_ALARM_MASK.
Continue program	The alarm is no longer displayed when the alarm cause has been removed. No further operator action required.

Alarm No.	
17191	Channel %1 block %2 T= %3, does not exist, program %4
Explanation	A tool identifier which the NCK does not recognize was programmed.
Reaction	Alarm display Interface signals are set Correction block with reorganization
Remedy	If the program pointer is at an NC block that contains the specified T identifier: If the program is incorrect, resolve the error with a correction block and continue the program. If the data record is missing, then create a data record, i.e. load the data record of the tool with all defined D numbers to the NCK (via HMI, with overstore), then continue the program. If the program pointer is at an NC block that does not contain the specified T identifier: The error already occurred earlier in the program when programming T, the alarm is however is only output with the change command. If the program is incorrect, e.g. T5 is programmed instead of T55, then the actual block can be corrected with a correction block; i.e. if only M06 is there, then the block can be corrected to become T55 M06. The incorrect T5 line remains in the program until it is terminated by a RESET or end of program. In complex program structures with indirect programming, it may not be possible to correct the program. Then, it is only possible to provide help locally using an overstore block - in the example with T55. If the data record is missing, then create a data record, i.e. load the data record for the tool with all defined D numbers to the NCK (via HMI, with overstore), program T with overstore and then continue the program.
Continue program	Cancel the alarm with NC START and continue processing.

Alarm No.	
17192	TO unit %1 invalid tool name of %2, duplo no. %3. No further replacement tools in %4 possible
Explanation	%1 = TO units number, %2 = tool identifier, %3 = duplo number of the tool to be renamed, %4 = group identifier The tool with the specified tool identifier, duplo number cannot accept the group identifier. Reason: The maximum number of permitted replacement tools has already been defined. By allocating a name for the tool, a new allocation or reallocation of the tool takes place in a tool group, which already has the maximum permitted number of replacement tools for this machine.
Reaction	Alarm display Interface signals are set

Alarm No.	
Remedy	Define fewer replacement tools. Unload replacement tools that are no longer required and delete their data in the NCK. Contact the machine manufacturer and request that the maximum number should be modified.
Continue program	The alarm is no longer displayed when the alarm cause has been removed. No further operator action required.

Alarm No.	
17193	Channel %1 block %2 the active tool is no longer on tool holder no. / spindle no. %3, program %4
Explanation	%1 = channel number, %2 = block number, label, %3 = tool holder no., spindle no., %4 = program name The tool on the specified tool holder / spindle on which the last tool change was performed as the master tool holder or master spindle, has been replaced. Example:N10 SETHTH(1) N20 T="Tool1" ;tool change to master tool holder 1 N30 SETMTH(2) N40 T1="Tool2" ;tool holder1 is only adjacent tool holder. ;Exchanging the tool does not result in the offset being deselected. N50 D5 ;New offset selection. There is currently no active tool to which D can refer.
Reaction	Alarm display Interface signals are set
Remedy	Change program: Set the required spindle as the main spindle or the tool holder as the master tool holder. Then reset any possible main spindle or master tool holder.
Continue program	Cancel the alarm with NC START and continue processing.

Alarm No.	
17194	Channel %1 block %2 no suitable tool found
Explanation	%1 = channel number, %2= block number, label An attempt was made to access a tool which has not been defined. The specified tool does not permit any access A tool with the required properties is not available.
Reaction	Alarm display Interface signals are set Correction block with reorganization
Remedy	Check access to tool: Is the language command parameterization correct? Does the state of the tool prevent access?
Continue program	Cancel the alarm with NC START and continue processing.

Alarm No.	
17200	Channel % 1 block % 2 tool data cannot be deleted
Explanation	%1 = channel number, %2= block number, label An attempt has been made to delete the tool data for a tool currently being used for machining from the part program. Tool data for tools involved in the current machining operation may not be deleted. This applies both for the tool preselected with T or that has been changed in place of another, and also for tools for which the constant grinding wheel peripheral speed or tool monitoring is active.
Reaction	Alarm display Interface signals are set Correction block with reorganization
Remedy	Check access to tool offset memory by means of $TC_DP1[t,d] = 0 or deselect tool.
Continue program	Cancel the alarm with NC START and continue processing.

Alarm No.	
17202	Channel %1 block %2 magazine data cannot be deleted
Explanation	%1 = channel number, %2= block number, label You have attempted to delete magazine data that presently cannot be deleted. It is not possible to delete a magazine that currently has the state "Tool is moving". A tool adapter currently assigned to a magazine location cannot be deleted. A tool adapter cannot be deleted if machine data $MN_MM_NUM_TOOL_ADAPTER has the value -1.
Reaction	Alarm display Interface signals are set Correction block with reorganization
Remedy	If an attempt to delete a magazine is rejected $TC_MAP1[m]=0 ; Delete magazine with m=magazine no. $TC_MAP1[0]=0 ; Delete all magazines $TC_MAP6[m]=0 ; Delete magazines and the tools contained in them, make sure that the magazine does not have the "Tool is moving" state when the magazine is called. If an attempt to delete a tool adapter is rejected $TC_ADPTT[a]=-1 ; Delete adapter with the number a $TC_ADPTT[0]=-1 ; Delete all adapters, then it has to be removed from the magazine location or the magazine locations by clearing the data.
Continue program	Cancel the alarm with NC START and continue processing.

Alarm No.	
17212	Channel %1 tool management: Load manual tool %3, duplo no. %2 onto spindle / tool holder %4
Explanation	%1 = channel number, %2 = duplo number, %3 = tool identifier, %4 = spindle / tool holder number Indicates that the specified manual tool must be brought to the specified tool holder or spindle before the program is continued. A manual tool is a tool whose data is registered in the NCK, but which is not assigned to a magazine location. As a result, it is not fully accessible for the purpose of automatic tool changes by the NCK and generally speaking, also the machine. The specified manual tool can also be a tool in a multitool. Then, the MT should be loaded.
Reaction	Alarm display

Alarm No.	
Remedy	Ensure that the specified tool is placed on the tool holder. The alarm is automatically canceled once the tool change ON command has been acknowledged by the PLC.
Continue program	The alarm is no longer displayed when the alarm cause has been removed. No further operator action required.

Alarm No.	
17214	Channel %1 tool management: Remove manual tool %3 from spindle / tool holder %2.
Explanation	%1 = channel number, %2 = tool holder number %3 = tool identifier, Indication that the specified manual tool must be removed from the specified tool holder or spindle before the program is continued. A manual tool is a tool whose data is registered in the NCK, but which is not assigned to a magazine location. As a result, it is not fully accessible for the purpose of automatic tool changes by the NCK and generally speaking, also the machine. The specified manual tool can also be in a multitool. Then, the MT should be removed.
Reaction	Alarm display
Remedy	Ensure that the specified tool is removed from the tool holder. The alarm is automatically canceled once the tool change ON command has been acknowledged by the PLC. Manual tools can only be used efficiently if appropriately supported by the PLC program.
Continue program	The alarm is no longer displayed when the alarm cause has been removed. No further operator action required.

Alarm No.	
17215	Channel %1 tool management: Remove manual tool %3 from buffer %2.
Explanation	%1 = channel number, %2 = buffer location %3 = tool identifier, Indication that the specified manual tool must be removed from the specified buffer before the program is continued. A manual tool is a tool whose data is registered in the NCK, but which is not assigned to a magazine location. As a result, it is not fully accessible for the purpose of automatic tool changes by the NCK and generally speaking, also the machine.
Reaction	Alarm display
Remedy	Ensure that the specified tool is removed from the tool holder. The alarm is automatically canceled once the tool change OFF command has been acknowledged by the PLC. Manual tools can only be used efficiently if appropriately supported by the PLC program.
Continue program	The alarm is no longer displayed when the alarm cause has been removed. No further operator action required.

Alarm No.	
17216	Channel %1 tool management: Remove manual tool from tool holder %4 and load manual tool %3, duplo no. %2.
Explanation	%1 = channel number, %2 = duplo number, %3 = tool identifier, %4 = tool holder (spindle) number Indicates that the specified manual tool must be loaded in the specified tool holder or spindle before the program is continued and that the manual tool located there must be removed. A manual tool is a tool whose data is registered in the NCK, but which is not assigned to a magazine location. As a result, it is not fully accessible for the purpose of automatic tool changes by the NCK and generally speaking, also the machine.
Reaction	Alarm display

Alarm No.	
Remedy	Make sure that the manual tools are exchanged. The alarm is automatically canceled once the tool change ON command has been acknowledged by the PLC. Manual tools can only be used efficiently if appropriately supported by the PLC program.
Program continuation	The alarm is no longer displayed when the alarm cause has been removed. No further operator action required.

Alarm No.	
17218	Channel %1 block %2 tool number %3 cannot become a manual tool.
Explanation	%1 = channel number, %2 = block number, %3 = tool number The specified tool has an owner location or there is a location in a real magazine reserved for this tool. As a consequence it cannot become a manual tool.
Reaction	Alarm display Interface signals are set Correction block with reorganization
Remedy	Correct the NC program By programming "DELRMRES" ensure that there is no reference to a real magazine location
Program continuation	Clear the alarm with NC START or the RESET key and continue the program.

Alarm No.	
17220	Channel %1 block %2 tool does not exist
Explanation	%1 = channel number, %2= block number, label If an attempt is made to access a tool using a T number, the tool name or the tool name and duplo no. that has not (yet) been defined, e.g. if tools are to be placed at magazine locations by programming $TC_MPP6 = "toolNo". This is possible only when both the magazine location and the tool specified by "toolNo" have been defined.
Reaction	Alarm display Interface signals are set NC start disable
Remedy	Correct the NC program
Continue program	Clear the alarm with NC start or RESET key and continue the program.

Alarm No.	
17224	Channel %1 block %2 tool T/D=%3 - tool type %4 is not permitted
Explanation	%1 = channel number, %2 = block number, label, %3 = contested T/D number, %4 = contested tool type It is not possible on this system to select tool offsets for tools of the specified tool type. The multitude of tool types can be restricted by the machine manufacturer or by the individual control models. Only use tool types permitted for this system. Check whether an error occurred when defining the tool.
Reaction	Alarm display Interface signals are set Interpreter stop NC start disable

Alarm No.	
Remedy	Correct NC program or tool data
Continue program	Clear the alarm with NC START or the RESET key and continue the program.

Alarm No.	
17230	Channel %1 block %2 duplo no. already assigned
Explanation	%1 = channel number, %2= block number, label The attempt was made to write the duplo number of a tool using a duplo number that has already been assigned to another tool (different T number).
Reaction	Alarm display Interface signals are set Interpreter stop NC start disable
Remedy	Correct the NC program
Continue program	Clear the alarm with NC START or the RESET key and continue the program.

Alarm No.	
17240	Channel %1 block %2 illegal tool definition
Explanation	%1 = channel number, %2= block number, label An attempt was made to change tool data, which would subsequently destroy the data consistency or would result in a contradictory definition.
Reaction	Alarm display Interface signals are set Interpreter stop NC start disable
Remedy	Correct the NC program
Continue program	Clear the alarm with NC START or the RESET key and continue the program.

Alarm No.	
17242	Channel %1 block %2 manual tool cannot be set as the function is not active
Explanation	%1 = channel number, %2= block number, label The manual tool function is not active. The tool cannot accept the state "manual tool" ($TC_TP8[Tool-No], bit 15=1 or "H8000"). The "manual tool" function is activated using machine data $MC_TOOL_CHANGE_ERROR_MODE, bit 1.
Reaction	Alarm display Interface signals are set Correction block with reorganization
Remedy	Correct the NC program
Continue program	Clear the alarm with NC START or the RESET key and continue the program.

Alarm No.	
17250	Channel %1 block %2 illegal magazine definition
Explanation	%1 = channel number, %2= block number, label An attempt was made to change magazine data, which would subsequently destroy the data consistency or would result in a contradictory definition.
Reaction	Alarm display Interface signals are set Interpreter stop NC start disable
Remedy	Correct the NC program
Continue program	Clear the alarm with NC START or the RESET key and continue the program.

Alarm No.	
17255	%?C(channel %1) block %2 magazine location hierarchies have been deleted
Explanation	%1 = channel number, %2= block number, label If $TC_MAMP2, bit 15 is changed, any existing hierarchies are changed because of the change in meaning.
Reaction	Alarm display
Remedy	Redefine magazine location hierarchies
Continue program	Clear the alarm with NC START or the RESET key and continue the program.

Alarm No.	
17260	Channel %1 block %2 illegal magazine location definition
Explanation	%1 = channel number, %2 = block number An attempt was made to change magazine location data, which would subsequently destroy the data consistency or would result in a contradictory definition. Example: If parameter $TC_MPP1 (= location type) is written with "spindle / tool holder location, then a conflict is possible with the adjacent machine data $MN_MM_NUM_TOOLHOLDERS. In this case, the remedy is either – if the control model permits it – to increase the value of $MN_MM_NUM_TOOLHOLDERS, or to correct the magazine definition. A tool must not be in two different magazine locations simultaneously - and a tool must not be contained in a multitool and a magazine location simultaneously.
Reaction	Alarm display Interface signals are set Correction block with reorganization
Remedy	Correct the NC program
Continue program	Clear the alarm with NC START or the RESET key and continue the program.

Alarm No.	
17262	Channel %1 block % illegal tool-adapter operation
Explanation	%1 = channel number, %2= block number, label This alarm is generated if you attempt to define or cancel the assignment between a tool adapter and a magazine location and the selected location already has another tool adapter and/or is already holding a tool or, if you are canceling the assignment, there is still another tool in the location. If machine data $MC_MM_NUM_SUMCORR has the value -1, then adapters cannot be created using a write operation to an adapter that has still not been defined. While the machine data has this value, you can only write adapter data to adapters which have already been (automatically) assigned to magazine locations.
Reaction	Alarm display Interface signals are set Correction block with reorganization
Remedy	Assign a maximum of one adapter to a magazine location. There must be no tool on the magazine location. Machine data $MC_MM_NUM_SUMCORR with value -1: If the alarm is generated when you are writing one of the system parameters $TC_ADAPTx (x=1,2,3,T), then you must change the write operation to ensure that it includes only adapter data which is already assigned to magazine locations.
Continue program	Clear the alarm with NC START or the RESET key and continue the program.

Alarm No.	
20150	Channel %1 tool management: PLC terminates interrupted command
Explanation	%1 = channel number Indication that the PLC has terminated an interrupted command (with alarm output) from the tool management - "tool change".
Reaction	Alarm display Interface signals are set
Remedy	For information only
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
20160	Channel %1 tool management: PLC can only terminate incorrectly canceled commands
Explanation	%1 = channel number Indication that the PLC wanted to cancel an active command from the tool management (tool change); or that there is no command active to be canceled. NCK rejects because the channel status is either "active" (cancel is then not allowed), or "reset" (then there is nothing to cancel).
Reaction	Alarm display Interface signals are set
Remedy	For information only
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
22066	Channel %1 tool management: Tool change not possible because tool %2 with duplo no. %3 is not in magazine %4
Explanation	%1 = channel number, %2 = string (identifier), %3 = duplo number, %4 = magazine number The desired tool change is not possible. The specified tool is not contained in the specified magazine. (NCK cannot contain tools that are not assigned to a magazine. No operations (movement, change) can be performed with these tools).
Reaction	NC start disable Alarm display Interface signals are set NC stop for alarm
Remedy	Please inform the authorized personnel / service department. Ensure that the specified tool is contained in the required magazine - or program a different tool, which should be changed. Check whether machine data $MC_RESET_MODE_MASK, $MC_START_MODE_MASK - and in turn the coupled machine data $MC_TOOL_RESET_NAME - match the actual definition data.
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
22067	Channel %1 tool management: Tool change not possible because no tool ready for use in tool group %2
Explanation	%1 = channel number, %2 = string (identifier) The desired tool change is not possible. The specified tool group does not contain a "ready to use" replacement tool which could be loaded. The tool monitoring function may have set all potentially suitable tools to the "disabled" status.
Reaction	NC start disable Alarm display Interface signals are set NC stop for alarm
Remedy	At the time of the requested tool change, ensure that there is a tool that can be used in the specified tool group. This can be achieved, e.g. by replacing disabled tools, or by manually releasing a disabled tool. Check whether the magazine data is correctly defined. Have all intended tools in the group been defined with the specified identifier and loaded?
Continue program	Clear the alarm with the RESET key. Restart the part program.

Alarm No.	
22068	Channel %1 tool management: No tool ready for use in tool group %3
Explanation	%1 = channel number, %2 = block number, label, %3 = string (identifier) The specified tool group does not contain a "ready to use" replacement tool which could be loaded. The tool monitoring function may have set all potentially suitable tools to the "disabled" status. The alarm can occur in conjunction with alarm 14710 (error when generating the INIT block). In this special situation, for instance, the NCK attempts to replace the disabled tool located in the spindle by an available replacement tool (which in this particular error case is not available). The operator must resolve this conflict, for instance by removing the tool located in the spindle using a movement command from the spindle (e.g. with an operator action via the HMI).
Reaction	NC start disable Alarm display Interface signals are set
Remedy	At the time of the requested tool change, ensure that there is a tool that can be used in the specified tool group. This can be achieved, for example, by replacing disabled tools, or by manually releasing a disabled tool. Check whether the magazine data is correctly defined. Have all intended tools in the group been defined with the specified identifier and loaded?
Continue program	Clear the alarm with NC START or the RESET key and continue the program.

Alarm No.	
22069	Channel %1 block %2 tool management: No tool available in tool group %3, program %4
Explanation	%1 = channel number, %2 = block number, label, %3 = string (identifier), %4 = program name The specified tool group does not contain a "ready to use" replacement tool which could be loaded. The tool monitoring function may have set all potentially suitable tools to the "disabled" status. Parameter %4 = program name facilitates the identification of the program containing the programming command (tool selection) that caused the error. This can be a subprogram or cycle, etc., which can no longer be identified from the display. If the parameter has not bee specified, then it is the currently displayed program.
Reaction	Alarm display Interface signals are set Correction block with reorganization
Remedy	At the time of the requested tool change, ensure that there is a tool that can be used in the specified tool group, e.g. by replacing disabled tools, or by manually releasing a disabled tool. Check whether the magazine data is correctly defined. Have all intended tools in the group been defined with the specified identifier and loaded?
Continue program	Clear the alarm with NC START or the RESET key and continue the program.

Alarm No.	
22070	TO unit %1 Please load tool T= %2 into magazine. Repeat data backup
Explanation	%1 = TO unit, %2 = T number of the tool Only issued if tool management is active. A data backup of the tool/magazine data has been started. The system has detected that the buffer magazine still contains one or more tools. During backup, these tools lose the information assigning them to a magazine and a location in the magazine. Therefore, it makes sense that all tools have been loaded the magazine at the time the data backup is performed. If this does not apply, when re-importing data, the magazine locations have the "reserved" status. You may have to reset this status manually. In the case of tools with a fixed-location coding, the loss of information about their location in the magazine is equivalent to a general empty location search on any subsequent change back to the magazine.
Reaction	Interface signals are set. Alarm display
Remedy	Make sure that there are no tools stored in the buffer magazine before you start to back up data. Repeat the data backup after removing the tools from the buffer magazine.
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
22071	TO unit %1 tool %2 duplo no. %3 is active but not in the current wear group
Explanation	%1 = TO units, %2 = T number of the tool, %3 = duplo number The "Wear group" function is active. In addition, the "Set tool to active status" setting is applied if a new wear group is activated. This setting can also be programmed with language command SETTA or started via similar functions on the OPI. It has been detected that more than one tool from the tool group has the "active" status. The tool which has the "active" status in an "inactive" wear grouping is named in the alarm. The alarm is for information purposes. It can be suppressed by setting bit 5 of MD 11410 SUPPRESS_ALARM_MASK.
Reaction	Alarm display Set interface signals
Remedy	Before you start the machining operation, make sure that the "active" status is not set for any of the tools in the magazine. You can do this by programming command SETTIA.
Continue program	Clear the alarm with the cancel key. No further operator action required.

Alarm No.	
400601	Incorrect configuration of loading points
Explanation	The PLC configuration in DB4 does not match the NC Configuration.
Reaction	Alarm display
Remedy	Correct tool management commissioning.
Continue program	Power-down the control and power-up again

Alarm No.	
400602	Incorrect spindle configuration
Explanation	The PLC configuration in DB4 does not match the NC Configuration.
Reaction	Alarm display
Remedy	Correct tool management commissioning.
Continue program	Power-down the control and power-up again

Alarm No.	
400603	Incorrect turret configuration
Explanation	The PLC configuration in DB4 does not match the NC Configuration.
Reaction	Alarm display
Remedy	Correct tool management commissioning.
Continue program	Power-down the control and power-up again

Alarm No.	
400604	Set change with M06 in machine data
Explanation	Change is possible only with M06 for the magazine type used (box-type, chain). Check for invalid settings when using turret magazines.
Reaction	Alarm display
Remedy	In the channel-specific data 22550: TOOL_CHANGE_MODE should be set to the value 1.
Continue program	Internal

Alarm No.	
410141	Number of loading points too high
Explanation	The PLC configuration in DB 4 does not match the NC Configuration.
Reaction	Alarm display
Remedy	Correct tool management commissioning.
Continue program	Switch control system OFF and ON again.

Alarm No.	
410142	Number of tool holders too high
Explanation	The PLC configuration in DB 4 does not match the NC Configuration.
Reaction	Alarm display
Remedy	Correct tool management commissioning.
Continue program	Switch control system OFF and ON again.

Alarm No.	
410143	Number of turrets too high
Explanation	The PLC configuration in DB 4 does not match the NC Configuration.
Reaction	Alarm display
Remedy	Correct tool management commissioning.
Continue program	Switch control system OFF and ON again.

Alarm No.	
410151	Magazine data for tool management missing in PLC
Explanation	No magazine data available in the PLC. Commissioning has not been completed although tool management option has been activated.
Reaction	Alarm display
Remedy	Press the "Create PLC data" softkey via SINUMERIK Operate when commissioning tool management.
Continue program	Internal

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
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)
TMFD	Tool management flat D number
TMMO	Tool monitoring function or monitor
TMMG	Tool magazine management
TM	Tool Management

A.2 Documentation overview

General documentation

- SINUMERIK — Sales brochure
  - SINUMERIK 840D sl
  - SINUMERIK 828D
  - SINUMERIK 828D BASIC T
- Catalog NC 62 SINUMERIK 840D sl type 1B
- Catalog NC 82 SINUMERIK 828D BASIC T/BASIC M SINAMICS S120 combi motors 1FK7 and 1PH8
- Catalog PM 21 SIMOTION, SINAMICS S120 and motors for production machines
- SINUMERIK — Configuration Manual EMC design guidelines
- SINUMERIK 840D sl 828D — System Manual Ctrl-Energy

User documentation

- SINUMERIK 840D sl 828D — Operating Manual
  - Universal
  - Turning
  - Milling
- SINUMERIK 840D sl 828D — Programming Manual
  - Basics
  - Job planning
  - Measuring cycles
- SINUMERIK 840D sl 828D — Programming Manual
  - ISO turning
  - ISO milling
- SINUMERIK 840D sl SINAMICS S120 — Diagnostics Manual
- SINUMERIK 828D SINAMICS S120 — Diagnostics Manual

Manufacturer / service documentation

- SINUMERIK 840D sl — Manual
  - NCU
  - Operation components and networking
- SINUMERIK 828D — Manual Commissioning Manual Service Manual
- SINUMERIK 840D sl — Commissioning Manual
  - CNC: NCK, PLC, drive
  - Basic software and operating software
- SINUMERIK 840D sl SINAMICS S120 — List Manual
  - Machine data
  - Interface signals
  - Variables
- SINUMERIK 828D SINAMICS S120 — List Manual
  - Machine data
  - Interface signals
  - Parameters
  - Variables
- SINUMERIK 840D sl SINAMICS S120 — System Manual Guidelines for machine configuration

Manufacturer / service documentation

- SINUMERIK 840D sl 828D — Function Manual
  - Basic functions
  - Extended functions
  - Special functions
  - Synchronized actions
  - ISO dialects
- SINUMERIK 840D sl — Function Manual Tool management
- SINAMICS S120 — Function Manual Drive functions
- SINUMERIK 840D sl SINAMICS S120 — Function Manual Safety Integrated
- SINUMERIK 828D SINAMICS S120 — Function Manual Safety Integrated

Info / training

- SINUMERIK — Training documents
  - Easy milling with ShopMill
  - Turning made easy with ShopTurn
- SINUMERIK — Manual Tool and mold making

Electronic documentation

DOConCD

My Documentation Manager

Industry Mall

Glossary

Access rights

Programs and other data are protected internally by a system of access rights based on seven levels: Three password levels for system manufacturers, machine manufacturers and users as well as a maximum of four keyswitch settings, which can be evaluated via the PLC.

Alarms

All messages and alarms are displayed on the operator panel in plain text. Alarms additionally with date and time as well as the appropriate symbol for the reset criterion. Alarms and messages are displayed separately.

Approach fixed machine point

Approach motion towards one of the predefined fixed machine points.

Archiving

Reading out data and/or directories to an external memory device.

Asynchronous subprogram (ASUP)

A part program which can be started asynchronously (independently) to the current program status by an interrupt signal (e.g. "rapid NC input" signal).

Auxiliary functions

Auxiliary functions can be used to transfer parameters to the PLC in -> part programs, where they trigger reactions which are defined by the machine manufacturer.

Axes

In accordance with their functional scope, the CNC axes are subdivided into:

Axis identifier

In accordance with DIN 66217, axes for a right-handed, rectangular -> coordinate system are identified using X, Y, Z. The identifiers A, B, C are used for rotary -> axes turning around X, Y, Z. Other letters can be used to identify additional parallel axes.

Axis/spindle interchange

An axis/spindle is permanently assigned to a specific channel via machine data. Using program commands it is possible to enable an axis/spindle and assign it to another channel.

Backup

Copies of the contents of storage medium (hard disk) are stored to an external memory device for the purpose of backing up and/or archiving data.

Basic Coordinate System

Cartesian coordinate system, which is mapped by transforming the machine coordinate system.
In the -> part program, the programmer uses the axis names of the Basic Coordinate System. The basic coordinate system exists in parallel to the -> machine coordinate system when no -> transformation is active. The difference between the systems relates to the axis identifiers.

Block

Block is the term given to any files required for creating and processing programs.

Part of a part program that is demarcated by a line feed. A distinction is made between main blocks and subblocks.

Block

Block is the term given to any files required for creating and processing programs.

Part of a part program that is demarcated by a line feed. A distinction is made between main blocks and subblocks.

Block search

The block search function allows any point in the part program to be selected, at which machining must start or be continued. The function is provided for the purpose of testing part programs or continuing machining after a program abort.

Booting

Loading the system program after power on.

Channel

A channel is characterized by the fact that it can process a part program independently of other channels. A channel exclusively controls the axes and spindles assigned to it. Parts that programs run on various channels can be coordinated by -> synchronization.

Channel structure

The channel structure makes it possible to process the programs of individual channels simultaneously and asynchronously.

Contour monitoring

The following error is monitored within a definable tolerance band as a measure of contour accuracy. Overloading of the drive, for example, may result in an unacceptably large following error. In such cases, an alarm is output and the axes are stopped.

D number

Number for the tool offset memory.

Data block

Data unit of the PLC that HIGHSTEP programs can access.
Data unit of the NC: Data blocks contain data definitions for global user data. This data can be initialized directly when it is defined.

Data word

A data unit, two bytes in size, within a PLC data block.

Dimensions, metric and inches

Position and pitch values can be programmed in inches in the machining program. The control is set to a basic system regardless of the programmable dimensions (G70/G71).

Editor

The editor makes it possible to create, edit, extend, join, and import programs, texts and program blocks.

File type

Possible types of files, e.g. part programs, zero offsets, R parameters, etc.

Fixed machine point

A point defined uniquely by the machine tool, such as the reference point.

Fixed-point approach

Machine tools can approach fixed points such as a tool change point, loading point, pallet change point, etc. The coordinates of these points are stored in the control. The control traverses the relevant axes, if possible in rapid traverse.

Frame

A frame is an arithmetic rule that transforms one Cartesian coordinate system into another Cartesian coordinate system. A frame contains the components work offset, rotation, scaling, mirroring.

Identifier

In accordance with DIN 66025, words are supplemented using identifiers (names) for variables (arithmetic variables, system variables, user variables), subprograms, key words, and words with multiple address letters. These supplements have the same meaning as the words with respect to block format. Identifiers must be unique. It is not permissible to use the same identifier for different objects.

Increment

Travel path length specification based on the number of increments. The number of increments can be stored as setting data or be selected by means of a suitably labeled key (i.e. 10, 100, 1000, 10000).

Keyswitch

The keyswitch is the operating switch of the CPU. The keyswitch is operated using a key that can be withdrawn.
The keyswitch on the machine control panel has four settings, to which functions are assigned by the operating system of the control. Further, the keyswitch has three differently colored keys, which can be removed in the specified positions.

Languages

The display texts of the operator navigation and the system messages and alarms are available in five languages:
German, English, French, Italian and Spanish.
Two of the specified languages are available in the control and can be selected (commissioning area).

Machine axes

Axes that are physically present in the machine tool.

Machine control panel

An operator panel on a machine tool with operating elements such as keys, rotary switches, etc., and simple indicators such as LEDs. It is used to directly influence the machine tool via the PLC.

Machine Coordinate System

A coordinate system, which is related to the axes of the machine tool.

Machine zero

Fixed point of the machine tool to which all (derived) measuring systems can be traced back.

Machining channel

Via a channel structure, parallel sequences of movements, such as positioning a loading gantry during machining, can shorten unproductive times. Here, a CNC channel must be regarded as a separate CNC control system with decoding, block preparation and interpolation.

Macro technique

Individual instructions in the programming language can be linked to create one overall instruction. This condensed instruction sequence is called by a user-defined name in the CNC program and the macro command executed in accordance with the individual instructions.

Magazine

In the tool management system, a distinction is made between:

Main block

A block preceded with ":" that contains all information to start the operating sequence in a -> part program.

Main program

-> Part program identified by a number or identifier in which further main programs, subroutines or -> cycles may be called.

Main run

The part program blocks which have been decoded and prepared in the preprocessing run are executed in the main run.

MDI

Operating mode of the control: Manual Data Input. In the MDI mode, individual program blocks or block sequences with no reference to a main program or subroutine can be input and executed immediately afterwards through actuation of the NC start key.

Messages

All messages programmed in the part program and all alarms recognized by the system are displayed on the operator panel in plain text. Alarms and messages are displayed separately.

Mirroring

Mirroring reverses the signs of the coordinate values of a contour, with respect to an axis. It is possible to mirror several axes simultaneously.

NC

Numerical Control: Numerical control (NC) includes all components of machine tool control: NCK, PLC, HMI, Com.

NCK

Numerical Control Kernel: Component of the NC that executes the part programs and basically coordinates the motion operations for the machine tool.

NRK

Numerical Robotic Kernel (operating system of the NCK)

OEM

The scope for implementing individual solutions (OEM applications) has been provided for machine manufacturers, who wish to create their own user interface or integrate technological functions in the control.

Offset memory

Data range in the control, in which the tool offset data is stored.

Oriented spindle stop

Stops the workpiece spindle with a specified orientation angle, e.g. to perform an additional machining operation at a specific position.

Override

Manual or programmable intervention features, which enable the user to override programmed feedrates or speeds in order to adapt them to a specific workpiece or material.

Part program

Series of instructions to the NC that act in concert to produce a particular workpiece. Likewise, this term applies to execution of a particular machining operation on a given raw part.

PLC

Programmable Logic Controller: Programmable logic controller, component of the NC controller: Programmable controller for processing the control logic of the machine tool.

PLC program memory

The PLC user program, the user data and the basic PLC program are stored together in the PLC user memory. The PLC user memory can be expanded up to 128 KB with memory expansions.

R parameter

Arithmetic parameter that can be set or queried by the programmer of the part program for any purpose in the program.

Reference point

Point on the machine tool with which the measuring system of the machine axes is referenced.

Reference point approach

If the utilized distance measuring system is not an absolute encoder then it is necessary to perform a reference point approach to ensure that the actual values returned by the measuring system match the machine coordinate values.

Replacement tool

Generally, a tool group contains several tools. Only the identifier is specified in the part program for the tool change. The tool with the "active" status is generally selected as the new tool. But if this is disabled, then one of the other -> twin tools, i.e. the replacement tool, is selected instead.

Replacement tool, tool group

Replacement tools have the same identifier and only differ in the duplo number. The replacement tools assigned to one identifier are also referred to as a tool group.

REPOS

- Repositioning to the contour per operator input
  The REPOS function can use the direction keys to reposition at the point of interruption.
- Repositioning to the contour per program
  Several approach strategies can be selected using program commands: Approach point of interruption, approach start of block, approach end of block, approach a point on the path between start of block and interruption.

Safety functions

The control has continually active monitoring functions, which detect malfunctions in the CNC, the programmable controller (PLC) and the machine at an early stage, in order to minimize the risk of damage to the tool, workpiece or machine. In the event of a fault, the machining operation is interrupted and the drives stopped. The cause of the malfunction is logged and output as an alarm. At the same time, the PLC is notified that a CNC alarm is pending.

Setting data

Data which communicates the properties of the machine tool to the NC, as defined by the system software.

Softkey

A key whose name appears on an area of the screen. The choice of softkeys displayed is dynamically adapted to the operating situation. The freely assignable function keys are assigned defined functions in the software.

Spindles

- Spindle = toolholder
  Toolholder is generally the location for the machining tool. However, the term “spindle” is frequently used in this general context.
- Main spindle = master spindle
  This is the spindle with the number defined by machine data MD $MC_SPIND_DEF_MASTER_SPIND. Language command SETMS(n) can be programmed to declare the spindle with number n as the master spindle. A channel has exactly one master spindle.
- Secondary spindle
  This term refers to all spindles that are not the master spindle.

Standard cycles

Standard cycles are provided for machining tasks which are frequently repeated:

Subblock

Block prefixed by "N" containing information for a machining step, such as a position parameter.

Subprogram

Sequence of statements of a part program that can be called repeatedly with different defining parameters. Cycles are a type of subprogram.

Synchronization

Instructions in part programs for coordination of the operations in different channels at specific machining points.

Synchronized axes

- Auxiliary function output
  During workpiece machining, technological functions can be output from the CNC program to the PLC. These auxiliary functions are used for example to control additional equipment for the machine tool, such as quills, grabbers, clamping chucks, etc.
- Fast auxiliary function output
  For time-critical switching functions, the acknowledgement times for the auxiliary functions can be minimized and unnecessary hold points in the machining process can be avoided.

System variable

A variable that exists without any input from the programmer of a part program. It is defined by a data type and the variable name preceded by the character $.

Tool nose radius compensation

Contour programming assumes that the tool is pointed. Because this is not actually the case in practice, the curvature radius of the tool used must be communicated to the control which then takes it into account. The curvature center is maintained equidistantly around the contour, offset by the curvature radius.

Tool offset

By programming a T function (5 decades, integer) in the block, you can select the tool. Every T number can be assigned up to 12 cutting edges (D addresses). The number of tools to be managed in the control is set while configuring.

Tool radius compensation

In order to be able to directly program a required workpiece contour, the control must traverse a path equidistant to the programmed contour, taking into account the radius of the tool used (G41/G42).

Transformation

Programming in a Cartesian coordinate system, execution in a non-Cartesian coordinate system (e.g. with machine axes as rotary axes).

User interface

The user interface (UI) is the display medium for a CNC in the form of a screen. It is laid out with eight horizontal and eight vertical softkeys.

User memory

All programs and data, such as part programs, subprograms, comments, tool offsets, and work offsets/frames, as well as channel and program user data, can be stored in the shared CNC user memory.

User program -> part program

User-defined variables

Users can define variables in the -> part program or data block (global user data) for their own use. A definition contains a data type specification and the variable name. See also -> System variable.

Variable definition

A variable definition includes the specification of a data type and a variable name. The variable names can be used to access the value of the variables.

Working memory

The working memory is a RAM memory in the -> CPU which is accessed by the processor to access the user program during program execution.

Workpiece

The part to be produced/machined by the machine tool or a workpiece as directory in which programs and other data are saved. Workpieces should be resaved in a directory.

Workpiece Coordinate System

The workpiece zero is the origin of the workpiece coordinate system. In machining operations programmed in workpiece coordinate system, the dimensions and directions refer to this system.

Workpiece zero

The workpiece zero forms the origin for the workpiece coordinate system. It is defined in terms of distances to the machine zero.

Index

$

$A_MONIFACT, 328
$A_MYMLN, 333
$A_MYMN, 333
$A_TOOLMLN, 327
$A_TOOLMN, 325
$A_USEDD, 340
$A_USEDND, 338
$A_USEDT, 339
$AC_MONMIN, 329
$AC_MSNUM, 354, 377
$AC_MTHNUM, 354, 377
$A-MONIFACT, 166
$P_AD[n], 353
$P_ADT[n], 353
$P_D_BEFORE_SEARCH, 357
$P_DL_BEFORE_SEARCH, 358
$P_DLNO, 351
$P_ISTEST, 367
$P_MAGA, 349
$P_MAGDISL, 342
$P_MAGDISS, 342
$P_MAGHLT, 348
$P_MAGNA, 349
$P_MAGNDIS, 342
$P_MAGNH, 348
$P_MAGNHLT, 348
$P_MAGNREL, 344
$P_MAGNS, 343
$P_MAGREL, 344
$P_MAGS, 343
$P_MSNUM, 354, 377
$P_MTHNUM, 355, 377
$P_MTHNUM_BEFORE_SEARCH, 357
$P_MTHSDC, 356
$P_TC, 352
$P_TCANG[n], 352
$P_TCDIFF[n], 352
$P_TH_OF_D, 285, 356
$P_TOOL, 350, 351
$P_TOOLD, 336
$P_TOOLL[n], 351
$P_TOOLND, 327
$P_TOOLNDL, 337
$P_TOOLNG, 333
$P_TOOLNO, 350
$P_TOOLNT, 336
$P_TOOLP, 350
$P_TOOLR, 351
$P_TOOLT, 336
$P_USEKT, $TC_TP11, 319
$P_VDITCP[x], 266
$TC_CARRx, 263
$TC_DPCx[t,D], 228
$TC_DPx[t,D], 226
$TC_MAMPx [n], 253
$TC_MAP10, 243
$TC_MAP3, 242
$TC_MAP8, 243
$TC_MAPCx[n], 243
$TC_MDPx[n,m], 250
$TC_MLSR[x,y], 260
$TC_MOPCx[t,D], 230
$TC_MOPx[t,D], 229
$TC_MPP1, 246
$TC_MPP5, 247
$TC_MPP6, 247
$TC_MPPCx[n,m], 248
$TC_MPPx[n,m], 244
$TC_MPTH[n,m], 249
$TC_SCPx[t,D], 231
$TC_TP1 and $TC_TP2, 235
$TC_TP3 to TP6, 235
$TC_TP7, 235
$TC_TP8, 236
$TC_TPCx[t], 239
$TC_TPGx[t], 239
$TC_TPx[t], 233

A

Access protection, 27
Acknowledgement
  Simplified, 206
Activate (internally)
  Wear group, 34
Activate wear group, 33
Adapter transformation, 185
Adding tools during machining, 94
Asynchronous transfer, 201

B

Background magazine, 34
Block execution, 85

Block search, 107, 109
Block search with calculation, 107
Block search, program test, 95
Block splitting, 85
Buffer, 29

C

Chain and box-type magazines, 30
Changing the acknowledgement data, 204
CHKDM, 180, 269
CHKDNO, 180
Code carrier data
    Data conversion, 394
Code carrier data formats ,
Code carrier data, dialog data
    Assignment, 402
Code carrier description file ,
Commissioning, 197
Consider adjacent location, 35
CRCEDN, 374
CREACE, 374
CREATO, 374
CRTOCE, 374
Cutting edge selection after tool change, 365

D

D numbers of replacement tools, 268
Data, 17
DB 1071, 198
DB 1072, 198
DB 1073, 198
DB 71, 198
DB 72, 198
DB 73, 198
DB 74, 198
Decoupling the tool management from the spindle number, 119
DefToolDat.txt, 382, 406
DELDL, 271
DELECE, 374
DELETO, 374
DELMLOWNER, 318
DELMLRES, 317
DELMT, 273
DELT, 273
DELTC, 304
Disable (internally)
    Wear group, 34
Disable monitored tools, 171
Disable wear group, 33
DL - Programming, 182
Duplo number, 17, 18
DZERO, 271

E

Empty spindle, 91
End acknowledgement, 203

F

Failure search strategy, 138
Free adapter data records, 187
Free selection of D numbers for every T, 179

G

GETACTT, 285
GETACTTD, 180, 269
GETDNO, 180, 270
GETEXET, 279
GETFREELOC, 311
GETSELT, 277
GETT, 274

H

HMI
    Overview, 21

I

IDENTCONNECTIONCONFIGURATION, 389
Identifiers, 17, 18
Internally assigned T numbers, 17

L

Loading magazine, 29
Loading points, 29
Loading stations, 29
Location coding, 75
Location-dependent offsets, 231
Location-dependent offsets, coarse, 231

M

Magazine configuration, 26

Magazine-location-related adapter data records, 187
Magazines
  Real, 26
Main spindle, 86
Manual tools, 94
Manufacturer's configuration, 196
MCIS-C script language ,
MMCSEM, 374
Monitoring status, 164
Monitoring types, 163
Multiple T selection, 91

N

NCK
  Overview, 21
NEWMT, 272
NEWT, 271

O

oem_sltmlistconfig.xml, 382
Operator panels, 17
OPI, 22, 223
OPI block AD, 262
OPI block C/S, 278, 280
OPI block TD, 233
OPI block TG, 239
OPI block TM, 241
OPI block TMC, 253, 260
OPI block TMV, 372
OPI block TO, 227, 229
OPI block TOE, TOET, 232
OPI block TOS, 231
OPI block TP, 245
OPI block TPM, 250
OPI block TS, 229
OPI block TT, 249
OPI block TU, 240
OPI block TUM, 244
OPI block TUP, 248
OPI block TUS, 230
Orientable tool holder, 263
Overview of data blocks, 198

P

ParamTM, 431
PI services, 374
PLC
  Overview, 21
PLC - NCK interfaces, 24
PLC at unloading, 156
PLC in test mode, 117
PLC services, 210
Positioning, 161
POSM, 289
POSM (multitool), 291
Predecoding, 85
Prepare to change tool in a secondary spindle, 87
Preparing and changing a tool, 74
Prewarning limit, 164
Program test, 116
Programming data
  Tool and cutting edge data,
Programming T/M06, 73
Programming the tool selection, 364
Protection levels, 27

Q

Quantity, 163

R

Relative D no. for each T, 178
Relocating, 159
Relocation by the PLC, 159
Replace tool search strategy, 144
Replacement tool, 85
Replacement tools, 92
RESETMON ,
RESETMON (multitool), 302
Retroload program, 153

S

Search for tool, 137
Search procedure for empty locations, 143
Search strategies, 137
Search strategy for empty locations, 140
Secondary spindle, 87
Service life, 163
SETDNO, 180, 270
SETMS, 287
SETMTH, 124, 287
SETPIECE, 167, 275
SETTA (multitool), 300
SETTIA, 298
SETTIA (multitool), 299
Setting-up offset, 181
Setting-up offsets, 231

Several magazines in one channel or one TO unit, 126
Several spindles / tool holders, 125
Several spindles in one channel or TO unit, 118
Special cases
    Empty spindle, 91
    Multiple T selection, 91
Spindle number, 119
Spindle/buffer DB 72, 87
SSL, 109
Subprogram replacement technique, 358
SUPPRESS_ALARM_MASK, 545
Synchronization, 203
Synchronized actions, 100
System variables, 223
systemconfiguration.ini, 382, 390

T

T number
    Internal, 18
T=location, 63
T=location number, 368
T=location, automatic tool selection, 322
TC_VAR, 117
TCA, 305
TCI, 310
TDIdentCfg.xml, 391
tdiidentcfg.xml, 382
Time monitoring, 166
TMCRTC, 374
TMCRTO, 374
TMFDPL, 374
TMFPBP, 374
TMGETT, 373, 375
TMMVTL, 374
TMPCIT, 374
TMPOSM, 374
TMRASS, 375
TOA area, 23
Tool change, 72
    Programming, 364
Tool change at the secondary spindle, 86
Tool change in the spindle, 87
Tool change of the main spindle, 86
Tool change preparation in a main spindle, 86
Tool change with turret, 92
Tool changes in the NCK using synchronized actions, 100
Tool changing errors, 93
Tool changing with T command, 74
Tool command
    Power failure., 196
Tool holder, 75
Tool holder numbers, 123
Tool Ident Connection
    Configuration,
    PLC connection,
Tool identifier, 17
Tool life decrementation, 166
Tool life monitoring, 165
Tool return transport, 75
Tool search, 137, 138
Tool search in wear group ,
TOOL_MANAGEMENT_TOOLHOLDER, 119
TOOLGNT, 323
TOOLGT, 323
Tool-related data, 233
ToolSize.mcc, 423
toolSpec.xml, 382
    Configuration,
Transport acknowledgement, 203
Traverse axes while tool is being changed, 87
TSEARC, 373, 374
Turret DB 73, 92
Turret magazine, 32

V

VDI signal, 170

W

Wear group, 33
Wear monitoring, 167
Wkonvert wizard
    Installation,
wkonvert.mcc, 382, 421
wkonvert.mcx, 382
wkonvert.txt
    Description file, 398
Workpiece count monitoring, 166