### Special functions

- **Preface**
- **F2: Multi-axis transformations**
- **G1: Gantry axes**
- **K6: Contour tunnel monitoring**
- **M3: Coupled axes**
- **R3: Extended stop and retract**
- **S9: Setpoint exchange - 840D sl only**
- **T3: Tangential control - 840D sl only**
- **TE01: Installation and activation of loadable compile cycles**
- **TE02: Simulation of compile cycles**
- **TE1: Clearance control - 840D sl only**
- **TE3: Speed/torque coupling, master-slave - 840D sl only**
- **TE4: Handling transformation package - 840D sl only**
- **TE6: MCS coupling - 840D sl only**
- **TE7: Continue machining at the contour (retrace support) - 840D sl only**
- **TE8: Cycle-independent path-synchronous switching signal output - 840D sl only**
- **TE9: Axis pair collision protection**

- **V2: Preprocessing**
- **W5: 3D tool radius compensation - 840D sl only**
- **W6: Path length evaluation - 840D sl only**

- **Z3: NC/PLC interface signals**

### Function Manual

**Valid for**

- **Controls**
  - SINUMERIK 840D sl / 840DE sl
  - SINUMERIK 828D

- **Software**
  - CNC software 4.5 SP1

**07/2012**

6FC5397-2BP40-3BA0
Warning notice system

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

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

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

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

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

**NOTICE**
- Indicates that an unintended result or situation can occur if the relevant information is not taken into account.

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

Qualified Personnel

The product/system described in this documentation may be operated only by personnel qualified for the specific task in accordance with the relevant documentation, 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.
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

This publication is intended for:
- Project engineers
- Technologists (from machine manufacturers)
- System startup engineers (Systems/Machines)
- 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 version

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

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

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

Technical Support

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

Installation

Structure of this Function Manual:

- Inner title (page 3) with the title of the Function Manual, the SINUMERIK controls as well as the software and the version for which this version of the Function Manual is applicable and the overview of the individual functional descriptions.
- Description of the functions in alphabetical order (e.g. A2, A3, B1, etc.)
- Appendix with:
  - List of abbreviations
  - Documentation overview
- Index of terms

Note

For detailed descriptions of data and alarms see:

- For machine and setting data:
  Detailed description of machine data (only electronically on DOConCD or DOConWEB)
- For NC/PLC interface signals:
  – Function Manual, Basic Functions; NC/PLC Interface Signals (Z1)
  – Function Manual, Basic Functions; NC/PLC Interface Signals (Z2)
  – Function Manual, Special Functions; NC/PLC Interface Signals (Z3)
- For alarms:
  Diagnostics Manual

Notation of system data

The following notation is applicable for system data in this documentation:

| Signal/Data          | Notation                                                                 | Example                                                                 |
|----------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------|
| NC/PLC interface     | ... NC/PLC interface signal: <signal address> (<signal name>)             | When the new gear stage is engaged, the following NC/PLC interface signals are set by the PLC program: DB31, ... DBX16.0-2 (actual gear stage A to C) DB31, ... DBX16.3 (gear is changed) |
| signals              |                                                                          |                                                                          |
| Machine data         | ... machine data: <Type><Number> <Complete Designator> (<Meaning>)       | Master spindle is the spindle stored in the machine data: MD20090 $MC_SPIND_DEF_MASTER_SPIND (position of deletion of the master spindle in the channel) |
| Setting data         | ... setting data: <Type><Number> <Complete Designator> (<Meaning>)       | The logical master spindle is contained in the setting data: SD42800 $SC_SPIND_ASSIGN_TAB[0] (spindle number converter) |
Note
Signal address
The description of functions include as <signal address> of an NC/PLC interface signal, only the address valid for SINUMERIK 840D sl. The signal address for SINUMERIK 828D should be taken from the data lists "Signals to/from ..." at the end of the particular description of functions.

Quantity structure
Explanations concerning the NC/PLC interface are based on the absolute maximum number of sequential components:
- Mode groups (DB11)
- Channels (DB21, etc.)
- Axes/spindles (DB31, etc.)

Data types
The following elementary data types are used in the control system:

| Type   | Meaning                                    | Range of values                                      |
|--------|--------------------------------------------|-----------------------------------------------------|
| INT    | Signed integers                            | -2147483648 ... +2147483647                         |
| REAL   | Figures with decimal point acc. to IEEE    | ±(2.2*10⁻³⁰⁸ ... 1.8*10⁺³⁰⁸)                        |
| BOOL   | Truth values TRUE (1) and FALSE (0)        | 1, 0                                                |
| CHAR   | ASCII characters                            | Corresponding to code 0 to 255                      |
| STRING | Character string, number of characters in [...] | Maximum of 200 characters (no special characters) |
| AXIS   | Axis names only                             | All axis identifiers in the channel                 |
| FRAME  | Geometrical parameters for moving, rotating, scaling, and mirroring |                                               |

Arrays can only be formed from similar elementary data types. Up to 3-dimensional arrays are possible.
Contents

Preface ................................................................................................................................................. 3

1 F2: Multi-axis transformations ........................................................................................................ 23

1.1 Brief description ................................................................................................................................ 23
1.1.1 5-axis Transformation .................................................................................................................. 23
1.1.2 3-axis and 4-axis transformation .................................................................................................. 25
1.1.3 Orientation transformation with a swiveling linear axis .............................................................. 26
1.1.4 Universal milling head .................................................................................................................... 27
1.1.5 Orientation axes ............................................................................................................................. 28
1.1.6 Cartesian manual travel .................................................................................................................. 29
1.1.7 Cartesian PTP travel ....................................................................................................................... 29
1.1.8 Generic 5-axis transformation ........................................................................................................ 29
1.1.9 Online tool length offset ............................................................................................................... 30
1.1.10 Activation via parts/program/softkey ......................................................................................... 30
1.1.11 Orientation compression ............................................................................................................. 30

1.2 5-axis transformation ....................................................................................................................... 31
1.2.1 Kinematic transformation .............................................................................................................. 31
1.2.2 Machine types for 5-axis transformation ...................................................................................... 31
1.2.3 Configuration of a machine for 5-axis transformation ................................................................. 33
1.2.4 Tool orientation ............................................................................................................................. 37
1.2.5 Singular positions and handling ..................................................................................................... 42

1.3 3-axis and 4-axis transformations ................................................................................................... 45

1.4 Transformation with swiveled linear axis ....................................................................................... 47

1.5 Cardan milling head ......................................................................................................................... 52
1.5.1 Fundamentals of cardan milling head ............................................................................................. 52
1.5.2 Parameterization ............................................................................................................................. 54
1.5.3 Traverse of the cardan milling head in JOG mode ........................................................................ 55

1.6 Programming of the 3- to 5-axis transformation ........................................................................... 56

1.7 Generic 5-axis transformation and variants ..................................................................................... 57
1.7.1 Functionality .................................................................................................................................. 57
1.7.2 Description of machine kinematics ............................................................................................... 58
1.7.3 Generic orientation transformation variants .................................................................................. 59
1.7.4 Parameterization of orientable toolholder data ............................................................................. 61
1.7.5 Extension of the generic transformation to six axes - 840D sl only ............................................. 66
1.7.6 Extension of the generic transformation to seven axes - 840D sl only ........................................ 70
1.7.7 Cartesian manual travel with generic transformation ..................................................................... 74

1.8 Restrictions for kinematics and interpolation ..................................................................................... 77
1.8.1 Singularities of orientation ............................................................................................................ 78

1.9 Orientation ........................................................................................................................................... 81
1.9.1 Basic orientation ............................................................................................................................ 81
1.9.2 Orientation movements with axis limits ....................................................................................... 83
1.9.3 Orientation compression ............................................................................................................... 85

Special functions
Function Manual, 07/2012, 6FC5397-2BP40-3BA0
## Contents

| Section | Title                                                                 | Page |
|---------|-----------------------------------------------------------------------|------|
| 1.9.4   | Smoothing of orientation characteristic                               | 89   |
| 1.9.5   | Orientation relative to the path                                      | 92   |
| 1.9.6   | Programming of orientation polynomials                                 | 96   |
| 1.9.7   | Tool orientation with 3-/4-/5-axis transformations                      | 99   |
| 1.9.8   | Orientation vectors with 6-axis transformation                         | 99   |
| 1.10    | Orientation axes                                                      | 100  |
| 1.10.1  | JOG mode                                                              | 102  |
| 1.10.2  | Programming for orientation transformation                            | 103  |
| 1.10.3  | Programmable offset for orientation axes                              | 105  |
| 1.10.4  | Orientation transformation and orientable tool holders                | 107  |
| 1.10.5  | Modulo display of orientation axes                                    | 107  |
| 1.11    | Orientation vectors                                                   | 108  |
| 1.11.1  | Polynomial interpolation of orientation vectors                        | 108  |
| 1.11.2  | Rotations of orientation vector                                       | 112  |
| 1.11.3  | Extended interpolation of orientation axes                             | 116  |
| 1.12    | Online tool length offset                                             | 122  |
| 1.13    | Examples                                                              | 126  |
| 1.13.1  | Example of a 5-axis transformation                                     | 126  |
| 1.13.2  | Example of a 3-axis and 4-axis transformation                          | 129  |
| 1.13.2.1| Example of a 3-axis transformation                                     | 129  |
| 1.13.2.2| Example of a 4-axis transformation                                     | 129  |
| 1.13.3  | Example of a universal milling head                                   | 130  |
| 1.13.4  | Example for orientation axes                                          | 131  |
| 1.13.5  | Examples for orientation vectors                                       | 133  |
| 1.13.5.1| Example for polynomial interpretation of orientation vectors           | 133  |
| 1.13.5.2| Example of rotations of orientation vector                            | 134  |
| 1.13.6  | Examples for generic axis transformations                              | 135  |
| 1.13.6.1| Example of a generic 6-axis transformation                             | 137  |
| 1.13.6.2| Example of a generic 7-axis transformation                             | 138  |
| 1.13.6.3| Example for the modification of rotary axis motion                     | 138  |
| 1.13.7  | Example: Compression of an orientation                                | 139  |
| 1.14    | Data lists                                                            | 140  |
| 1.14.1  | Machine data                                                          | 140  |
| 1.14.1.1| General machine data                                                   | 140  |
| 1.14.1.2| Channelspecific machine data                                          | 140  |
| 1.14.2  | Setting data                                                          | 144  |
| 1.14.2.1| General setting data                                                   | 144  |
| 1.14.2.2| Channelspecific setting data                                          | 144  |
| 1.14.3  | Signals                                                               | 145  |
| 1.14.3.1| Signals from channel                                                  | 145  |

2 **G1: Gantry axes** ........................................................................................................ 147

2.1 Brief description ........................................................................................................ 147
2.2 "Gantry axes" function .............................................................................................. 148
2.2.1 Definition of a gantry grouping .......................................................................... 148
2.2.2 Monitoring the synchronism difference .................................................................. 149
2.2.3 Extended monitoring of the synchronism difference ............................................. 150
2.2.4 Referencing and synchronization of gantry axes .................................................. 150
2.2.5 Control dynamics ................................................................................................... 151
## Special functions

| Section | Title | Page |
|---------|-------|------|
| 4.1.5   | Status of coupling | 202  |
| 4.1.6   | Dynamics limit     | 202  |
| 4.1.7   | Supplementary conditions | 203  |
| 4.1.8   | Examples           | 203  |
| 4.2     | Curve tables - 840D sl only | 205  |
| 4.2.1   | Product brief      | 205  |
| 4.2.1.1 | Function           | 205  |
| 4.2.1.2 | Preconditions      | 205  |
| 4.2.2   | General functionality | 206  |
| 4.2.3   | Memory organization | 207  |
| 4.2.4   | Commissioning      | 209  |
| 4.2.4.1 | Memory configuration | 209  |
| 4.2.4.2 | Tool radius compensation | 210  |
| 4.2.4.3 | Specification of memory type | 210  |
| 4.2.5   | Programming        | 211  |
| 4.2.6   | Access to table positions and table segments | 218  |
| 4.2.7   | Activation/deactivation | 222  |
| 4.2.8   | Modulo-leading axis special case | 223  |
| 4.2.9   | Behavior in AUTOMATIC, MDA and JOG modes | 224  |
| 4.2.10  | Effectiveness of PLC interface signals | 224  |
| 4.2.11  | Diagnosing and optimizing utilization of resources | 225  |
| 4.2.12  | Supplementary conditions | 229  |
| 4.2.13  | Examples           | 230  |
| 4.3     | Master value coupling - 840D sl only | 232  |
| 4.3.1   | Product brief      | 232  |
| 4.3.1.1 | Function           | 232  |
| 4.3.1.2 | Preconditions      | 232  |
| 4.3.2   | General functionality | 232  |
| 4.3.3   | Programming        | 236  |
| 4.3.4   | Behavior in AUTOMATIC, MDA and JOG modes | 240  |
| 4.3.5   | Effectiveness of PLC interface signals | 242  |
| 4.3.6   | Special characteristics of the axis master value coupling function | 242  |
| 4.3.7   | Supplementary conditions | 243  |
| 4.4     | Electronic gear (EG) - 840D sl only | 244  |
| 4.4.1   | Product brief      | 244  |
| 4.4.1.1 | Function           | 244  |
| 4.4.1.2 | Preconditions      | 245  |
| 4.4.2   | Definition of an EG axis group | 253  |
| 4.4.3   | Activating an EG axis group | 254  |
| 4.4.4   | Deactivating an EG axis group | 258  |
| 4.4.5   | Deleting an EG axis group | 259  |
| 4.4.6   | Interaction between rotation feedrate (G95) and electronic gearbos | 260  |
| 4.4.7   | Response to POWER ON, RESET, operating mode change, block search | 260  |
| 4.4.8   | System variables for electronic gearbos | 261  |
| 4.4.9   | Examples           | 263  |
| 4.4.9.1 | Example using linear couplings | 263  |
| 4.4.9.2 | Extended example with non-linear components | 266  |
| 4.5     | Generic coupling   | 273  |
| 4.5.1   | Brief description  | 273  |
| 4.5.1.1 | Function           | 273  |
Contents

4.5.1.2 Requirements ............................................................................................................................. 274
4.5.2 Fundamentals ............................................................................................................................... 277
4.5.2.1 Coupling module ....................................................................................................................... 277
4.5.2.2 Keywords and coupling characteristics .................................................................................... 279
4.5.2.3 System variables ....................................................................................................................... 281
4.5.3 Creating/deleting coupling modules ............................................................................................ 283
4.5.3.1 Creating a coupling module (CPDEF) .................................................................................... 283
4.5.3.2 Delete coupling module (CPDEL) .......................................................................................... 284
4.5.3.3 Defining leading axes (CPLDEF or CPDEF+CPLA) ............................................................... 285
4.5.3.4 Delete leading axes (CPLDEL or CPDEL+CPLA) ................................................................. 286
4.5.4 Switching coupling on/off .......................................................................................................... 288
4.5.4.1 Switching on a coupling module (CPON) .............................................................................. 288
4.5.4.2 Switch off coupling module (CPOF) ....................................................................................... 289
4.5.4.3 Switching on leading axes of a coupling module (CPLON) ..................................................... 290
4.5.4.4 Switching off leading axes of a coupling module (CPOF) ....................................................... 291
4.5.4.5 Implicit creation and deletion of coupling modules ................................................................. 292
4.5.5 Programming coupling characteristics ....................................................................................... 292
4.5.5.1 Coupling rule (CPLNUM, CPLDEN, CPLTID) ........................................................................ 292
4.5.5.2 Coupling relationship (CPLSETVAL) ..................................................................................... 295
4.5.5.3 Co-ordinate reference (CPFRS) ............................................................................................. 296
4.5.5.4 Block change behavior (CPBC) ............................................................................................... 297
4.5.5.5 Synchronized position of the following axis when switching on (CPFPOS+CPON) ............... 298
4.5.5.6 Synchronized position of the leading axis when switching on (CPLPOS) ............................. 300
4.5.5.7 Synchronization mode (CPFMSON) ......................................................................................... 301
4.5.5.8 Behavior of the following axis at switch-on (CPFMON) ........................................................ 303
4.5.5.9 Behavior of the following axis at switch-off (CPFMOF) ........................................................ 304
4.5.5.10 Position of the following axis when switching off (CPFPOS+CPOF) ................................. 305
4.5.5.11 Condition at RESET (CPMRESET) ..................................................................................... 306
4.5.5.12 Condition at parts program start (CPMSTART) ................................................................... 307
4.5.5.13 Status during part program start in search run via program test (CPMPRT) ......................... 308
4.5.5.14 Offset / scaling (CPLINTR, CPLINSC, CPLOUTTR, CPLOUTSC) ........................................... 310
4.5.5.15 Synchronism monitoring stage 1 (CSYNCP, CPSYNFIP, CPSYNCOV, CPSYNFIV) ............ 313
4.5.5.16 Synchronous operation monitoring stage 2 (CSYNCP2, CPSYNFIP2) ............................. 317
4.5.5.17 Reaction to stop signals and commands (CPMBRAKE) ...................................................... 322
4.5.5.18 Response to certain NC/PLC interface signals (CPMVDI) .................................................. 323
4.5.5.19 Alarm suppression (CPMALARM) ...................................................................................... 328
4.5.6 Coupling cascading ..................................................................................................................... 329
4.5.7 Compatibility ............................................................................................................................... 330
4.5.7.1 Adaptive cycles ....................................................................................................................... 330
4.5.7.2 Coupling types (CPSETTYPE) .............................................................................................. 331
4.5.7.3 Projected coupling (CPRES) .................................................................................................. 336
4.5.8 Cross-channel coupling, axis replacement .................................................................................. 337
4.5.9 Behavior with rotary axes ............................................................................................................ 337
4.5.10 Behavior during POWER ON, ................................................................................................. 339
4.5.11 Disturbance characteristic ........................................................................................................ 340
4.5.11.1 Rapid stop ............................................................................................................................. 340
4.5.12 Tracking the deviation from synchronism .................................................................................. 341
4.5.12.1 Fundamentals ......................................................................................................................... 341
4.5.12.2 Measuring the deviation from synchronism ......................................................................... 342
4.5.12.3 Entering the deviation from synchronism directly ............................................................... 345
4.5.12.4 Synchronism correction ........................................................................................................ 345
4.5.12.5 Diagnostics for synchronism correction ................................................................................ 346
### Contents

| Section                                                                 | Page |
|------------------------------------------------------------------------|------|
| 4.5.12.6 Resetting synchronism correction                              | 353  |
| 4.5.12.7 Limitations and constraints                                   | 355  |
| 4.5.13 Examples                                                        | 356  |
| 4.5.13.1 Programming examples                                         | 356  |
| 4.5.13.2 Adapt adaptive cycle                                          | 358  |
| 4.6 Dynamic response of following axis                                | 359  |
| 4.6.1 Parameterized dynamic limits                                     | 359  |
| 4.6.2 Programmed dynamic limits                                       | 359  |
| 4.6.2.1 Programming (VELOLIMA, ACCLIMA)                               | 359  |
| 4.6.2.2 Examples                                                       | 362  |
| 4.6.2.3 System variables                                               | 363  |
| 4.7 General supplementary conditions                                   | 364  |
| 4.8 Data lists                                                         | 365  |
| 4.8.1 Machine data                                                     | 365  |
| 4.8.1.1 NC-specific machine data                                      | 365  |
| 4.8.1.2 Channelspecific machine data                                  | 365  |
| 4.8.1.3 Axis/spindlespecific machine data                             | 366  |
| 4.8.2 Setting data                                                     | 366  |
| 4.8.2.1 Axis/spindle-specific setting data                            | 366  |
| 4.8.3 System variables                                                 | 367  |
| 4.8.4 Signals                                                         | 370  |
| 4.8.4.1 Signals to axis/spindle                                       | 370  |
| 4.8.4.2 Signals from axis/spindle                                     | 370  |
| 5 R3: Extended stop and retract                                       | 373  |
| 5.1 Brief description                                                  | 373  |
| 5.2 Control-managed ESR - 840D sl only                                | 374  |
| 5.2.1 Extended stop and retract (ESR)                                  | 374  |
| 5.2.2 Drive-independent reactions                                     | 375  |
| 5.2.3 Power failure detection and bridging                            | 376  |
| 5.2.4 NC-controlled extended stop                                     | 377  |
| 5.2.5 Retract                                                         | 379  |
| 5.2.6 Trigger sources                                                  | 384  |
| 5.2.7 Logic gating functions: Source and reaction linking             | 385  |
| 5.2.8 Activation                                                      | 385  |
| 5.2.9 Configuring aids for ESR                                        | 386  |
| 5.2.10 Control system response                                        | 389  |
| 5.2.10.1 Axis behavior depending on the enable signals                | 389  |
| 5.2.10.2 POWER OFF/POWER ON                                            | 391  |
| 5.2.10.3 Operating mode change, NC Stop, reset                        | 391  |
| 5.2.10.4 Part program start, NC start                                 | 392  |
| 5.2.10.5 Alarm behavior                                               | 392  |
| 5.2.10.6 Block search, REPOS                                          | 393  |
| 5.3 ESR executed autonomously in the drive                            | 393  |
| 5.3.1 Fundamentals                                                    | 393  |
| 5.3.2 Configuring stopping in the drive                                | 394  |
| 5.3.3 Configuring retraction in the drive                              | 395  |
| 5.3.4 Configuring generator operation in the drive                    | 397  |
| 5.3.5 ESR is enabled via system variable                              | 398  |
| 5.3.6 Triggering ESR via system variable                              | 399  |
## Contents

| Section                                                                 | Page |
|------------------------------------------------------------------------|------|
| 5.3.7 Feedback of the ESR status                                       | 391  |
| 5.3.8 Acknowledge ESR reactions                                        | 392  |
| 5.3.9 Configuring ESR in the part program                              | 393  |
| 5.3.10 ESR and Safety Integrated (840D sl)                             | 397  |
| 5.3.11 ESR and Safety Integrated (828D)                                | 398  |
| 5.4 Boundary conditions                                                | 400  |
| 5.5 Examples                                                           | 401  |
| 5.5.1 NC-controlled reactions                                          | 401  |
| 5.5.2 Retraction while thread cutting                                  | 402  |
| 5.5.3 Rapid lift using ASUB and fast input                             | 403  |
| 5.5.4 Rapid lift, absolute and incremental                             | 403  |
| 5.5.5 Lift fast with linear relation of axes                           | 404  |
| 5.6 Data lists                                                          | 405  |
| 6.8.1 Machine data                                                     | 405  |
| 6.8.1.1 Channelspecific machine data                                   | 405  |
| 6.8.1.2 Axis/spindlespecific machine data                              | 405  |
| 6.8 Data lists                                                          | 405  |
| 6.8.1 Machine data                                                     | 405  |
| 5.6.1 Machine data                                                     | 405  |
| 5.6.1.1 Channel specific machine data                                  | 405  |
| 5.6.1.2 Axis/spindle specific machine data                             | 405  |
| 5.6.2 System variables                                                 | 405  |
| 5.6.3 Signals                                                          | 406  |
| 5.6.3.1 Signals to channel                                              | 406  |
| 5.6.3.2 Signals to axis/spindle                                        | 406  |
| 5.6.3.3 Signals from axis/spindle                                      | 406  |
| 5.6.4 Interrupts                                                       | 413  |
| 5.6.5 Position control loop                                            | 413  |
| 5.6.6 Reference points                                                 | 413  |
| 5.6.7 Supplementary conditions                                         | 414  |
| 6.8.1.1 Machine data                                                   | 415  |
| 5.5.1.1 Machine data                                                   | 415  |
| 5.5 NC-controlled reactions                                             | 401  |
| 5.5.2 Rapid lift using ASUB and fast input                              | 403  |
| 5.5.3 Rapid lift, absolute and incremental                             | 403  |
| 5.5.4 Lift fast with linear relation of axes                            | 404  |
| 5.5.5 Limit angle                                                      | 427  |
| 7.3.7 Feedback of the ESR status                                       | 391  |
| 7.3.8 Acknowledge ESR reactions                                        | 392  |
| 7.3.9 Configuring ESR in the part program                              | 393  |
| 7.3.10 ESR and Safety Integrated (840D sl)                             | 397  |
| 7.3.11 ESR and Safety Integrated (828D)                                | 398  |
| 7.4 Limit angle                                                        | 427  |
## Special functions

| Section | Title                                                                 | Page |
|---------|----------------------------------------------------------------------|------|
| 7.5     | Supplementary conditions                                           | 428  |
| 7.6     | Examples                                                            | 429  |
| 7.7     | Data lists                                                          | 431  |
| 7.7.1   | Machine data                                                        | 431  |
| 7.7.1.1 | Axis/spindlespecific machine data                                   | 431  |
| 7.7.2   | System variables                                                    | 431  |
| 8       | TE01: Installation and activation of loadable compile cycles        | 433  |
| 8.1     | Loading compile cycles                                              | 435  |
| 8.1.1   | Loading a compile cycle with SINUMERIK Operate                     | 435  |
| 8.1.2   | Loading a compile cycle with HMI Advanced                           | 436  |
| 8.1.3   | Loading a compile cycle from an external computer with WinSCP3     | 436  |
| 8.2     | Interface version compatibility                                     | 438  |
| 8.3     | Software version of a compile cycle                                 | 440  |
| 8.4     | Activating the technological functions in the NCK                  | 441  |
| 8.5     | Function-specific startup                                           | 442  |
| 8.6     | Creating alarm texts                                               | 443  |
| 8.6.1   | Creating alarm texts with SINUMERIK Operate                         | 443  |
| 8.6.2   | Creating alarm texts with HMI Advanced                              | 444  |
| 8.7     | Upgrading a compile cycle                                           | 445  |
| 8.8     | Deleting a compile cycle                                            | 446  |
| 8.9     | Data lists                                                          | 447  |
| 8.9.1.1 | NC-specific machine data                                            | 447  |
| 8.1.1   | Loading a compile cycle with SINUMERIK Operate                      | 435  |
| 8.1.2   | Loading a compile cycle with HMI Advanced                           | 436  |
| 8.1.3   | Loading a compile cycle from an external computer with WinSCP3     | 436  |
| 8.2     | Interface version compatibility                                     | 438  |
| 8.3     | Software version of a compile cycle                                 | 440  |
| 8.4     | Activating the technological functions in the NCK                  | 441  |
| 8.5     | Function-specific startup                                           | 442  |
| 8.6     | Creating alarm texts                                               | 443  |
| 8.6.1   | Creating alarm texts with SINUMERIK Operate                         | 443  |
| 8.6.2   | Creating alarm texts with HMI Advanced                              | 444  |
| 8.7     | Upgrading a compile cycle                                           | 445  |
| 8.8     | Deleting a compile cycle                                            | 446  |
| 8.9     | Data lists                                                          | 447  |
| 8.9.1.1 | NC-specific machine data                                            | 447  |
| 9       | TE02: Simulation of compile cycles                                  | 449  |
| 9.1     | Brief description                                                   | 449  |
| 9.1.1   | Function                                                            | 449  |
| 9.1.2   | OEM transformations                                                 | 450  |
| 10      | TE1: Clearance control - 840D sl only                              | 453  |
| 10.1    | Brief description                                                   | 453  |
| 10.1.1  | General                                                             | 453  |
| 10.1.2  | Function description                                                | 454  |
| 10.2    | Clearance control                                                   | 456  |
| 10.2.1  | Control dynamics                                                    | 456  |
| 10.2.2  | Velocity feedforward control                                        | 458  |
| 10.2.3  | Control loop structure                                              | 459  |
| 10.2.4  | Compensation vector                                                 | 460  |
| 10.3    | Technological features of clearance control                          | 463  |
| 10.4    | Sensor collision monitoring                                         | 465  |
| 10.5    | Startup                                                             | 466  |
| 10.5.1  | Activating the technological function                               | 466  |
| 10.5.2  | Configuring the memory                                              | 466  |
| 10.5.3  | Parameter settings for input signals (840D sl)                      | 466  |
10.5.4 Parameters of the programmable compensation vector .......................................................... 467
10.5.5 Parameter settings for clearance control .................................................................................. 469
10.5.6 Starting up clearance control .................................................................................................... 470
10.6 Programming .............................................................................................................................. 473
10.6.1 Activating and deactivating clearance control (CLC) .............................................................. 473
10.6.2 Closed-loop control gain (CLC_GAIN) .................................................................................... 479
10.6.3 Limiting the control range (CLC_LIM) ...................................................................................... 481
10.6.4 Direction-dependent traversing motion disable ...................................................................... 482
10.6.5 Voltage offset, can be set on a block-specific basis (CLC_VOFF) ............................................. 484
10.6.6 Voltage offset definable by synchronized action ........................................................................ 485
10.6.7 Selection of the active sensor characteristic (CLC_SEL) ......................................................... 486
10.7 Function-specific display data ...................................................................................................... 487
10.7.1 Channel-specific GUD variables .............................................................................................. 487
10.7.2 OPI variable ............................................................................................................................. 489
10.8 Function-specific alarm texts ...................................................................................................... 491
10.9 Supplementary conditions .......................................................................................................... 492
10.9.1 I/O modules ............................................................................................................................... 492
10.9.1.1 External smoothing filters ...................................................................................................... 493
10.9.2 Function-specific supplementary conditions ............................................................................ 493
10.10 Data lists .................................................................................................................................. 496
10.10.1 Machine data .......................................................................................................................... 496
10.10.1.1 NC-specific machine data ....................................................................................................... 496
10.10.1.2 Channelspecific machine data .............................................................................................. 496
10.10.1.3 Axis/spindlespecific machine data ......................................................................................... 497
10.10.2 Drive parameters (SINAMICS S120) ....................................................................................... 498
10.10.3 Signals .................................................................................................................................. 498
10.10.3.1 Signals to channel .................................................................................................................. 498
10.10.3.2 Signals from channel ............................................................................................................. 498
11 TE3: Speed/torque coupling, master-slave - 840D sl only ............................................................ 499
11.1 Brief description .......................................................................................................................... 499
11.2 Coupling diagram ....................................................................................................................... 500
11.3 Configuring a coupling ................................................................................................................ 501
11.4 Torque compensatory controller ............................................................................................... 504
11.5 Tension torque ............................................................................................................................ 507
11.6 Closing/opening a coupling ......................................................................................................... 511
11.7 Response on activation/deactivation ........................................................................................... 513
11.8 Supplementary conditions .......................................................................................................... 517
11.8.1 Speed/torque coupling (SW 6 and higher) ................................................................................ 517
11.8.2 Axial NC/PLC interface signals ............................................................................................... 518
11.8.3 Interaction with other functions ............................................................................................... 519
11.9 Examples .................................................................................................................................. 525
11.9.1 Master-slave coupling between AX1=Master and AX2=Slave .................................................. 525
11.9.2 Close coupling via the PLC ...................................................................................................... 525
11.9.3 Close/separate coupling via part program ................................................................................. 526
11.9.4 Release the mechanical brake .................................................................................................. 527
13 TE6: MCS coupling - 840D sl only ................................................................. 589
13.1 Brief description ..................................................................................... 589
13.2 Description of MCS coupling functions ............................................. 591
13.2.1 Defining coupling pairs ................................................................... 591
13.2.2 Switching the coupling ON/OFF ..................................................... 592
13.2.3 Tolerance window ............................................................................ 593
13.2.4 Memory configuration: Block memory ........................................... 593
13.3 Description of collision protection ......................................................... 594
13.3.1 Defining protection pairs ................................................................. 594
13.3.2 Switching collision protection ON / OFF ......................................... 594
13.3.3 Configuring example ....................................................................... 596
13.4 User-specific configurations ................................................................. 597
13.5 Special operating states ....................................................................... 598
13.6 Boundary conditions .......................................................................... 599
13.7 Data lists ............................................................................................. 600
13.7.1 Machine data .................................................................................. 600
13.7.1.1 Channelspecific machine data .................................................... 600
13.7.1.2 Axis/spindlespecific machine data ............................................. 600
14 TE7: Continue machining at the contour (retrace support) - 840D sl only ................................................................. 601
14.1 Brief description ................................................................................... 601
14.2 Function description ......................................................................... 603
14.2.1 Function .......................................................................................... 603
14.2.2 Definition of terms ........................................................................ 604
14.2.3 Functional sequence (principle) ....................................................... 605
14.2.4 Maximum retraceable contour area ............................................... 608
14.3 Startup ................................................................................................. 609
14.3.1 Activation ........................................................................................ 609
14.3.2 Definition of the RESU working plane .......................................... 609
14.3.3 Memory configuration: Block memory ........................................... 609
14.3.4 Memory configuration: Heap memory ............................................ 610
14.3.5 RESU main program memory area ................................................. 612
14.3.6 Storage of the RESU subroutines .................................................. 613
14.3.7 ASUB enable .................................................................................. 613
14.3.8 PLC user program ......................................................................... 614
14.4 Programming ....................................................................................... 615
14.4.1RESU Start/Stop/Reset (CC_PREPRE) ........................................... 615
14.5 RESU-specific part programs ............................................................... 617
14.5.1 Overview ......................................................................................... 617
14.5.2 Main program (CC_RESU.MPF) ...................................................... 618
14.5.3 INI program (CC_RESU_INI.SPF) .................................................. 619
14.5.4 END program (CC_RESU_END.SPF) ............................................ 620
14.5.5 Retrace support ASUB (CC_RESU_BS_ASUP.SPF) ..................... 621
14.5.6 RESU ASUB (CC_RESU_ASUP.SPF) .............................................. 622
14.6 Retrace support .................................................................................. 623
14.6.1 General ......................................................................................... 623
Contents

14.6.2 Block search with calculation on contour ................................................................. 623
14.6.3 Reposition ............................................................................................................... 624
14.6.4 Temporal conditions concerning NC start .............................................................. 625
14.6.5 Block search from last main block ......................................................................... 625
14.7 Function-specific display data .................................................................................... 627
14.7.1 Channel-specific GUD variables ............................................................................. 627
14.8 Function-specific alarm texts ..................................................................................... 628
14.9 Boundary conditions ................................................................................................. 629
14.9.1 Function-specific supplementary conditions .......................................................... 629
14.9.1.1 Continue machining within subprograms .............................................................. 629
14.9.1.2 Continue machining within program loops ......................................................... 630
14.9.1.3 Machining continuation on full circles ............................................................... 630
14.9.1.4 Automatically generated contour elements ......................................................... 630
14.9.2 Supplementary conditions for standard functions .................................................. 631
14.9.2.1 Axis replacement ................................................................................................. 631
14.9.2.2 Traversing movements of the channel axes ......................................................... 631
14.9.2.3 Block numbers .................................................................................................. 631
14.9.2.4 Block search ...................................................................................................... 632
14.9.2.5 Transformations ................................................................................................. 632
14.9.2.6 Compensations ................................................................................................. 633
14.9.2.7 Frames ............................................................................................................... 633
14.9.2.8 Tool offsets ....................................................................................................... 633
14.10 Data lists ................................................................................................................... 634
14.10.1 Machine data ......................................................................................................... 634
14.10.1.1 General machine data ...................................................................................... 634
14.10.1.2 Channel-specific machine data ....................................................................... 634
14.10.2 Supplementary conditions for standard functions .................................................. 635
14.10.3 Boundary conditions ............................................................................................ 635
15 TE8: Cycle-independent path-synchronous switching signal output - 840D si only ............... 635
15.1 Brief description ......................................................................................................... 635
15.2 Functional description ............................................................................................... 636
15.2.1 General .................................................................................................................. 636
15.2.2 Calculating the switching positions ...................................................................... 636
15.2.2.1 Block-related switching signal output ............................................................... 636
15.2.2.2 Path length-related switching signal output ...................................................... 638
15.2.3 Calculating the switching instants ......................................................................... 639
15.2.4 Switching frequency and switching position distance ......................................... 639
15.2.5 Approaching switching position .......................................................................... 640
15.2.6 Programmed switching position offset .................................................................. 641
15.2.7 Response to part program interruption ................................................................. 641
15.3 Startup ....................................................................................................................... 642
15.3.1 Activation ............................................................................................................. 642
15.3.2 Memory configuration ......................................................................................... 642
15.3.3 Parameterizing the digital on-board outputs ......................................................... 642
15.3.4 Parameterizing the switching signal ...................................................................... 643
15.3.5 Parameterization of the geometry axes ................................................................. 644
15.4 Programming ............................................................................................................ 645
15.4.1 Activating the block-related switching signal output (CC_FASTON) ......................... 645
15.4.2 Activating the path length-related switching signal output (CC_FASTON_CONT) .... 646
15.4.3 Deactivation (CC_FASTOFF) ................................................................................ 647
| 15.5 | Function-specific alarm texts | ................................. | 648 |
| 15.6 | Supplementary conditions | ................................................................. | 649 |
| 15.6.1 | Block search | ........................................................................ | 649 |
| 15.6.2 | Transformations | ................................................................. | 650 |
| 15.6.3 | Compensations | ................................................................. | 650 |
| 15.6.4 | Tool radius compensation (TRC) | ................................................................. | 651 |
| 15.6.5 | Continuous-path mode | ................................................................. | 651 |
| 15.6.6 | Software cams | ................................................................. | 651 |
| 15.7 | Data lists | ................................................................. | 652 |
| 15.7.1 | Machine data | ................................................................. | 652 |
| 15.7.1.1 | General machine data | ................................................................. | 652 |
| 15.7.1.2 | Channelspecific machine data | ................................................................. | 652 |
| **16** | TE9: Axis pair collision protection | ................................................................. | 653 |
| 16.1 | Brief description | ................................................................. | 653 |
| 16.1.1 | Brief description | ................................................................. | 653 |
| 16.2 | Functional description | ................................................................. | 654 |
| 16.3 | Startup | ................................................................. | 655 |
| 16.3.1 | Enabling the technology function (option) | ................................................................. | 655 |
| 16.3.2 | Activating the technology function | ................................................................. | 655 |
| 16.3.3 | Definition of an axis pair | ................................................................. | 656 |
| 16.3.4 | Retraction direction | ................................................................. | 656 |
| 16.3.5 | Offset for the machine coordinate systems | ................................................................. | 657 |
| 16.3.6 | Protection window | ................................................................. | 657 |
| 16.3.7 | Orientation | ................................................................. | 658 |
| 16.3.8 | Protection window extension | ................................................................. | 658 |
| 16.3.9 | Activating the protection function | ................................................................. | 658 |
| 16.3.10 | Axis-specific acceleration | ................................................................. | 659 |
| 16.3.11 | Monitoring status (GUD) | ................................................................. | 659 |
| 16.4 | Limitations and constraints | ................................................................. | 660 |
| 16.4.1 | Precedence of function-specific acceleration | ................................................................. | 660 |
| 16.4.2 | Axis container | ................................................................. | 660 |
| 16.5 | Examples | ................................................................. | 662 |
| 16.5.1 | Collision protection | ................................................................. | 662 |
| 16.5.2 | Collision protection and distance limiter | ................................................................. | 664 |
| 16.6 | Data lists | ................................................................. | 666 |
| 16.6.1 | Option data | ................................................................. | 666 |
| 16.6.2 | Machine data | ................................................................. | 666 |
| 16.6.2.1 | NC-specific machine data | ................................................................. | 666 |
| 16.6.2.2 | Axis/Spindle-specific machine data | ................................................................. | 666 |
| 16.6.3 | User data | ................................................................. | 666 |
| 16.6.3.1 | Global user data (GUD) | ................................................................. | 666 |
| **17** | V2: Preprocessing | ................................................................. | 667 |
| 17.1 | Brief description | ................................................................. | 667 |
| 17.2 | Program handling | ................................................................. | 670 |
| 17.3 | Program call | ................................................................. | 673 |
## Contents

| Section | Description | Page |
|---------|-------------|------|
| 17.4    | Constraints | 676   |
| 17.5    | Examples    | 678   |
| 17.5.1  | Preprocessing individual files | 678 |
| 17.5.2  | Preprocessing in the dynamic NC memory | 679 |
| 17.6    | Data lists  | 680   |
| 17.6.1  | Machine data | 680 |
| 17.6.1.1| General machine data | 680 |
| 17.6.1.2| Channelspecific machine data | 680 |

### 18 W5: 3D tool radius compensation - 840D sl only

| Section | Description | Page |
|---------|-------------|------|
| 18.1    | Brief description | 681 |
| 18.1.1  | General     | 681   |
| 18.1.2  | Machining modes | 683 |
| 18.2    | Circumferential milling | 684 |
| 18.2.1  | Corners for circumferential milling | 685 |
| 18.2.2  | Behavior at outer corners | 686 |
| 18.2.3  | Behavior at inside corners | 690 |
| 18.3    | Face milling | 694 |
| 18.3.1  | Cutter shapes | 694 |
| 18.3.2  | Orientation | 696 |
| 18.3.3  | Compensation on path | 698 |
| 18.3.4  | Corners for face milling | 700 |
| 18.3.5  | Behavior at outer corners | 701 |
| 18.3.6  | Behavior at inside corners | 702 |
| 18.3.7  | Monitoring of path curvature | 703 |
| 18.4    | Selection/deselection of 3D TRC | 704 |
| 18.4.1  | Selection of 3D TRC | 704 |
| 18.4.2  | Deselection of 3D TRC | 705 |
| 18.5    | Constraints | 706 |
| 18.6    | Examples | 707 |
| 18.7    | Data lists | 709 |
| 18.7.1  | General machine data | 709 |
| 18.7.2  | Channelspecific machine data | 709 |

### 19 W6: Path length evaluation - 840D sl only

| Section | Description | Page |
|---------|-------------|------|
| 19.1    | Brief description | 711 |
| 19.2    | Data | 712 |
| 19.3    | Parameterization | 713 |
| 19.3.1  | General activation | 713 |
| 19.3.2  | Data groups | 713 |
| 19.4    | Examples | 714 |
| 19.4.1  | Traversal per part program | 714 |
| 19.5    | Data lists | 715 |
| 19.5.1  | Machine data | 715 |
| 19.5.1.1| NC-specific machine data | 715 |
| 19.5.1.2| Axis/spindlespecific machine data | 715 |
# Contents

20  **Z3: NC/PLC interface signals** ................................................................. 717

20.1 3-Axis to 5-Axis Transformation (F2) ...................................................... 717

20.1.1 Signals from channel (DB21, ...) ............................................................. 717

20.2 Gantry Axes (G1) ..................................................................................... 719

20.2.1 Signals to axis/spindle (DB31, ...) ............................................................ 719

20.2.2 Signals from axis/spindle (DB31, ...) ......................................................... 720

20.3 Axis Couplings and ESR (M3) ................................................................. 724

20.3.1 Signals to axis (DB31, ...) ....................................................................... 724

20.3.2 Signals from axis (DB31, ...) ................................................................... 724

20.4 Extended stop and retract (R3) ................................................................. 726

20.4.1 Signals from axis/spindle (DB31, ...) ......................................................... 726

20.5 Setpoint Exchange (S9) ............................................................................ 727

20.5.1 Signals to axis/spindle (DB31, ...) ............................................................ 727

20.5.2 Signals from axis/spindle (DB31, ...) ......................................................... 727

20.6 Tangential Control (T3) ............................................................................ 728

20.6.1 Special response to signals .................................................................... 728

20.7 Clearance Control (TE1) .......................................................................... 729

20.7.1 Signals to channel (DB21, ...) ................................................................. 729

20.7.2 Signals from channel (DB21, ...) ............................................................. 730

20.8 Speed/Torque Coupling, Master-Slave (TE3) ............................................ 732

20.8.1 Signals to axis/spindle (DB31, ...) ............................................................ 732

20.8.2 Signals from axis/spindle (DB31, ...) ......................................................... 733

20.9 Handling Transformation Package (TE4) .................................................. 734

20.9.1 Signals from channel (DB21, ...) ............................................................. 734

20.10 MCS Coupling (TE6) .............................................................................. 736

20.10.1 Signals to axis/spindle (DB31, ...) ............................................................ 736

20.10.2 Signals from axis/spindle (DB31, ...) ......................................................... 736

20.11 Retrace Support (TE7) ............................................................................ 738

20.11.1 Signals to channel .................................................................................. 738

20.11.2 Signals from channel ............................................................................. 739

A  **Appendix** ............................................................................................... 741

A.1 List of abbreviations .................................................................................. 741

A.2 Overview .................................................................................................. 751

**Glossary** .................................................................................................. 753

**Index** ....................................................................................................... 775
F2: Multi-axis transformations

1.1 Brief description

**Note**

The transformations described below require that individual names are assigned to machine axes, channels and geometry axes when the transformation is active. Compare machine data:

- MD10000 $MN_AXCONF_MACHAX_NAME_TAB (machine axis name)
- MD20080 $MC_AXCONF_CHANAX_NAME_TAB (name of the channel axis in the channel)
- MD20060 $MC_AXCONF_GEOAX_NAME_TAB (name of the geometry axis in the channel)

Besides this no unambiguous assignments are present.

1.1.1 5-axis Transformation

**Function**

The "5-Axis Transformation" machining package is designed for machining sculptured surfaces that have two rotary axes in addition to the three linear axes X, Y, and Z. This package thus allows an axially symmetrical tool (milling cutter, laser beam) to be oriented in any desired relation to the workpiece in the machining space.

The path and path velocity are programmed in the same way as for 3-axis tools. The tool orientation is programmed additionally in the traversing blocks.

The real-time transformation performs the calculation of the resulting motion of all 5 axes. The generated machining programs are therefore not machine specific. Kinematic-specific post-processors are not used for the 5-axis machining operation.

A selection of various transformations is available for adapting the control to various machine kinematics. Part program commands can be issued in operation to switch over between two transformations parameterized during start-up.

This package therefore covers the three possible basic machine configurations which differ in terms of tool and workpiece orientation:

- Orientation of tool with two-axis swivel head (machine type 1)
- Orientation of workpiece with two-axis rotary table (machine type 2)
- Orientation of workpiece and tool with single-axis rotary table and swivel head (machine type 3)

The calculation also includes tool length compensation.

Since the orientation in relation to the workpiece surface is stored in a separate FRAME, a tool retraction operation with vertical orientation to the workpiece is also possible.
F2: Multi-axis transformations

1.1 Brief description

Tool orientation

Tool orientation can be specified in two ways:

- **Machine-related orientation**
  The machine-related orientation is dependent on the machine kinematics.

- **Workpiece-related orientation**
  The workpiece-related orientation is not dependent on the machine kinematics.
  It is programmed by means of:
  - Euler angles
  - RPY angles
  - Vector components

  The direction of the tool is described in the workpiece coordinate system with the part orientation. It is possible to program a specific component of the tool in its orientation to the workpiece. In most cases, this will be a longitudinal axis of the tool with the tool tip (Tool Center Point, TCP), which is also referred to as TCP-programming.

System variables for orientation

Part programs and synchronized actions can access the system variables that provide information on the following, in read only mode:

- End orientation of block (run-in value)
- Setpoint orientation
- Actual value orientation
- Switching between setpoint and actual value orientation
- Status for variables of actual value orientation

Special cases of 5-Axis transformation

The following transformations are to be entered as special cases of the general 5-Axis transformation:

- **3-axis and 4-axis transformation**
  There are 2 or 3 linear axes and a rotary axis.

- **Swivelling linear axis**
  One of the rotary axis rotates the 3rd linear axis.

- **Universal milling head**
  The two rotary axes are positioned at a projectable angle in relation to one another.

Knowledge of the general 5-axis transformation is a prerequisite for all of these transformations.
1.1.2 3-axis and 4-axis transformation

Function

The 3- and 4-Axis transformations are distinguished by the following characteristics:

| Transformation          | Features                  |
|-------------------------|---------------------------|
| 3-axis Transformation   | 2 linear axes             |
|                         | 1 rotary axis             |
| 4-Axis transformation   | 3 linear axes             |
|                         | 1 rotary axis             |

Both types of transformation belong to the orientation transformations. Orientation of the tool must be programmed explicitly. The orientation of the tool is executed in a plane perpendicular to the rotary axis.

Figure 1-1 Schematic diagram of 3-axis transformation

Figure 1-2 Schematic diagram of a 4-axis transformation with moveable workpiece
1.1.3 Orientation transformation with a swiveling linear axis.

Function

The orientation transformation with swiveling linear axis is similar to the 5-axis transformation of Machine Type 3, though the 3rd linear axis is not always perpendicular to the plane defined by the other two linear axes.

Features of kinematics

- Kinematics with three linear axes and two orthogonal rotary axes.
- Rotary axes are parallel to two of the three linear axes.
- The first rotary axis is moved by two Cartesian linear axes. It rotates the third linear axis, which moves the tool. The tool is aligned parallel to the third linear axis.
- The second rotary axis rotates the workpiece.
- The kinematics comprise a moved workpiece and a moved tool.

The following figure shows the interrelations for one of the possible axis sequences, for which transformation is possible.

![Figure 1-3 Schematic diagram of a machine with swiveling linear axis](image-url)
1.1.4 Universal milling head

Function

A machine tool with a universal milling head has got at least 5 axes:

- 3 linear axes
  - for linear movement [X, Y, Z]
  - move the machining point to any random position in the working area
- 2 rotary swivelling axes
  - are arranged under a configurable angle (mostly 45 Degree)
  - enable the tool to define orientations in space
  (are limited to a hemisphere in a 45 degree arrangement)

Figure 1-4 Schematic diagram of a machine tool with universal milling head
1.1.5 Orientation axes

Model for describing change in orientation

There is no such simple correlation between axis motion and change in orientation in case of robots, hexapodes or nutator kinematics, as in the case of conventional 5-axes machines. For this reason, the change in orientation is defined by a model that is created independently of the actual machine. This model defines three virtual orientation axes which can be visualized as rotations about the coordinate axes of a rectangular coordinate system.

For the purpose of 6-axis transformation, a third degree of freedom for orientation, describing the rotation of the tool about itself, has been introduced.

Real-time transformation

The Cartesian coordinates are converted from basic to machine coordinate system by means of a real-time transformation process.

These Cartesian coordinates comprise:

- **Geometry axes**
  Geometry axes describe the machining point.

- **Orientation axes**
  Orientation axes describe the orientation of a tool in space.

Tool orientation

You can define the orientation of the tool in space as follows using linear interpolation, large circle interpolation and by means of orientation vectors:

- **Direct programming of rotary axis positions A, B, C**

  5-axis transformation by programming:
  - The Euler- or RPY angle in degrees through $A_2, B_2, C_2$
  - The direction vector over $A_3, B_3, C_3$

- **Programming using lead angle** LEAD and **tilt angle** TILT
1.1.6 Cartesian manual travel

Function

The "Cartesian Manual Operation" function can be used to set one of the following coordinate systems as reference system for JOG motion to be selected separately for translation and orientation as:

- Basic coordinate system (BCS)
- Workpiece coordinate system (WCS)
- Tool coordinate system (TCS)

1.1.7 Cartesian PTP travel

Function

The "Cartesian PTP Travel" [PTP = Point-to-point movement (Point to Point)] function can be used to program a position in a cartesian coordinate system (workpiece coordinate system). The machine however moves in its machine coordinates.

The function can be used, for example, to traverse a singularity. Cartesian positions, supplied by a CAD system, need not be converted to machine axis values.

It must also be noted that axes take longer to traverse in the Cartesian coordinate system with active transformation and programmed feedrate than when they are traversed directly.

1.1.8 Generic 5-axis transformation

Function

The generic 5-axis transformation function differs from earlier 5-axis transformation versions insofar as it is no longer restricted with respect to the directions of rotary axes.

The basic orientation of the tool is no longer predefined in machine data as was the case in earlier versions of orientation transformations, but can now be programmed freely.
1.1 Brief description

1.1.9 Online tool length offset

Function

The system variable $AA_TOFF[]$ can be used to overlay the effective tool lengths in 3-D in runtime. For an active orientation transformation (TRAORI) or for an active tool carrier that can be oriented, these offsets are effective in the particular tool axes.

If the tool orientation changes, the tool length offsets that apply are rotated so that the pivot point for the orientation movement always refers to the corrected tool tip.

1.1.10 Activation via parts/program/softkey

The machine data relevant to the kinematic transformation has thus far been activated mostly through POWER ON.

Transformations MDs can also be activated via the part program / softkey and it is not necessary to boot the control system.

References:
Function Manual, Extended Functions; Kinematic Transformations (M1), Section: Cartesian PTP travel

1.1.11 Orientation compression

During the execution of NC programs containing blocks with relatively short traverse paths, the interpolation time can lead to a reduction in tool path velocity and a corresponding increase in machining time.

COMPON, COMPCURV, COMPCAD

You can run NC programs with short traverse paths without reducing the tool path velocity by activating "compressors" COMPON, COMPCURV or COMPCAD. The compressor also smooths the programmed movements and consequently tool path velocity.

Programming of direction vectors

Programming of tool orientation that is independent of the kinematics, can be achieved through programming of direction vectors. NC programs with such direction vectors can be executed with compressors COMPON, COMPCURV and COMPCAD.
1.2 5-axis transformation

1.2.1 Kinematic transformation

Task of orientation transformation

The task of orientation transformation is to compensate movements of the tool nose, which result from changes in orientation, by means of appropriate compensating movements of the geometry axes. The orientation movement is therefore decoupled from the movement on the workpiece contour. Various machine kinematics each require their own orientation transformation.

Fields of application

The "5-axis transformation" machining package is provided for machine tools, which have two additional rotary axes (rotation about the linear axes) in addition to three linear axes X, Y and Z: This package thus allows an axially symmetrical tool (milling cutter, laser beam) to be oriented in any desired relation to the workpiece in every point of the machining space.

The workpiece is always programmed in the rectangular workpiece coordinate system; any programmed or set frames rotate and shift this system in relation to the basic system. The kinematic transformation then converts this information into motion commands of the real machine axes.

The kinematic transformation requires information about the design (kinematics) of the machine, which are stored in machine data.

The kinematic transformation does not act on positioning axes.

1.2.2 Machine types for 5-axis transformation

Kinematics of machines for 5-axis transformation

5-axis machines are generally equipped with three linear and two rotary axes: the latter may be implemented as a two-axis swivel head, a two-axis rotary table or as a combination of single-axis rotary table and swivel head. These types of machine are characterized by:

1. Three linear axes form a right-handed, Cartesian coordinate system.
2. Rotary axes are parallel to the traversing direction of one of the linear axes.

Example:
- A parallel to X
- B parallel to Y
- C parallel to Z
3. Rotary axes are positioned vertically one above the other.

4. Rotary axes turn
   - Tool with two-axis swivel head (machine type 1)
   - Workpiece with two-axis rotary table (machine type 2)
   - Tool and workpiece with single-axis rotary table and swivel head (machine type 3)

5. The following applies to machine types 1 and 2:
   - Rotary axis 1 is treated as the 4th machine axis of the transformation.
   - Motion of 1st rotary axis changes the orientation of the 2nd rotary axis.
   - Rotary axis 2 is treated as the 5th machine axis of the transformation.
   - Motion of 2nd rotary axis does not change the orientation of the 1st rotary axis.

6. The following applies to machine type 3:
   - 1. Rotary axis (4th machine axis of transformation) turns the tool.
   - 2. Rotary axis (5th machine axis of transformation) turns the tool.

7. Initial tool position:
   - in negative Z direction.

---

**Note**

Transformations that do not fulfill all the conditions mentioned here (3 and 4-axis transformations, orientation transformation with swivelling linear axes, universal milling head) are described in separate sub chapters.
1.2.3 Configuration of a machine for 5-axis transformation

To ensure that the 5-axis transformation can convert the programmed values to axis motions, certain information about the mechanical design of the machine is required; this information is stored in machine data:

- Machine type
- Axis assignment
- Geometry information
- Assignment of direction of rotation

Machine type

The machine types have been designated above as types 1 to 3 and are stored in the following machine data as a two-digit number:

`MD24100 $MC_TRAFO_TYPE_1 (definition of channel transformation 1)`

...  

`MD24480 $MC_TRAFO_TYPE_10 (definition of channel transformation 10)`

The following table contains a list of machine types, which are suitable for 5-axis transformation.

| Axis sequence | Machine type 1 with swivelling / rotary tool | Machine type 2 with swivelling / rotary workpiece | Machine type 3 with swivelling / rotary tool/workpiece |
|---------------|---------------------------------------------|--------------------------------------------------|------------------------------------------------------|
| AB            | 16                                          | 32                                               | 48                                                   |
| AC            | x                                           | 33                                               | 49                                                   |
| BA            | 18                                          | 34                                               | 50                                                   |
| BC            | x                                           | 35                                               | 51                                                   |
| CA            | 20                                          | x                                                | x                                                    |
| CB            | 21                                          | x                                                | x                                                    |

Combinations that are not meaningful, whose C-axis corresponds to a rotation of the tool about its longitudinal axis (symmetry axis), are marked by x.

Identification of axis sequence

The axis sequence is identified in the following way:

- AB means: A is 4th axis, B is 5th axis of transformation
- For machine type 3, the swivel axis of the tool is the 4th axis of the transformation and the rotary axis of the workpiece is the 5th axis of the transformation.
Axis assignment

The axis assignment at the start of the 5-axis transformation defines the axis that will be mapped by the transformation internally onto a channel axis. Thus, the following is defined in the machine data below:

MD24110 $MC_TRAFO_AXES_IN_1 (Axis assignment for transformation 1)

... 

MD24482 $MC_TRAFO_AXES_IN_10 (Axis assignment for transformation 10)

Geometry information

Information concerning machine geometry is required so that the 5-axis transformation can calculate axis values: This information is stored in the machine data (in this case, for the first transformation in the channel):

MD24500 $MC_TRAFO5_PART_OFFSET_1 (workpiece-oriented offset)

- for machine type 1 (two-axis swivel head)
  Vector from machine reference point to table zero point (zero vector)
- for machine type 2 (two-axis rotary table)
  Vector from last table swivel joint to zero point of table

Figure 1-6 Machine data MD24500 $MC_TRAFO5_PART_OFFSET_1 for machine type 2

- for machine type 3 (single-axis swivel head and single-axis rotary table)
  Vector from joint of table to zero point of table

MD24560 $MC_TRAFO5_JOINT_OFFSET_1 (vector of the kinematic offset of 5-axis transformation 1)

Vector from the first to the second swivel joint (machine type 1 and 2)
Vector from machine zero point to the swivel joint of the table (machine type 3)

MD24510 $MC_TRAFO5_ROT_AX_OFFSET_1 (position offset of rotary axes 1/2/3 5-axis transformation 1) angle offset of the first or second rotary axis
F2: Multi-axis transformations

1.2 5-axis transformation

Figure 1-7 Schematic diagram of CA kinematics, moved tool

Figure 1-8 Schematic diagram of CB kinematics, moved workpiece
Assignment of direction of rotation

The sign interpretation setting for a rotary axis is stored in the sign machine data for 5-axis transformation.

MD24520 \$MC\_TRAFO5\_ROT\_SIGN\_IS\_PLUS\_1[n] (sign of rotary axis 1/2/3 for 5-axis transformation 1)

MD24620 \$MC\_TRAFO5\_ROT\_SIGN\_IS\_PLUS\_2[n] (sign of rotary axis 1/2/3 for 5-axis transformation 2)

Transformation types

Ten transformation types per channel can be configured in the following machine data:

MD24100 \$MC\_TRAFO\_TYPE\_1 ...MD24480 \$MC\_TRAFO\_TYPE\_10 (definition of transformation 1 in channel … definition of transformation 10 in channel)

Of these eight types, a maximum of two may be 5-axis transformations.

Activation

Activation of the 5-axis transformation is described in the section "Activation and Application of 3- to 5-axis Transformation".
1.2.4 Tool orientation

Programming

The orientation of the tool can be programmed in a block directly by specifying the rotary axes or indirectly by specifying the Euler angle, RPY angle and direction vector. The following options are available:

- directly as rotary axes A, B, C
- indirectly for 5-axis transformation:
  - via Euler or RPY angles in degrees via A2, B2, C2
  - Indirectly for 5-axis transformation via direction vector A3, B3, C3

The identifiers for Euler angles and direction vectors can be set in machine data:

- Euler angles via:
  MD10620 $MN_EULER_ANGLE_NAME_TAB (name of Euler angles)
- Direction vector via:
  MD10640 $MN_DIR_VECTOR_NAME_TAB (name of direction vectors)

The tool orientation can be located in any block. Above all, it can be programmed alone in a block, resulting in a change of orientation in relation to the tool tip which is fixed in its relationship to the workpiece.
Euler or RPY

The following machine data can be used to switch between Euler and RPY angles:

MD21100 $MC_ORIENTATION_IS_EULER (angle definition for orientation programming)

Orientation reference

A tool orientation at the start of a block can be transferred to the block end in two different ways:

- in the workpiece coordinate system with command ORIWKS
- in the machine coordinate system with command ORIMKS

ORIWKS command

The tool orientation is programmed in the workpiece coordinate system (WCS) and is thus not dependent on the machine kinematics.

In the case of a change in orientation with the tool tip at a fixed point in space, the tool moves along a large arc on the plane stretching from the start vector to the end vector.

ORIMKS command

The tool orientation is programmed in the machine coordinate system and is thus dependent on the machine kinematics.

In the case of a change in orientation of a tool tip at a fixed point in space, linear interpolation takes place between the rotary axis positions.

The orientation is selected via NC language commands ORIWKS and ORIMKS.
Figure 1-11  Change in cutter orientation while machining inclined edges

1) Machine the surface F1 along the edge of F2
2) Retract
3) Change in orientation B: +90° A: -30°
4) Approach the surface F2
5) Machining of the surface F2 along the edge of F1
ORIMKS constitutes the basic setting

The basic setting can be changed via the following machine data:

MD20150 MC_GCODE_RESET_VALUES (RESET position of G groups)

MD20150 $MC_GCODE_RESET_VALUES [24] = 1 ⇒ ORIWK is basic setting

MD20150 $MC_GCODE_RESET_VALUES [24] = 2 ⇒ ORIWK is basic setting

Illegal tool orientation

If tool position is programmed in relation to the following functions, alarm 12130 "Illegal tool orientation" is output when Euler angles and direction vectors are selected and the NC program then stops (this alarm can also occur in connection with G331, G332 and G63).

- G04: Dwell time
- G33: Thread cutting with constant lead
- G74: Approaching a reference point
- G75: Approaching a fixed point
- REPOS: Repositioning to the contour
- REPOSQ: Repositioning to the contour
- REPOSH: Repositioning to the contour
To remedy this situation, tool orientation can be programmed with axis end values.

Alarm 17630 or 17620 is output for G74 and G75 if a transformation is active and the axes to be traversed are involved in the transformation. This applies irrespective of orientation programming.

If the start and end vectors are inverse parallel when ORIWKs is active, then no unique plane is defined for the orientation programming, resulting in the output of alarm 14120.

If a transformation switch (switch On, switch Off or change transformation) is undertaken, alarm 14400 will be generated.

In the reverse situation, i.e. a tool radius offset is selected or deselected when a transformation is active, no alarm message is output.

Multiple input of tool orientation

According to DIN 66025, only one tool orientation may be programmed in a block, e.g. with direction vectors:

```
N50 A3=1 B3=1 C3=1
```

If tool orientation is entered multiply, i.e. with direction vectors and with Euler angles, error message 12240 "Channel X block Y tool orientation xx defined more than once" is displayed and the NC part program stops.

```
N60 A3=1 B3=1 C3=1 A2=0 B2=1 C2=3
```

Tool orientation using orientation vectors

Polynomials can also be programmed for the modification of the orientation vector.

This method produces an extremely smooth change in speed and acceleration at the block changes for rotary axes when the tool orientation has to be programmed over several blocks.

The interpolation of orientation vectors can be programmed with polynomials up to the 5th degree. Polynomial interpolation of orientation vectors is described in the "Polynomial Interpolation of Orientation Vectors" section.

Note

Further explanations of tool orientation using orientation vectors and their handling in machines are given in:

Reference:
Function Manual, Basic Machine; Tool Offset; Orientable Toolholders (W1)
1.2.5 Singular positions and handling

Extreme velocity increase

If the path runs in close vicinity to a pole (singularity), one or several axes may traverse at a very high velocity. Alarm 10910 "Irregular velocity run in a path axis" is then triggered. The programmed velocity is then reduced to a value, which does not exceed the maximum axis velocity.

Behavior at pole

Unwanted behavior of fast compensating movements can be controlled by making an appropriate selection of the following machine data (see following Figure):

- MD24530 $MC_TRAFO5_NON_POLE_LIMIT_1 (definition of pole area for 5-axis transformation 1)
- MD24630 $MC_TRAFO5_NON_POLE_LIMIT_2 (definition of pole area for 5-axis transformation 2)
- MD24540 $MC_TRAFO5_POLE_LIMIT_1 (closing angle tolerance for interpolation by pole for 5-axis transformation)
- MD24640 $MC_TRAFO5_POLE_LIMIT_2 (closing angle tolerance for interpolation by pole for 5-axis transformation)

Note

Singularities are dealt with differently in SW 5.2 and higher: Only one relevant machine data item exists, $MC_TRAFO5_POLE_LIMIT (see Section "Singularities of orientation (Page 78)" or "Programming Manual, Production Planning").

Definition of the pole range for 5-axis transformation

This machine data identifies a limit angle of the 5th axis of the first MD24530 $MC_TRAFO5_NON_POLE_LIMIT_1 or the second MD24630 $MC_TRAFO5_NON_POLE_LIMIT_2 5-axis transformation with the following properties:

If the path runs past the pole at an angle lower than the value set here, it crosses through the pole.

With the 5-axis transformation, a coordinate system consisting of circles of longitude and latitude is spanned over a spherical surface by the two orientation axes of the tool.

If, as a result of orientation programming (i.e. the orientation vector is positioned on one plane), the path passes so close to the pole that the angle is less than the value defined in this machine data, then a deviation from the specified interpolation is made so that the interpolation passes through the pole.
End angle tolerance for interpolation by pole for 5-axis transformation

This machine data identifies a limit angle for the 5th axis of the first MD24540 $MC_TRAFO5_NON_POLE_LIMIT_1 or the second MD24640 $MC_TRAFO5_NON_POLE_LIMIT_2 5-axis transformation with the following properties:

With interpolation through the pole point, only the fifth axis moves; the fourth axis remains in its start position. If a movement is programmed which does not pass exactly through the pole point, but is to pass within the tolerance defined by the following machine data in the vicinity of the pole, a deviation is made from the specified path because the interpolation runs exactly through the pole point.

- MD24530 $MC_TRAFO5_NON_POLE_LIMIT_1
- MD24630 $MC_TRAFO5_NON_POLE_LIMIT_2

As a result, the position at the end point of the fourth axis (pole axis) deviates from the programmed value.

This machine data specifies the angle by which the pole axis may deviate from the programmed value with a 5-axis transformation if a switchover is made from the programmed interpolation to interpolation through the pole point. In the case of a greater deviation, an error message is output and the interpolation is not executed.

Figure 1-13 5-axis transformation; orientation path in pole vicinity. Example for machine type 1: 2-axis swivel head with rotary axis RA 1 (4th axis of transformation) and rotary axis RA 2 (5th axis of transformation)
Behavior during large circle interpolation at pole position

The following machine data can be used to set the response for large circle interpolation in pole position as follows:

MD21108 $MC_POLE_ORI_MODE

Does not define the treatment of changes in orientation during large circle interpolation unless the starting orientation is equal to the pole orientation or approximates to it and the end orientation of the block is outside the tolerance circle defined in the following machine data.

- MD24530 $MC_TRAFO5_NON_POLE_LIMIT_1
- MD24630 $MC_TRAFO5_NON_POLE_LIMIT_2

The position of the polar axis is arbitrary in the polar position. For the large circle interpolation, however, a specified orientation is required for this axis.

The following machine data is coded decimally.

MD21108 $MC_POLE_ORI_MODE

The units define the behavior if start orientation coincides with pole position and the decade the behavior if start orientation is within the tolerances defined by the following machine data.

- MD24530 $MC_TRAFO5_NON_POLE_LIMIT_1
- MD24630 $MC_TRAFO5_NON_POLE_LIMIT_2

All setting values are described in "Channel-specific Machine Data".
1.3 3-axis and 4-axis transformations

Introduction

3-axis and 4-axis transformations are special types of the 5-axis transformation initially described. Orientation of the tool is possible only in the plane perpendicular to the rotary axis. The transformation supports machine types with movable tool and movable workpiece.

Kinematics variants

The variants specified in the following table apply both for 3-axis and 4-axis transformations.

| Variants of 3-axis and 4-axis transformations |
|-----------------------------------------------|
| Machine type | swiveling/rotary | rotary axis is parallel | orientation plane | $MC\_TRAFO\_TYPE_n$ | Tool orientation in zero position |
|-------------|-----------------|------------------------|------------------|------------------|-------------------------------|
| 1 Tool      | X               | Y - Z                  | 16               | Z                |
|             | Y               | X - Z                  | 18               |                  |
|             | Z               | X - Y                  | 20               | Y                |
|             | any             | any *                  | 24               | Any              |
| 2 workpiece | X               | Y - Z                  | 32, 33           | Z                |
|             | Y               | X - Z                  | 34, 35           |                  |
|             | Any             | any *                  | 40               | Any              |

*Note: on types 24 and 40 *

In the case of transformation types 24 and 40, the axis of rotation and tool orientation can be set so that the change in orientation takes place at the outside of a taper and not in a plane.

Zero position

Tool orientation at zero position is the position of the tool with G17 as the active working plane and position of the rotary axis at 0 degrees.

Axis assignments

The three translatory axes included in the transformation are assigned to any channel axes via machine data $MC\_TRAFO\_GEOAX\_ASSIGN\_TAB_n[0..2]$ and $MC\_TRAFO\_AXES\_IN_n[0..2]$. The following must apply for the assignment of channel axes to geometry axes for the transformation:

$MC\_TRAFO\_GEOAX\_ASSIGN\_TAB_n[0] = MC\_TRAFO\_AXES\_IN_n[0]$
$MC\_TRAFO\_GEOAX\_ASSIGN\_TAB_n[1] = MC\_TRAFO\_AXES\_IN_n[1]$
$MC\_TRAFO\_GEOAX\_ASSIGN\_TAB_n[2] = MC\_TRAFO\_AXES\_IN_n[2]$

The axes with corresponding index must be assigned to each other.
Parameter assignment procedure

- Enter the type of transformation according to the previous table as machine data:
  \$MC\_TRAFO\_TYPE\_n
- Assign channel axes to the geometry axes of the transformation.
- For a 3-axis transformation, set the values for the axis, which is not required:
  - \$MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_n[geoax] = 0
  - \$MC\_TRAFO\_AXES\_IN\_n[geoax] = 0
  - \$MC\_TRAFO\_AXES\_IN\_n[4] = 0; → there is no 2nd rotary axis
- For a 4-axis transformation, set the following for the 3 linear axes
  - \$MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_n[geoax] = ...
  - \$MC\_TRAFO\_AXES\_IN\_n[geoax] = ...
  - \$MC\_TRAFO\_AXES\_IN\_n[4] = 0; → there is no 2nd rotary axis

Complete examples of a 3-axis and 4-axis transformation can be found in "Example for 3- and 4-axis Transformation" section.
### 1.4 Transformation with swiveled linear axis

#### General information

The "transformation with swiveling linear axis forms" a transformation group of its own. It can be used when a kinematic as described in the Section "Orientation transformation with a swiveling linear axis. (Page 26)" is present:

- Three Cartesian linear axes (X, Y, Z) and two orthogonal rotary axes (A, B).
- The rotary axes are parallel to two of the three linear axes.
- The first rotary axis (A) is moved by two Cartesian linear axes. It rotates the third linear axis (Z) that moves the tool.
- The tool is aligned parallel to the third linear axis (Z).
- The second rotary axis (B) rotates the workpiece.

#### Additional requirement:

- The first rotary axis (A) may only sweep a very small swivel range (swivel range $< \pm 90^\circ$).

#### Note

All the axis values used in the text relate to the designations of the example machine in the following figure "Machine with swiveling linear axis Z".

---

![Diagram of a machine with swiveling linear axis Z](image)

**Figure 1-14** Example: Machine with swiveling linear axis Z
Pole

The transformation with swiveling linear axis has a pole for a tool orientation parallel to the second rotary axis (B). Singularity occurs in the pole position because the third linear axis (Z) is parallel to the plane of the first two linear axes (X, Y), thus excluding the possibility of compensating movements perpendicular to this plane.

Parameterization

Kinematic variants

The kinematic variant of the machine is set in the machine data:

MD24100, ..., MD25190 $MC_TRAFO_TYP_n = \langle\text{type}\rangle$, with n = 1, 2, 3, ...

| Kinematics | <type> |
|------------|--------|
| 1. Rotary axis | 2. rotary axis | swiveled linear axis | Bits 6 - 0 |
| A | B | Z | 10,00 000 |
| A | C | Y | 10,00 001 |
| B | A | Z | 10,00 010 |
| B | C | X | 10,00 011 |
| C | A | Y | 10,00 100 |
| C | B | X | 10,00 101 |

Machine kinematics

The machine kinematics is set for the 1st ($MC_TRAFO5 \ldots _1$) and/or 2nd ($MC_TRAFO5 \ldots _2$) 5-axis transformation in the channel set with the following machine data:

- Vector (po, see following figure) from the second rotary axis to workpiece table zero:
  - MD24500 $MC_TRAFO5\_PART\_OFFSET\_1$
  - MD24600 $MC_TRAFO5\_PART\_OFFSET\_2$

- Axis positions of the two rotary axes at the initial position of the machine:
  - MD24510 $MC_TRAFO5\_ROT\_AX\_OFFSET\_1$
  - MD24610 $MC_TRAFO5\_ROT\_AX\_OFFSET\_2$

- Sign with which the rotary axis positions are included in the transformation:
  - MD24520 $MC_TRAFO5\_ROT\_SIGN\_IS\_PLUS\_1$
  - MD24620 $MC_TRAFO5\_ROT\_SIGN\_IS\_PLUS\_2$

- Vector (jo) from machine zero to the second rotary axis:
  - MD24560 $MC_TRAFO5\_JOINT\_OFFSET\_1$
  - MD24660 $MC_TRAFO5\_JOINT\_OFFSET\_2$
- Vector (to) from the toolholder (flange) to the first rotary axis (measured at machine initial position):
  - MD24550 $MC\_TRAFO5\_BASE\_TOOL\_1$
  - MD24650 $MC\_TRAFO5\_BASE\_TOOL\_2$

- Vector (ro) from machine zero to the first rotary axis (measured at the machine initial position):
  - MD24562 $MC\_TRAFO5\_TOOL\_ROT\_AX\_OFFSET\_1$
  - MD24662 $MC\_TRAFO5\_TOOL\_ROT\_AX\_OFFSET\_2$

Determining the machine data values

As an aid for determining the values for the above-mentioned machine data, the following two sketches clarify the relationships between the vectors.

**Note**

**Requirement**

The machine has been traversed so that the toolholding flange aligns with the table zero (*). Is this technically not possible, vector to must be corrected by the deviations.

---

**Figure 1-15** Projections of the vectors to be set in the machine data
1.4 Transformation with swiveled linear axis

**Note**

A physically identical point on the 1st rotary axis (e.g. point of intersection between the tool axis and the 1st rotary axis) must be assumed for both views.

---

**Figure 1-16** Machine in the zero position

**Figure 1-17** Front view: Vectors for machine in the zero position
Determination of the machine data values

Perform the following operation:

1. Determine, as shown in the lower part for vector jo in the "Vectors for machine in zero position" figure, the X and Y components for all vectors.

2. As shown in the upper part for vector ro in the "Vectors for machine in zero position" figure, determine the Z component for all vectors.

3. Enter the X, Y and Z components of the vectors (po, jo, to, ro) in the relevant machine data.

The procedure can be used all settable kinematic variants.

Note

For the appropriate machine geometry or position of the machine zero, both individual components as well as complete vectors can become zero.

Programming

The switch on/off of the transformation in the part program or synchronized action is described in Section "Programming of the 3- to 5-axis transformation (Page 56)".

Tool orientation

For a transformation with swiveling linear axis, the same statements as for the 5-axis transformation with regard to the tool orientation apply similarly (see Section "Tool orientation (Page 37)").
1.5 Cardan milling head

1.5.1 Fundamentals of cardan milling head

**Note**

The following description of the cardan milling head transformation has been formulated on the assumption that the reader has already read and understood the general 5-axis transformation described in Section "5-axis transformation (Page 31)". Please note that where no specific statements relating to the cardan milling head are made in the following section, the statements relating to general 5-axis transformation apply.

**Applications**

A cardan milling head is used for machining contours of sculptured parts at high feedrates. An excellent degree of machining accuracy is achieved thanks to the rigidity of the head.

![Schematic representation of the cardan milling head versions](image)

**Figure 1-19  Schematic representation of the cardan milling head versions**

**Configuring the nutator angle $\phi$**

The angle of the inclined axis can be configured in a machine data:

- $\text{MC_TRAFO5_NUTATOR_AX_ANGLE}_1$: for the first orientation transformation
- $\text{MC_TRAFO5_NUTATOR_AX_ANGLE}_2$: for the second orientation transformation

The angle must lie within the range of 0 degrees to +89 degrees.
Tool orientation

Tool orientation at zero position can be specified as follows:

- parallel to the first rotary axis or
- perpendicular to it, and in the plane of the specified axis sequence

Types of kinematics

The axis sequence of the rotary axes and the orientation direction of the tool at zero position are set for the different types of kinematics using the following machine data:

$MC_{TRAFO\_TYPE\_1} \ldots MC_{TRAFO\_TYPE\_10}$

Axis designation scheme

As for the other 5-axis transformations, the following applies:

the rotary axis ...

...A is parallel to X: $A'$ is below the angle $\phi$ to the X axis

...B is parallel to Y: $B'$ is below angle $\phi$ to the Y axis

...C is parallel to Z: $C'$ is below angle $\phi$ to the Z axis

Angle definition

Figure 1-20  Position of axis $A'$

Axis $A'$ is positioned in the plane spanned by the rectangular axes of the designated axis sequence. If, for example, the axis sequence is CA', then axis $A'$ is positioned in plane Z-X. The angle $\phi$ then is the angle between axis $A'$ and the X axis.
1.5 Cardan milling head

1.5.2 Parameterization

Setting the type of transformation

The transformation type is set with the machine data of the corresponding transformation data block:

MD24100, ... MD25190 $MC_TRAFO_TYP_n, with n = 1, 2, 3, ...

| Bit | <value> | Description |
|-----|---------|-------------|
| 7   | 1       | Activation of the transformation for cardan milling head |
| 6 - 5 | Moving component |
|      | 00 | Movable tool |
|      | 01 | Moving workpiece |
|      | 10 | Movable tool and workpiece |
| 4 - 3 | Direction of orientation of tool in position zero |
|      | 00 | X direction |
|      | 01 | Y direction |
|      | 10 | Z direction |
| 2 - 0 | Axis sequence |
|      | 000 | AB' or A'B |
|      | 001 | AC' or A'C |
|      | 010 | BA' or B'A |
|      | 011 | BC' or B'C |
|      | 100 | CA' or C'A |
|      | 101 | CB' or C'B |

The following transformation types can be set:

| Axis sequence: Bits 0 - 2 | Moving component: Bits 6 - 5 |
|--------------------------|-------------------------------|
|                          | Tool 00 | Workpiece 01 | Tool/workpiece 10 |
| Zero position 1) | Zero position 1) | Zero position 1) |
| X   Y   Z | X   Y   Z | X   Y   Z |
| 00 01 10 | 00 01 10 | 00 01 10 |

| AB' / A'B 000 | x x - | - - - | - - - |
| AC' / A'C 001 | x - x | - - - | - - - |
| BA' / B'A 010 | x x - | - - - | - x x |
| BC' / B'C 011 | - x x | - x x | - - - |
Active machining plane

The tool orientation at the zero position can be set not only in the Z direction. For this reason, ensure that the active working plane is set so that the tool length compensation acts in the tool orientation direction.

The active machining plane should always be the plane according to which the tool orientation is set in position zero.

Other settings

The geometry information used by the cardan milling head transformation for calculation of the axis values is set in the same way as that of the other 5-axis transformations.

1.5.3 Traverse of the cardan milling head in JOG mode

JOG

In JOG mode, the linear axes can be traversed normally. It is, however, difficult to set the orientation correctly by traversing these axes.
1.6 Programming of the 3- to 5-axis transformation

Switch on

The 3- to 5-axis transformations, including the transformations with swiveled linear axis and caridan milling head, are enabled with the TRAORI(<transformation-no.>) command. The enable of the transformation sets the NC/PLC interface signal:

DB21, ... DBX33.6 = 1 (transformation active)

Deactivation

With the TRAFOOF command disables the currently active 3- to 5-axis transformation. The disable of the transformation resets the NC/PLC interface signal:

DB21, ... DBX33.6 = 0 (transformation inactive)

Switch-over

If a transformation is already active in the channel, the TRAORI(<transformation-no.>) with a new transformation number command can be used to switch to another transformation.

Reset / program end

The control behavior after startup, program end or NC reset is set in the machine data:

MD20110 $MC_RESET_MODE_MASK, bit 7 = <value>

| <value> | Meaning |
|---------|---------|
| 0       | Initial setting for active transformation after reset / program end according to $MC_TRAFO_RESET_VALUE |
| 1       | The active transformation remains active over reset / program end |

Option

The "5-axis transformation" function, together with its special forms, is an option.

References

A detailed description of the machine data can be found in:

Parameter Manual, Detailed Machine Data Description
1.7 Generic 5-axis transformation and variants

1.7.1 Functionality

Scope of functions

The scope of functions of generic 5-axis transformation covers implemented 5-axis transformations (see Section "5-axis transformation (Page 31)") for perpendicular rotary axes as well as transformations for the cardan milling head (one rotary axis parallel to a linear axis, the second rotary axis at any angle to it, see Section "Cardan milling head (Page 52)”).

Applications

In certain cases, it may not be possible to compensate the conventional transformation machine accuracy, e.g. if:

- the rotary axes are not exactly mutually perpendicular or
- one of the two rotary axes is not positioned exactly parallel to the linear axes

In such cases, generic 5-axis transformation can produce better results.

Programming example

for generic 5-axis transformation is shown in Section "Example for Generic 5-axis Transformation".

Activation

Generic 5-axis transformation can also be activated like any other orientation transformation using the TRAORI() or TRAORI(n) command (where n is the number of the transformation). Furthermore, the basic transformation can be transferred in the call in three other parameters, e.g. TRAORI(1, 1.1, 1.5, 8.9).

A transformation can be deselected implicitly by selecting another transformation or explicitly with TRAFOOF.
1.7.2 Description of machine kinematics

Machine types

Like the existing 5-axis transformations, there are three different variants of generic 5-axis transformation:

1. Machine type: Rotatable tool
   - Both rotary axes change the orientation of the workpiece. The orientation of the workpiece is fixed.

2. Machine type: Rotatable workpiece
   - Both rotary axes change the orientation of the workpiece. The orientation of the tool is fixed.

3. Machine type: Rotatable tool and rotatable workpiece - one rotary axis changes the tool orientation and the other the workpiece orientation.

Configurations

As previously, the machine configurations are defined in the following machine data (see Section “Configuration of a machine for 5-axis transformation (Page 33)“):

$\text{MC\_TRAFO\_TYPE\_1, ..., \_8}$

Additional types have been introduced for generic 5-axis transformation:

Table 1-1 Overview of machine types for the generic 5-axis transformation

| Machine type               | 1         | 2         | 3         |
|----------------------------|-----------|-----------|-----------|
| Swivel/rotatable:          | tool      | workpiece | Tool/workpiece |
| Transformation types       | 24        | 40        | 56        |

Rotary axis direction

The direction of the rotary axis is defined by the following machine data:

$\text{MC\_TRAFO5\_AXIS1\_n}$ (1st rotary axis) and

$\text{MC\_TRAFO5\_AXIS2\_n}$ (2nd rotary axis)

where n is 1 or 2 for the first or second 5-axis transformation in the system respectively. The machine data specified above are fields with three values, which describe that axis direction vectorially (similar to the description of rotary axes for orientable toolholder). The absolute value of the vectors is insignificant; only the defined direction is relevant.
Example:

1. A-axis is the rotary axis (parallel to the x direction):
   MD24570 $MC_TRAFO5_AXIS1_1[0] = 1.0 (direction first rotary axis)
   MD24570 $MC_TRAFO5_AXIS1_1[1] = 0.0
   MD24570 $MC_TRAFO5_AXIS1_1[2] = 0.0

2. B-axis is the rotary axis (parallel to the y direction):
   MD24572 $MC_TRAFO5_AXIS2_1[0] = 0.0 (direction 2nd rotary axis)
   MD24572 $MC_TRAFO5_AXIS2_1[1] = 1.0
   MD24572 $MC_TRAFO5_AXIS2_1[2] = 0.0

1.7.3 Generic orientation transformation variants

Extension

Generic orientation transformation for 5-axis transformation has been extended with the following variants for 3- and 4-axis transformation:

Variant 1

4-axis transformations

A 4-axis transformation is characterized by the exclusive use of the first rotary axis as an entry axis of the transformation. The following applies:

MD24110 $MC_TRAFO_AXES_IN_1[4] = 0 (axis assignment for transformation 1) or
MD24210 $MC_TRAFO_AXES_IN_2[4] = 0 (axis assignment for transformation 2)

Variant 2

3-axis transformations

In a 3-axis transformation, one of the geometry axes is not present, by entering a zero in the field:

MD24120 $MC_TRAFO_GEOAX_ASSIGN_TAB_1[n] (assignment between geometry axis and channel axis for transformation 1)
MD24220 $MC_TRAFO_GEOAX_ASSIGN_TAB_2[n] (assignment between geometry axis and channel axis for transformation 2)
1.7 Generic 5-axis transformation and variants

Transformation types
Both variants of generic 3- or 4-axis transformation are described by the following transformation types:

- 3- or 4-axis transformation with rotatable tool
  $MC\_TRAFO\_TYPE\_n = 24$
- 3- or 4-axis transformation with rotatable workpiece
  $MC\_TRAFO\_TYPE\_n = 40$

In conventional 3-axis or 4-axis transformations, the transformation type also defined the basic tool orientation in addition to the position of the rotary axis, which could then no longer be influenced.

Effects on orientations
Generic 3-axis or 4-axis transformation has the following effect on the various orientations:

The resulting tool orientation is defined according to the hierarchy specified for generic 5-axis transformation.

Priority:

- high: programmed orientation,
- medium: tool orientation and
- low: basic orientation

Allowance is made, in particular, for the following orientations:

- A programmed tool orientation
- A basic tool orientation, modified by orientable toolholders.

Note
Further information on programmable tool orientation and on basic tool orientation can be found in:

Reference:
Function Manual, Basic Machine; Tool Offset; Orientable Toolholders (W1)
Programming Manual, Fundamentals
Comparison

Besides the 3- and 4-axis transformations mentioned in Section "3- and 4-axis Transformations", the following differences should be noted:

- **Position of the rotary axis:**
  - can be arbitrary
  - need not be parallel to a linear axis
- **Direction of the rotary axis**
  - Must be defined by the following machine data:
    MD24570 $MC_TRAFO5_AXIS1_1[n]$ (direction first rotary axis) or
    MD24670 $MC_TRAFO5_AXIS1_2[n]$ (direction first rotary axis)
- **Basic tool orientation**
  - Must be defined by the following machine data:
    MD24574 $MC_TRAFO5_BASE_ORIENT_1[n]$ (workpiece orientation) or
    MD24674 $MC_TRAFO5_BASE_ORIENT_2[n]$ (workpiece orientation)
- **Selection of a generic 3-/4-axis transformation**
  - Optional tool orientation can be transferred as in the case of a generic 5-axis transformation.

1.7.4 Parameterization of orientable toolholder data

Application

Machine types for which the table or tool can be rotated, can either be operated as true 5-axis machines or as conventional machines with orientable toolholders. In both cases, machine kinematics is determined by the same data, which, due to different parameters, previously had to be entered twice - for toolholder via system variables and for transformations via machine data. The new transformation type 72 can be used to specify that these two machine types access identical data.
Transformation type 72

The following machine data can be used to define a generic 5-axis transformation for transformation type 72 with kinematic data read from the data for an orientable toolholder.

MD24100 $MC_TRAFO_TYPE_1 (definition of transformation 1 in the channel) or
MD24200 $MC_TRAFO_TYPE_2 (definition of transformation 2 in the channel)

From this number data is made available via machine data MD24582 $MC_TRAFO5_TCARR_NO_1 (TCARR-Number for the first 5-axis transformation) for the first or MD24682 $MC_TRAFO5_TCARR_NO_2 (TCARR-Number for the second 5-axis transformation) for the second orientation transformation. The corresponding transformation type can then be derived from the content of kinematic type with parameter $TC_CARR23 - see following table.

Table 1-2 Machine types for generic 5-axis transformation

| Machine type | 1  | 2  | 3  | 4                                      |
|--------------|----|----|----|----------------------------------------|
| Swivel/      | tool | workpiece | Tool/workpiece | Type 3 or orientable toolholder       |
| rotatable:   |     |               |               |                                        |
| Kinematic type: | T | P | M | T, P, M                                |
| Transformation type: | 24 | 40 | 56 | 72 from content of $TC_CARR23         |

Note

The transformation only takes place if the orientable toolholder concerned is available and the value of $TC_CARR23 contains a valid entry for type M, P or T kinematics in lower or upper case.

Transformation machine data for the first orientation transformation listed in the tables below are equally valid for the second orientation transformation. All other machine data that may affect the transformation characteristics and do not appear in the tables below, remain valid and effective:

MD24110/MD24210 $MC_TRAFO_AXES_IN_1/2 (axis assignment for transformation) or
MD24574/MD24674 $MC_TRAFO5_BASE_ORIENT_1/2 (basic tool orientation)

If in the tables below a second additive parameter appears in brackets for the parameters of the orientable toolholder (e.g. $STC_CARR24 (+ $TC_TCARR64)), the sum of both values will only be effective if the fine offset specified in setting data is active when the data is transferred from the orientable toolholder.

SD42974 $SC_TOCARR_FINE_CORRECTION = TRUE (fine offset TCARR on/off)
Activation

The most significant parameter values of an orientable toolholder for a transformation can be activated in the part program with NEWCONFIG. Alternatively, the machine data concerned for transformation type 72 can be activated via the HMI user interface.

Assignment for all types of transformation

The assignments between the toolholder data for writing the linear offsets and the corresponding machine data for kinematic transformations are determined by the transformation type. The following assignment of all other parameters is identical for all three possible types of transformation:

| Assignment for all types of transformation together identical |   |   |
|--------------------------------------------------------------|---|---|
| MD24100 $MC_TRAFO_TYPE_1 (definition of transformation 1 in the channel) | 24 | T |
|                                                               | 40 | P |
|                                                               | 56 | M |
| MD24570 $MC_TRAFO5_AXIS1_1[0] (direction first rotary axis)    | $TC_CARR7 |
| MD24570 $MC_TRAFO5_AXIS1_1[1]                                 | $TC_CARR8 |
| MD24570 $MC_TRAFO5_AXIS1_1[2]                                 | $TC_CARR9 |
| MD24572 $MC_TRAFO5_AXIS2_1[0] (direction second rotary axis)  | $TC_CARR10 |
| MD24572 $MC_TRAFO5_AXIS2_1[1]                                 | $TC_CARR11 |
| MD24572 $MC_TRAFO5_AXIS2_1[2]                                 | $TC_CARR12 |
| MD24510 $MC_TRAFO5_ROT_AX_OFFSET_1[0] (position offset of rotary axes 1/2/3 for 5-axis transformation) 1) | $TC_CARR24 (+$TC_TCARR64) |
| MD24510 $MC_TRAFO5_ROT_AX_OFFSET_1[1]                         | $TC_CARR25 (+$TC_TCARR65) |
| MD24520 $MC_TRAFO5_ROT_SIGN_IS_PLUS_1[0] (sign of rotary axis 1/2/3 for 5-axis transformation) 1) | TRUE* |
| MD24520 $MC_TRAFO5_ROT_SIGN_IS_PLUS_1[1]                      | TRUE* |

*) Machine data MD24520/MD24620 $MC_TRAFO5_ROT_SIGN_IS_PLUS_1/2 are redundant. They are used to invert the direction of rotation of the assigned rotary axis. However, this can also be achieved by inverting the direction of axis vector $MC_TRAFO5_AXIS1/2_1/2. It is for this reason that there is no corresponding parameter for the orientable toolholder. For the purpose of absolute clarity, the following machine data must be ignored:

MD24520/MD24620 TRAFO5_ROT_SIGN_IS_PLUS_1/2
Assignments for transformation type 24

Toolholder data assignments dependent on transformation type 24

| Transformation type "T" (in accordance with MD24100 $MC_TRAFO_TYPE_1 = 24) |
|-------------------------------------------------|
| MD24500 $MC_TRAFO5_PART_OFFSET_1[0] (translation vector of 5-axis transformation 1) | $TC_CARR1 (+$TC_TCARR41) |
| MD24500 $MC_TRAFO5_PART_OFFSET_1[1] | $TC_CARR2 (+$TC_TCARR42) |
| MD24500 $MC_TRAFO5_PART_OFFSET_1[2] | $TC_CARR3 (+$TC_TCARR43) |
| MD24560 $MC_TRAFO5_JOINT_OFFSET_1[0] (vector of the kinematic offset of 5-axis transformation 1) | $TC_CARR4 (+$TC_TCARR44) |
| MD24560 $MC_TRAFO5_JOINT_OFFSET_1[1] | $TC_CARR5 (+$TC_TCARR45) |
| MD24560 $MC_TRAFO5_JOINT_OFFSET_1[2] | $TC_CARR6 (+$TC_TCARR46) |
| MD24550 $MC_TRAFO5_BASE_TOOL_1[0] (vector of basic tool when 5-axis transformation is active) | $TC_CARR15 (+$TC_TCARR55) |
| MD24550 $MC_TRAFO5_BASE_TOOL_1[1] | $TC_CARR16 (+$TC_TCARR56) |
| MD24550 $MC_TRAFO5_BASE_TOOL_1[2] | $TC_CARR17 (+$TC_TCARR57) |

Assignments for transformation type 40

Toolholder data assignments dependent on transformation type 40

| Transformation type "P" (in accordance with MD24100 $MC_TRAFO_TYPE_1 = 40) |
|-------------------------------------------------|
| MD24550 $MC_TRAFO5_BASE_TOOL_1[0] | $TC_CARR4 (+$TC_TCARR44) |
| MD24550 $MC_TRAFO5_BASE_TOOL_1[1] | $TC_CARR5 (+$TC_TCARR45) |
| MD24550 $MC_TRAFO5_BASE_TOOL_1[2] | $TC_CARR6 (+$TC_TCARR46) |
| MD24560 $MC_TRAFO5_JOINT_OFFSET_1[0] | $TC_CARR15 (+$TC_TCARR55) |
| MD24560 $MC_TRAFO5_JOINT_OFFSET_1[1] | $TC_CARR16 (+$TC_TCARR56) |
| MD24560 $MC_TRAFO5_JOINT_OFFSET_1[2] | $TC_CARR17 (+$TC_TCARR57) |
| MD24500 $MC_TRAFO5_PART_OFFSET_1[0] | $TC_CARR18 (+$TC_TCARR58) |
| MD24500 $MC_TRAFO5_PART_OFFSET_1[1] | $TC_CARR19 (+$TC_TCARR59) |
| MD24500 $MC_TRAFO5_PART_OFFSET_1[2] | $TC_CARR20 (+$TC_TCARR60) |
Assignments for transformation type 56

Toolholder data assignments dependent on transformation type 56

| Transformation type "M" (in accordance with MD24100 $MC\_TRAFO\_TYPE\_1 = 56)                                      |                                      |
|-------------------------------------------------------------------------------------------------|-------------------------------------|
| MD24560 $MC\_TRAFO5\_JOINT\_OFFSET\_1[0] (vector of the kinematic offset of 5-axis transformation 1) | TC\_CARR1 (+TC\_TCARR41)            |
| MD24560 $MC\_TRAFO5\_JOINT\_OFFSET\_1[1]                                                   | TC\_CARR2 (+TC\_TCARR42)            |
| MD24560: TRAFO5\_JOINT\_OFFSET\_1[2]                                                       | TC\_CARR3 (+TC\_TCARR43)            |
| MD24550 $MC\_TRAFO5\_BASE\_TOOL\_1[0]                                                      | TC\_CARR4 (+TC\_TCARR44)            |
| MD24550 $MC\_TRAFO5\_BASE\_TOOL\_1[1]                                                      | TC\_CARR5 (+TC\_TCARR45)            |
| MD24550 $MC\_TRAFO5\_BASE\_TOOL\_1[2]                                                      | TC\_CARR6 (+TC\_TCARR46)            |
| MD24558 $MC\_TRAFO5\_JOINT\_OFFSET\_PART\_1[0]                                             | TC\_CARR15 (+TC\_TCARR55)           |
| MD24558 $MC\_TRAFO5\_JOINT\_OFFSET\_PART\_1[1]                                             | TC\_CARR16 (+TC\_TCARR56)           |
| MD24558 $MC\_TRAFO5\_JOINT\_OFFSET\_PART\_1[2]                                             | TC\_CARR17 (+TC\_TCARR57)           |
| MD24500 $MC\_TRAFO5\_PART\_OFFSET\_1[0]                                                    | TC\_CARR18 (+TC\_TCARR58)           |
| MD24500 $MC\_TRAFO5\_PART\_OFFSET\_1[1]                                                    | TC\_CARR19 (+TC\_TCARR59)           |
| MD24500 $MC\_TRAFO5\_PART\_OFFSET\_1[2]                                                    | TC\_CARR20 (+TC\_TCARR60)           |

Example of parameterization

The first 5-axis transformation is to obtain its data from machine data and the second, in contrast, is to be parameterized using the data from the 3rd orientable toolholder.

| MD24100 $MC\_TRAFO\_TYPE\_1 = 24 | ; first 5-axis transformation       |
| MD24200 $MC\_TRAFO\_TYPE\_2 = 72 | ; second 5-axis transformation     |
| MD24682 $MC\_TRAFO5\_TCARR\_NO\_2 = 3; | ; parameterize data of the third orientable toolholder |
1.7.5 Extension of the generic transformation to six axes - 840D sl only

Application

With the maximum 3 linear axes and 2 rotary axes, the motion and direction of the tool in space can be completely described with the generic 5-axis transformation. Rotations of the tool around itself, as is important for a tool that is not rotation-symmetric or robots, require an additional rotary axis. The previous generic 5-axis transformation will therefore be extended by a 3rd rotary axis and further functions added.

- Extension to 3 linear axes and 3 rotary axes, i.e. 6 axes.
- General use of the generic orientation transformation with unchanged parameterization of machine data.
- Cartesian manual travel also for the generic transformation.

Kinematics for the 6-axis transformation

The 6-axis transformation is based on the generic 5-axis transformation and is extended by transformation type 57. Therefore, four different machine kinematics exist that are differentiated through the specification of the transformation type in the following machine data:

\[ \text{MD24100 } \$\text{MC\_TRAFO\_TYPE\_1} = \text{transformation type} (\text{definition of transformation 1 in the channel}) \]

Table 1-3 Overview of machine types for the generic 6-axis transformation

| Machine type       | 1       | 2       | 3       | 4       |
|--------------------|---------|---------|---------|---------|
| swivel/rotatable   | tool    | workpiece | Tool/ workpiece | Tool/ workpiece |
| Transformation types | 24      | 40      | 56      | 57      |
| Orientation in space, rotation of the axes | Unchanged. All three axes rotate the tool | Unchanged. All three axes rotate the workpiece | Tool by 2 axes, workpiece by one rotary axis | Tool by one axis, workpiece by two rotary axes |
In all four cases, the first rotary axis is the one which closest to the workpiece and the third rotary axis the one which closest to the tool in the kinematic chain.

**Note**

The four specified transformation types only cover those kinematics in which the three linear axes form a rectangular Cartesian coordinate system, i.e. no kinematics are covered in which at least one rotary axis lies between two linear axes in the kinematic chain.

Dedicated machine data exist for each general transformation or for each orientation transformation that are differentiated by the suffixes _1, _2 etc. (e.g. MD24100 $MC\_TRAFO\_TYPE\_1, MD24200 $MC\_TRAFO\_TYPE\_2 etc.). In the following, only the names for the first transformation are specified, i.e. those with the suffix _1. If a transformation other than the first is parameterized, the correspondingly modified names must be used.

**Configuration**

For configuration of a 6-axis transformation the extensions of the following machine data are required:

- The channel axis index of the 3rd rotary axis must be entered in the following machine data:
  - MD24110 $MC\_TRAFO\_AXES\_IN\_1[5]$ (axis assignment for transformation)
- The direction of the 3rd rotary axis must be specified in the following machine data:
  - MD24573 $MC\_TRAFO5\_AXIS3\_1[0..2]$ (direction 3rd rotary axis)
- An orientation normal vector with a length not equal to zero and which is not parallel or anti-parallel to the orientation vector defined in machine data MD24574 $MC\_TRAFO5\_BASE\_ORIENT\_1[0..2]$ (basic tool orientation), must be specified in machine data MD24576 $MC\_TRAFO6\_BASE\_ORIENT\_NORMAL\_1[0..2]$ (tool normal vector).

The previous offsets (vector):

- MD24550 $MC\_TRAFO5\_BASE\_TOOL\_1[0..2]$  (vector of the basic tool with activation of the 5-axis transformation) 1)
- MD24560 $MC\_TRAFO5\_JOINT\_OFFSET\_1[0..2]$  (vector of the kinematic offset of 5-axis transformation 1)
- MD24558 $MC\_TRAFO5\_JOINT\_OFFSET\_PART\_1[0..2]$  (vector of kinematic offset in table)
- MD24500 $MC\_TRAFO5\_PART\_OFFSET\_1[0..2]$  (translation vector 5-axis transformation 1)
The following machine data is added as **new offset (vector)**, describing the offset between the second and third rotary axis:

- MD24561 $MC_TRAFO6_JOINT_OFFSET_2_3_1[0..2]
  (vector of kinematic offset)

**Note**

Existing machine data blocks are compatible for transfer, without any changes having to be made in the machine data. The new machine data therefore do not have to be specified for a 3-/4-/5-axis transformation.

---

### Programming of orientation

With the extension of the generic orientation transformation to 6 axes, all three degrees of freedom of the orientation can be freely selected. They can be uniquely defined through the position of a rectangular Cartesian coordinate system. One axis direction, that of the third axis, (typically in the Z direction) defines the orientation.

Two degrees of freedom are required for the specification of this direction. The third degree of freedom is defined via a **rotation** around this direction, e.g. through the specification of an angle THETA or a direction vector for one of the two other axes of the coordinate system, see Section "Rotations of orientation vector (Page 112)".

The new addresses AN3, BN3, CN3 define the direction of the second axis, of the coordinate system (typically the Y axis) of the orientation normal vector. The programmed orientation normal vector should be perpendicular to the orientation and is only possible when both programmed vectors are not parallel or anti-parallel. Otherwise, alarm 4342 is output.

The direction of the first axis, the X axis, is then uniquely defined.

---

### Default setting of the orientation normal vector

The default setting of the orientation normal vector in the transformation can also be defined as for the default setting of the orientation in one of three ways:

**Specification for the activation of the transformation**

1. Vector components are transferred as parameters 8 to 10:
   - Parameter 1: Transformation No.
   - Parameter 2 - 4: Orientation vector,
   - Parameter 5 - 7: Rotary axis offsets

2. If **no** orientation normal vector has been specified and a tool is active, the vector is taken from the tool data.

3. If **no** orientation normal vector has been specified and also **no** tool is active, the vector defined in the following machine data is used.

   MD24567 $MC_TRAFO6_BASE_ORIENT_NORMAL_1[0..2] (tool normal vector)
The position of the orientation coordinate system of a standard tool depends on the active plane G17, G18, G19 according to the following table:

Table 1-4 Position of the orientation coordinate system

|                         | G17 | G18 | G19 |
|-------------------------|-----|-----|-----|
| Direction of the orientation vector | Z   | Y   | X   |
| Direction of the orientation normal vector | Y   | X   | Z   |

Note

The orientation vector of a tool can also be defined via system the variables $TC_DPV or $TC_DPV3 - $TC_DPV5 in tool data - see Function Manual Basic Machine, Tool Corrections (W1), Section: Sum and setup offsets.

This option is expanded in order to specify the orientation normal vector, using system variables $TC_DPVN3 - $TC_DPVN5. The meaning of the vector components is similar to the meaning of the components of the tool orientation:

$TC_DPVN3 is the component in the direction of tool length L1,

$TC_DPVN4 is the component in the direction of tool length L2,

$TC_DPVN5 is the component in the direction of tool length L3,

The following machine data must have the value 3 in order to allow the new tool parameters to be used:

MD18114 $MN_MM_ENABLE_TOOL_ORIENT (assign orientation to tool cutting)

The coordinate system is not rotated through the programming of a rotation of the tool with AN3, BN3, CN3 or THETA.

Programming example

See Section "Example of a generic 6-axis transformation (Page 137)".
1.7.6 Extension of the generic transformation to seven axes - 840D sl only

Application

The generic 5-/6-axis transformation with transformation type 24 is extended by a 7th or 6th axis, which rotates the workpiece. The work space of the transformation can be expanded in this way.

Requirement

For generic 7-axis transformation there must be at least 6 or 7 axes.

Function

Another 7th axis is required in connection with the generic 6-axis transformation which rotates the workpiece. This 7th axis is considered only along with transformation type 24 (generic 6-axis transformation having 3 rotary axes that move the tool).

The position of the 7th axis is specified according to a strategy of the CAD system and settled with the Cartesian position (X, Y, Z) by the generic transformation in such a way that the axes always approach the TCP position programmed with reference to the workpiece, independently of the position of the 7th axis. If ORIWK is active, the end orientation programmed with reference to the workpiece is also rotated by the 7th axis. This way it is possible to program the orientation in relation to the workpiece.

The transformation uses the 7th axis as the observed input variable.

To configure the 7th axis, the channel machine data of the 5-/6-axis transformation is extended by one field containing the 3 components of the direction vector of the 7th axis and an axis offset.

This gives the following advantages:

- The contour and the orientation at the workpiece can be programmed in relation to the workpiece.
- The programmed feed is maintained in the contour, even if the 7th axis also moves.
- All the contour-related control functions can be used.
- The displayed WCS position corresponds to the programmed position.
- The transformation is configured as in generic 6-axis transformation. One can switch between a 6-axis and a 7-axis transformation smoothly.
- In case of large radius circular interpolation, the release of singularities incorporating the 7th axis.
Notations

Dedicated machine data exist for each general transformation and for each orientation transformation that are differentiated by the suffixes _1, _2 etc. (e.g. $MC.TRAFO.TYPE_1, $MC.TRAFO.TYPE_2 etc.). In the following, only the names for the first transformation are specified, i.e. those with the suffix _1. If a transformation other than the first is parameterized, the correspondingly modified names must be used.

Description of the kinematics

The 7-axis transformation builds on the generic 5-/6-axis transformation.

Note

The 7-axis transformation also covers kinematics in which the 6th axis is not available. In the following pages, we speak exclusively about a 7th axis or about a 7-axis transformation, even when it is actually the 6th axis in connection with a 5-axis kinematics.

The 7-axis transformation types only cover those kinematics in which the three linear axes form a rectangular Cartesian coordinate system, i.e. no kinematics are covered in which at least one rotary axis lies between two linear axes in the kinematic chain.

There is only one machine kinematics for which a 7th axis can be configured. It is designated by the Transformation Type 24:

$MC.TRAFO.TYPE_1 = 24  \text{ Rotary tool: Three (or two) axes rotate the tool; the 7th axis rotates the workpiece.}$

The extensions of the following machine data are required to configure a generic 7-axis transformation:

| Machine data                              | extension                                                                 |
|-------------------------------------------|---------------------------------------------------------------------------|
| $MC.TRAFO_AXES.IN_1[9]$                   | The channel axis index of the 4th rotary axis is recorded here.            |
| $MC.TRAFO_AXES.IN_1[10]$ and              | This machine data is to be assigned with default value of 0. (default setting) |
| $MC.TRAFO_AXES.IN_1[11]$                  | This machine data is not evaluated by the generic 7-axis transformation.   |
| $MC.TRAFO7.EXT_AXIS1[0..2]$               | The direction of the 4th rotary axis is specified here.                    |
| $MC.TRAFO7.EXT_AX_OFFSET_1[0..2]$         | A position offset of the 4th rotary axis is recorded here.                |
1. Programming the Cartesian position

The position of the 7th axis must be programmed in the workpiece coordination system in addition to the Cartesian position. The Cartesian position is thus programmed in relation to the constant workpiece. The 7-axis transformation converts the WCS position via the rotation of the 7th axis in the basic coordinate system. Possibly programmed or set frames are normally settled before the 7-axis transformation.

2. Programming of orientation

All programming options of the generic 5-/6-axis transformation are available while programming the orientation. The 7th axis must always be programmed additionally.

Two different response types can be set in this context via the G code.
- The position of the 7th axis does not influence the programmed orientation.
- The programmed end-orientation is rotated with the 7th axis.
Orientation

1. Orientation with axis interpolation

   If the 7th axis should have no influence on the programmed orientation, the G codes of Groups 25 and 51 must be set accordingly:

   G code group 25: ORIMKS
   G code group 51: ORIAXES (if MD21104 $MC_ORI_IPO_WITH_G_CODE = 1 is set).

   The programmed positions of the rotary axes are not changed by the position of the 7th axis in this case, but approached directly. The orientation is programmed in relation to the machine.

   Example

   Program code

   | TRAORI(1)  |
   | ORIAXES    |
   | ORIMKS     |
   | G1 X500 Y300 Z800 C15 A5 C1=10 E1=120 |

1. Orientation and large circle interpolation

   If traversing is to be done with large radius circular interpolation, the end orientation is rotated with the 7th axis.

   G code group 25: ORIWKS
   G code group 51: ORIVECT (if MD21104 $MC_ORI_IPO_WITH_G_CODE = 1 is set).

   In this case the orientation must be programmed in relation to the workpiece. The programmed orientation is thus related to the fixed workpiece. The position of the 7th axis is thus not contained in the programmed orientation.

   Example

   Program code

   | TRAORI(1)  |
   | ORIVECT    |
   | ORIWKS     |
   | G1 X500 Y800 Z100 A3=0 B3=1 C3=0 AN3=0 BN3=0 CN3=–1 E1=–90 |
Frames

The basic coordinate system sits on the 7th axis. It is also rotated when the 7th axis rotates. This way the workpiece coordinate system (WCS) does not remain stationary when the workpiece is rotated over the 7th axis. A workpiece position rotated to the zero position of the 7th axis can be compensated by an axial frame offset of the 7th axis.

Traversing with the 7th axis in the JOG mode

Only the compensatory movements for the linear axes are created if the 7th axis is traversed in the JOG mode with active 7-axis transformation. The position at the workpiece is kept constant in this way. As the rotary axes go into the transformation only as input axes, they are not influenced by the 7th axis during the JOG travel. The orientation at the workpiece is thus kept variable.

1.7.7 Cartesian manual travel with generic transformation

Note

The use of the "Handling transformation package" option is necessary for the "Cartesian manual travel" function.

Functionality

The "Cartesian manual travel" function, as a reference system for JOG mode, allows axes to be set independently of each other in Cartesian coordinate systems:

- Basic coordinate system (BCS)
- Workpiece coordinate system (WCS)
- Tool coordinate system (TCS)

The following machine data not only activates the function, but also sets the permitted coordinate systems.

MD21106 $MC_CART_JOG_SYSTEM (coordinate systems for Cartesian JOG)

For JOG motion, one of the three reference systems can be set not only for the translation/movement of the geometry axes, but also for tool orientation/movement of the orientation axes via the setting data SD42650 $SC_CART_JOG_MODE (coordinate systems for Cartesian manual traverse) independently from one another.
Activation

The following machine data not only activates the function, but also sets the permitted coordinate systems.

MD21106 $MC_CART_JOG_SYSTEM (coordinate systems for Cartesian JOG)

The following setting data sets the virtual kinematics used for traversing motion of the orientation:

SD42660 $SC_ORI_JOG_MODE (definition of virtual kinematics for JOG)

As opposed to the generic 5-/6-axis transformation, only kinematics can be set in which the rotary axes are perpendicular to one another.

The traversing of the geometry and orientation axes is performed via the VDI interface signals of the geometry or orientation axes.

Translations

A translatory movement can be used to move the tool tip (TCP) 3-dimensionally in parallel to the axes of the set reference system. Traversing is performed via the VDI interface signals of the geometry axes.

Note

For further information about the representation of the translations for the Cartesian manual travel in the corresponding coordinate systems, see:

References:
Function Manual Extension Functions; Kinematic Transformation (M1)

Tool orientation

The tool can be aligned to the workpiece surface via an orientation movement. The motions of the orientation axes are triggered by the PLC via the VDI interface signals of the orientation axes. The virtual orientation axes execute rotations around the fixed directions of the relevant reference system. Virtual kinematics is defined by the following setting data via the active transformation:

SD42660 $SC_ORI_JOG_MODE = 0
Rotations of the orientations

Rotation of the orientation axes is defined by additional settings of the following setting data:

**SD42660 $MC_ORI_JOG_MODE**

The options are as follows:

**Rotations with JOG**

With JOG, the rotations around the specified directions of the respective reference system can be performed with Euler angle or RPY angle.

**SD42660 $SC_ORI_JOG_MODE = 1:** When jogging, **Euler angles** are traversed, i.e.:
- the first axis rotates around the z direction,
- the second axis rotates around the x direction,
- the third axis (if present) rotates around the new z direction.

**SD42660 $SC_ORI_JOG_MODE = 2:** When jogging, **RPY angles** are traversed with rotation sequence XYZ, i.e.:
- the first axis rotates around the x direction,
- the second axis rotates around the y direction,
- the third axis (if present) rotates around the new z direction.

**SD42660 $SC_ORI_JOG_MODE = 3:** When jogging, **RPY angles** are traversed with rotation sequence ZYX, i.e.:
- the first axis rotates around the z direction,
- the second axis rotates around the y direction,
- the third axis (if present) rotates around the new x direction.

**Rotation sequence of the rotary axes**

Rotation sequence of the rotary axes is set via the following setting data:

- **SD42660 $SC_ORI_JOG_MODE = 4:**
  - via machine data MD21120 $MC_ORIAX_TURN_TAB_1 (definition of reference axes for ORI axes)

- **SD42660 $SC_ORI_JOG_MODE = 5:**
  - via machine data MD21130 $MC_ORIAX_TURN_TAB_2 (definition of reference axes for ORI axes)

For further explanations of the orientation movements (see Section "Orientation (Page 81)" and "Orientation axes (Page 100)").

**Note**

For further information about the programming of rotations please refer to:

**References:**

Programming Manual, Job Planning; Section: "Transformations"
1.8 Restrictions for kinematics and interpolation

For systems where there are less than six axes available for transformation, the following restrictions must be taken into account.

5-axis kinematics

For 5-axis kinematics there are two degrees of freedom for orientation. The assignment of orientation axes and tool vector direction must be selected so that there is no rotation around the tool vector. As a result, only two orientation angles are required to describe the orientation. If the axis is traversed using ORIVECT, the tool vector performs pure swiveling motion.

3-and 4-axis kinematics

For 3- and 4-axis kinematics, only one degree of freedom is available for orientation. The respective transformation determines the relevant orientation angle. In this case, it only makes sense to traverse the orientation axis using ORIAXES. In this case, the orientation axis is directly and linearly interpolated.

Interpolation of the tool orientation over several blocks by means of orientation vectors

If the orientation of a tool is programmed over several consecutive part program blocks by directly entering the appropriate rotary axis positions, then undesirable discontinuous changes of the orientation vector are obtained at the block transitions. This results in discontinuous velocity and acceleration changes of the rotary axes. This means that no continuous velocity and acceleration of the orientation axes over several blocks can be achieved using large circle interpolation.

Continuous block transitions

As long as only linear blocks (G1) are programmed, then the orientation axes also behave just like linear axes. In this case, motion with continuous acceleration is achieved through polynomial interpolation. Significantly better results can be achieved by programming the orientation in space using orientation vectors (see Section "Polynomial interpolation of orientation vectors (Page 108)").
1.8.1 Singularities of orientation

Description of problem

As described in Section "Singularities and how to treat them", singularities (poles) are constellations in which the tool is orientated becomes parallel to the first rotary axis. If the orientation is changed when the tool is in or close to a singularity (as is the case with large-circle interpolation ORIWKS ), the rotary axis positions must change by large amounts to achieve small changes in orientation. In extreme cases, a jump in the rotary axis position would be needed.

Such a situation would be treated as follows:

There is only one relevant machine data, which circles the pole as usual:

MD24540 $MC_TRAFO5_POLE_LIMIT_1 (closing angle tolerance for interpolation by pole for 5-axis transformation)

or

MD24640 $MC_TRAFO5_POLE_LIMIT_2 (closing angle tolerance for interpolation by pole for 5-xis transformation)

For further information about the handling of singular positions, see:

References:

Programming Manual, Job Planning, Transformations; Section: Cartesian PTP travel
Example for machine type 1

Rotatable tool

Both rotary axes change the orientation of the workpiece. The orientation of the workpiece is fixed.

2-axis swivel head with rotary axis RA 1 (4th transformation axis) and rotary axis RA 2 (5th transformation axis)

Figure 1-22 Generic 5-axis transformation; end point of orientation inside tolerance circle.
End point within the circle

If the end point is within the circle, the first axis comes to a standstill and the second axis moves until the difference between target and actual orientation is minimal. However, since the first rotary axis does not move, the orientation will generally deviate from the programmed value (see previous figure). However, the programmed orientation can at least be reached exactly if the first rotary axis happens to be positioned correctly.

Note

In the previous Figure the resulting path is a straight line because the position of the first rotary axis is constant on that path. This representation is always correct, irrespective of the angle between the two rotary axes. The orientation vector only moves in a plane, however, if the two rotary axes and the basic orientation are all mutually perpendicular. In all other cases, the orientation vector describes the outside of a cone.

End point outside the circle

If the orientation interpolation describes a path through the circle, while the end point is outside the circle, the end point is approached with axis interpolation. This applies in particular if the interpolation starting point is located inside the circle. Path deviations from the programmed setpoint orientation are thus unavoidable.
1.9 Orientation

1.9.1 Basic orientation

Differences to the previous 5-axis transformations

In the 5-axis transformations implemented to date, basic orientation of the tool was defined by the type of transformation.

Generic 5-axis transformation can be used to enable any basic tool orientation, i.e. space orientation of the tool is arbitrary, with axes in their initial positions.

If an orientation is programmed by means of Euler angles, RPY angles (A2, B2, C2) or vectors (A3, B3, C3), basic orientation is taken into consideration, i.e. the rotary axes are positioned so that a tool positioned in basic orientation is traversed to the programmed orientation.

If the rotary axes are programmed directly, basic orientation has no effect.

Definition

There are three different ways to define basic orientation:

1. Via the transformation call
2. Via the orientation of the active tool
3. Via machine data

Via the transformation call

For 1.:

When the transformation is called, the direction vector of the basic orientation can be specified in the call, e.g. TRAORI(0, 0., 1., 5.). The direction vector is defined by parameters 2 to 4; the vector in the example therefore has the value (0., 1., 5.).

The first parameter specifies the transformation number. The number can be omitted if the first transformation is to be activated. To enable the parameters to be identified correctly when specifying an orientation, a blank space has to be inserted instead of the transformation number, e.g. TRAORI(, 0., 1., 5.).

Note

The orientation data is absolute; it will not be modified by any active frame.

The absolute value of the vector is insignificant; only the direction is relevant. Non-programmed vector elements can be set to zero.
Please note that if all three vector components are zero (because they have been set explicitly so or not specified at all), the basic orientation is not defined by data in the TRAORI(...) call, but by one of the methods described below.

If a basic orientation is defined by the above method, it cannot be altered while a transformation is active. The orientation can be changed only by selecting the transformation again.

**Via the orientation of the active tool**

**For 2.:**

The basic orientation is determined by the tool

- if it has not been defined through specification of a direction vector in the transformation call
- and if a tool is already active.

The orientation of a tool is dependent on the selected plane. It is parallel to Z at G17, parallel to Y at G18 and parallel to X at G19.

It can be modified arbitrarily by orientable toolholders, see:

References:

Function Manual, Basic Machine; Tool Offset, Section: Orientable toolholders (W1)

If the tool is changed when a transformation is active, the basic orientation is also updated. The same applies if the orientation of a tool changes as the result of a change in plane (plane changes are equivalent to tool changes, as they also alter the assignment between tool length components and individual axes).

If the tool is de-selected, thereby canceling the definition of tool orientation, the basic orientation programmed in machine data becomes operative.

**Via machine data**

**For 3.:**

If the basic orientation is not defined by either of the two variants described above, it is specified with reference to the following machine data.

$MC_TRAFO5_BASE_ORIENT_n (basic tool orientation)

This machine data must not be set to a zero vector or else an alarm will be generated during control run-up when a transformation is active.
1.9 Orientation

If a basic orientation is programmed in machine data $MC\_TRAFO5\_BASE\_ORIENT\_n$ when a transformation is active and a tool is subsequently activated, the basic orientation is re-defined by the tool.

**Note**

The range of settable orientations depends on the directions of the rotary axes involved and the basic orientation. The rotary axes must be mutually perpendicular if all possible orientations are to be used. If this condition is not met, "dead" ranges will occur.

**Examples:**

1. Extreme example: A machine with rotatable tool has a C axis as its first rotary axis and an A axis as its second. If the basic orientation is defined in parallel to the A axis, the orientation can only be changed in the X-Y plane (when the C axis is rotating), i.e. orientation with a Z component unequal to zero is not possible in this instance. The orientation does not change when the A axis rotates.

2. Realistic example: A machine with nutator kinematics (cardan head) with an axis inclined at less than $45^\circ$ in a basic orientation parallel to the Z axis can only assume orientations within a semi-circle: The top semi-circle with basic orientation towards +Z and the bottom with basic orientation towards -Z.

### 1.9.2 Orientation movements with axis limits

**Calculate rotary axis position**

If the final orientation in a 5-axis transformation is programmed indirectly in an NC block by means of a Euler, RPY angle or direction vector, it is necessary to calculate the rotary axis positions that produce the desired orientation. This calculation has no unique result.

There are always at least two essentially different solutions. In addition, any number of solutions can result from a modification to the rotary axis positions by any multiple of 360 degrees.

The control system chooses the solution which represents the shortest distance from the current starting point, allowing for the programmed interpolation type.
Determining permissible axis limits

The control system attempts to define another permissible solution if the axis limits are violated, by approaching the desired axis position along the shortest path. The second solution is then verified, and if this solution also violates the axis limits, the axis positions for both solutions are modified by multiples of 360 until a valid position is found.

The following conditions must be met in order to monitor the axis limits of a rotary axis and modify the calculated end positions:

- A generic 5-axis transformation of type 24, 40 or 56 must be active.
- The axis must be referenced.
- The axis must not be a modulo rotary axis.
- The following machine data may not be equal to zero:
  
  MD21180 $MC_ROT_AX_SWL_CHECK_MODE (check software limits for orientation axes)

The following machine data specifies the conditions under which the rotary axis positions may be modified:

MD21180 $MC_ROT_AX_SWL_CHECK_MODE

Value 0: No modification permitted (default, equivalent to previous behavior).
Value 1: Modification is only permitted if axis interpolation is active (ORIAXES or ORIMKS).
Value 2: Modification is always permitted, even if vector interpolation (large circle interpolation, conical interpolation, etc.) was active originally.

Switch-over to axis interpolation

If the axis positions have to be changed from the originally determined value, the system switches to rotary axis interpolation because the original interpolation path, e.g. large circle interpolation or conical interpolation, can no longer be maintained.

Example

An example is shown in Chapter "Example for Generic 5-axis Transformation" for modifying the rotary axis motion of a 5-axis machine with a rotatable tool.
1.9.3 Orientation compression

Function

The compressor functions COMPON, COMPCURV and COMPCAD can also be used to compress NC programs containing orientations programmed with the help of direction vectors, to a definable tolerance.

Precondition

A precondition for the compression of orientations is the availability of the "Orientation Transformation" option.

Conditions

The orientation movement is compressed in the following cases:

- Active orientation transformation ($TRORI$)
- Active large radius circular interpolation (i.e. tool orientation is changed in the plane which is determined by start and end orientation).

Large circle interpolation is performed under the following conditions:

- MD21104 $MC\_ORI\_IPO\_WITH\_G\_CODE = 0$,
  ORIWKs is active and orientation is programmed with the help of vectors (with $A3, B3, C3$ or $A2, B2, C2$).
- MD21104 $MC\_ORI\_IPO\_WITH\_G\_CODE = 1$ and ORI VECT or ORIPLANE is active.

The tool orientation can be programmed either as a direction vector or with rotary axis positions. If one of the G-codes ORICONxx or ORICURVE is active, or if polynomials for the orientation angles ($PO[PHI]$ and $PO[PSI]$) are programmed, no large circle interpolation will be executed.

Parameter assignment

NC blocks can only be compressed if deviations are allowed between the programmed contour and interpolated contour or between the programmed orientation and interpolated orientation.

Compression tolerances can be used to set the maximum permissible deviation. The higher the tolerances, the more blocks can be compressed. However, the higher the tolerances, the more the interpolated contour or orientation can deviate from the programmed values.

Axis accuracy

For each axis, the compressor creates a spline curve which deviates from the programmed end points of each axis by a maximum of the value set with the following machine data.

MD33100 $MA\_COMPRESS\_POS\_TOL$ (maximum tolerance with compression)
Contour accuracy

The maximum contour deviations (geo axes) and tool orientation are specified via the following setting data:

SD42475 $SC_COMPRESS_CONTUR_TOL (maximum contour deviation for compressor)

SD42476 $SC_COMPRESS_ORI_TOL (maximum angular deviation for tool orientation compressor)

SD42477 $SC_COMPRESS_ORI_ROT_TOL (maximum angular deviation for the angle of rotation of the tool) (only available on 6-axis machines)

Note

It is only possible to specify a maximum angular deviation for tool orientation if an orientation transformation (TRAORI) is active.

Compression mode

The manner in which the tolerances are to be considered is set via the unit position in the machine data:

MD20482 $MC_COMPRESSOR_MODE (mode of compression)

| Value | Meaning |
|-------|---------|
| 0     | The tolerances specified with MD33100 $MA_COMPRESS_POS_TOL are observed for all the axes (geo and orientation axes). |
| 1     | The contour tolerance specified with SD42475 $SC_COMPRESS_CONTUR_TOL is effective for the geometry axes. The axis-specific tolerances MD33100 $MA_COMPRESS_POS_TOL are effective for the orientation axes. |
| 2     | The axis-specific tolerances MD33100 $MA_COMPRESS_POS_TOL are effective for the geometry axes. The maximum angular deviations SD42476 $SC_COMPRESS_ORI_TOL or SD42477 $SC_COMPRESS_ORI_ROT_TOL are effective for the orientation axes. |
| 3     | The contour tolerance specified with SD42475 $SC_COMPRESS_CONTUR_TOL is effective for the geometry axes. The maximum angular deviations SD42476 $SC_COMPRESS_ORI_TOL or SD42477 $SC_COMPRESS_ORI_ROT_TOL are effective for the orientation axes. |
With the **tens digit** of MD20482 you set whether blocks with programmed tool orientation and/or value assignments (for example, X=100 ...) are to be compressed or not:

| Value | Meaning |
|-------|---------|
| x0x   | All blocks with programmed tool orientation and/or value assignments will be compressed (default setting). Notice: This behavior is incompatible with earlier SW versions! |
| x1x   | Only the blocks with programmed tool orientation will be compressed. The blocks with value assignments will **not** be compressed. |
| x2x   | Only the blocks with value assignments will be compressed. The blocks with programmed tool orientation will **not** be compressed. |
| x3x   | All blocks with programmed tool orientation and/or value assignments will **not** be compressed. This setting provides a behavior that is fully compatible with earlier SW versions. |

The **hundreds position** of MD20482 is used to select which blocks outside the linear blocks (G1) should be compressed:

| Value | Meaning |
|-------|---------|
| 0xx   | Circular blocks and G0 blocks are **not** compressed. This is compatible with earlier SW versions. |
| 1xx   | Circular blocks will be linearized and compressed by COMPCAD. Advantage: The compressor function works more accurate and therefore creates normally better surfaces. Disadvantage: The compressor function is more sensitive to bugs in the NC programs. For reasons of compatibility it might therefore be necessary to keep the setting 0xx. |
| 2xx   | G0 blocks are compressed - it is possible that a different tolerance becomes effective (see MD20560 $MC_G0_TOLERANCE_FACTOR or NC command STOLF). Advantage: As a result that the tolerance has been set higher and the compression of G0 infeed motion, these can be more quickly and more fluidly executed. |
| 3xx   | Combination of the two previous options: Circular blocks as well as G0 blocks are compressed. |
Programming

Tool orientation
If orientation transformation (TRAORI) is active, for 5-axis machines, tool orientation can be programmed in the following way (independent of the kinematics):

- Programming of the direction vectors via:
  \[ A_3 = \ldots \quad B_3 = \ldots \quad C_3 = \ldots \]

- Programming of the Euler angles or RPY-angles via:
  \[ A_2 = \ldots \quad B_2 = \ldots \quad C_2 = \ldots \]

Rotation of the tool
For six-axis machines you can program the tool rotation in addition to the tool orientation.

The angle of rotation is programmed with:

\[ \text{THETA} = \ldots \]

Note
NC blocks in which additional rotation is programmed, can only be compressed if the angle of rotation changes linearly, meaning that a polynomial with \( \text{PO[THT]}(\ldots) \) for the angle of rotation should not be programmed.

General structure of an NC block that be compressed
The general structure of an NC block that can be compressed can therefore look like this:

\[
\begin{align*}
N & \ldots X = \ldots \quad Y = \ldots \quad Z = \ldots \quad A_3 = \ldots \quad B_3 = \ldots \quad C_3 = \ldots \quad \text{THETA} = \ldots \quad F = \ldots \\
N & \ldots X = \ldots \quad Y = \ldots \quad Z = \ldots \quad A_2 = \ldots \quad B_2 = \ldots \quad C_2 = \ldots \quad \text{THETA} = \ldots \quad F = \ldots 
\end{align*}
\]

or

Programming tool orientation using rotary axis positions
Tool orientation can be also specified using rotary axis positions, e.g. with the following structure:

\[
\begin{align*}
N & \ldots X = \ldots \quad Y = \ldots \quad Z = \ldots \quad A = \ldots \quad B = \ldots \quad C = \ldots \quad \text{THETA} = \ldots \quad F = \ldots 
\end{align*}
\]

In this case, compression is executed in two different ways, depending on whether large radius circular interpolation is executed. If no large radius circular interpolation takes place, then the compressed change in orientation is represented in the usual way by axial polynomials for the rotary axes.
Activation/deactivation

Compressor functions are activated using the modal G codes COMPON, COMPCURV or COMPCAD. COMPOF terminates the compressor function.

Programming example

See "Example: Compression of an orientation (Page 139)"

References

The COMPON, COMPCURV and COMPCAD compression functions are described in: Function Manual Basic Functions; Continuous Path Mode, Exact Stop, LookAhead (B1), Chapter: "NC block compression"

1.9.4 Smoothing of orientation characteristic

Introduction

With many of the NC programs for 5-axis machining created with CAD/CAM systems it happens that although the contour characteristic is sufficiently smooth in accordance with the underlying geometry the orientation characteristic contains more or less large fluctuations. These fluctuations in orientation result in very unsmooth running of the orientation axes with permanent acceleration and braking. The compensating motions that the linear axes then have to carry out require that the linear axes also have to be accelerated and braked permanently. Due to these unnecessary accelerations, the possible path velocity is strongly limited and consequently the machining time unnecessarily prolonged.

Function

The "Smoothing the orientation characteristic (ORISON)" function can be used to smooth oscillations affecting orientation over several blocks. The aim is to achieve a smooth characteristic for both the orientation and the contour.

Prerequisites

The "Smoothing the orientation characteristic (ORISON)" function is only available in systems with 5-/6-axis transformation.
Parameter assignment

Number of blocks
Smoothing of the orientation characteristic is carried out by means of a settable number of blocks:

MD28590 $MC_MM_ORISON_BLOCKS = <Value>

For most applications, 10 blocks should be sufficient. The minimum value that should be entered is 4.

Note
If smoothing of the orientation characteristic is activated without sufficient block memory having been configured for it (MD28590 < 4), an alarm message will be output and the function cannot be executed.

Maximum block path length
The orientation characteristic is only smoothed in blocks whose traversing distance is shorter than the settable maximum block path length:

MD20178 $MC_ORISON_BLOCK_PATH_LIMIT

Blocks with longer traversing distances interrupt smoothing and are traversed as programmed.

Maximum tolerance
Smoothing of the orientation characteristic is carried out with the specified maximum tolerance being observed (maximum angular displacement of tool orientation in degrees):

SD42678 $SC_ORISON_TOL

Maximum path distance
The maximum distance over which smoothing is carried is the specified path distance:

SD42680 $SC_ORISON_DIST

Activation/deactivation
The function is activated/deactivated in the part program with the following modal G commands:

ORISON: Smoothing of the orientation characteristic ON
ORISOF: Smoothing of the orientation characteristic OFF
Example

| Program code | Comments |
|--------------|----------|
| ...          |          |
| TRAORI()     | ; Activation of orientation transformation. |
| ORISON       | ; Activation of orientation smoothing. |
| $SC\_ORISON\_TOL=1.0 | ; Orientation tolerance smoothing = 1.0 degrees. |
| G91          |          |
| X10 A3=1 B3=0 C3=1 |          |
| X10 A3=1 B3=0 C3=1 |          |
| X10 A3=1 B3=0 C3=1 |          |
| X10 A3=1 B3=0 C3=1 |          |
| X10 A3=1 B3=0 C3=1 |          |
| X10 A3=1 B3=0 C3=1 |          |
| X10 A3=1 B3=0 C3=1 |          |
| X10 A3=1 B3=0 C3=1 |          |
| X10 A3=1 B3=0 C3=1 |          |
| X10 A3=-1 B3=0 C3=1 |          |
| X10 A3=-1 B3=0 C3=1 |          |
| X10 A3=-1 B3=0 C3=1 |          |
| X10 A3=-1 B3=0 C3=1 |          |
| X10 A3=-1 B3=0 C3=1 |          |
| X10 A3=-1 B3=0 C3=1 |          |
| ORISOF       | ; Deactivation of orientation smoothing. |
| ...          |          |

The orientation is pivoted through 90 degrees on the XZ plane from -45 to +45 degrees. Due to the smoothing of the orientation characteristic the orientation is no longer able to reach the maximum angle values of -45 or +45 degrees.


1.9.5 Orientation relative to the path

Functionality

Irrespective of certain technological applications, the previous programming of tool orientation is improved in that the programmed relative orientation in relation to the total path is maintained. The required deviations from the ideal orientation path can be specified if, for example, a corner occurs in the contour.

Tool orientation can be modified not only via configurable machine data, but also via new language commands in the part program. In this way, it is possible to maintain the relative orientation not only at the block end, but also throughout the entire trajectory. The desired orientation is achieved:

- By settable orientation methods with ORIPATH, specifying how interpolation is to be performed relative to the path.
- Whether the tool orientation should either always run continuously with specifiable deviations from the orientation relative to the path at a block transition, or whether the orientation jump should be smoothed in a dedicated, inserted intermediate block. In this case, path motion is stopped in the contour corner.
- There are two options of 6-axis transformations:
  - Like tool rotation, tool orientation is interpolated relative to the path using ORIPATH, ORIPATHS.
  - The orientation vector is programmed and interpolated in the usual manner. The rotation of the orientation vector is initiated relative to the path tangent using ORIROTC.

Note

Orientation relative to the path interpolation with ORIPATH or ORIPATHS and ORIROTC, cannot be used in conjunction with the function "Orientation smoothing". For this OSOF must be active in the part program. Otherwise alarm 10980 "Orientation smoothing not possible" is generated.

Activate orientation relative to the path

The extended function "Orientation relative to the path" is activated with the following machine data:

MD21094 $MC_ORIPATH_MODE > 0 (setting for path relative orientation ORIPATH)

The tool orientation relative to the path is activated in the part program by programming ORIPATH. A kink in the orientation path, e.g. as can occur at a corner in the contour, can be smoothed with ORIPATH.
Orientation at block transition

In case of the following machine data, tool orientation is always continuous at a block transition:

MD21094 $MC_ORIPATH_MODE = 0

With the following machine data a jump in tool orientation can occur at a block transition:

MD21094 $MC_ORIPATH_MODE > 0

A jump in orientation always occurs when either the path tangent or the surface normal vector does not change smoothly at a block transition.

Deviation from the desired orientation

During the interpolation of the block, the orientation may deviate more or less from the desired relative orientation. The orientation achieved in the previous block is transferred to the programmed end orientation using large circular interpolation. The resulting deviation from the desired relative orientation has two main causes:

1. The end orientation of the previous block refers to the tangent and the normal vector at the end of the previous block. Both can differ from this at the start of the current block. Therefore, the start orientation in the current block does not have the same alignment with respect to the tangent and the normal vector as at the end of the previous block.

2. Not only the tangent, but also the normal vector can change throughout the entire block. This is the case, when circles, splines or polynomials are programmed for the geometry axes, or when not only a start, but also an end value is programmed for the normal vector. In this case, the tool orientation must change accordingly during the interpolation of the block, in order to have the same reference to the path tangent and to the surface normal vector in each path point.
Set orientation relative to the path

The following machine data is used to set in which way the orientation relative to the path is to be interpolated.

MD21094 $MC.ORIPATH_MODE (setting for path relative orientation ORIPATH)

With ORIPATH the behavior of tool orientation interpolation relative to the path can be activated for various functions:

Meaning of units  
activate proper orientation relative to the path

0: The tool orientation only has the reference to the path tangent and to the normal vector programmed with LEAD and TILT at the end of the block, whereas, during the block, the orientation does not follow the path tangent (previous behavior).

1: Reference of the tool orientation to the path tangent and to the surface normal vector programmed with LEAD/TILT is maintained throughout the entire block.

Meaning of decades  
Interpretation of the turning angle TILT

0: LEAD = rotation about the direction perpendicular to the tangent and normal vector (forward angle)

TILT = rotation of orientation around the normal vector

1: LEAD = rotation about the direction perpendicular to the tangent and normal vector (forward angle)

TILT = rotation of orientation around the direction of path tangent (sideways angle)

Meaning of centuries  
Retracting movement with re-orientation

0: There is no retracting movement

There is a retracting movement in the tool coordinate system, i.e. the direction programmed by the retracting vector is interpreted in a coordinate system, which is specified in the following way:

1: Current tool direction (z coordinate) and orientation change (x coordinate)

2: Active plane (z coordinate is normal vector to the active plane) and orientation change (x coordinate)

Smoothing of the orientation jump ORIPATHS

Smoothing of the orientation jump is done within the setting data SD42670 $SC.ORIPATH_SMOOTH_DIST (path distance to smoothing orientation) of the specified path. The programmed reference of the orientation to the path tangent and normal vector is then no longer maintained within this distance. If this distance is set too small, the path velocity may have to be reduced significantly.
A velocity jump of the orientation axes can also be smoothed. In the case where the orientation path does not perform a jump, but whose first derivation is not smooth, the resulting velocity jump can be smoothed. The setting data SD42672 \$SC_ORIPATH_SMOOTH_TOL > 0 (tolerance for smoothing the orientation) is used to specify how much the orientation may deviate from the "tangential" alignment. This orientation smoothing is only performed if G code ORIPATHS ORIPATHS is active and setting data SD42672 SC_ORIPATH_SMOOTH_TOL > 0.

**Insertion of intermediate blocks for the smoothing of the orientation path**

If the following setting data is set, a separate intermediate block is inserted for the smoothing of the orientation path:

SD42670 \$SC_ORIPATH_SMOOTH_DIST = 0.0 (path distance for smoothing the orientation)

This means that the path motion then stops at the corner of the contour and only then the jump in the tool orientation is executed. The orientation change is then only performed with continuous acceleration when ORIPATHS is active. Otherwise the orientation is transferred from the start orientation to the end orientation by means of linear large circle interpolation.

**Execute tool retracting movement**

A tool retracting movement can be performed during this re-orientation. The direction and path length of the retracting movement is programmed via the vector using the components A8=x, B8=y and C8=z. If the length of this vector is exactly zero, no retracting movement is executed.

In which coordinate system the tool retracting vector is interpreted, depends on the value of the following machine data:

MD21094 \$MC_ORIPATH_MODE (setting for path relative orientation ORIPATH)

This specifies in which coordinate system the retracting vector is interpreted.

1. Tool coordinate system: z coordinate defined by current tool direction.
2. Workpiece coordinate system: z coordinate defined by active plane.

Normally the retracting movement is performed simultaneously to the orientation change. A factor can be programmed with the identifier ORIPLF = r , which defines a "safety clearance". In this way, tool orientation only changes when the tool has retracted by r * retraction path. The programmed retraction factor must be in the interval 0 = r ≤ 1, in order to avoid alarm 14126.

**Path relative interpolation of the rotation ORIROTC**

With 6-axis transformations, in addition to the complete interpolation of the tool orientation relative to the path and the rotation of the tool, there is also the option that only the rotation of the tool relative to the path tangent is interpolated. The tool orientation can be programmed and interpolated independently of this. This is activate by the G-code ORIROTC in the 54th G-code group. Tool orientation direction can be programmed as usual with direction vectors, Euler or RPY angle. Their interpolation method can be specified as usual with the G codes ORIVECT, ORIAXES, ORICONxx, ORICURVE, see Section "Rotations of orientation vector (Page 112)".

Special functions

Function Manual, 07/2012, 6FC5397-2BP40-3BA0 95
1.9 Orientation

1.9.6 Programming of orientation polynomials

Functionality

Orientation polynomials and even axis polynomials can be programmed with different types of polynomials regardless of the type of polynomial interpolation currently active. This can be applied to:

- Linear interpolation with G-code G01
- Polynomial interpolation with G-code POLY
- Circular interpolation with G-code G02, G03 or CIP
- Involute interpolation with G-code INVCW or INVCCW

This enables a number of polynomials to be programmed for one contour at the same time.

Note

For further information about programming axis polynomials with PO[X], PO[Y], PO[Z] and orientation polynomials such as PO[PHI], PO[PSI], PO[THT] and PO[XH], PO[YH], PO[ZH], please see:

References:

Programming Manual, Job Planning

Two different types of orientation polynomials are defined:

1. Polynomials for angles with reference to the plane defined by the start and end orientation (type 1 orientation polynomials).
2. Polynomials for curves in space on a reference point on the tool (type 2 orientation polynomials).

Type 1 polynomials

Orientation polynomials of type 1 are polynomials for angles

PO[PHI]: in the plane between start and end orientation
PO[PSI]: describing the tilt of the orientation from the plane between start and end orientation

Type 2 polynomials

Orientation polynomials of type 2 are polynomials for coordinates

PO[XH]: x coordinate of the reference point on the tool
PO[YH]: y coordinate of the reference point on the tool
PO[ZH]: z coordinate of the reference point on the tool
Polynomials for angle of rotation and rotation vectors

For 6-axis transformations, the rotation of the tool around itself can be programmed for tool orientation. This rotation of a third rotary axis is described either by an angle of rotation or by a rotation vector, which is perpendicular to the tool direction in the plane.

In addition, a polynomial for rotation with PO[THT] of the orientation vector can be programmed in these three cases. This is always possible if the kinematic transformation applied, supports rotary angles.

Angle of rotation with ORIPATH and ORIPATHS

For path relative orientation interpolation relative to the path ORIPATH or ORIPATHS, the additional rotation can be programmed with the angle THETA=<...>. Polynomials up to the 5th degree can also be programmed with PO[THT]=(...) for this angle of rotation.

The three possible angles, i.e. lead angle, tilt angle and angle of rotation, have the following meaning with respect to the rotation effect:

- **LEAD**: Angle relative to the surface normal vector in the plane put up by the path tangent and the surface normal vector
- **TILT**: Rotation of orientation in the z direction or rotation about the path tangent
- **THETA**: Rotation around the tool direction. This is only possible when the tool orientation has a total of three degrees of freedom (see Section "Extension of the generic transformation to six axes - 840D sl only (Page 66)").

How the angles LEAD and TILT are to be interpreted, can be set with the following machine data:

MD21094 $MC_ORIPATH_MODE (setting for path relative orientation ORIPATH)

In addition to the constant angles programmed with LEAD and TILT, polynomials can be programmed for lead angle and tilt angle. Polynomials are programmed with the PHI and PSI angles:

- **PO[PHI] = (a2, a3, a4, a5)** Polynomial for the LEAD angle
- **PO[PSI] = (b2, b3, b4, b5)** Polynomial for the TILT angle

Polynomials can be programmed up to the 5th degree for both angles. The angle values at the block end are programmed with the NC addresses LEAD = <...> bzw. TILT = <...>.

The higher polynomial coefficients, which are zero, can be omitted when programming. For example **PO[PHI] = (a2)** programs a parabola for the lead angle LEAD.
Rotations of rotation vectors with ORIROTC

The rotation vector is interpolated relative to the path tangent with an offset that can be programmed using the THETA angle.

A polynomial up to the 5th degree can also be programmed with PO[THT]=(c2, c3, c4, c5) for the offset angle.

**Note**

If ORIAXES is active, i.e. the tool orientation is interpolated via axis interpolation, the orientation of the rotation vector relative to the path is only fulfilled at the end of the block.

For further information about programming, please see:

**References:**

Programming Manual, Production Planning; transformations, interpolation type (ORIPATH, ORIPATHS)

### Supplementary conditions

It is only useful to program orientation polynomials for specific interpolation types, which affect both contour and orientation. A number of supplementary conditions must be met to avoid illegal programming settings:

Orientation polynomials cannot be programmed,

- if ASPLINE, BSPLINE, CSPLINE spline interpolations are active.

  Polynomials for type 1 orientation angles are possible for every type of interpolation except spline interpolation, i.e. linear interpolation with rapid traverse G00 or with feedrate G01 and polynomial POLY and circular/involute interpolation G02, G03, CIP, CT, INVCW and INVCCW.

  In contrast, type 2 orientation polynomials are only possible if linear interpolation with rapid traverse G00 or with feedrate G01 or polynomial interpolation POLY is active.

- if the orientation is interpolated using ORIAXES axis interpolation.

  In this case, polynomials can be programmed directly with PO[A] and PO[B] for orientation axes A and B.

If ORICURVE is active, the Cartesian components of the orientation vector are interpolated and only type 2 orientation polynomials are possible. However, type 1 orientation polynomials are not permitted.

Only type 1 orientation polynomials are possible for large circle interpolation and taper interpolation with ORIVECT, ORIPLANE, ORICONxxx. However, type 2 orientation polynomials are not permitted.

### Alarms

If an illegal polynomial is programmed, the following alarms are generated:

- **Alarm 14136:** Orientation polynomial is generally not allowed.
- **Alarm 14137:** Polynomials PO[PHI] and PO[PSI] are not permitted.
- **Alarm 14138:** Polynomials PO[XH], PO[YH], PO[ZH] are not permitted.
- **Alarm 14139:** Polynomial for angle of rotation PO[THT] is not permitted.
1.9.7 Tool orientation with 3-/4-/5-axis transformations

Tool direction can be read with the following system variables:

- $P\_TOOLO[n]$ Tool orientation active in the **interpreter** cannot be applied in synchronous actions
- $AC\_TOOLO\_ACT[n]$ Active setpoint orientation in the **interpolator**
- $AC\_TOOLO\_END[n]$ End orientation of the current block
- $AC\_TOOLO\_DIFF$ Residual angle of tool orientation in the active block
- $VC\_TOOLO[n]$ Actual value orientation direction
- $VC\_TOOLO\_DIFF$ Angle between actual and setpoint orientation
- $VC\_TOOLO\_STAT$ Status of calculations of actual value orientation

1.9.8 Orientation vectors with 6-axis transformation

With 6-axis transformations, the complete orientation is described by two vectors that are perpendicular to one another.

- The **first** vector points in the direction of the tool (see Section "Tool orientation with 3-/4-/5-axis transformations (Page 99)"), while
- the **second** is in the plane perpendicular to this and describes rotations of the tool around itself.

Both vectors can be read via system variables and also via the OPI interface.

The reading of the direction of rotation vector with the following system variables is only meaningful for a 6-axis transformation.

- $P\_TOOLROT[n]$ Rotation direction vector active in the **Interpreter** not applicable in synchronous actions
- $AC\_TOOLR\_ACT[n]$ Active rotation direction vector in the **interpolator**
- $AC\_TOOLR\_END[n]$ End orientation of the active block
- $AC\_TOOLR\_DIFF$ Residual angle of the direction of rotation vector in the active block in degrees
- $VC\_TOOLR[n]$ Actual value of the rotation direction vector
- $VC\_TOOLR\_DIFF$ Angle between actual value and setpoint of the direction of rotation vector in degrees
- $VC\_TOOLR\_STAT$ Calculation status of the actual value of the direction of rotation vector

**Reference:**
Parameter Manual, System Variables

For further information about the programming of polynomials for axis movements with orientation vectors (see Section "Orientation vectors (Page 108)").
1.10 Orientation axes

Direction

The directions around which axes are rotated are defined by the axes of the reference system. In turn, the reference system is defined by ORIMKS and ORIWKS commands:

- ORIMKS: Reference system = Basic coordinate system
- ORIWKS: Reference system = Workpiece coordinate system

Order of rotation

The order of rotation for the orientation axes is defined by the following machine data:

MD21120 $MC_ORIA_X_TURN_TAB_1[0..2]$ (definition of reference axes for ORI axes)

1. First rotation around the axis of the reference system, defined in the following machine data:
   
   MD21120 $MC_ORIA_X_TURN_TAB_1[0]$

2. Second rotation around the axis of the reference system, defined in the following machine data:
   
   MD21120 $MC_ORIA_X_TURN_TAB_1[1]$

3. Third rotation around the axis of the reference system, defined in the following machine data:
   
   MD21120 $MC_ORIA_X_TURN_TAB_1[2]$

Direction of the tool vector

The direction of the tool vector in the initial machine setting is defined in the following machine data:

MD24580 $MC_TRAFO5_TOOL_VECTOR_1$ (orientation vector direction) or
MD24680 $MC_TRAFO5_TOOL_VECTOR_2$ (orientation vector direction)

Assignment to channel axes

Machine data MD24585 $MC_TRAFO5_ORIA_ASSIGN_TAB_1[0..2]$ (ORI/channel assignment Transformation 1) are used to assign up to a total of 3 virtual orientation axes to the channel, which are set as input variables in machine data

$MC_TRAFO_AXES_IN_n[4..6]$ (axis assignment for Transformation n).
For assigning channel axes to orientation axes, the following applies:

- $MC\_TRAFO5\_ORIAAX\_ASSIGN\_TAB\_n[0] = MC\_TRAFO\_AXES\_IN\_n[4]$
- $MC\_TRAFO5\_ORIAAX\_ASSIGN\_TAB\_n[1] = MC\_TRAFO\_AXES\_IN\_n[5]$
- $MC\_TRAFO5\_ORIAAX\_ASSIGN\_TAB\_n[2] = MC\_TRAFO\_AXES\_IN\_n[6]$

Orientation transformation 1:

- MD24585 $MC\_TRAFO5\_ORIAAX\_ASSIGN\_TAB\_1[n]$ n = channel axis [0..2]

Orientation transformation 2:

- MD24685 $MC\_TRAFO5\_ORIAAX\_ASSIGN\_TAB\_2[n]$ n = channel axis [0..2]
- MD24110 $MC\_TRAFO5\_AXES\_IN\_1[n]$ (axis assignment n = channel axis [0..7] for transformation)
- MD24410 $MC\_TRAFO5\_AXES\_IN\_4[n]$ (axis assignment for transformation 4)
- MD24432 $MC\_TRAFO5\_AXES\_IN\_5[n]$ (axis assignment n = channel axis [0..7] for transformation 5)
- MD24462 MC\_TRAFO5\_AXES\_IN\_8[n] (axis assignment for transformation 8)

Example

For orientation axes, see Section "Example for orientation axes (Page 131)". 
1.10 Orientation axes

1.10.1 JOG mode

It is not possible to traverse orientation axes in JOG mode until the following conditions are fulfilled:

- The orientation axis must be defined as such, that is, a value must be set in the following machine data:
  
  MD24585 $MC_TRAFO5_ORIAX_ASSIGN_TAB (ORI/channel axis assignment Transformation 1)

- A transformation must be active (TRAORI command)

Axis traversal using traverse keys

When using the traverse keys to move an axis continuously (momentary-trigger mode) or incrementally, it must be noted that only one orientation axis can be moved at a time.

If more than one orientation axis is moved, alarm 20062 "Channel 1 axis 2 already active" is generated.

Traversal using handwheels

More than one orientation axis can be moved simultaneously via the handwheels.

Feedrate in JOG

When orientation axes are traversed manually, the channel-specific feedrate override switch or the rapid traverse override switch in rapid traverse override is applied.

Until now, velocities for traversal in JOG mode have always been derived from the machine axis velocities. However, geometry and orientation axes are not always assigned directly to a machine axis.

For this reason, new machine data have been introduced for geometry and orientation axes, allowing separate velocities to be programmed for these axis types:

- MD21150 $MC_JOG_VELO_RAPID_ORI[n]  
  (conventional fast traverse for ORI axes)
- MD21155 $MC_JOG_VELO_ORI[n]  
  (conventional ORI axis speed)
- MD21160 $MC_JOG_VELO_RAPID_GEO[n]  
  (conventional fast traverse for GEO axes)
- MD21165 $MC_JOG_VELO_GEO[n]  
  (conventional GEO axis speed)

Appropriate speed values for the axes must be programmed in these data.

Acceleration

Acceleration for the orientation axes can be set by means of the following machine data:

MD21170 $MC_ACCEL_ORI[n] (acceleration for orientation axes)
1.10 Orientation axes

### Programming for orientation transformation

The values can only be programmed in conjunction with an orientation transformation.

**Programming of orientation**

Orientation axes are programmed by means of axis identifiers A2, B2 and C2. Euler and RPY values are distinguished on the basis of G-group 50:

- **ORIEULER**: Orientation programming on the basis of Euler angles (default)
- **ORIRPY**: Orientation programming via RPY angles
- **ORIVIRT1**: Orientation programming on the basis of virtual orientation axes (definition 1)
- **ORIVIRT2**: Orientation programming on the basis of virtual orientation axes (definition 2)

The type of interpolation is distinguished on the basis of G-group 51:

- **ORIAXES**: Orientation programming of linear interpolation of orientation axes or machine axes
- **ORIVECT**: Orientation programming of large circle interpolation of orientation axes (interpolation of the orientation vector)

Machine data MD21102 $MC_ORI_DEF_WITH_G_CODE (definition of ORI axes via G-code) is used to specify whether MD21100 $MC_ORIENTATION_IS_EULER (angle definition for orientation programming) is active (default) or G-group 50.

The following four variants are available for programming orientation:

1. A, B, C:
   - Machine axis parameter designation
2. A2, B2, C2:
   - Angle programming of virtual axes
3. A3, B3, C3:
   - Vector component designation
4. LEAD, TILT:
   - Specification of lead and side angles with reference to path and surface

References:
Programming Manual Fundamentals
1.10 Orientation axes

Note
The four variants of orientation programming are mutually exclusive. If mixed values are programmed, alarm 14130 or alarm 14131 is generated.

Exception:
For 6-axis kinematics with a 3rd degree of freedom for orientation, C2 may also be programmed for variants 3 and 4. C2 in this case describes the rotation of the orientation vector about its own axis.

Example
For orientation axes for kinematics with 6 or 5 transformed axes, see Section "Example for orientation axes (Page 131)".

Interpolation type
The following machine data is used to specify which interpolation type is used:
MD21104 $MC_ORI_IPO_WITH_G_CODE (G-code for orientation interpolation):
- ORIMKS or ORIWKS (for description, see Section "Tool orientation (Page 37)")
- G-code group 51 with the commands ORIAXES or ORIVECT
  - ORIAXES:
    Linear interpolation of machine axes or orientation axes.
  - ORIVECT:
    Orientation is controlled by the orientation vector being swivelled in the plane spanned by the start and end vectors (large circle interpolation). With 6 transformation axes a rotation around the orientation vector is executed in addition to the swivel movement.
    With ORIVECT the orientation axes are always traversed on the shortest possible path.

Value range
Value range for orientation axes:
- 180 degrees < A2 < 180 degrees
- 90 degrees < B2 < 90 degrees
- 180 degrees < C2 < 180 degrees
All possible rotations can be represented with this value range. Values outside the range are normalized by the control system to within the range specified above.

Feedrate when programming ORIAXES
Feedrate for an orientation axis can be limited via the FL[ ] instruction (feed limit).
1.10.3 Programmable offset for orientation axes

How the programmable offset works

The additional programmable offset for orientation axes acts in addition to the existing offset and is specified when transformation is activated. Once transformation has been activated, it is no longer possible to change this additive offset and no zero offset will be applied to the orientation axes in the event of an orientation transformation.

The programmable offset can be specified in two ways.

1. Direct programming of the offset with TRAORI() when transformation is activated.
2. Automatic transfer of the offset from the zero offset active for the orientation axes when transformation is activated. This automatic transfer is configured via machine data.

Programming offset directly

When transformation is activated, the offset can be programmed directly as TRAORI(n, x, y, z, a, b). The following parameters are available as an option:

- n: Number of transformation n = 1 or 2
- x, y, z: Components of the vector for the basic orientation of the tool (generic 5-axis transformation only).
- a, b: Offset for rotary axes

These optional parameters can be omitted. However, if they are used for programming purposes, the correct sequence must be observed. If for example only one rotary axis offset is to be entered, TRAORI,,,,, a, b) is to be programmed.

For further information about programming, please see:

Reference:
Programming Manual, Production Planning; Transformations
Programming offset automatically

As the offset is transferred automatically from the currently active zero offset on the orientation axes, the effects of zero offset on rotary axes are always the same, both with and without active transformation. Automatic take-over of offset from zero offset is possible via machine data MD24590 $MC_TRAFO5_ROT_OFFSET_FROM_FR_1 = TRUE (Offset of rotary transformation axes from NPV) for the first machine data, or MD24690 $MC_TRAFO5_ROT_OFFSET_FROM_FR_2 = TRUE (Offset of rotary transformation axes from NPV) for the second transformation in the channel.

Note
There is no difference between a zero offset on the orientation axes programmed during active transformation and the previous offset.

If automatic transfer of offset has been activated and a rotary axis offset is programmed at the same time, the programmed offset value takes priority.

Orientable toolholder with additive offset

On an orientable toolholder, the offset for both rotary axes can be programmed with the system variables $TC_CARR24 and $TC_CARR25. This rotary axis offset can be transferred automatically from the zero offset effective at the time the orientable toolholder was activated.

Automatic transfer of offset from zero offset is made possible via the following machine data: MD21186 $MC_TOCARR_ROT_OFFSET_FROM_FR = TRUE (offset of TOCARR rotary axes from NPV)

Note
For more information about orientable toolholders, please see:

Reference:
Function Manual, Basic Machine; Tool Offset (W1)
1.10.4 Orientation transformation and orientable tool holders

Note
Orientation transformation and orientable tool holders can be combined. The resulting orientation of the tool is produced by linking the orientation transformation and the orientable tool holder.

1.10.5 Modulo display of orientation axes

Function
The positions of orientation axes can be displayed for the BCS and WCS display in a settable modulo area. Whether the concerned machine axes are linear or rotary is not relevant in this context, i.e. this display option can be enabled even for normal generic 5-/6-axis transformation.

Requirements
- Orientation axes must be available. This is the case if an orientation transformation is active (e.g. generic 5-/6-axis transformation).
- The following machine data must also be set for OEM transformations:
  MD24585 $MC_TRAFO5_ORIAX_ASSIGN_TAB_1[0..2]

Parameterization
The modulo display of orientation axes is activated as follows:
MD21132 $MC_ORI_DISP_IS_MODULO[0...2] = TRUE
The modulo range is defined with the help of the following machine data:
- MD21134 $MC_ORI_MODULO_RANGE[0...2]
  (Size of the modulo range for the display of the orientation axes)
- MD21136 $MC_ORI_MODULO_RANGE_START[0...2]
  (Starting position of the modulo range for the display of the orientation axes)

Please note the following:
- The machine data becomes effective with NewConfig.
- The machine data does not have any influence or effects on:
  - any axis positions that can be programmed for these axes.
  - The traversing movements of these axes.
  - The display of the MCS values of these axes
1.11 Orientation vectors

1.11.1 Polynomial interpolation of orientation vectors

Programming of polynomials for axis motions

The rotary axes are normally subjected to linear interpolation in case of orientation changes with the help of rotary axis interpolation. However, it is also possible to program the polynomials as usual for the rotary axes. This allows a generally more homogeneous axis motion to be produced.

Note
Further information about programming polynomial interpolation with POLY and on interpolation of orientation vectors is given in:

Reference:
Programming Manual; Production Planning

A block with POLY is used to program polynomial interpolation. Whether the programmed polynomials are then interpolated as polynomial, depends on whether the G-code POLY is active or not.

- The G-code is not active: The programmed axis end points are traversed linearly.
- The G-code is active: The programmed polynomials are interpolated as polynomials.

MD10674

If machine data MD10674 $MN_PO_WITHOUT_POLY = FALSE (polynomial programming without G-function POLY programmable) it can be specified, whether the following programming is possible:

- PO[... ] or PO( ... ) is possible only if POLY is active, or
- PO[ ] or PO( ) polynomials are also possible without active G-code POLY.

As default, MD10674: PO_WITHOUT_POLY = FALSE set and with MD10674 $MN_PO_WITHOUT_POLY = TRUE the following programming is always possible:

- PO[... ] = (...), regardless of whether POLY is active or not.

Orientation polynomials can be programmed in conjunction with different interpolation types and are described in Section "Programming of Orientation Polynomials".
POLYPATH:

In addition to the modal G function POLY, the predefined subprogram POLYPATH(argument) can be used to activate polynomial interpolation selectively for different axis groups. The following arguments are allowed for the activation of polynomial interpolation:

- "AXES": For all path axes and supplementary axes
- "VECT": For orientation axes
- "AXES", "VECT": For path axes, supplementary axes and orientation axes
- (without argument): deactivates polynomial interpolation for all axis groups

Normally, the polynomial interpolation is activated for all axis groups.

Programming of orientation vectors

An orientation vector can be programmed in each block. If polynomials are programmed for the orientation, the interpolated orientation vector is generally turned not in the plane between start and end vector, but it can be rotated at random from this plane.

Orientation vectors can be programmed as follows:

1. Programming of rotary axis positions with A, B and C or with the actual rotary axis identifiers.
2. Programming in Euler angle or RPY angle via A2, B2, C2.
3. Programming of the direction vector via A3, B3, C3.
4. Programming using lead angle LED and tilt angle TILT

Selection of type of interpolation

The type of interpolation of orientation axes is selected with G-code of group 51 and is independent of the programming type of the end vector:

- ORIAXES: Linear interpolation of the machine axes or using polynomials for active POLY or
- ORIVECT: Interpolation of the orientation vector using large circle interpolation

If ORIAXES is active, the interpolation of the rotary axis can also take place using polynomials like polynomial interpolation of axes with POLY.

On the other hand, if ORIVECT is active, "normal" large circle interpolation is carried out through linear interpolation of the angle of the orientation vector in the plane that is defined by the start and end vector.
Polynomials for 2 angles

Additional programming of polynomials for 2 angles that span the start vector and end vector can also be programmed as complex changes in orientation with ORIVECT.

The two PHI and PSI angles are specified in degrees.

POLY
Activation of polynomial interpolation for all axis groups.

POLYPATH ( )
Activation of polynomial interpolation for all axis groups. "AXES" and "VECT" are possible groups.

The coefficients \(a_n\) and \(b_n\) are specified in degrees.

\[
\text{PO}[\text{PHI}]= (a_2, a_3, a_4, a_5) \\
\text{PO}[\text{PSI}]= (b_2, b_3, b_4, b_5)
\]

The angle PHI is interpolated according to \(\text{PHI}(u) = a_0 + a_1u + a_2u^2 + a_3u^3 + a_4u^4 + a_5u^5\).

The angle PSI is interpolated according to \(\text{PSI}(u) = b_0 + b_1u + b_2u^2 + b_3u^3 + b_4u^4 + b_5u^5\).

PL
Length of the parameter interval where polynomials are defined. The interval always starts at 0.

Theoretical value range for PL: 0.0001 ... 99999.9999

The PL value applies to the block that contains it. PL=1 is applied if no PL value is programmed.

Rotation of the orientation vector

Changes in orientation are possible with ORIVECT independent of the type of end vector programming. The following situations apply:

**Example 1:** Components of end vectors are programmed directly.

N... POLY A3=a B3=b C3=c PO[PHI] = (a2, a3, a4, a5) PO[PSI] = (b2, b3, b4, b5)

**Example 2:** The end vector is determined by the position of the rotary axes.

N... POLY Aa Bb Cc PO[PHI] = (a2, a3, a4, a5) PO[PSI] = (b2, b3, b4, b5)

The angle PHI describes the rotation of the orientation vector in the plane between start and end vectors (large circle interpolation, see figure below). The orientation is interpolated exactly as in Example 1.

![Rotation of the orientation vector in the plane between start and end vector](image-url)
PHI and PSI angle

Programming of polynomials for the two angles PO[PHI] and PO[PSI] is always possible. Whether the programmed polynomials are actually interpolated for PHI and PSI depends on:

- **POLYPATH("VECT")** and **ORIVECT** are **active**, then the polynomials will be interpolated.
- **If POLYPATH("VECT")** and **ORIVECT** are **not active**, the programmed orientation vectors are traversed at the end of the block by a "normal" large circle interpolation. This means that the polynomials for the two angles PHI and PSI are ignored in this case.

![Diagram](image)

**Figure 1-24** Movement of the orientation vector in plan view

The angle PSI can be used to generate movements of the orientation vector perpendicular to large circle interpolation plane (see previous figure)

**Maximum polynomials of the 5th degree permitted**

5th degree polynomials are the maximum possible for programming the PHI and PSI angles. Here the constants and linear coefficient are defined by the initial value or end value of the orientation vector.

Higher degree coefficients can be omitted from the coefficient list (...,...) if these are all equal to zero.

The length of the parameter interval in which the polynomials are defined can also be programmed with PL.
Points to note

If no polynomial for angle PSI is programmed, the orientation vector is always interpolated in the plane defined by the start and end vector.

The PHI angle in this plane is interpolated according to the programmed polynomial for PHI. As a result the orientation vector moves through a "normal" large circle interpolation in the plane between the start and end vector and the movement is more or less irregular depending on the programmed polynomial.

In this way, the velocity and acceleration curve of the orientation axes can be influenced within a block, for example.

Note

Further information on polynomial interpolation for axis motion and general programming is given in:

Reference:
Programming Manual; Production Planning

Supplementary conditions

Polynomial interpolation of orientation vectors is only possible for control variants in which the following functions are included in the functional scope:

- Orientation transformation
- Polynomial interpolation

1.11.2 Rotations of orientation vector

Functionality

Changes in tool orientation are programmed by specifying an orientation vector in each block, which is to be reached at the end of the block. The end orientation of each block can be programmed in the following way:

1. programming the vector directly, or
2. programming the rotary axis positions.

The second option depends on machine kinematics. Interpolation of the orientation vector between the start and end values can also be modified by programming polynomials.
Programming of orientation directions

The following options are available for programming tool orientation:

1. Direct programming of rotary axis positions (the orientation vector is derived from machine kinematics).
2. Programming in Euler angles via A2, B2, C2 (angle C2 is irrelevant).
3. Programming in RPY angles via A2, B2, C2.
4. Programming the direction vector via A3, B3, C3 (the length of the vector is irrelevant).

Switching between Euler and RPY angle programming can be selected via the following machine data or via the G-codes {ORIEULER and ORIRPY}:

MD21100 $MC_ORIENTATION_IS_EULER (angle definition for orientation programming)

Programming of orientation direction and rotation

While the direction of rotation is already defined when you program the orientation with RPY angles, additional parameters are needed to specify the direction of rotation for the other orientations:

1. Direct programming of the rotary axes
   A supplementary rotary axis for direction of rotation has to be defined.
2. Programming in Euler angles via A2, B2, C2
   Angle C2 must be programmed additionally. The complete orientation is thus defined, including tool rotation.
3. Programming in RPY angles via A2, B2, C2
   Additional settings are not required.
4. Programming of the direction vector via A3, B3, C3
   The rotation angle is programmed with THETA=<value>.

Note

The following cases do not allow for a programmed rotation:

Multiple programming of the direction of rotation is not allowed and results in an alarm. If the Euler angle C2 and the angle of rotation THETA are programmed simultaneously, the programmed rotation is not executed.

If machine kinematics are such that the tool cannot be rotated, any programmed rotation is ignored. This is the case with a normal 5-axis machine, for example.
Rotation of the orientation vector

The following options are available for interpolating rotation of the orientation vector by programming the vector directly:

- Linear interpolation, i.e. the angle between the current rotation vector and the start vector is a linear function of the path parameter.
- Non-linear due to additional programming of a polynomial for the angle of rotation $\theta$ of 5th degree maximum, in the format:
  $$\text{PO}[\text{THT}] = (d_2, d_3, d_4, d_5)$$

Interpolation of the angle of rotation

Higher degree coefficients can be omitted from the coefficient list (...) if these are all equal to zero.

In such cases, the end value of the angle and the constant and linear coefficient $d_n$ of the polynomial cannot be programmed directly.

The linear coefficient $d_n$ is defined by means of the end angle $\theta_e$ in degrees.

The start angle $\theta_s$ is derived from the start value of the rotation vector, resulting from the end value of the previous block. The constant coefficient of the polynomial is defined by the starting angle of the polynomial.

The rotation vector is always perpendicular to the current tool orientation and forms the angle $\Theta$ in conjunction with the basic rotation vector.

Note

During machine configuration, the direction in which the rotation vector points at a specific angle of rotation can be defined, when the tool is in the basic orientation.

Formula

In general, the angle of rotation is interpolated with a 5th degree polynomial:

$$\theta_u = \theta_s + d_1 u + d_2 u^2 + d_3 u^3 + d_4 u^4 + d_5 u^5$$  \hspace{1cm} (14)

For the parameter interval $0 \ldots 1$, this produces the following values for linear coefficients:

$$d_1 = \theta_e - \theta_s - d_2 - d_3 - d_4 - d_5$$  \hspace{1cm} (15)
Interpolation of the rotation vector

The programmed rotation vector can be interpolated in the following way, using modal G-codes:

- **ORIROTA** (orientation rotation absolute):
  The angle of rotation \( \theta \) is interpreted with reference to an absolute direction in space. The basic direction of rotation is defined by machine data.

- **ORIROTR** (orientation rotation relative):
  The angle of rotation \( \theta \) is interpreted relative to the plane defined by the start and end orientation.

- **ORIROTT** (orientation rotation tangential):
  The angle of rotation \( \theta \) is interpreted relative to the change in orientation. That means the rotation vector interpolation is tangential to the change in orientation for \( \theta = 0 \).

  This is different to **ORIROTR**, only if the change in orientation does not take place in one plane. This is the case if at least one polynomial was programmed for the "tilt angle" \( \psi \) for the orientation. An additional angle of rotation \( \theta \) can then be used to interpolate the rotation vector such that it always produces a specific angle referred to the change in orientation.

Activation of rotation

A rotation of the orientation vector is programmed with the identifier \( \theta \). The following options are available for programming:

- \( \theta = <\text{value}> \) Programming of an angle of rotation at the end of the block.
- \( \theta = \varphi \) Programmed angles \( \varphi \) can be interpreted as absolute (G90 is active) or as relative (G91 is active incremental).
- \( \theta = \text{AC}(...) \) Switch over to absolute dimensions per block
- \( \theta = \text{IC}(...) \) Switch over to incremental dimensions per block
- \( \text{PO}[	heta] = (...) \) Programming of a polynomial for rotation angle \( \theta \).

The angle \( \theta \) is programmed in degrees.

Interpolation of the rotation vector is defined by modal G-codes:

- **ORIROTA** Angle of rotation to an absolute direction of rotation
- **ORIROTR** Angle of rotation relative to the plane between the start and end orientation
- **ORIROTT** Angle of rotation relative to the change of the tangential rotationvector of the orientation vector to the orientation change
- **ORIROTC** Angle of rotation relative to the change of the tangential rotationvector of the orientation vector to the path tangent
- **PL** Length of the parameter interval where polynomials are defined. The interval always starts at 0. If no PL has been programmed, PL = 1 will be applied.

These G-codes define the reference direction of the angle of rotation. The meaning of the programmed angle of rotation changes accordingly.
Boundary conditions

The angle of rotation or rotation vector can only be programmed in all four modes if the interpolation type \texttt{ORIROTA} is active.

1. Rotary axis positions
2. Euler angles via \(A_2, B_2, C_2\)
3. RPY angles via \(A_2, B_2, C_2\)
4. Direction vector via \(A_3, B_3, C_3\)

If \texttt{ORIROTR} or \texttt{ORIROTT} is active, the angle of rotation can only be programmed directly with \texttt{THETA}.

The other programming options must be excluded in this case, since the definition of an absolute direction of rotation conflicts with the interpretation of the angle of rotation in these cases. Possible programming combinations are monitored and an alarm is output if applicable.

A rotation can also be programmed in a separate block without an orientation change taking place. In this case \texttt{ORIROTR} and \texttt{ORIROTT} are irrelevant. In this case the angle of rotation is always interpreted with reference to the absolute direction (\texttt{ORIROTA}).

A programmable rotation of the orientation vector is only possible when an orientation transformation (\texttt{TRAORI}) is active.

A programmed orientation rotation is only interpolated if the machine kinematics allow rotation of the tool orientation (e.g. 6-axis machines).

1.11.3 Extended interpolation of orientation axes

Functionality

To execute a change in orientation along the peripheral surface of a cone located in space, it is necessary to perform an extended interpolation of the orientation vector. The vector around which the tool orientation is to be rotated must be known. The start and end orientation must also be specified. The start orientation is given by the previous block and the end orientation must either be programmed or defined by other conditions.
Required definitions

Generally, the following data is required:

- The start orientation is defined by the end orientation of the previous block.
- The end orientation is defined either by specifying the vector (with A3, B3, C3), the Euler angles or RPY angles (with A2, B2, C2) or by programming the positions of the rotary axis (with A, B, C).
- The rotary axis of the taper is programmed as a (normalized) vector with A6, B6, C6.
- The opening angle of the cone is programmed degrees with the identifier (nutation angle).

The value range of this angle is limited to the interval between 0 degrees and 180 degrees. The values 0 degrees and 180 degrees must not be programmed. If an angle is programmed outside the valid interval, an alarm is generated.

In the special case where NUT = 90 degrees, the orientation vector in the plane is interpolated perpendicular to the direction vector (large circle interpolation).

The sign of the programmed opening angle specifies whether the traversing angle is to be greater or less than 180 degrees.

In order to define the cone, the direction vector or its opening angle must be programmed. Both may not be specified at the same time.

- A further option is to program an intermediate orientation that lies between the start and end orientation.

Programming

ORIPLANE orientation interpolation in a plane: Interpolation in a plane (large circle interpolation)
ORICONCW orientation interpolation on a cone clockwise: Interpolation on the peripheral surface of a cone in the clockwise direction
ORICONCCW orientation interpolation on a cone counter clockwise: Interpolation on the peripheral surface of a cone in the counter-clockwise direction.

Programming of the direction vector is carried out using the identifiers A6, B6, C6 and is specified as a (normalized) vector.

Note

Programming of an end orientation is not absolutely necessary. If no end orientation is specified, a full outside cone with 360 degrees is interpolated.
The opening angle of the taper is programmed with NUT= <angle>, where the angle is specified in degrees.

**Note**

An end orientation must be specified. A complete outside cone with 360 degrees cannot be interpolated in this way. The sign of the opening angle defines whether the traversing angle is to be greater or less than 180 degrees.

The identifiers have the following meanings:

- **NUT = +...** Traverse angle smaller than or equal to 180 degrees
- **NUT = -...** Traverse angle greater than or equal to 180 degrees

A positive sign can be omitted when programming.

**Settings for intermediate orientation**

**ORICONIO**

orientation interpolation on a cone with intermediate orientation: Interpolation on a conical peripheral surface with intermediate orientation setting

If this G-code is active, it is necessary to specify an intermediate orientation with A7, B7, C7 which is specified as a (normalized) vector.

**Note**

Programming of the end orientation is absolutely necessary in this case.

The change in orientation and the direction of rotation is defined uniquely by the three vectors Start, End and Intermediate orientation.

All three vectors must be different from each other. If the programmed intermediate orientation is parallel to the start or end orientation, a linear large circle interpolation of the orientation is executed in the plane that is defined by the start and end vector.

**Angle of rotation and opening angle**

The following may be programmed additionally for the angle of the cone:

- **PHI** angle of rotation for orientation about the direction axis
- **PSI** Opening angle of the cone

Besides this polynomials of the 5th degree (max.) can be programmed as follows:

PO[PHI] = (a2, a3, a4, a5) Constant and linear coefficients are defined by start and end orientation respectively.

PO[PSI] = (b2, b3, b4, b5)
Further interpolation options

It is possible to interpolate the orientation on a cone which connects tangentially to the previous change of orientation. This orientation interpolation is achieved by programming the G-codes ORICONTO.

ORICONTO
orientation interpolation on a cone with tangential orientation: Interpolation on a peripheral surface of the cone with tangential transition

A further option for orientation interpolation is to describe the change in orientation through the path of a 2nd contact point on the tool.

ORICURVE
orientation interpolation with a second curve: Interpolation of orientation with specification of motion of two contact points of the tool.

The coordinates for movement of the 2nd contact point of the tool must be specified. This additional curve in space is programmed with XH, YH, ZH.

Besides the two end values, additional polynomials can be programmed in the following form:

PO[XH] = (xe, x2, x3, x4, x5): (xe, ye, ze) the end point of the curve, and
PO[YH] = (ye, y2, y3, y4, y5): xi, yi, zi the coefficients of the polynomials
PO[ZH] = (ze, z2, z3, z4, z5): of the 5th degree maximum.

This type of interpolation can be used to program points (G1) or polynomials (POLY) for the two curves in space.

Note

Circles or involutes are specifically not allowed. It is also possible to activate a spindle interpolation with BSPLINE. The programmed end points of both curves in space are then interpreted as nodes.

Other types of splines (ASPLINE and CSPLINE) and the activation of a compressor (COMPON, COMPCURV, COMPCAD) are not permitted here.

Supplementary conditions

The extended interpolation of orientations requires that all necessary orientation transformations be considered, since these belong to the functional scope.
Activation

A change in orientation on any peripheral surface of a cone in space is activated with the G-code of group 51 through extended interpolation of the orientation vector, using the following commands:

- **ORIPLANE**: Interpolation in a plane with specification of the end orientation (same as ORIVECT)
- **ORICONCW**: Interpolation on a peripheral surface of a taper in clockwise direction with specification of the end orientation and taper direction or opening angle of the cone.
- **ORICONCCW**: Interpolation on the peripheral surface of a cone in the counterclockwise direction. Specification of the end orientation and cone direction or opening angle of the taper.
- **ORICONIO**: Interpolation on a peripheral surface of a cone with specification of end orientation and an intermediate orientation.
- **ORICURVE**: Interpolation of orientation with specification of motion of two contact points of the tool.
- **ORIPATH**: Tool orientation in relation to the path.
- **ORIPATHS**: Tool orientation in relation to the path, when, for example, a kink in the orientation path, e.g. at a corner in the contour, is to be smoothed, see Section "Orientation relative to the path (Page 92)".

Examples

Various changes in orientation are programmed in the following program example:

| Program code | Comment |
|--------------|---------|
| N10 G1 X0 Y0 F5000 | ; Orientation transformation active. |
| N20 TRAORI | ; Interpolate tool orientation as a vector |
| N30 ORIVECT | ; Select large circle interpolation |
| N40 ORIPLANE | ; Orientation in the Y/Z plane by 45 degrees rotated, at the block end the orientation (0, 1 / (√2)), 1 / (√2) is reached. |
| N50 A3=0 B3=0 C3=1 | ; Outside of the taper with the direction |
| N60 A3=0 B3=1 C3=1 | ; The orientation vector is on a |
| N70 ORICONCW | ; The rotation angle is 270 degrees. |
| N80 A6=0 B6=0 C6=1 A3=1 B3=0 C3=1; (0, 0, 1) to the orientation | ; (1 / √2), 0, 1 / (√2) |
| | ; Interpolated clockwise, |
| | ; The rotation angle is 270 degrees. |
### 1.11 Orientation vectors

| Program code | Comment |
|--------------|---------|
| N90 A6=0 B6=0 C6=1 | ; The tool orientation traverses a complete rotation on the same outside surface of the cone |

...
1.12  **Online tool length offset**

**Functionality**

Effective tool length can be changed in real time so that the length changes are also considered for changes in orientation of the tool. The system variable $AA_TOFF[<geometry axis identifier>] includes tool length compensations in 3-D according to the three tool directions.

None of the tool parameters are changed. The actual compensation is performed internally by means of transformations using an orientable tool length compensation.

The number of active compensation directions must be the same as the number of active geometry axes. All offsets can be active at the same time.

**Application**

The online tool length compensation function can be used for:

- Orientation transformations (TRAORI)
- Orientable tool carriers (TCARR)

**Note**

The online tool length offset is an option. This function is only practical in conjunction with an active orientation transformation or an active orientable toolholder.

**Reference:**

Function Manual, Basic Functions; Tool Offset, Section: Orientable toolholders (W1)

**Block preparation**

In the case of block preparation in run-in, the tool length offset currently active in the main run is considered. In order to utilize the maximum permissible axis velocities as far as possible, it is necessary to halt the block preparation with a stop preprocessing command (STOPRE) while a tool offset is being generated.

The tool offset is always known at the time of run-in when the tool length offsets are not changed after program start or if more blocks have been processed after changing the tool length offsets than the IPO buffer can accommodate between run-in and main run. This ensures that correct axis velocities are applied quickly.
The dimension for the difference between the currently active compensation in the interpolator and the compensation that was active at the time of block preparation can be polled in the system variable $AA_TOFF_PREP_DIFF[].

Note
Changing the effective tool length using online tool length offset produces changes in the compensatory movements of the axes involved in the transformation in the event of changes in orientation. The resulting velocities can be higher or lower depending on machine kinematics and the current axis position.

**MD21190 $MC_TOFF_MODE (operation of tool offset)**

The following machine data can be used to set whether the content of the synchronization variable $AA_TOFF[] is to be approached as an absolute value or whether an integrating behavior is to take place.

**MD21190 $MC_TOFF_MODE**

The integrating behavior of $AA_TOFF[] allows 3D remote control. The integrated value is available via the system variable $AA_TOFF_VAL[].

The following machine data and setting data are available for configuring online tool length compensation:

| Machine data / setting data | Meaning for online tool length offset |
|-----------------------------|--------------------------------------|
| MD21190 $MC_TOFF_MODE      | The contents of $AA_TOFF[] are traversed as an absolute value or integrated |
| MD21194 $MC_TOFF_VELO      | Speed of online tool length offset |
| MD21194 $MC_TOFF_ACCEL     | Acceleration of online tool length offset |
| SD42970 $SC_TOFF_LIMIT     | Upper limit of tool length offset value $AA_TOFF |

With the acceleration margin, 20% is reserved for the overlaid movement of online tool length offset, which can be changed via the following machine data:

**MD20610 $MC_ADD_MOVE_ACCEL_RESERVE (acceleration margin for overlaid movements)**

**Activation**

The TOFFON instruction can be used to activate online tool length offset from the part program for at least one tool direction, if the option is available. During activation an offset value can be specified for the relevant direction of compensation and this is immediately traversed.

Example: TOFFON(Z, 25).
Repeated programming of the instruction TOFFON( ) with an offset causes the new offset to be applied. The offset value is added to variables $AA_TOFF[ ]$ as an absolute value.

**Note**
For further information about programming plus programming examples, please see:

**Reference:**
Programming Manual, Job Planning; Transformations

As long as online tool length offset is active, the VDI signal on the NCK → PLC interface in the following interface signal is set to 1:

DB21, ... DBX318.2 (TOFF active)

While a correction movement is active, the VDI → signal in the following interface signal is set to 1:

DB21, ... DBX318.3 (TOFF movement active)

**Reset**
Compensation values can be reset with the TOFFOF( ) command. This instruction triggers a preprocessing stop.

Accumulated tool length compensations are cleared and incorporated in the basic coordinate system. The run-in is synchronized with the current position in main run. Since no axes can be traversed here, the values of $AA_IM[ ]$ do not change. Only the values of the variables $AA_IW[ ]$ and $AA_IB[ ]$ are changed. These variables now contain the deselected share of tool length compensation.

Once "Online tool length offset" has been deselected for a tool direction, the value of system variable $AA_TOFF[ ]$ or $AA_TOFF_VAL[ ]$ is zero for this tool direction. The following interface signal is set to 0:

DB21, ... DBX318.2 (TOFF active)

**Alarm 21670**
An existing tool length offset must be deleted via TOFFOF( ) so that alarm 21670 "Channel %1 block %2, illegal change of tool direction active due to $AA_TOFF active" is suppressed:

- When the transformation is deactivated with TRAFOOF
- On switch-over from CP to PTP travel.
- If a tool length offset exists in the direction of the geometry axis during geometry replacement.
- If a tool length offset is present during change of plane.
- When changing from axis-specific manual travel in JOG mode to PTP as long as a tool length compensation is active. There is no switchover to PTP.
**Mode change**

Tool length compensation remains active even if the mode is changed and can be executed in any mode.

If a tool length compensation is interpolated on account of $AA_{TOFF}[ ]$ during mode change, the mode change cannot take place until the interpolation of the tool length compensation has been completed. Alarm 16907 "Channel %1 action %2 ALNX possible only in stop state" is issued.

**Behavior with REF and block search**

Tool length offset is not considered during reference point approach REF in JOG mode.

The instructions TOFFON( ) and TOFFOF( ) are not collected and output in an action block during block search.

**System variable**

In the case of online tool length offset, the following system variables are available to the user:

| System variable         | Meaning for online tool length offset                                                                 |
|-------------------------|-------------------------------------------------------------------------------------------------------|
| $AA_{TOFF}[ ]$          | Position offset in the tool coordinate system                                                        |
| $AA_{TOFF\_VAL}[ ]$     | Integrated position offset in the WCS                                                              |
| $AA_{TOFF\_LIMIT}[ ]$   | Query whether the tool length offset value is close to the limit                                     |
| $AA_{TOFF\_PREP\_DIFF}[ ]$ | Magnitude of the difference between the currently active value of $AA_{TOFF}[ ]$ and the value prepared as the current motion block. |

**Reference:**
Parameter Manual, System Variables

**Supplementary conditions**

The online tool length offset function is an option and is available during "generic 5-axis transformation" by default and for "orientable toolholders".

If the tool is not perpendicular to the workpiece surface during machining or the contour contains curvatures whose radius is smaller than the compensation dimension, deviations compared to the actual offset surface are produced. It is not possible to produce exact offset surfaces with one tool length compensation alone.
1.13 Examples

1.13.1 Example of a 5-axis transformation

CHANDATA(1)

$MA\_IS\_ROT\_AX[AX5] = TRUE
$MA\_SPIND\_ASSIGN\_TO\_MACHAX[AX5] = 0
$MA\_ROT\_IS\_MODULE[AX5]=0

;-----------------------------------------------------------------------------------------------------
; general 5-axis transformation
;
; kinematics: 1. rotary axis is parallel to Z
; 2. rotary axis is parallel to X
; Movable tool
;-----------------------------------------------------------------------------------------------------

$MC\_TRAFO\_TYPE\_1 = 20

$MC\_ORIENTATION\_IS\_EULER = TRUE

$MC\_TRAFO\_AXES\_IN\_1[0] = 1
$MC\_TRAFO\_AXES\_IN\_1[1] = 2
$MC\_TRAFO\_AXES\_IN\_1[2] = 3
$MC\_TRAFO\_AXES\_IN\_1[3] = 4
$MC\_TRAFO\_AXES\_IN\_1[4] = 5

$MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_1[0]=1
$MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_1[1]=2
$MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_1[2]=3

$MC\_TRAFO5\_PART\_OFFSET\_1[0] = 0
$MC\_TRAFO5\_PART\_OFFSET\_1[1] = 0
$MC\_TRAFO5\_PART\_OFFSET\_1[2] = 0
$MC\_TRAFO5\_ROT\_AX\_OFFSET\_1[0] = 0
$MC\_TRAFO5\_ROT\_AX\_OFFSET\_1[1] = 0
$MC\_TRAFO5\_ROT\_SIGN\_IS\_PLUS\_1[0] = TRUE

$MC\_TRAFO5\_ROT\_SIGN\_IS\_PLUS\_1[1] = TRUE
$MC\_TRAFO5\_NON\_POLE\_LIMIT\_1 = 2.0

$MC\_TRAFO5\_POLE\_LIMIT\_1 = 2.0
$MC\_TRAFO5\_BASE\_TOOL\_1[0] = 0.0
$MC\_TRAFO5\_BASE\_TOOL\_1[1] = 0.0
$MC\_TRAFO5\_BASE\_TOOL\_1[2] = 5.0

$MC\_TRAFO5\_JOINT\_OFFSET\_1[0] = 0.0
$MC\_TRAFO5\_JOINT\_OFFSET\_1[1] = 0.0
$MC\_TRAFO5\_JOINT\_OFFSET\_1[2] = 0.0

CHANDATA(1)

M17

Program example for general 5-axis transformation:

| Program code | Comment |
|--------------|---------|
| ; Definition of tool T1 | |
| $TC\_DP1[1,1] = 10 | ; Type |
| $TC\_DP2[1,1] = 0 | |
| $TC\_DP3[1,1] = 0 | ; z length compensation vector G17 |
| $TC\_DP4[1,1] = 0. | ; y |
| $TC\_DP5[1,1] = 0. | ; x |
| $TC\_DP6[1,1] = 0. | ; Radius |
| $TC\_DP7[1,1] = 0 | |
| $TC\_DP8[1,1] = 0 | |
| $TC\_DP9[1,1] = 0 | |
| $TC\_DP10[1,1] = 0 | |
| $TC\_DP11[1,1] = 0 | |
| $TC\_DP12[1,1] = 0 | |

Approach initial position:

N100 G1 x1 y0 z0 a0 b0 F20000 G90 G64 T1 D1 G17 ADIS=.5 ADISPOS=3
F2: Multi-axis transformations

1.13 Examples

Orientation vector programming:

N110 TRAORI(1)
N120 ORIWKS
N130 G1 G90
N140 a3 = 0 b3 = 0 c3 = 1 x0
N150 a3 = 0 b3 =-1 c3 = 0
N160 a3 = 1 b3 = 0 c3 = 0
N170 a3 = 1 b3 = 0 c3 = 1
N180 a3 = 0 b3 = 1 c3 = 0
N190 a3 = 0 b3 = 0 c3 = 1

Euler angles program:

N200 ORIMKS
N210 G1 G90
N220 a2 = 0 b2 = 0 x0
N230 a2 = 0 b2 = 90
N240 a2 = 90 b2 = 90
N250 a2 = 90 b2 = 45
N260 a2 = 0 b2 =-90
N270 a2 = 0 b2 = 0

Axis programming:

N300 a0 b0 x0
N310 a45
N320 b30

TOFRAME:

N400 G0 a90 b90 x0 G90
N410 TOFRAME
N420 z5
N430 x3 y5
N440 G0 a0 b0 x1 y0 z0 G90

N500 TRAFOOF
m30
1.13.2 Example of a 3-axis and 4-axis transformation

1.13.2.1 Example of a 3-axis transformation

Example: For the schematically represented machine (see "Figure 1-1 Schematic diagram of 3-axis transformation (Page 25)"), the 3-axis transformation can be projected as follows:

| Program code | Comment |
|--------------|---------|
| $MC\_TRAFO\_TYPE\_n = 18$ | |
| $MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_n[0] = 1$ ; Assignment of channel axes to geometry axes | |
| $MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_n[1] = 0$ | |
| $MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_n[2] = 3$ | |
| $MC\_TRAFO\_AXES\_IN\_n[0] = 1$ ; X axis is channel axis 1 | |
| $MC\_TRAFO\_AXES\_IN\_n[1] = 0$ ; Y axis is not used | |
| $MC\_TRAFO\_AXES\_IN\_n[2] = 3$ ; Z axis is channel axis 3 | |
| $MC\_TRAFO\_AXES\_IN\_n[4] = 0$ ; There is no second rotary axis | |

1.13.2.2 Example of a 4-axis transformation

Example: For the schematically represented machine (see "Figure 1-2 Schematic diagram of a 4-axis transformation with moveable workpiece (Page 25)"), however, with an additional axis (Y), the 4-axis transformation can be configured as follows:

| Program code | Comment |
|--------------|---------|
| $MC\_TRAFO\_TYPE\_n = 18$ | |
| $MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_n[0] = 1$ | |
| $MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_n[1] = 2$ | |
| $MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_n[2] = 3$ | |
| $MC\_TRAFO\_AXES\_IN\_n[0] = 1$ ; X axis is channel axis 1 | |
| $MC\_TRAFO\_AXES\_IN\_n[1] = 2$ ; Y axis is channel axis 2 | |
| $MC\_TRAFO\_AXES\_IN\_n[2] = 3$ ; Z axis is channel axis 3 | |
| $MC\_TRAFO\_AXES\_IN\_n[4] = 0$ ; There is no second rotary axis | |
### Example of a universal milling head

#### General

The following two subsections show the main steps which need to be taken in order to activate a transformation for the universal milling head.

#### Machine data

; machine kinematics CA' with tool orientation in zero position in the z direction

$MC\_TRAFO\_TYPE\_1 = 148$

$MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_1[0] = 1$

$MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_1[1] = 2$

$MC\_TRAFO\_GEOAX\_ASSIGN\_TAB\_1[2] = 3$

; angle of second rotary axis

$MC\_TRAFO5\_NUTATOR\_AX\_ANGLE\_1 = 45$

#### Program

| Program code | Comment |
|--------------|---------|
| ; Definition of tool T1 | |
| $TC\_DP1[1,1] = 120$ | ; Type |
| $TC\_DP2[1,1] = 0$ | ; |
| $TC\_DP3[1,1] = 20$ | ; Z length offset vector G17 |
| $TC\_DP4[1,1] = 8.$ | ; Y |
| $TC\_DP5[1,1] = 5.$ | ; X |
| TRAORI(1) | ; Activation of transformation |
| ORIMKS | ; Orientation reference to MCS |
| G0 X1 Y0 Z0 A0 B0 F20000 G90 G64 T1 D1 G17 | |
| ; Programming of direction vector | |
| G1 G90 | |
| a3 = 0 b3 = 1 c3 = 0 | |
| ; Programming in Euler angles | |
| G1 G90 | |
| a2 = 0 b2 = 0 X0 | |
| ; Programming of rotary axis motion | |
| G1 X10 Y5 Z20 A90 C90 | |
| m30 | |
1.13.4 Example for orientation axes

Example 1:

3 orientation axes for the 1st orientation transformation for kinematics with 6 transformed axes. Rotation must be done in the following sequence:

- firstly about the Z axis.
- then about the Y axis and
- finally about the Z axis again.

The tool vector must point in the X direction.

| Program code | Comment |
|--------------|---------|
| CHANDATA(1)  |         |
| $MC\_ TRAFO5\_ TOOL\_ VECTOR\_ 1=0      ; Tool vector in X direction |
| $MC\_ TRAFO5\_ ORIAX\_ ASSIGN\_ TAB\_ 1[0]=4; Channel index of first orientation axis |
| $MC\_ TRAFO5\_ ORIAX\_ ASSIGN\_ TAB\_ 1[1]=5; Channel index of second orientation axis |
| $MC\_ TRAFO5\_ ORIAX\_ ASSIGN\_ TAB\_ 1[2]=6; Channel index of third orientation axis |
| $MC\_ ORIAX\_ TURN\_ TAB\_ 1[0]=3; Z direction |
| $MC\_ ORIAX\_ TURN\_ TAB\_ 1[1]=2; Y direction |
| $MC\_ ORIAX\_ TURN\_ TAB\_ 1[2]=3; Z direction |

CHANDATA(1)
M17

Figure 1-26 3 orientation axes for the 1st orientation transformation for kinematics with 6 transformed axes
Example 2:
3 orientation axes for the 2nd orientation transformation for kinematics with 5 transformed axes. Rotation must be done in the following sequence:

- firstly about the X axis.
- then about the Y axis and
- finally about the Z axis.

The tool vector must point in the Z direction.

| Program code | Comment |
|--------------|---------|
| $\text{MC\_TRAFO5\_TOOL\_VECTOR\_Z}=2$ | ; Tool vector in Z direction |
| $\text{MC\_TRAFO5\_ORIAX\_ASSIGN\_TAB\_1[0]}=4$ | ; Channel index of first orientation axis |
| $\text{MC\_TRAFO5\_ORIAX\_ASSIGN\_TAB\_1[1]}=5$ | ; Channel index of second orientation axis |
| $\text{MC\_TRAFO5\_ORIAX\_ASSIGN\_TAB\_1[2]}=0$ | ; Channel index of third orientation axis |
| $\text{MC\_ORIAX\_TURN\_TAB\_1[0]}=1$ | ; X direction |
| $\text{MC\_ORIAX\_TURN\_TAB\_1[1]}=2$ | ; Y direction |
| $\text{MC\_ORIAX\_TURN\_TAB\_1[2]}=3$ | ; Z direction |

Figure 1-27 3 orientation axes for the 2nd orientation transformation for kinematics with 5 transformed axes

The rotation through angle C2 about the Z'' axis is omitted in this case, because the tool vector orientation can be determined solely from angles A2 and B2 and no further degree of freedom is available on the machine.

References:
Programming Manual, Production Planning
1.13.5 Examples for orientation vectors

1.13.5.1 Example for polynomial interpretation of orientation vectors

Orientation vector in Z-X plane

The orientation vector is programmed directly in the examples below. The resulting movements of the rotary axes depend on the particular kinematics of the machine.

| Program code | Comment                                      |
|--------------|----------------------------------------------|
| N10 TRAORI   |                                              |
| N20 POLY     | ; Polynomial interpolation is possible.      |
| N30 A3=0 B3=0 C3=1 | ; Orientation in +Z direction (start vector) |
| N40 A3=1 B3=0 C3=0 | ; Orientation in +X direction (end vector)   |

In N40, the orientation vector is rotated in the Z-X plane which is spanned by the start and end vector. Here, the PHI angle is interpolated in a line in this plane between the values 0 and 90 degrees (large circle interpolation).

The additional specification of the polynomials for the two angles PHI and PSI means that the interpolated orientation vector can lies anywhere between the start and end vector.

PHI angle using polynomial PHI

In contrast to the example above, the PHI angle is interpolated using the polynomial PHI(u) = (90-10)u + 10*u^2 between the values 0 and 90 degrees.

The angle PSI is not equal to zero and is interpolated according to the following polynomial:

PSI(u) = -10*u + 10*u^2

The maximum "tilt" of the orientation vector from the plane between the start and end vector is obtained in the middle of the block (u = 1/2).

| Program code | Comment                                      |
|--------------|----------------------------------------------|
| N10 TRAORI   |                                              |
| N20 POLY     | ; Polynomial interpolation is possible.      |
| N30 A3=0 B3=0 C3=1 | ; Orientation in +Z direction (start vector) |
| N40 A3=1 B3=0 C3=0 | PO[PHI]=(10) ; in +X direction (end vector)   |
|              | PO[PSI]=(10)                                |
1.13.5.2 Example of rotations of orientation vector

Rotations with angle of rotation THETA

In the following example, the angle of rotation is interpolated in linear fashion from starting value 0 degrees to end value 90 degrees. The angle of rotation changes according to a parabola or a rotation can be executed without a change in orientation. Tool orientation is rotated from the Y direction to the X direction.

| Program code | Comment |
|--------------|---------|
| N10 TRAORI   | ; Activation of orientation transformation |
| N20 G1 X0 Y0 Z0 F5000 | ; |
| N30 A3=0 B3=0 C3=1 THETA=0 | ; In Z direction with angle of rotation 0 |
| N40 A3=1 B3=0 C3=0 THETA=90 | ; In X direction and rotation |
| N50 A3=0 B3=1 C3=0 PO\{(THT)=(180,90) | ; By 90 degrees |
| N60 A3=0 B3=1 C3=0 THETA=IC(-90) | ; In Y direction and rotation |
| N70 ORIROTT  | ; Angle of rotation relative to |
| N80 A3=1 B3=0 C3=0 THETA=30 | ; Rotation vector in angle |

N40 Linear interpolation of angle of rotation from starting value 0 degrees to end value 90 degrees.

N50 The angle of rotation changes from 90 degrees to 180 degrees in accordance with the parabola.

\[ \theta(u) = 90 + u^2 \]

N60 A rotation can also be programmed without a change in orientation taking place.

N80 Tool orientation is rotated from the Y direction to the X direction. The change in orientation takes place in the X-Y plane and the rotation vector describes an angle of 30 degrees to this plane.
**1.13.6 Examples for generic axis transformations**

The following example is based on a machine with rotatable tool on which the first rotary axis is a C axis and the second a B axis (CB kinematics). The basic orientation defined in the machine data is the bisecting line between the X and Z axes.

Relevant machine data is as follows:

```
CHANDATA(1)
$MC_TRAFO_TYPE_1 = 24 ; General 5-axis transformation
$MC_TRAFO5_AXIS1_1[0] = 0.0
$MC_TRAFO5_AXIS1_1[1] = 0.0
$MC_TRAFO5_AXIS1_1[2] = 1.0 ; 1. Rotary axis is parallel to Z.
$MC_TRAFO5_AXIS2_1[0] = 0.0
$MC_TRAFO5_AXIS2_1[1] = 1.0
$MC_TRAFO5_AXIS2_1[2] = 0.0 ; 2. Rotary axis is parallel to Y.
$MC_TRAFO5_BASE_ORIENT_1[0] = 1.0
$MC_TRAFO5_BASE_ORIENT_1[1] = 0.0
$MC_TRAFO5_BASE_ORIENT_1[2] = 1.0

M30
```

Example program:

| Program code | Comment |
|--------------|---------|
| N10          | $TC_DP1[1,1] = 120 ; End mill |
| N20          | $TC_DP3[1,1] = 0 ; Length offset vector |
| N30          | N40      | ; Definition of tool carrier |
| N50          | $TC_CARR7[1] = 1 ; Component of the 1st rotary axis |
| N60          | $TC_CARR11[1] = 1 ; Component of the 2nd rotary axis |
| N70          | $TC_CARR13[1] = -45 ; Angle of rotation of 1st axis |
| N80          | $TC_CARR14[1] = 0 ; Angle of rotation of 2nd axis |
| N90          | N100     | X0 Y0 Z0 B0 C0 F10000 ORIWKGS G17 |
| N110         | TRAORI() ; Selection of basic transformation orientation |
| N120         | C3=1 ; Orientation parallel to Z |
|             | ; Set → B-45 C0 |
| N130         | T1 D1 ; Basic orientation is now parallel to Z |
| N140         | C3=1 ; Orientation parallel to Z |
|             | ; Set → B0 C0 |
### 1.13 Examples

| Program code | Comment |
|--------------|---------|
| N150 G19     | ; Basic orientation is now parallel to X |
| N160 C3=1    | ; Orientation parallel to Z |
|              | ; Set → B-90 C0 |
| N170 G17 TCARR=1 TCOABS | ; Basic orientation is now angle-halving Y-Z |
| N180 A3=1    | ; Orientation parallel to X |
|              | ; Set → B-90 C-135 |
| N190 B3=1 C3=1 | ; Orientation parallel to basic orientation → B0 C0 |
| N200 TRAORI(2.0, 3.0, 6.0) | ; Pass basic orientation in call |
| N210 A3=2 B3=3 C3=6 | ; Orientation parallel to basic orientation → B0 C0 |
| N220 TOFRAME  | ; Z axis points in the direction of the orientation |
| N230 G91 Z7   | ; 7 mm in new Z direction |
|              | ; Traverse → X2 Y3 Z6 |
| N240 C3=1     | ; Orientation parallel to New Z axis → B0 C0 |
| N250 M30      |         |
1.13.6.1 Example of a generic 6-axis transformation

Activation of a 6-axis transformation with subsequent orientation changes and traversing:

| Program code | Comment |
|--------------|---------|
| N10 A0 B0 X0 Y0 Z0; | ; |
| N20 TRAORI(1,...,0,0,0,0,1,0) ; | ; Transfer the orientation vector |
| ; and the orientation normal vector, |
| ; transformation selection |
| N30 T1 D1 X10 Y20 Z30 A3=0.5 | ; Change of orientation, rotation |
| C3=1 BN3=1 ORIPLANE ORIWK5 | ; and traversing motion |
| N40 B3=0.5 C3=1 AN3=-1 | ; Rotation programmed, |
| ; Orientation constantly |
| N50 M30 | |

A tool, of which the orientation differs from the default, is defined in the following example.

With G17, the orientation vector is in the X-Z plane and its inclination to the Z axis is 26.565 degrees because of $\tan(26.565) = 0.5 = \frac{\text{TC_DPV3}[2,2]}{\text{TC_DPV5}[2,2]}$.

The orientation normal vector is also specified. As only $\text{TC_DPVN4}[2,2]$ is not equal to zero, it points in the Y direction. Orientation vector and orientation normal vector are perpendicular to one another.

An orthogonalization is therefore **not necessary**, and therefore the programmed orientation normal vector is not modified.

| Program code | Comment |
|--------------|---------|
| N100 $\text{TC_DP1}[2,2] = 120$; | ; End mill |
| N110 $\text{TC_DP3}[2,2]$= 20; | ; Length offset vector |
| N120 $\text{TC_DPV}[2,2]$ = 0; | ; Tool cutting edge orientation |
| N130 $\text{TC_DPV3}[2,2]$ = 1; | ; X component tool cutting edge orientation |
| N140 $\text{TC_DPV4}[2,2]$ = 0; | ; Y component tool cutting edge orientation |
| N150 $\text{TC_DPV5}[2,2]$= 0.5; | ; Z component tool cutting edge orientation |
| N160 $\text{TC_DPV3}[2,2]$= 0; | ; X component orientation normal vector |
| N170 $\text{TC_DPV4}[2,2]$= 1; | ; Y component orientation normal vector |
| N180 $\text{TC_DPV5}[2,2]$= 0; | ; Z component orientation normal vector |
| N200 TRAORI(); | ; Pass basic orientation in call |
| N210 A3=5 C3=10 BN3=1; | ; Place rotary axes in initial state |
| N220 C3=1; | ; Orientation in the Z direction -> tool |
| ; rotated by 26.565 degrees |
| N230 THETA=IC(90); | ; Orientation normal vector |
| ; Incremental by 90 degrees |
| ; rotate Vector points in negative |
| ; X direction |
| N240 M30 | |
### 1.13.6.2 Example of a generic 7-axis transformation

**Example of a generic 7-axis transformation**

Activation of a 7-axis transformation with subsequent orientation changes and traversing:

| Program | Comment |
|---------|---------|
| N10 TRAFOOF | |
| N20 a0 b0 c0 x0 y0 z0 e=0 | |
| N30 $MC_TRAFO5_AXIS1_1[2] = 1 ; 1. Rotary axis shows in Z direction | |
| N40 $MC_TRAFO5_AXIS1_2[0] = 1 ; 2. Rotary axis shows in X direction | |
| N50 $MC_TRAFO5_AXIS1_3[2] = 1 ; 3. Rotary axis shows in Z direction | |
| N60 $MC_TRAFO7_EXT_AXIS1_1[0] = 1 ; 7. Axis shows in X direction | |
| N70 $MC_TRAFO_BASE_ORIENT_1[2] = 1 ; Orientation vector | |
| N80 $MC_TRAFO_BASE_ORIENT_NORMAL_1[1] = 1 ; Orientation normal vector | |
| N90 NEWCONF | |
| N100 traori() | |
| N110 G1 t1 d1 x10 y0 z50 c3=1 an3=1 bn3=1 | |
| | orivect oriwks G19 F10000 |
| N120 G2 y50 z0 b3=1 e=DC(90) CR=50 ; 1. Quadrant | |
| N130 G2 y0 z–50 c3=–1 e=DC(180) CR=50 ; 2. Quadrant | |
| N140 G2 y–50 z0 b3=–1 e=DC(270) CR=50 ; 3. Quadrant | |
| N150 G2 y0 z50 c3=1 e=DC(0) CR=50 ; 4. Quadrant | |
| N200 M30 | |

**Note**

While traversing the quadrant in the example, only the 7th axis turns by 360 degrees. The machine remains in the fixed position.

### 1.13.6.3 Example for the modification of rotary axis motion

The machine is a 5-axis machine of machine type 1 (two-axis swivel head with CA kinematics) on which both rotary axes rotate the tool (transformation type 24). The first rotary axis is a modulo axis parallel to Z (C axis); the second rotary axis is parallel to Y (B axis) and has a traversing range from -5 degrees to +185 degrees.

To allow modification at any time, the following machine data has the value 2:

**MD21180 $MC_ROT_AX_SWL_CHECK_MODE** (check software limits for orientation axes)

| Program | Comment |
|---------|---------|
| N10 X0 Y0 Z0 B0 C0 | ; basic orientation 5-axis transformation |
| N20 TRAORI( ) | ; Rotary axis positions B-1 and C10 |
| N30 B-1 C10 | |
| N40 A3=1 C3=1 ORIWKS ; large circle interpolation in WCS | |
| N50 M30 | |
At the start of block N40 in the example program, the machine is positioned at rotary axis positions B-1 C10. The programmed end orientation can be achieved with either of the axis positions B-45 C0 (1st solution) or B45 C180 (2nd solution).

The first solution is selected initially, because it is nearest to the starting orientation and, unlike the second solution, can be achieved using large circle interpolation (ORIWKS). However, this position cannot be reached because of the axis limits of the B axis.

The second solution is therefore used instead, i.e. the end position is B45 C180. The end orientation is achieved by axis interpolation. The programmed orientation path cannot be followed.

### Example: Compression of an orientation

In the example program below, a circle approached by a polygon definition is compressed. The tool orientation moves on the outside of the taper at the same time. Although the programmed orientation changes are executed one after the other, but in an unsteady way, the compressor function generates a smooth motion of the orientation.

#### Programming

```plaintext
DEF INT NUMBER=60
DEF REAL RADIUS=20
DEF INT COUNTER
DEF REAL ANGLE
N10 G1 X0 Y0 F5000 G64

$SC_COMPRESS_CONTUR_TOL=0.05 ; Maximum deviation of the contour = 0.05 mm
$SC_COMPRESS_ORI_TOL=5 ; Maximum deviation of the orientation = 5 degrees

TRAORI
COMPCURV

; The movement describes a circle generated from polygons. The orientation moves on a taper around the Z axis with an opening angle of 45 degrees.

N100 X0 Y0 A3=0 B3=-1 C3=1
N110 FOR COUNTER=0 TO NUMBER
N120 ANGLE=360*COUNTER/NUMBER
N130 X=RADIUS*cos(angle) Y=RADIUS*sin(angle)
A3=sin(angle) B3=-cos(angle) C3=1
N140 ENDFOR
```
### 1.14 Data lists

#### 1.14.1 Machine data

##### 1.14.1.1 General machine data

| Number  | Identifier: $MN_\_ | Description                                                  |
|---------|-------------------|--------------------------------------------------------------|
| 10620   | EULER_ANGLE_NAME_TAB | Name of Euler angles or names of orientation axes            |
| 10630   | NORMAL_VECTOR_NAME_TAB | Name of normal vectors                                         |
| 10640   | DIR_VECTOR_NAME_TAB  | Name of direction vectors                                     |
| 10642   | ROT_VECTOR_NAME_TAB  | Name of rotation vectors                                      |
| 10644   | INTER_VECTOR_NAME_TAB| Name of intermediate vector components                      |
| 10646   | ORIENTATION_NAME_TAB | Identifier for programming a 2nd orientation path             |
| 10648   | NUTATION_ANGLE_NAME  | Name of orientation angle                                     |
| 10670   | STAT_NAME           | Name of position information                                  |
| 10672   | TU_NAME             | Name of position information of the axes                     |
| 10674   | PO_WITHOUT_POLY     | Permits programming of PO[] without POLY having to be active |

##### 1.14.1.2 Channelspecific machine data

| Number  | Identifier: $MC_\_ | Description                                                   |
|---------|-------------------|--------------------------------------------------------------|
| 20150   | GCODE_RESET_VALUES[n] | Reset G groups                                               |
| 20152   | GCODE_RESET_MODE[n] | Setting after RESET/end of part program                      |
| 20482   | COMPRESS_MODE      | Mode of the compressor                                       |
| 20621   | HAN DHW.ORIA X_MAX_INCR_SIZE | Limitation of handwheel increment                          |
| 20623   | HANDHW.ORIA X_MAX_INCR_VSIZE | Orientation velocity overlay                            |
| 21094   | ORIPATH_MODE       | Setting for path relative orientation                        |
| 21100   | ORIENTATION_IS_EULER | Angle definition for orientation programming              |
| 21102   | ORI_DEF_WITH_G_CODE | Definition of orientation angles A2, B2, C2                |
| 21104   | ORI IPO_WITH_G_CODE | Definition of interpolation type for orientation            |
| 21106   | CART_JOG_SYSTEM    | Coordinate system for Cartesian JOG                          |
| 21108   | POLE_ORI_MODE      | Behavior during large circle interpolation at pole position |
| 21120   | ORIA X_TURN_TAB_1[n] | Assignment of rotation of orientation axes about the reference axes, definition 1 [n = 0..2] |
| 21130   | ORIA X_TURN_TAB_2[n] | Assignment of rotation of orientation axes about the reference axes, definition 2 [n = 0..2] |
## F2: Multi-axis transformations

### 1.14 Data lists

| Number   | Identifier: $MC_\text{Description}$                                                                 |
|----------|-----------------------------------------------------------------------------------------------------|
| 21132    | ORI_DISP_IS_MODULO[n] Modulo display of the orientation axes positions [n = 0..2]                  |
| 21134    | ORI_DISP_MODULO_RANGE Size of the module range for the display of the orientation axes            |
| 21136    | ORI_DISP_MODULO_RANGE_START Starting position of the module range for the display of the orientation axes |
| 21150    | JOG_VELO_RAPID_ORI[n] Rapid traverse in jog mode for orientation axes in the channel [n = 0..2]    |
| 21155    | JOG_VELO_ORI[n] Orientation axis velocity in jog mode [n = 0..2]                                   |
| 21160    | JOG_VELO_RAPID_GEO[n] Rapid traverse in jog mode for geometry axes in the channel [n = 0..2]       |
| 21165    | JOG_VELO_GEO[n] Geometry axis velocity in jog mode [n = 0..2]                                      |
| 21170    | ACCEL_ORI[n] Acceleration for orientation axes [n = 0..2]                                         |
| 21180    | ROT_AX_SWL_CHECK_MODE Check software limits for orientation axes                                   |
| 21186    | TOCARR_ROT_OFFSET_FROM_FR Offset of TOCARR rotary axes                                             |
| 21190    | TOFF_MODE Operation of online offset in tool direction                                               |
| 21194    | TOFF_VELO Speed of online offset in tool direction                                                   |
| 21196    | TOFF_ACCEL Acceleration of online offset in tool direction                                           |
| 24100    | TRAFO_TYPE_1 Definition of transformation 1 in channel                                                |
| 24110    | TRAFO_AXES_IN_1[n] Axis assignment for transformation 1 [axis index]                                |
| 24120    | TRAFO_GEOAX_ASSIGN_TAB_1[n] Assignment geometry axis to channel axis for transformation 1 [geometry no.] |
| 24200    | TRAFO_TYPE_2 Definition of transformation 2 in channel                                                |
| 24210    | TRAFO_AXES_IN_2[n] Axis assignment for transformation 2 [axis index]                                 |
| 24220    | TRAFO_GEOAX_ASSIGN_TAB_2[n] Assignment geometry axis to channel axis for transformation 2 [geometry no.] |
| 24300    | TRAFO_TYPE_3 Definition of transformation 3 in channel                                                |
| 24310    | TRAFO_AXES_IN_3[n] Axis assignment for transformation 3 [axis index]                                 |
| 24320    | TRAFO_GEOAX_ASSIGN_TAB_3[n] Assignment geometry axis to channel axis for transformation 3 [geometry no.] |
| 24400    | TRAFO_TYPE_4 Definition of transformation 4 in channel                                                |
| 24410    | TRAFO_AXES_IN_4[n] Axis assignment for transformation 4 [axis index]                                 |
| 24420    | TRAFO_GEOAX_ASSIGN_TAB_4[n] Assignment geometry axis to channel axis for transformation 4 [geometry no.] |
| 24430    | TRAFO_TYPE_5 Definition of transformation 5 in channel                                                |
| 24432    | TRAFO_AXES_IN_5[n] Axis assignment for transformation 5 [axis index]                                 |
| 24434    | TRAFO_GEOAX_ASSIGN_TAB_5[n] Assignment geometry axis to channel axis for transformation 5 [geometry no.] |
| 24440    | TRAFO_TYPE_6 Definition of transformation 6 in channel                                                |
| 24442    | TRAFO_AXES_IN_6[n] Axis assignment for transformation 6 [axis index]                                 |
| 24444    | TRAFO_GEOAX_ASSIGN_TAB_6[n] Assignment geometry axis to channel axis for transformation 6 [geometry no.] |
| Number | Identifier: $MC_\text{Description}$ | Description |
|--------|-----------------------------------|-------------|
| 24450  | TRAFO_TYPE_7                      | Definition of transformation 7 in channel |
| 24452  | TRAFO_AXES_IN_7[n]                | Axis assignment for transformation 7 [axis index] |
| 24454  | TRAFO_GEOAX_ASSIGN_TAB_7[n]       | Assignment geometry axis to channel axis for transformation 7 [geometry no.] |
| 24460  | TRAFO_TYPE_8                      | Definition of transformation 8 in channel |
| 24462  | TRAFO_AXES_IN_8[n]                | Axis assignment for transformation 8 [axis index] |
| 24464  | TRAFO_GEOAX_ASSIGN_TAB_8[n]       | Assignment geometry axis to channel axis for transformation 8 [geometry no.] |
| 24470  | TRAFO_TYPE_9                      | Definition of transformation 9 in channel |
| 24472  | TRAFO_AXES_IN_9[n]                | Axis assignment for transformation 9 [axis index] |
| 24474  | TRAFO_GEOAX_ASSIGN_TAB_9[n]       | Assignment geometry axis to channel axis for transformation 9 [geometry no.] |
| 24480  | TRAFO_TYPE_10                     | Definition of transformation 10 in channel |
| 24482  | TRAFO_AXES_IN_10[n]               | Axis assignment for transformation 10 [axis index] |
| 24484  | TRAFO_GEOAX_ASSIGN_TAB_10[n]      | Assignment geometry axis to channel axis for transformation 10 [geometry no.] |
| 24500  | TRAFO5_PART_OFFSET_1[n]           | Offset vector for 5-axis transformation 1 [n = 0..2] |
| 24510  | TRAFO5_ROT_AX_OFFSET_1[n]         | Position offset of rotary axis 1/2 for 5-axis transformation 1 [axis no.] |
| 24520  | TRAFO5_ROT_SIGN_IS_PLUS_1[n]      | Sign of rotary axis 1/2 for 5-axis transformation 1 [axis no.] |
| 24530  | TRAFO5_NON_POLE_LIMIT_1           | Definition of pole range for 5-axis transformation 1 |
| 24540  | TRAFO5_POLE_LIMIT_1               | End angle tolerance with interpolation through pole for 5-axis transformation 1 |
| 24550  | TRAFO5_BASE_TOOL_1[n]             | Vector of base tool for activation of 5-axis transformation 1 [n = 0..2] |
| 24558  | TRAFO5_JOINT_OFFSET_PART_1[n]     | Vector of kinematic offset in table for 5-axis transformation 1 [n = 0..2] |
| 24560  | TRAFO5_JOINT_OFFSET_1[n]          | Vector of kinematic offset for 5-axis transformation 1 [n = 0..2] |
| 24561  | TRAFO6_JOINT_OFFSET_2_3_1[n]      | Vector of kinematic offset for 6-axis transformation 2_3_1 |
| 24562  | TRAFO5_TOOL_ROT_AX_OFFSET_1[n]    | Offset of focus of 1st 5-axis transformation with swiveled linear axis. |
| 24564  | TRAFO5_NUTATOR_AX_ANGLE_1         | Angle of 2nd rotary axis for the universal milling head |
| 24570  | TRAFO5_AXIS1_1[n]                 | Vector for the first rotary axis and the first orientation transformation [n = 0..2] |
| 24572  | TRAFO5_AXIS2_1[n]                 | Vector for the second rotary axis and the first transformation [n = 0..2] |
| 24673  | TRAFO5_AXIS3_1[n]                 | Direction of third rotary axis for general 6-axis transformation (Transformer type 24, 40, 56, 57) |
### F2: Multi-axis transformations

#### 1.14 Data lists

| Number | Identifier | Description |
|--------|------------|-------------|
| 24574  | TRAFO5_BASE_ORIENT_1[n] | Basic orientation for the first transformation \([n = 0..2]\) |
| 24576  | TRAFO6_BASE_ORIENT_NORMAL_1[n] | Tool normal vector for the first transformation \([n = 0..2]\) |
| 24580  | TRAFO5_TOOL_VECTOR_1 | Tool vector direction for the first 5-axis transformation |
| 24582  | TRAFO5_TCARR_NO_1 | TCARR number for the first 5-axis transformation |
| 24585  | TRAFO5_ORIAX_ASSIGN_TAB_1[n] | Assignment of orientation axes to channel axes for orientation transformation 1 \([n = 0..2]\) |
| 24590  | TRAFO5_ROT_OFFSET_FROM_FR_2 | Offset of transf. rotary axes from WO |
| 24594  | TRAFO7_EXT_ROT_AX_OFFSET_1 | Angle offset of the 1st external rotary axis |
| 24595  | TRAFO7_EXT_AXIS1_1 | Direction of the 1st external rotary axis |
| 24600  | TRAFO5_PART_OFFSET_2[n] | Offset vector for 5-axis transformation 2 \([n = 0..2]\) |
| 24610  | TRAFO5_ROT_AX_OFFSET_2[n] | Position offset of rotary axis 1/2 for 5-axis transformation 2 \([\text{axis no.}]\) |
| 24620  | TRAFO5_ROT_SIGN_IS_PLUS_2[n] | Sign of rotary axis 1/2 for 5-axis transformation 2 \([\text{axis no.}]\) |
| 24630  | TRAFO5_NON_POLE_LIMIT_2 | Definition of pole range for 5-axis transformation 2 |
| 24640  | TRAFO5_POLE_LIMIT_2 | End angle tolerance with interpolation through pole for 5-axis transformation 2 |
| 24650  | TRAFO5_BASE_TOOL_2[n] | Vector of base tool with activation of 5-axis transformation 2 \([n = 0..2]\) |
| 24658  | TRAFO5_JOINT_OFFSET_PART_2[n] | Vector of kinematic offset in table for 5-axis transformation 2 \([n = 0..2]\) |
| 24660  | TRAFO5_JOINT_OFFSET_2[n] | Vector of kinematic offset for 5-axis transformation 2 \([n = 0..2]\) |
| 24661  | TRAFO6_JOINT_OFFSET_2_3_2[n] | Vector of kinematic offset for 6-axis transformation 2_3_2 |
| 24662  | TRAFO5_TOOL_ROT_AX_OFFSET_2[n] | Offset of focus of 2nd 5-axis transformation with swiveled linear axis. |
| 24664  | TRAFO5_NUTATOR_AX_ANGLE_2 | Angle of the 2nd rotating axis for the universal milling head |
| 24670  | TRAFO5_AXIS1_2[n] | Vector for the first rotary axis and the second orientation transformation \([n = 0..2]\) |
| 24672  | TRAFO5_AXIS2_2[n] | Vector for the second rotary axis and the first transformation \([n = 0..2]\) |
| 24673  | TRAFO5_AXIS3_2[n] | Direction of third rotary axis for generic 6-axis transformation (type 24, 40, 56, 57) |
| 24674  | TRAFO5_BASE_ORIENT_2[n] | Basic orientation for the second transformation \([n = 0..2]\) |
| 24676  | TRAFO6_BASE_ORIENT_NORMAL_2[n] | Tool normal vector for the second transformation \([n = 0..2]\) |
| 24680  | TRAFO5_TOOL_VECTOR_2 | Tool vector direction for the second 5-axis transformation 2 |
### 1.14 Data lists

#### 1.14.2 Setting data

##### 1.14.2.1 General setting data

| Number | Identifier: $SN_ | Description |
|--------|------------------|-------------|
| 41110  | JOG_SET_VELO     | Geometry axes |
| 41130  | JOG_ROT_AX_SET_VELO | Orientation axes |

##### 1.14.2.2 Channelspecific setting data

| Number | Identifier: $SC_ | Description |
|--------|------------------|-------------|
| 42475  | COMPRESS_CONTOUR_TOL | Max. contour deviation for compressor |
| 42476  | COMPRESS_ORI_TOL   | Max. angular displacement of tool orientation for the compressor |
| 42477  | COMPRESS_ORI_ROT_TOL | Max. angular displacement of tool rotation for the compressor |
| 42650  | CART_JOG_MODE     | Coordinate system for Cartesian manual travel |
| 42660  | ORI_JOG_MODE      | Definition of virtual kinematics for JOG |
| 42670  | ORIPATH_SMOOTH_DIST | Smoothing path of the orientation |
| 42672  | ORIPATH_SMOOTH_TOL | Smoothing tolerance of the orientation |
| 42970  | TOFF_LIMIT        | Upper limit for offset value $AA_TOFF |

---

| Number | Identifier: $MC_ | Description |
|--------|------------------|-------------|
| 24682  | TRAFO5_TCARR_NO_2 | TCARR number for the second 5-axis transformation 2 |
| 24685  | TRAFO5_ORIAX_ASSIGN_TAB_2[n] | Assignment of orientation axes to channel axes for orientation transformation 2 $[n = 0..2]$ |
| 24694  | TRAFO7_EXT_ROT_AX_OFFSET_2 | Angle offset of the 2nd external rotary axis |
| 24695  | TRAFO7_EXT_AXIS1_2 | Direction of the 2nd external rotary axis |
| 25294  | TRAFO7_EXT_ROT_AX_OFFSET_3 | Angle offset of the 3rd external rotary axis |
| 25295  | TRAFO7_EXT_AXIS1_3 | Direction of the 3rd external rotary axis |
| 25394  | TRAFO7_EXT_ROT_AX_OFFSET_4 | Angle offset of the 4th external rotary axis |
| 25395  | TRAFO7_EXT_AXIS1_4 | Direction of the 4th external rotary axis |
| 28580  | MM_ORIPATH_CONFIG | Configuration for path relative orientation ORIPATH |
### 1.14.3 Signals

#### 1.14.3.1 Signals from channel

| Signal name                                           | SINUMERIK 840D sl | SINUMERIK 828D |
|-------------------------------------------------------|------------------|----------------|
| Activate PTP traversal                                | DB21, .. DBX29.4 | -              |
| Transformation active                                 | DB21, .. DBX33.6 | -              |
| Number of active G function of G function group 25    | DB21, .. DBB232  | -              |
| PTP traversal active                                  | DB21, .. DBX317.6| -              |
| Activate online tool length offset                    | DB21, .. DBX318.2| -              |
| Activate offset motion                                | DB21, .. DBX318.3| -              |
F2: Multi-axis transformations

1.14 Data lists
2.1 Brief description

For gantry machines, each of various machine elements, such as the gantry and the transverse beams, are moved by several axes that operate in parallel. The axes that together move a machine part, are designated as gantry axes or gantry grouping. Because of the mechanical structure, the gantry axes are rigidly connected with each other and so must always be traversed synchronously by the control.

![Diagram of gantry machine](image)

**Figure 2-1** Example: Gantry-type milling machine with gantry and transverse beams

**Guide axis**

The guide axis of the gantry grouping is the axis that represents the gantry grouping. Only this axis is programmed to perform the traversing movements of the gantry grouping.

**Synchronous axes**

The synchronous axes of the gantry grouping are the axes that because of their coupling with the guide axis are also automatically traversed by the control. A guide axis can be assigned any number of synchronous axes.

**Synchronization difference**

The synchronous operation difference is the deviation of the axial actual value of a synchronous axis from its ideal position referred to the actual value of the guide axis. The control continually monitors the synchronous operation difference. A message is displayed if the alarm limit is exceeded. The complete gantry grouping is stopped when the alarm limit is exceeded. The limit values can be parameterized as machine data.
2.2 "Gantry axes" function

2.2.1 Definition of a gantry grouping

Definition

The axes of a gantry grouping are specified via the following axial machine data:

| MD37100 $MA_GANTRY_AXIS_TYPE[AX1] = xy |
|-----------------------------------------|
| x | Tens decimal place: Type of gantry axis (guide or synchronous axis) |
| y | Ones decimal place: ID of the gantry grouping |

A maximum of eight gantry groupings (gantry grouping ID: 1 - 8) can be defined. The gantry grouping ID must be unique in all channels or in all NCUs in accordance with the assigned axis.

In principle, a gantry grouping can be assigned any number of synchronous axes.

Example

Definition of a gantry grouping with ID=1, guide axis AX1 and synchronous axis AX2

- MD37100 $MA_GANTRY_AXIS_TYPE[AX1] = 01 (guide axis)
- MD37100 $MA_GANTRY_AXIS_TYPE[AX2] = 11 (synchronous axis)

Supplementary conditions

The following supplementary conditions apply to a gantry grouping:

- A gantry grouping must not contain a spindle.
- A synchronous axis must not be a concurrent POS axis.
- A synchronous axis must not belong to a transformation.
- A synchronous axis must not be a following axis of another axis coupling.
- A synchronous axis must not be a guide axis of another axis coupling.
- All axes of a gantry grouping must be of the same axis type, linear or rotary: MD30300 $MA_IS_ROT_AX (rotary axis/spindle)

NOTICE

Drive optimization

At a SINAMICS S120 drive unit, a maximum of three drives can be optimized or measured at the same time (speed controller optimization / function generator). Therefore, for a coupling with more than three coupled drives at the same time, we recommend that these are distributed over several drive units.
2.2.2 Monitoring the synchronism difference

Limit values for monitoring

2 limit values can be specified for the synchronism difference.

Gantry warning limit

The gantry warning limit is set using the following machine data:

MD37110 $MA_GANTRY_POS_TOL_WARNING (gantry warning limit)

The "Alarm limit exceeded" message is displayed if the synchronism difference exceeds the gantry warning limit. In addition, the NC/PLC-interface signal is set:

DB31, ... DBX101.3 = 1 (gantry warning limit exceeded)

After the alarm limit has been fallen below, the message and interface signal are automatically reset.

Note

Gantry warning limit

If the "Alarm limit exceeded" message is not to be displayed, then a value of 0 should be entered into MD37110.

Gantry trip limit

The gantry trip limit is set using the following machine data:

- for the synchronized gantry grouping:
  MD37120 $MA_GANTRY_POS_TOL_ERROR

- for the non-synchronized gantry grouping:
  MD37130 $MA_GANTRY_POS_TOL_REF

Alarm 10653 "Error limit exceeded" is displayed if the synchronism difference exceeds the gantry trip limit. In addition, the NC/PLC-interface signal is set:

DB31, ... DBX101.2 = 1 (gantry trip limit exceeded)

The alarm is also displayed if the gantry grouping is jammed (no controller enable, gantry grouping in the "Hold" state).
### 2.2.3 Extended monitoring of the synchronism difference

**Activation of the extended monitoring**

An extended monitoring of the synchronism difference can be activated using the following machine data:

\[
\text{MD37150 $MA\_GANTRY\_FUNCTION\_MASK$, Bit 0 = 1}
\]

For the extended monitoring, a synchronism difference between the leading and synchronous axis, obtained when tracking or when the gantry grouping is opened, is taken into account.

The extended monitoring becomes active after the NC boots after the first referencing (incremental encoder) or synchronization (absolute encoder).

**Exceeding the gantry trip limit**

If, when the extended monitoring is active, the trip limit of the synchronism difference is exceeded, Alarm 10653 "Error limit exceeded" is displayed.

In order to be able to reset the alarm, proceed as follows:

1. Deactivating extended monitoring:
   \[
   \text{MD37150 $MA\_GANTRY\_FUNCTION\_MASK$, Bit 0 = 0}
   \]
2. Deleting the synchronism difference displayed in the machine data:
   \[
   \text{MD37135 $GANTRY\_ACT\_POS\_TOL\_ERROR = 0}
   \]
3. Cancel alarm
4. Re-reference or re-synchronize the axes of the gantry grouping
5. Reactivating extended monitoring:
   \[
   \text{MD37150 $MA\_GANTRY\_FUNCTION\_MASK$, Bit 0 = 1}
   \]

### 2.2.4 Referencing and synchronization of gantry axes

**Use case**

In cases where an incremental measuring system is being used for the leading or the synchronous axis, after the NC boots, the measuring systems must be referenced, maintaining the axis coupling.

After every axis in the gantry grouping has approached its reference point, any misalignment that may exist between the axes must be eliminated (gantry synchronization process). Once this has been performed, the NC/PLC interface signal is set:

\[
\text{DB31, ... DBX101.5 = 1 (gantry grouping is synchronized)}
\]

For the sequence when referencing or synchronizing gantry axes, see Section [Referencing and synchronization of gantry axes](Page 152).
2.2.5 Control dynamics

Use case

From the user perspective, a gantry grouping is exclusively traversed via the leading axis. The NC generates the setpoints of the synchronous axes directly from the setpoints of the leading axis in time synchronism and outputs these to them. To minimize the synchronous operation differences, the control system dynamics of all axes of a gantry grouping must be set identical (see Section “Start-up of gantry axes (Page 162)”).

Note

Identical control dynamics must be set for all axes of a gantry grouping.

Disturbance characteristic

If faults occur, which cause an axis of the gantry to be stopped, then the complete gantry grouping is always stopped.

2.2.6 Opening the gantry grouping

Description

The axis coupling within a gantry grouping can be opened (dissolved) using the following machine data:

MD37140 $MA_GANTRY_BREAK_UP = 1 (invalidate gantry grouping)

When the setting becomes active, the axes of the gantry grouping can be individually traversed in the JOG, AUTOMATIC and MDA modes.

The monitoring functions of the synchronism difference and/or the alarm and trip limits are not active.

The NC/PLC interface signal "Gantry grouping is synchronized" is reset:

DB31, ... DBX101.5 = 0

⚠️ CAUTION

If the gantry axes remain mechanically coupled, there is a risk of damage to the machine when the leading or synchronous axes are traversed in this operating state!
2.3 Referencing and synchronization of gantry axes

2.3.1 Introduction

Misalignment after starting

Immediately after the machine is switched on, the leading and synchronous axes may not be ideally positioned in relation to one another (e.g. misalignment of a gantry). Generally speaking, this misalignment is relatively small so that the gantry axes can still be referenced.

In special cases (e.g. gantry axes were stopped owing to a disturbance, power failure or EMERGENCY STOP), the dimensional offset must be checked for permissible tolerance values and a compensatory motion executed if necessary before the axes are traversed.

To execute this compensatory motion, the gantry grouping must be invalidated by means of the following machine data:

MD37140 $MA_GANTRY_BREAK_UP (invalidate gantry grouping)

Gantry synchronization process

All gantry axes must first be referenced and then synchronized after the control system is switched on. During gantry synchronization, all gantry axes approach the reference position of the gantry grouping in the decoupled state. The reference position of the gantry grouping for referencing the gantry axes corresponds to the reference position of the leading axis:

MD34100 $MA_REFP_SET_POS (reference point value/destination point for distance-coded system)

Otherwise, the reference position is the current actual position of the leading axis.

These operations for referencing and synchronizing the gantry axes are executed automatically in accordance with a special flowchart.
Referencing process

The flowchart for referencing gantry axes using an incremental measuring system is as follows:

Section 1:
Referencing of the leading axis

The axis-specific referencing of the gantry axis will be started by the active machine function REF upon the leading axis' interface signal from the PLC user program:

DB31, ... DBX4.7/4.6 (traversing key plus/minus)

The leading axis approaches the reference point (operational sequence as for reference point approach).

References:
Function Manual Basic Functions; Reference Point Approach (R1)

The appropriate synchronous axes traverse in synchronism with the leading axis. Interface signal "Referenced/synchronized" of the leading axis is output to indicate that the reference point has been reached.

Section 2:
Referencing of the synchronous axes

As soon as the leading axis has approached its reference point, the synchronous axis is automatically referenced (as for reference point approach).

References:
Function Manual Basic Functions; Reference Point Approach (R1)

The dependency between the leading axis and synchronous axis is inverted in the control for this phase so that the leading axis now traverses in synchronism with the synchronous axis. IS "Referenced/synchronized" of the synchronous axis is output to indicate that the reference point has been reached. The gantry axis dependency then reverts to its previous status. If a further synchronous axis is defined in the grouping, then this is also referenced in the way described above.

Section 3:
Gantry synchronization process

Once all axes in the gantry grouping have been referenced, they must be synchronized with the defined reference position. The actual position of each gantry axis is first compared to the defined reference position of the leading axis.
The next step in the operating sequence depends on the difference calculated between the actual values of the leading and synchronous axes:

- difference is **smaller** than the gantry warning limit:
  
  MD37110 $MA_GANTRY_POS_TOL_WARNING (gantry warning limit)
  
  The gantry synchronization process is started **automatically**. The message "Synchronization in progress gantry grouping x" is output during this process.
  
  The message "Synchronization running gantry grouping x" can be suppressed with:
  
  MD37150 $MA_GANTRY_FUNCTION_MASK Bit 2 = 1
  
  All gantry axes traverse at a specific position value **in the decoupled state** at the velocity set in the machine data:
  
  MD34040 $MA_REFP_VELO_SEARCH_MARKER (creep velocity)
  
  The position value is defined by the leading axis:
  
  MD34100 $MA_REFP_SET_POS (reference point/destination point for distance-coordinated system)
  
  The absolute encoders and distanced-coded encoders of the leading axis will be set to the current actual position of the leading axis or to the reference point by the following machine data:
  
  MD34330 $MA_REFP_STOP_AT_ABS_MARKER (Distancecoded linear measuring system without destination point)
  
  For this operation, the axes traverse at the same velocity as set for reference point approach:
  
  MD34070 $MA_REFP_VELO_POS (reference point positioning velocity)
  
  As soon as all gantry axes have reached their target position (ideal position), IS "Gantry grouping is synchronized" is set to "1" followed by re-activation of the gantry axis coupling. The position actual value of all axes in the gantry grouping must now be identical. The gantry synchronization process is now complete.

- Difference is **higher** than the gantry warning limit for at least one synchronous axis:
  
  IS "Gantry synchronization read to start" is set to "1" and the message "Wait for synchronization start of gantry grouping x" is output. The gantry synchronization process is not started automatically in this case, but must be started explicitly by the operator or from the PLC user program. The process is initiated by IS "Start gantry synchronization" on the leading axis. The signal is set on the leading axis. The operational sequence is then the same as that described above.
The following flowchart illustrates the referencing and synchronization processes.

**Figure 2-2**  Flowchart for referencing and synchronization of gantry axes
Synchronization process

A synchronization process is always required in the following cases:

- after the reference point approach of all axes included in a grouping,
- if the axes become de-synchronized (s below).

Operational sequence failure

If the referencing process described above is interrupted as a result of disturbances or a RESET, proceed as follows:

- **Abort within section 1 or 2:**
  Restart reference point with leading axis (see section 1)

- **Abort in section 3:**
  In cases where the gantry axes have not yet been referenced (IS "Referenced/Synchronized" = 1), the gantry synchronization process can be started again with IS "Synchronize gantry grouping".

Restart gantry synchronization

Synchronization of the gantry axes can be started with IS "Start gantry synchronization" under the following conditions only:

- JOG/REF mode must be active. The following interface signal must be set:
  - DB11, ... DBX5.2 = 1 (active machine function REF)
  - DB31, ... DBX 101.5 = 0 (gantry grouping is synchronized)
  - All grouping axes operate within the tolerance windows:
    - DB31, ... DBX 101.4 = 1 (gantry synchronization process ready to start)
  - Axes are not referenced in the relevant NC channel
    - DB21, ... DBX33.0 = 0 (referencing active)

If the gantry synchronization process is **not started from the referencing process** by means of IS "Start gantry synchronization process", then the reference position is not specified as target position for the synchronous axes:

MD34100 $MA_REFP_SET_POS (reference point value/destination point for distancecoded system)

Instead, the **actual position of the leading axis** is specified as the target position and is approached in the uncoupled state.
2.3 Referencing and synchronization of gantry axes

**Note**

For the leading axis, automatic synchronization can be locked using the following NC/PLC interface signal:

\[ \text{DB31, ... DBX29.5} = 1 \] (no automatic synchronization process)

This always makes sense if no axis enabling signal has yet been issued for the axes. In this case, the synchronization process should also be started explicitly with the NC/PLC interface signal:

\[ \text{DB31, ... DBX29.4} = 1 \] (start gantry synchronization process)

**Loss of synchronization**

The gantry grouping becomes desynchronized as a result of:

- "Tracking" the gantry axes
- Loss of the reference position of a gantry axis, e.g. by "Parking" (no measuring system active)
- Re-referencing of gantry axis
- The gantry grouping is opened (dissolved) by:
  
  \[ \text{MD37140 $MA\_GANTRY\_BREAK\_UP = 0$} \] (invalidate gantry axis grouping)

The corresponding NC/PLC interface signal is reset:

- \[ \text{DB31, ... DBX60.4 or DBX60.5 == 0} \] (referenced/synchronized 1 or 2 respectively)
- \[ \text{DB31, ... DBX 101.5 == 0} \] (gantry grouping is synchronized)

If, in operation, gantry grouping synchronization is lost due to a fault, then synchronization can be restarted using the NC/PLC interface signal:

\[ \text{DB31, ... DBX29.4 == 1} \] (start gantry synchronization process)

Requirement is that the following applies to all axes of the gantry grouping:

\[ \text{DB31, ... DBX60.4 or DBX60.5 == 1} \] (referenced/synchronized 1 or 2 respectively)

In this case, the synchronizing axes traverse the current actual position of the leading axis in the decoupled state.

If, when the gantry grouping is traversing, the signal "Emergency Stop" (DB10, DBX56.2) is set and again reset, and the gantry axes have drifted apart less than the standstill tolerance of the synchronous axes, then these are automatically resynchronized. Automatic synchronization can be suppressed using the NC/PLC interface signal for the leading axis:

\[ \text{DB31, ... DBX29.5} = 1 \] (no automatic synchronization process)
Selecting the reference point

To ensure that the shortest possible paths are traversed when the gantry axes are referenced, the reference point values from leading and synchronous axes should be the same in the machine data:

MD34100 $MA_REFP_SET_POS (reference point value/destination point for distance-coded system)

Allowance for deviations in distance between the zero mark and the reference point must be made for specific axes via the machine data:

MD34080 $MA_REFP_MOVE_DIST (reference point distance)
MD34090 $MA_REFP_MOVE_DIST_CORR (reference point offset/absolute offset)

Referencing direction selection

The zero mark search direction of the synchronous axis can be defined via the machine data:

MD37150 $MA_GANTRY_FUNCTION_MASK, Bit 1

| Bit | Value | Meaning |
|-----|-------|---------|
| 1   | 0     | Zero mark search direction of the synchronous axis analog to machine data: MD34010 $MA_REFP_CAM_DIR_IS_MINUS |
|     | 1     | Zero mark search direction of the synchronous axis the same as the leading axis |

During referencing, the reference point value of the leading axis is specified as the target position for all axes in the grouping for the synchronization compensatory motion. This position is then approached without axis coupling. The absolute encoders and distance-coded encoders of the leading axis will be set to the current actual position of the leading axis or to the reference point by the following machine data:

MD34330 $MA_REFP_STOP_AT_ABS_MARKER (Distance-coded linear measuring system without destination point)

If only one reference cam is used for the leading and synchronous axes, then this must be taken into account in the PLC user program.

2.3.2 Automatic synchronization

Automatic synchronization can take place:

- In referencing mode (see Section "Introduction (Page 152)"
- in other modes, as described below:

If a gantry grouping is switched to follow-up mode, monitoring of the actual values between the leading and synchronized axes is disabled. The grouping is no longer synchronized as a result. Independent of axes positions, the following interface signal will be set to 0 (from leading axis)

DB31, ... DBX101.5 (gantry grouping is synchronous)
If the gantry grouping is switched from follow-up mode to position control mode, axis synchronism is automatically restored provided the actual-value monitor does not detect a difference between the positions of the leading and synchronized axes greater than the setting in the machine data:

MD36030 $MA_STANDSTILL_POS_TOL (standstill tolerance)

In this case, a new setpoint is specified for the synchronized axis (axes) without interpolation. The positional difference detected earlier is then corrected by the position controller. The correction causes only the synchronized axis (axes) to move.

The motional sequence of the synchronized axis (axes) is analogous to the situation in which the grouping switches from the "Hold" state to position control mode. In this case, the position specified by the position controller before the grouping is halted is set again on condition that the zero speed monitor has not activated alarm 25040 (with follow-up as alarm reaction) in the meantime.

The same tolerance window is used for this mode of automatic synchronization as for the zero speed monitoring function:

MD36030 $MA_STANDSTILL_POS_TOL (standstill tolerance)

Parameter rate dependence loads with machine data:

MD36012 $MA_STOP_LIMIT_FACTOR (exact stop coarse/fine and standstill factor)

**Note**

The following interface signal blocks automatic synchronization in all modes except referencing mode:

DB31, ... DBX29.5 (no automatic synchronization)

Should the automatic synchronization be activated at this point, then the following interface signal must be reset:

DB31, ... DBX29.5 = 0 (no automatic synchronization)

Then switch one of the axes in the gantry grouping from follow-up mode to position-controlled mode. This is achieved with the interface signals:

DB31, ... DBX1.4 = 1 (follow-up mode)

DB31, ... DBX2.1 = 1 (servo enable)

### 2.3.3 Points to note

#### 2. Position measuring systems per gantry axis

Different types of position measuring systems can be mounted on the gantry axes of a grouping. Furthermore, each gantry axis is capable of processing two position measuring systems, it being possible to switch over from one system to the other at any time:

DB31, ... DBX1.5 (position measuring system 1)

DB31, ... DBX1.6 (position measuring system 2)
The maximum tolerance for position actual value switchover should be set to a lower value than the gantry warning limit:

MD36500 $MA_ENC_CHANGE_TOL (Max. tolerance for position actual value switchover)

The two position measuring systems must, however, have been referenced beforehand. The relevant measuring system must be selected before referencing is initiated. The operational sequence is then the same as that described above.

**Channelspecific referencing**

Gantry axes can also be referenced by channel with the following interface signal:

DB21, ... DBX1.0 (activate referencing)

The value of the leading axis’ machine data is used for the axis sequence for channel-specific referencing:

MD34110 $MA_REFP_CYCLE_NR (Axis sequence for channel-specific referencing)

After the reference point of the leading axis has been reached, the synchronized axes are referenced first as described above.

**Referencing from part program with G74**

The referencing and synchronization process for gantry axes can also be initiated from the part program by means of command G74. In this case, only the axis name of the leading axis may be programmed. The operational sequence is analogous to that described for axis-specific referencing.

**Position measuring system with distancecoded reference marks**

In order that return traverses do not have to be made over large distances, it is possible to use a position measuring system with distance-coded reference marks as a sole or second measuring system for gantry axes. In this way the measuring system is referenced after traversal of a short path (e.g. 20 mm). The procedure for referencing the gantry axes is the same as that described for the normal incremental measuring system.

**References:**

Function Manual Basic Functions; Reference Point Travel (R1)

**Absolute encoder**

During the course of the synchronization compensatory motion, all axes in the gantry axis grouping traverse to the reference point value of the leading axis defined in the machine data:

MD34100 $MA_REFP_SET_POS (reference point value/destination point for distancecoded system)

The absolute encoders and distance-coded encoders of the leading axis will be set to the current actual position of the leading axis or to the reference point by the following machine data:

MD34330 $MA_REFP_STOP_AT_ABS_MARKER (Distancecoded linear measuring system without destination point)
2.3 Referencing and synchronization of gantry axes

Activation of axis compensations

Compensation functions can be activated for both the leading axis and the synchronized axes. Compensation values are applied separately for each individual gantry axis. These values must therefore be defined and entered for the leading axis and the synchronized axes during start-up.

The compensations do not become operative internally in the control until the axis is referenced or the gantry grouping synchronized. The following applies:

| Compensation type       | Takes effect when                      | PLC interface signal            |
|-------------------------|----------------------------------------|----------------------------------|
| Backlash compensation   | Axis is referenced                     | "Referenced/Synchronized"        |
| LEC                     | Axis is referenced                     | "Referenced/synchronized"        |
| Sag compensation        | Gantry grouping is synchronized        | "Gantry grouping is synchronized"|
| Temperature compensation| Gantry grouping is synchronized        | "Gantry grouping is synchronized"|

If a movement by the synchronized axis (axes) is caused by an active compensation, a travel command is displayed for the synchronized axis (axes) independently of the leading axis.

Monitoring functions effective

Analogous to normal NC axes, the following monitoring functions do not take effect for gantry axes until the reference point is reached (IS "Referenced/Synchronized"):  
- Working area limits
- Software limit switch
- Protection zones

The axial machine data values are used as monitoring limit values for the synchronized axes as well.

Multi-channel block search

The cross-channel block search in Program Test mode (SERUPRO "Search Run by Program test") can be used to simulate the traversal of gantry axis groupings.

Note

Further information regarding multi-channel block search SERUPRO can be found under:  
References:  
Function Manual, Basic Functions; Mode Group, Channel, Program Mode (K1),  
Section: Program test
2.4 Start-up of gantry axes

General

Owing to the forced coupling which is normally present between guide and synchronous gantry axes, the gantry grouping must be started up as if it were an axis unit. For this reason, the axial machine data for the guide and synchronous axes must always be defined and entered jointly.

If the synchronous axis is overloaded due to a weaker dynamic response than the guide axis, alarm 10656 is displayed.

Special points to be noted with regard to starting up gantry axes are described below.

Traversing direction

As part of the start-up procedure, a check must be made to ensure that the direction of rotation of the motor corresponds to the desired traversing direction of the axis. Correct with the following axial machine data:

MD32100 $MA_AX_MOTION_DIR (traversing direction)

Entering gantry trip limits

For the monitoring of the actual position values of the synchronous axis in relation to the actual position of the guide axis, the limit values for termination, as well as for the guide and synchronous axes, should be entered corresponding to the specifications of the machine manufacturer:

- MD37120 $MA_GANTRY_POS_TOL_ERROR (gantry trip limit)
- MD37130 $MA_GANTRY_POS_TOL_REF (gantry trip limit for referencing)

Note

The control must then be switched off and then on again because the gantry axis definition and the trip limit values only take effect after power ON.

Response to setpoint changes and disturbances

Since digital drives respond well to disturbances and setpoint changes, there is no need for a compensatory control between the gantry axes. However, the gantry axes can only operate in exact synchronism if the parameters for the control circuits of the guide and synchronous axes are set to the same dynamic response value.

To ensure the best possible synchronism, the guide axis and synchronous axis must be capable of the same dynamic response to setpoint changes. The axial control loops (position, speed and current controllers) should each be set to the optimum value so that disturbances can be eliminated as quickly and efficiently as possible. The dynamic response adaptation function in the setpoint branch is provided to allow differing dynamic responses of axes to be matched without loss of control quality.
Axial optimization

The following control parameters must be set to the optimum axial value for both the guide axis and the synchronous axis:

- MD32200 $MA_POSCTRL_GAIN (servo gain factor)
- MD32620 $MA_FFW_MODE (feedforward control parameter)
- MD32610 $MA_VELO_FFW_WEIGHT (feedforward control factor for acceleration/speed)
- MD32650 $MA_AX_INERTIA (inertia for torque feedforward control)
- MD32800 $MA_EQUIV_CURRCTRL_TIME (equivalent time constant current control loop for feedforward control)
- MD32810 $MA_EQUIV_SPEEDCTRL_TIME (equivalent time constant speed control loop for feedforward control)

References

Extended Functions Function Manual; Compensations (K3)

Same settings

The following control parameters must be set to the same value for the guide axis and synchronous axis:

- MD33000 $MA_FIPO_TYPE (fine interpolator type)
- MD32400 $MA_AX_JERK_ENABLE (axial jerk limitation)
- MD32410 $MA_AX_JERK_TIME (time constant for the axial jerk filter)
- MD32420 $MA_JOG_AND_POS_JERK_ENABLE (basic position of axial jerk limitation)
- MD32430 $MA_JOG_AND_POS_MAX_JERK (axial jerk)

References

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

Dynamics matching

The guide axis and the coupled synchronous axis must be capable of the same dynamic response to setpoint changes. The same dynamic response means: The following errors are equal in magnitude when the axes are operating at the same speed.

The dynamic response adaptation function in the setpoint branch makes it possible to obtain an excellent match in the response to setpoint changes between axes which have different dynamic characteristics (control loops). The difference in equivalent time constants between the dynamically "weakest" axis and the other axis in each case must be specified as the dynamic response adaptation time constant.
Example

When the speed feedforward control is active, the dynamic response is primarily determined by the equivalent time constant of the "slowest" speed control loop.

Guide axis

MD32810 $MA_EQUIV_SPEEDCTRL_TIME [n] = 5 ms (equivalent time constant speed control loop for feedforward control)

Synchronous axis

MD32810 $MA_EQUIV_SPEEDCTRL_TIME [n] = 3 ms

- Time constant of dynamic response adaptation for synchronous axis:
  
  MD32910 $MA_DYN_MATCH_TIME [n] = 5 ms - 3 ms = 2 ms (time constant of dynamic response adaptation)

The dynamic response adaptation must be activated axially with the machine data:

MD32900 $MA_DYN_MATCH_ENABLE (dynamic response adaptation)

Check of dynamic response adaptation:

The following errors of the guide and synchronous axes must be equal in magnitude when the axes are operating at the same speed!

For the purpose of fine tuning, it may be necessary to adjust servo gain factors or feedforward control parameters slightly to achieve an optimum result.

Referencing gantry axes

The positions of the guide and synchronous axes reference points must first be set to almost identical values.

To ensure that the synchronization compensatory motion of the gantry axes is not started automatically, the gantry warning limit must be set to 0 at the first start-up before referencing:

MD37100 $MA_GANTRY_POS_TOL (gantry axis definition)

This will prevent a warning message being output during traversing motion.

In cases where an excessively high additional torque is acting on the drives due to misalignment between the guide and synchronous axes, the gantry grouping must be aligned before the axes are traversed. After this, the gantry axes must be referenced. For further details, see:

- Section "Referencing and synchronization of gantry axes (Page 152)"

References: Basic Functions Function Manual; Reference Point Approach (R1)

After the guide and synchronous axes have been referenced, the difference between them must be determined by comparing the actual position value (HMI: Operating area "Diagnostics" > "Service axes") and taken into account as the reference point offset:

- MD34080 $MA_REFP_MOVE_DIST (reference point distance)
- MD34090 $MA_REFP_MOVE_DIST_CORR (reference point offset / absolute offset)
The differences in distance between the zero mark and reference point must also be calculated for each gantry axis and adjusted. They are to be customized, via the following machine data, in such a way that the actual position values of the guide and synchronous axes are identical after execution of the compensatory motion:

- MD34080 $MA_REFP_MOVE_DIST (reference point distance)
- MD34090 $MA_REFP_MOVE_DIST_CORR (reference point offset / absolute offset)

**Synchronizing gantry axes**

The gantry synchronization is activated via the NC/PLC interface signal (see Section "Referencing and synchronization of gantry axes (Page 150)"):

DB31, ... DBX29.4 = 1 (start synchronization of gantry)

The completion of the synchronization is displayed via the NC/PLC interface signal:

DB31, ... DBX101.5 == 1 (gantry grouping is synchronous)

Once the axes have been synchronized, check that the dimensional offset between the guide and the synchronous axes is 0. If required, make corrections in the machine data mentioned above.

**Input of gantry warning limit**

Once the reference point values for the guide and synchronous axes have been optimized so that the gantry axes are perfectly aligned with one another after synchronization, the warning limit values for all axes must be entered in the following machine data:

MD37110 $MA_GANTRY_POS_TOL_WARNING (gantry warning limit)

To do this, the value must be increased incrementally until it is just below the alarm (limit exceeded) response limit. It is particularly important to check the acceleration phases.

This limit value also determines the position deviation value at which gantry synchronization is automatically started in the control.

**Calculating and activating compensations**

In cases where the gantry axes require compensation (backlash, sag, temperature or leadscrew error), the compensation values for the guide axis and the synchronous axis must be calculated and entered in the appropriate parameters or tables.

**References**

Extended Functions Function Manual; Compensations (K3)
2.4 Start-up of gantry axes

Function generator / measuring function

The activation of the function generator and measuring function for a synchronous axis is aborted with an error message. If an activation of the synchronous axis is absolutely necessary, e.g. to measure the machine, the guide and the synchronous axes must be temporarily interchanged.

Special cases

If individual axes have to be activated, the gantry groups must be temporarily canceled. As the second axis no longer travels in synchronism with the first axis, the activated axis must not be allowed to traverse beyond the positional tolerance.

If the gantry grouping is canceled, the following points must be noted:

- Always activate the traversing range limits and set them to the lowest possible values (position tolerance).
- Synchronize the gantry grouping first if possible and then execute a POWER-ON-RESET without referencing the axes again. This ensures that the traversing range limits always refer to the same position (i.e. that which was valid on power ON).
- Avoid using the step-change function. Position step changes are only permissible if they stay within the permitted tolerance.
- Always use an offset of 0 for the function generator and measuring function in contrast to the recommendations for normal axes.
- Set the amplitudes for function generator and measuring function to such low values that the activated axis traverses a shorter distance than the position tolerance allows. Always activate the traversing range limits as a check (see above).

References

Drive Functions Function Manual; Speed Control Loop (DD2)

Start-up support for gantry groupings

The start-up functions of the function generator and measuring are parameterized via the PI service. The traversing movement starts for all parameterized axes with NC start in JOG mode.

A window is displayed in the "Measuring function and function generator in gantry grouping" user interface. Two amplitude values, each with an offset and bandwidth, must be entered in this window. The first amplitude value applies to the measuring axis and the second to the other coupled axes.
2.5 Parameter assignment: Response to faults

Pulse suppression

The behavior of the gantry grouping with regard to faults that trigger pulse suppression can be set with the following axis-specific machine data:

MD30455 $MA_MISC_FUNCTION_MASK, bit 9 = <value>

| <value> | Meaning |
|---------|---------|
| 0       | When a fault occurs that triggers the pulse suppression (e.g. measuring-circuit fault), the pulses in all other axes of the gantry grouping will also be suppressed. Result: Coast down of all axes of the gantry grouping. |
| 1       | When a fault occurs that triggers the pulse suppression (e.g. measuring-circuit fault), the pulses for only this axis will be suppressed. The other axes of the gantry grouping will be stopped. The pulses for these axes are also then suppressed. To ensure that the pulses these axes are not suppressed in advance, especially for axes to be held at the standstill position (vertical axes), the following drive parameters must be taken into account:  
  - p1135[0...n] OFF3 ramp-down time  
  - p1217 holding brake application time  
  - p1227 zero speed detection monitoring time  
  - p1228 pulse suppression, delay time  
For a detailed description of drive parameters, refer to:  
References:  
SINAMICS S120/S150 Parameter Manual |
2.6 PLC interface signals for gantry axes

Special IS for gantry axes

The special NC/PLC interface signals of the coupled gantry axes are taken via the axial NC/PLC interface of the leading or synchronized axes. The table below shows all special gantry NC/PLC interface signals along with their codes and indicates whether the IS is evaluated on the leading axis or the synchronized axis.

| NC/PLC interface signal                  | Direction of transfer | DB31, ... DBX... | Leading axis | Synchronous axis |
|------------------------------------------|-----------------------|------------------|--------------|------------------|
| Start gantry synchronization             | PLC → NCK             | 29.4             | X            |                  |
| No automatic synchronization             | PLC → NCK             | 29.5             | X            |                  |
| Gantry axis                              | NCK → PLC             | 101.7            | 1            | 1                |
| Gantry leading axis                      | NCK → PLC             | 101.6            | 1            | 0                |
| Gantry grouping is synchronized          | NCK → PLC             | 101.5            | X            |                  |
| Gantry synchronization ready to start    | NCK → PLC             | 101.4            | X            |                  |
| Gantry warning limit exceeded            | NCK → PLC             | 101.3            |               | X                |
| Gantry trip limit exceeded               | NCK → PLC             | 101.2            |               | X                |

Effect of axial interface signals on gantry axes

a) Axial interface signals from PLC to axis (PLC → NCK)

The axial interface signals from the PLC to the axis are always referred to all gantry axes in the grouping. In this case, all gantry axes (leading and synchronized axis) have equal priority.

For example, all axes in the gantry groupings will be simultaneously shut down when the following interface signal is set to "0" from the leading axis:

DB31, ... DBX2.1 (servo enable)
The following table shows the effect of individual interface signals (from PLC to axis) on gantry axes:

| NC/PLC interface signal                          | DB31, ... DBX ... | Effect on                                      |
|-------------------------------------------------|-------------------|------------------------------------------------|
| Axis/spindle disable                            | 1.3               | On all axes in gantry grouping                 |
| Position measuring system 1/2                   | 1.5 and 1.6       | Axial 1)                                       |
| Controller enable                               | 2.1               | On all axes in gantry grouping 2)             |
| Delete distance to go (axial)                   | 2.2               | Axial                                          |
| Clamping in progress                            | 2.3               | Axial                                          |
| Reference point value 1-4                       | 2.4 - 2.7         | Axial                                          |
| Feed stop                                       | 4.3               | On all axes in gantry grouping                 |
| Hardware limit switch plus/minus                | 12.0 and 12.1     | Axial alarm: Brake request on all axes in gantry grouping |
| 2. Hardware limit switch plus / minus           | 12.2 and 12.3     | Axial                                          |
| Ramp-function generator fast stop (RFGFS)       | 20.1              | On all axes in gantry grouping                 |
| Select drive parameter set                      | 21.0 - 21.2       | Axial                                          |
| Enable Pulses                                   | 21.7              | Axial                                          |

1) DB31, ... DBX1.5 and 1.6 (position measuring system 1/2)

The switchover between position measuring systems 1 and 2 applies individually for each gantry axis. However, deactivation of both position measuring systems (known as the parking position) applies as a common signal for all gantry axes.

2) DB31, ... DBX2.1 (controller enable)

If the servo enable signal on one gantry axis is canceled, all axes in the gantry grouping are shut down simultaneously. The method by which shutdown is implemented (e.g. with fast stop) is identical for all gantry axes.

Either the "Follow-up" state (IS of one gantry axis = 1) or the "Stop" state (IS of all gantry axes = 0) is activated for all gantry axes, depending on interface signal:

DB31, ... DBX1.4 (follow-up mode)

b) Axial interface signals from axis to PLC (NCK → PLC)

Each of the axial, axis-to-PLC interface signals for the synchronized axis and the leading axis is always set on an axis-specific basis and output to the PLC.

Example:

DB31, ... DBX60.4 resp. 60.5 (referenced/synchronized 1/2).

Exception:

When the leading axis is traversed, the interface signal will also be set for the synchronizing axis:

DB31, ... DBX64.6 and 64.7 (traverse command plus resp. minus)
2.7 Miscellaneous points regarding gantry axes

Manual travel
It is not possible to traverse a synchronized axis directly by hand in JOG mode. Traverse commands entered via the traversing keys of the synchronized axis are ignored internally in the control. Rotation of the handwheel for the synchronized axis has no effect either.

Handwheel override
An overriding motion by means of the handwheel can only be applied to the leading axis in coupled axis mode. In this case, the synchronized axes traverse in synchronism with the leading axis.

DRF offset
A DRF offset can only be applied to the leading axis. In this case, the synchronized axes traverse in synchronism with the leading axis.

Programming in part program
Only the leading axis of a gantry axis grouping may be programmed in the part program. An alarm is generated while programming a synchronized axis, even when a gantry axis grouping is released (MD37140 $MA_GANTRY_BREAK_UP = 1).

PLC or command axes
Only the leading axis of the gantry grouping can be traversed by the PLC using FC 18 or as a command axis by means of synchronized actions.

References:
- Function Manual, Basic Functions, Basic PLC Program (P3)
- Function Manual, Synchronized Actions

PRESET
The PRESET function can only be applied to the leading axis. All axes in the gantry grouping are reevaluated internally in the control when PRESET is activated. The gantry axis then lose their reference and synchronization:

DB31, ... DBX101.5 (gantry grouping is synchronized) = 0

Channel assignment of the gantry axes
Please ensure that for a gantry grouping whose leading axis is known in several channels, its synchronized axes in these channels are also known. If this is not the case, Alarm 10651 is output with reason 60XX (XX is the objectionable gantry grouping).
Axis replacement

All axes in the gantry grouping are released automatically in response to a RELEASE command (leading axis).

A replacement of the leading axis of a closed gantry grouping is only possible, if all axes of the grouping are known in the channel in which they are to be transferred, otherwise alarm 10658 is signaled.

No automatic axis change and no automatic adjustment of the gantry axis conditions are undertaken while trying to reconnect a gantry grouping is released with MD37140 $MA_GANTRY_BREAK_UP = 1. The user is responsible for this. A check of the axis conditions is conducted after the break up and if necessary, a corresponding alarm 10658 is output.

Note

If a gantry grouping is to be closed again, the user must ensure that all axes of the grouping are in a channel with a corresponding axis condition.

Default for RESET

In an active gantry grouping, the following machine data parameterization is ignored for the synchronized axes:

MD30450 $MA_IS_CONCURRENT_POS_AX = 1 (Reset default: neutral axis/channel axis)

The state of the leading axis is assumed. The user is informed about the inappropriate configuration with display alarm 4300.

Display data

The position actual value display shows the actual values of both the leading axis and the synchronized axes. The same applies to the service display values in the "Diagnosis" operating area.

software limit switch

The SW limit switch monitor is processed for the leading axis only. If the leading axis crosses the limit switch, all axes in the gantry grouping are braked to a standstill.
Differences in comparison with the "Coupled motion" function

The main differences between the "gantry axes" and "coupled motion" functions are listed below:

- The axis coupling between the gantry axes must always be active. Separation of the axis coupling via part program is therefore not possible for gantry axes. In contrast, the coupled axis grouping can be separated by means of the part program and the axes then traversed individually.

- In the "Gantry Axes" function, the difference of the actual position values from the leading and synchronized axis is monitored continuously and the traversing motion is shut down if there are impermissible deviations. There is no monitoring for the "Coupled motion" function.

- Gantry axes must remain coupled even during referencing. For this reason, special procedures are applied for the reference point approach of gantry axes. In contrast, coupled-motion axes are referenced as individual axes.

- For the gantry axes to traverse without mechanical offset, the synchronized axes must be set like the leading axes from the control dynamics perspective. In contrast, the "coupled motion" function permits axes with different dynamic control response characteristics to be coupled.

References:
Function Manual, Basic Functions, Reference Point Approach (M3)

Block search with active coupling

Note
For an active coupling, it is recommended to only use block search type 5, "Block search via program test" (SERUPRO) for a block search.
2.8 Examples

2.8.1 Creating a gantry grouping

Introduction

The gantry grouping, the referencing of its axes, the orientation of possible offsets and, finally, the synchronization of the axes involved are complicated procedures. The individual steps involved in the process are explained below by an example constellation.

Constellation

Machine axis 1 = gantry leading axis, incremental measuring system
Machine axis 3 = gantry synchronized axis, incremental measuring system

Machine data

The following machine data describes the original values at the beginning of the procedure. Individual settings must be corrected or added later according to the information below.

Gantry machine data

Axis 1

MD37100 $MA_GANTRY_AXIS_TYPE = 1 (gantry axis definition)
MD37110 $MA_GANTRY_POS_TOL_WARNING = 0 (gantry warning limit)
MD37120 $MA_GANTRY_POS_TOL_ERROR = e.g. 1 (gantry trip limit)
MD37130 $MA_GANTRY_POS_TOL_REF = e.g. 100 mm (max. misalignment) (gantry trip limit for referencing)
MD37140 $MA_GANTRY_BREAK_UP = 0 (invalidate gantry axis grouping)

Axis 3

MD37100 $MA_GANTRY_AXIS_TYPE = 11
MD37110 $MA_GANTRY_POS_TOL_WARNING = 0
MD37120 $MA_GANTRY_POS_TOL_ERROR = e.g. 1 mm
MD37130 $MA_GANTRY_POS_TOL_REF = e.g. 100 mm (max. misalignment)
MD37140 $MA_GANTRY_BREAK_UP = 0
Reference point machine data (for first encoder each)

Axis 1
MD34000 $MA_REFP_CAM_IS_ACTIVE = TRUE
MD34010 $MA_REFP_CAM_DIR_IS_MINUS = e.g. FALSE
MD34020 $MA_REFP_VELO_SEARCH_CAM =
MD34030 $MA_REFP_MAX_CAM_DIST = corresponds to max. distance traversed
MD34040 $MA_REFP_VELO_SEARCH_MARKER =
MD34050 $MA_REFP_SEARCH_MARKER_REVERSE = e.g. FALSE
MD34060 $MA_REFP_MAX_MARKER_DIST = Difference betw. cam edge and 0 mark
MD34070 $MA_REFP_VELO_POS =
MD34080 $MA_REFP_MOVE_DIST = 0
MD34090 $MA_REFP_MOVE_DIST_CORR = 0
MD34092 $MA_REFP_CAM_SHIFT = 0
MD34100 $MA_REFP_SET_POS = 0
MD34200 $MA_ENC_REFP_MODE = 1

The reference point machine data (for the first encoder) of axis 3 must be specified analogously.
2.8.2 Setting of NCK PLC interface

Introduction

An automatic synchronization process during axis referencing must be disabled initially so as to prevent any damage to grouping axes that are misaligned.

Disabling of automatic synchronization

The PLC user program sets the following for the axis data block of axis 1:

DB31, ... DBX29.4 = 0 (do not start gantry synchronization)
DB31, ... DBX29.5 = 1 (no automatic synchronization process)

The NCK sets the following as a confirmation in the axis block of axis 1:

DB31, ... DBB101.4 = 0 (synchronization process not ready to start)
DB31, ... DBB101.6 = 1 (leading axis LA)
DB31, ... DBB101.7 = 1 (gantry axis)

The PLC user program sets for the axis data block of axis 3:

DB31, ... DBX29.4 = 0 (do not start gantry synchronization)

The NCK sets the following as a confirmation in the axis block of axis 3:

DB31, ... DBB101.4 = 0 (synchronization process not ready to start)
DB31, ... DBB101.6 = 0 (synchronized axis GA)
DB31, ... DBB101.7 = 1 (gantry axis)
2.8 Examples

2.8.3 Commencing start-up

Referencing

The following steps must be taken:

- Select "REF" operating mode
- Start referencing for axis 1 (master axis)
- Wait until message "10654 Channel 1 Waiting for synchronization start" appears.

At this point in time, the NCK has prepared axis 1 for synchronization and registers this to the interface signal:

- DB31, DB31, ... DBB101.4 = 1 (synchronization process ready to start)
- DB31, ... DBB101.6 = 1 (leading axis LA)
- DB31, ... DBB101.7 = 1 (gantry axis)

In addition, the following steps must be taken:

- RESET
- Read off values in machine coordinate system:
  e.g.
  \[ X = 0.941 \]
  \[ Y = 0.000 \]
  \[ XF = 0.000 \]
- Enter the X value of master axis 1 with inverted sign in the machine data of slave axis 3:
  \[ MD34090 \text{ $MA\_REFP\_MOVE\_DIST\_CORR = -0.941$ (reference point offset/absolute offset)} \]

Note

This machine data is effective after power ON. To avoid having to perform a power ON now, the value can also be entered in the following machine data:

\[ MD34080 \text{ $MA\_REFP\_MOVE\_DIST$ (reference point distance)} \]

This machine data is then valid after a RESET.

- Start referencing again for axis 1 (master axis) with the modified machine data
• Wait until message "10654 Channel 1 Waiting for synchronization start" appears
• At this point in time, the NCK has prepared axis 1 for synchronization and registers this to
  the interface signal:

  DB31, ... DBB101.4 = 1 (synchronization process ready to start)
  DB31, ... DBB101.6 = 1 (leading axis LA)
  DB31, ... DBB101.7 = 1 (gantry axis)

• Examine actual positions of machine. Case A or B might apply:

![Diagram](image)

Figure 2-3 Possible results after referencing of axis 1 (master axis)

If Case A applies, the synchronization process can be started immediately (see "start
synchronization process" step). If Case B applies, the offset "diff" must be calculated and
taken into account:
• Measuring of diff
• By using two appropriate, right-angled reference points R' and R" in the machine bed
  (right in picture), the difference in position in JOG can be traversed. The diff offset can
  then be read as the difference in the position display. The diff offset must be entered in
  the machine data of axis 3 (synchronized axis):

  MD34100 $MA_REFP_SET_POS

  Continue with Step 1 (see above).

• Start gantry synchronization. PLC sets:

  DB31, ... DBX29.4 = 1 (start synchronization of gantry)
2.8.4 Setting warning and trip limits

If the gantry grouping is set and synchronized, the following machine data must then be set:

MD37110 $MA_GANTRY_POS_TOL_WARNING (gantry warning limit)
MD37120 $MA_GANTRY_POS_TOL_ERROR (gantry trip limit)

Proceed as follows:

- Set the machine data for all axes with a large value to begin with:
  MD37120 $MA_GANTRY_POS_TOL_ERROR (gantry trip limit)
- Set a very small value in the machine data:
  MD37110 $MA_GANTRY_POS_TOL_WARNING (gantry warning limit)
  When you put a heavy, dynamic strain on the axes, the self-canceling alarm "10652 channel %1 axis %2 gantry warning limit exceeded" will be output repeatedly.
- Now increase the following machine data:
  MD37110 $MA_GANTRY_POS_TOL_WARNING (gantry warning limit)
  Repeat this until the alarm no longer appears. The interface indicates the status:
  DB31, ... DBB102.2 = 0 (trip limit not exceeded)
  DB31, ... DBB102.3 = 0 (warning limit not exceeded)
  DB31, ... DBB102.4 = 0 (synchronization process not ready to start)
  DB31, ... DBB102.5 = 1 (gantry grouping is synchronous)
  DB31, ... DBB102.6 = 1 (leading axis LA)
  DB31, ... DBB102.7 = 1 (gantry axis)

If the monitoring still only triggers very sporadically, it is possible to program a edge memory bit in the PLC user program.
- Enter the value calculated for the warning limit + a small safety provision in the following machine data:
  MD37120 $MA_GANTRY_POS_TOL_ERROR (gantry trip limit)
Error limit values

Values are entered in the following machine data:

MD37110 $MA_GANTRY_POS_TOL_WARNING (gantry warning limit)
MD37120 $MA_GANTRY_POS_TOL_ERROR (gantry trip limit)
MD37130 $MA_GANTRY_POS_TOL_REF (gantry trip limit for referencing)

These should have the following scales of magnitude at the end of the customizing process:

Note

The same procedure must be followed when starting up a gantry grouping in which the coupled axes are driven by linear motors and associated measuring systems.

The error limits entered into machine data MD37110 and MD37120 are considered as additional tolerance values of the actual-value difference of the master and following axis if the IS "Gantry is synchronous" is not present (e.g. to be resynchronized after canceling alarms without gantry).
## 2.9 Data lists

### 2.9.1 Machine data

#### 2.9.1.1 Axis/spindlespecific machine data

| Number | Identifier: $MA_ \_\_\_\_ | Description |
|--------|-----------------------------|-------------|
| 30300  | IS_ROT_AX                  | Rotary axis |
| 32200  | POSCTRL_GAIN                | $K_v$ factor |
| 32400  | AX_JERK_ENABLE              | Axial jerk limitation |
| 32410  | AX_JERK_TIME                | Time constant for axis jerk filter |
| 32420  | JOG_AND_POS_JERK_ENABLE     | Initial setting for axial jerk limitation |
| 32610  | JOG_AND_POS_MAX_JERK        | Axial jerk |
| 32620  | FFW_MODE                    | Feedforward control mode |
| 32650  | AX_INERTIA                  | Moment of inertia for torque feedforward control |
| 32800  | EQUIV_CURRCTRL_TIME         | Equivalent time constant, current control loop for feedforward control |
| 32810  | EQUIV_SPEEDCTRL_TIME        | Equivalent time constant, speed control loop for feedforward control |
| 32900  | DYN_MATCH_ENABLE            | Dynamic response adaptation |
| 32910  | DYN_MATCH_TIME              | Time constant for dynamic response adaptation |
| 33000  | FIPO_TYPE                   | Fine interpolator type |
| 34040  | REFP VELO SEARCH_MARKER     | Creep velocity |
| 34070  | REFP VELO_POS               | Reference point start velocity |
| 34080  | REFP MOVE DIST              | Reference point approach distance |
| 34090  | REFP MOVE DIST CORR         | Home position offset |
| 34100  | REFP SET POS                | Reference point value |
| 34330  | REFP STOP AT ABS_MARKER     | Distancecoded linear measuring system without destination point |
| 36012  | STOP LIMIT FACTOR           | Exact stop coarse/fine factor and zero speed |
| 36030  | STANDSTILL_POS_TOL          | Zero speed tolerance |
| 36500  | ENC_CHANGE_TOL              | Maximum tolerance for position actual value switchover |
| 37100  | GANTRY_AXIS_TYPE            | Gantry axis definition |
| 37110  | GANTRY_POS_TOL_WARNING      | Gantry warning limit |
| 37120  | GANTRY_POS_TOL_ERROR        | Gantry trip limit |
| 37130  | GANTRY_POS_TOL_REF          | Gantry trip limit for referencing |
| 37140  | GANTRY_BREAK_UP             | Invalidate gantry axis grouping |
### 2.9.2 Signals

#### 2.9.2.1 Signals from mode group

| Signal name | SINUMERIK 840D sl | SINUMERIK 828D |
|-------------|-------------------|----------------|
| Active machine function REF | DB11, DBX5.2 | DB3100, DBX1.2 |

#### 2.9.2.2 Signals from channel

| Signal name | SINUMERIK 840D sl | SINUMERIK 828D |
|-------------|-------------------|----------------|
| Referencing active | DB21, ... DBX33.0 | DB3300, DBX1.0 |

#### 2.9.2.3 Signals to axis/spindle

| Signal name | SINUMERIK 840D sl | SINUMERIK 828D |
|-------------|-------------------|----------------|
| Start gantry synchronization | DB31, ... DBX29.4 | DB380x, DBX5005.4 |
| No automatic synchronization | DB31, ... DBX29.5 | DB380x, DBX5005.5 |

#### 2.9.2.4 Signals from axis/spindle

| Signal name | SINUMERIK 840D sl | SINUMERIK 828D |
|-------------|-------------------|----------------|
| Referenced/synchronized 1, referenced/synchronized 2 | DB31, ... DBX60.4/5 | DB390x, DBX0.4/5 |
| Gantry trip limit exceeded | DB31, ... DBX101.2 | DB390x, DBX5005.2 |
| Gantry warning limit exceeded | DB31, ... DBX101.3 | DB390x, DBX5005.3 |
| Gantry synchronization ready to start | DB31, ... DBX101.4 | DB390x, DBX5005.4 |
| Gantry grouping is synchronized | DB31, ... DBX101.5 | DB390x, DBX5005.5 |
| Gantry guide axis | DB31, ... DBX101.6 | DB390x, DBX5005.6 |
| Gantry axis | DB31, ... DBX101.7 | DB390x, DBX5005.7 |
G1: Gantry axes

2.9 Data lists
3

K6: Contour tunnel monitoring

3.1 Brief description

3.1.1 Contour tunnel monitoring - 840D sl only

Function

The absolute movement of the tool tip in space is monitored. The function operates channel specific.

Model

A round tunnel with a definable diameter is defined around the programmed path of a machining operation. Axis movements are stopped as an option if the path deviation of the tool tip is greater than the defined tunnel as the result of axis errors.

Response

In the event of a recognized deviation, the system reacts as quick as possible. However, at least one interpolation cycle will pass, before one of the following reactions will occur:

- An alarm is triggered when the tunnel is violated and the axes continue to traverse.
- Violation of the tunnel triggers an alarm and the axis movements are decelerated.

Braking methods

If the monitoring tunnel is violated, one of the following methods can be used to decelerate:

- Deceleration ramp
- Speed setpoint zero and follow-up mode

Use

The function can be used for 2D and 3D paths. For 2D the monitoring surface is defined by parallel lines to the programmed path. The monitoring area is defined by 2 or 3 geometry axis.

Monitoring of synchronized axes, positioning axes, etc. that are not geometry axes is performed directly on the machine axis plane with the "Contour monitoring".
Example

The following figure is a diagram of the monitoring area shown by way of a simple example.

![Diagram of the monitoring area](image)

Figure 3-1  Position of the contour tunnel around the programmed path

As long as the calculated actual position of the tool tip remains inside the sketched tunnel, motion continues in the normal way. If the calculated actual position violates the tunnel, an alarm is triggered (in the default setting) and the axes are stopped by "Ramp Stop". This response to the violation of the tunnel can be disabled (alarm triggered but movement continued) or intensified (rapid stop) by means of a machine data setting.

Analysis

The calculated distance between the programmed path and the actual values can be routed to an analog output to analyze the progression of the contour errors during normal operation (quality control).

3.1.2  Programmable contour accuracy

Function

As an alternative to the function described in "Contour tunnel monitoring", i.e. monitoring of the machining accuracy and stopping machining if excessive deviations occur, another function is offered. With this function, the selected accuracy is always achieved with the path velocity being reduced if necessary. Detailed information about this function can be found under the subject "Programmable contour accuracy".
3.2 Contour tunnel monitoring - 840D sl only

Aim of the monitoring function
The aim of the monitoring function is to stop the movement of the axes if axis deviation causes the distance between the tool tip (actual value) and the programmed path (setpoint) to exceed a defined value (tunnel radius).

Tunnel size
The radius of the contour tunnel being monitored around the programmed path must be defined to implement the monitoring function:

MD21050 $MC_CONTOUR_TUNNEL_TOL (Response threshold for Contour tunnel monitoring)

If the machine data is set to 0.0, monitoring is not performed. The value of the machine data is transferred to the control for new configurations.

Parameterizable deceleration behavior
The deceleration behavior for the monitoring response can be set via the following machine data:

MD21060 $MC_CONTOUR_TUNNEL_REACTION (Reaction upon response of contour tunnel monitoring)

| Value | Meaning                                         |
|-------|------------------------------------------------|
| 0     | Display alarm and continue machining           |
| 1     | Deceleration according to the deceleration ramps (default setting) |
| 2     | Rapid stop (speed setpoint = 0)                |

Encoder switchover
Switching between two encoder systems usually causes a sudden change in the actual position of the tool tip. This change resulting from encoder switchover must not be so large as to cause the tool tip to violate the monitoring tunnel. The radius defined in MD21050 must be higher than the allowed tolerance on actual value switchover:

MD36500 $MA_ENC_CHANGE_TOL (Max. tolerance for position actual value switchover)

Activation
The monitoring will only become active if the following conditions are met:

- MD21050 is higher than 0.0.
- At least two geometry axes have been defined.
Shutting down

Monitoring can be stopped by enabling the machine data setting:

\[ MD21050 = 0.0. \]

Analysis output

The values of deviation of the actual value of the tool tip from the programmed path can – for analysis purposes – be output on a fast analog output (accuracy monitoring).

The assignment of an analog output to output the contour error is programmed in machine data:

\[ MD21070 \text{ $MC\_CONTOUR\_ASSIGN\_FASTOUT$} \]

| Value | Meaning               |
|-------|-----------------------|
| 0     | No output (default setting) |
| 1     | Output to output 1    |
| 2     | Output to output 2    |
| ...   | ...                   |
| 8     | Output to output 8    |

Scale

The tunnel radius set in MD21050 corresponds to a voltage of 10 V at the output.
### 3.3 Programmable contour accuracy

#### Function

The function "Programmable contour accuracy" limits the contour errors caused by control behavior and jerk filter to a specified value by reducing the path velocity on curved contours by the necessary amount. It allows the user to set a compromise between accuracy and productivity of a machining.

#### Note

The "LookAhead" function ensures that the velocity necessary for maintaining the required contour accuracy is not exceeded at any point along the path.

#### Configuration

The mode of operation and parameterization of the function is determined by the machine data:

**MD20470 $MC_CPREC_WITH_FFW** (programmable contour accuracy)

| Value | Meaning |
|-------|---------|
| 0     | The "Programmable contour accuracy" function has no effect when feedforward control is also active. |
| 1     | The "Programmable contour accuracy" function also acts for feedforward control. With active feedforward control, the reduction of the path velocity is calculated on the basis of the effective $K_v$ factor with feedforward control. |
| 2     | Like 1, the function, however, is parameterized with **MD32415 $MA_EQUIV_CPREC_TIME** (time constant for the programmable contour accuracy). The jerk filter is correctly taken into account. The **SD42450 $SC_CONTPREC** setting data determines the permissible contour errors (see "Parameterization"). |
| 3     | Like for 2, but any contour accuracy programmed with CTOL has priority over **SD42450 $SC_CONTPREC**. The jerk filter is correctly taken into account. The programmed CTOL contour tolerance determines the permissible contour errors (see "Parameterization"). **$SC_CONTPREC** is relevant only if CTOL has not be programmed. |

For the MD20470 = 2 or 3 functional versions, the control assumes that there is a jerk filter time constant (**MD32410 $MA_AX_JERK_TIME**) for which the setting of the control section with feedforward control generates a negligible contour error. This value must be entered in the machine data **MD32415 $MA_EQUIV_CPREC_TIME** (see "Parameterization").
To calculate the contour error based on the set jerk filter type (MD32402 $MA_AX_JERK_MODE), the following value is used:

- For active feedforward control, the difference:
  \[ \text{MD32410 } \text{MD}_A \text{X}_JERK\_TIME - \text{MD32415 } \text{MD}_A \text{MA}_EQUV\_CPREC\_TIME \]

- Without feedforward control, the complete value from MD32410 $MA_AX_JERK\_TIME

This procedure allows the commissioning engineer to first change from an initially precise, but possibly excessively hard, setting by increasing the jerk filter time constants to a softer setting with a controlled loss of accuracy.

Supplementary conditions:

- The function does not use the "belt stop" jerk filter type (MD32412 $MA_AX_JERK\_MODE = 3).

- The MD20470 = 2 or 3 functional versions are primarily intended for use with feedforward control. If one of the two functional versions is active for switched off feedforward control, a contour error that results from the \( K_f \) factor is added. This reduces the path velocity significantly faster.

**Note**

The MD20470 = 0 or 1 functional versions are no longer recommended. They only provide compatibility with older software versions.

**Parameterization**

**Contour accuracy**

The maximum contour error for the path of the geometry axes on curved contours is determined by:

- For MD20470 $MC\_CPREC\_WITH\_FFW = 2 with the setting data:
  \[ \text{SD42450 } \text{SC\_CONTPREC} \text{ (contour accuracy)} \]

- For MD20470 $MC\_CPREC\_WITH\_FFW = 3 with the contour tolerance programmed with CTOL.

The smaller the value and the lower the \( K_f \) factor of the geometry axes, the greater the path feedrate on curved contours is lowered.

**Minimum path feedrate**

The user can use the following setting data to specify a minimum path feedrate for the "Programmable contour accuracy" function:

\[ \text{SD42460 } \text{SC\_MINFEED} \text{ (minimum path feed with CPRECON)} \]

The feedrate will not limited below this value, unless a lower \( F \) value has been programmed or the dynamic limitations of the axes force a lower path velocity.
3.3 Programmable contour accuracy

**Time constant for the programmable contour accuracy**

The equivalent time constant for the MD20470 = 2 or 3 functional versions (see "Configuration") is entered in the machine data:

MD32415 $MA_EQUIV_CPREC_TIME (time constant for the programmable contour accuracy)

MD32415 must contain that jerk filter time constant (MD32410 $MA_AX_JERK_TIME) for which the contour error for active feedforward control is negligibly small.

**Programming**

The "programmable contour accuracy" can be activated and deactivated in the part program with the CPRECON and CPRECOF modal G functions.

Example:

| Program code | Comment |
|--------------|---------|
| N10 G0 X0 Y0 |         |
| N20 CPRECON  | ; Activate the "programmable contour accuracy". |
| N30 G1 G64 X100 F10000 | ; Machining with 10 m/min in the continuous-path mode. |
| N40 G3 Y20 J10 | ; Automatic feed limitation in circular block |
| N50 G1 X0 | ; Feedrate again without limitation (10 m/min). |
| ...         |         |
| N100 CPRECOF | ; Deactivate the "programmable contour accuracy". |
| N110 G0 ... |         |

The two CPRECON and CPRECOF modal G functions form the G function group 39 (programmable contour accuracy).

**Behavior for part program start and after reset / part program end**

For part program start and after reset / part program end, the configured control initial setting acts for the G function group 39:

MD20110 $MC_RESET_MODE_MASK (definition of initial control settings after RESET / TP End)

MD20112 $MC_START_MODE_MASK (definition of the basic setting of the control after part program start)
Supplementary conditions

Positioning axes

The function considers only the geometry axes of the path. It does not have any effect of the velocities for the positioning axes.

References

Information about MD32402 $MA_AX_JERK_MODE (filter type for axial jerk limitation) and MD32410 $MA_AX_JERK_TIME (time constant for the axial jerk filter), see:

Function Manual, Basic Functions; Acceleration (B2), Section: "Functions" > "Jerk filter (axis-specific)"

Information about CTOL, see:

Function Manual, Basic Functions; Continuous-Path Mode, Exact Stop, Look Ahead (B1), Section: "Contour/orientation tolerance"
3.4 Constraints

Availability of the "Contour tunnel monitoring" function

The function is an option ("Contour monitoring with tunnel function"), which must be assigned to the hardware via the license management.

Coupled motion

If coupled motion between two geometry axes is programmed with contour tunnel monitoring, this always results in activation of the contour tunnel monitoring. In this case, the contour tunnel monitoring must be switched off before programming the coupled motion:

```
MD21050 $MC_CONTOUR_TUNNEL_TOL = 0.0
```
## 3.5 Data lists

### 3.5.1 Machine data

#### 3.5.1.1 Channelspecific machine data

| Number | Identifier: $MC_\text{__}$ | Description |
|--------|-----------------------------|-------------|
| 20110  | RESET_MODE_MASK             | Determination of basic control settings after Reset / TP End |
| 20112  | START_MODE_MASK             | Definition of the initial control settings after part program start |
| 20470  | CPREC_WITH_FFW              | Programmable contour accuracy |
| 21050  | CONTOUR_TUNNEL_TOL          | Response threshold for contour tunnel monitoring |
| 21060  | CONTOUR_TUNNEL_REACTION     | Reaction to response of contour tunnel monitoring |
| 21070  | CONTOUR_ASSIGN_FASTOUT      | Assignment of an analog output for output of the contour error |

#### 3.5.1.2 Axis/spindlespecific machine data

| Number | Identifier: $MA_\text{__}$ | Description |
|--------|-----------------------------|-------------|
| 32402  | AX_JERK_MODE                | Filter type for axial jerk limitation |
| 32410  | AX_JERK_TIME                | Time constant for axial jerk filter |
| 32415  | EQUIV_CPREC_TIME            | Time constant for the programmable contour accuracy |
| 36500  | ENC_CHANGE_TOL              | Maximum tolerance for position actual value switchover |

### 3.5.2 Setting data

#### 3.5.2.1 Channelspecific setting data

| Number | Identifier: $SC_\text{__}$ | Description |
|--------|-----------------------------|-------------|
| 42450  | CONTPREC                    | Contour accuracy |
| 42460  | MINFEED                     | Minimum path feed for CPRECON |
M3: Coupled axes

4.1 Coupled motion

4.1.1 Brief description

4.1.1.1 Function

The "coupled motion" function enables the definition of simple axis links between a master axis and a slave axis, taking into consideration a coupling factor.

Coupled motion has the following features:

- Any axis of the NC can be defined as a master axis.
- Any axis of the NC can be defined as a coupled axis with a specific coupling factor.
- The master axis and coupled motion axis or axes together form a coupled axis grouping.
- Any number of coupled motion axes can be assigned to a master axis.
- A total of 2 leading axes may be assigned to each coupled motion axis.
- A coupled motion axis can be the master axis of a further coupled axis grouping.
- Traversing movements of the master axis are executed in synchronism on all slave axes based on the coupling factor.
- Coupled motion axes can be moved independently of the master axis while the coupling is active (overlaid movements).
- The master and coupled motion axes of a coupled axis grouping are defined, and the coupling switch on/switch off, by programming instructions in the part program or by synchronized action.
- Coupled motion is also possible in the following manual modes: JOG, JOG REF, JOG INC, etc.
4.1.1.2 Preconditions

Coupled motion function

The coupled motion function forms part of the NCK software.

Generic coupling

The coupled motion functionality is also available in the generic coupling.

However, for basic operation of generic coupling, the following restrictions apply:

- The maximum number of coupled motion groupings is limited to 4.
- Only 1 leading axes may be assigned to each coupled motion axis.
- Cascading is not possible.

Note

These restrictions do not apply when NCK software is supplied with the relevant options of generic coupling (refer to "Preconditions (Page 274)" in the "Brief Description" of Generic Coupling).

4.1.2 General functionality

The "Coupled motion" function allows the definition of simple axis couplings. Coupling is performed from one leading axis to one or more following axes, the so-called coupled motion axes. A separate coupling factor can be specified for each coupled motion axis.

Coupled axis grouping

The leading axis and all the coupled motion axes assigned to it together form a coupled axis grouping. If the leading axis is traversed, all coupled motion axes traverse in accordance with their coupling factors.

A coupled axis grouping can consist of any combination of linear and rotary axes.

Leading axes

Any axis of the NC, including simulated axes, can be used as leading axis.

Coupled axes

Any axis of the NC can be used as coupled motion axis.
Coupling factor

The ratio in which the coupled motion axis moves in relation to the leading axis is specified via the coupling factor.

Coupling factor $K = \text{motion of the coupled motion axis} / \text{motion of the leading axis}$

Negative coupling factors (motion of the coupled motion axis in the opposite direction) are also permitted.

![Diagram of coupled motion axes](image)

Figure 4-1 Application Example: Two-sided machining

**Multiple couplings**

Up to two leading axes can be assigned to one coupled motion axis. The traversing movement of the coupled motion axis then results from the sum of the traversing movements of the leading axes.

**Dependent coupled motion axis**

A coupled motion axis is a "dependent coupled motion axis" when it traverses as a result of a leading axis movement.

**Independent coupled motion axis**

A coupled motion axis is an "independent coupled motion axis" when it traverses as a result of a direct traverse instruction. The traversing movement resulting from the coupled motion axis is then the sum total of the traversing movements as a "dependent" and an "independent" coupled motion axis.
Coupled motion axis as leading axis

A coupled motion axis can at the same time be the leading axis of a further coupled axis grouping.

Coordinate system

Coupled axis motion is always executed in the base coordinate system (BCS).

Switch ON/OFF

Coupled motion can be activated/deactivated via the part programs and synchronous actions. In this context please ensure that the switch on and switch off is undertaken with the same programming:

- Switch on: Part program → Switch off: Part program
- Switch on: Synchronous action → Switch off: Synchronous action

Synchronization on-the-fly

If switch on is performed while the leading axis is in motion, the coupled motion axis is first accelerated to the velocity corresponding to the coupling. The position of the leading axis at the time the velocities of the leading and coupled motion axes are synchronized then serves as the start position for further coupled motion.

Operating modes

Coupled motion is effective in the AUTOMATIC, MDA and JOG modes.
Reference point approach

The following applies for referencing of axes of a coupled axis grouping:

- **Leading axes**
  
  When referencing the leading axis of a coupled axis grouping, the coupling to all coupled motion axes is retained. The coupled motion axes move in synchronism with the leading axis, as a function of their coupling factors.

- **Coupled motion axis: JOG/REF mode**
  
  When referencing a coupled motion axis of a coupled axis grouping, the coupling to the leading axis is cancelled. If the coupling is canceled, the following alarm is displayed:
  
  Alarm 16772 "Channel Channel No. block Block No. Axis Axis No. is following axis, coupling is opened."

  The coupling is not activated again until JOG/REF mode is cancelled.

  The display of this alarm can be suppressed using the following machine data:

  MD11410 $MN_SUPPRESS_ALARM_MASK, Bit 29 = 1 (mask supporting special alarm generation)

  **CAUTION**

  When the coupled motion axis is referenced, the coupling to the leading axis is cancelled. If referencing is executed immediately with the leading axis, i.e. without changing JOG/REF mode, the coupled motion axis does not traverse with the leading axis.

- **Coupled motion axis: Part program command G74**

  It is not possible to reference a coupled motion axis of a coupled axis grouping using the G74 programming instruction.

**Distance-to-go: Coupled motion axis**

The distance-to-go of a coupled motion axis refers to the total residual distance to be traversed from dependent and independent traversing.

**Delete distance-to-go: Coupled motion axis**

Delete distance-to-go for a coupled motion axis only results in aborting of the independent traversing movement of the leading axis.
Response to NC Start

The behavior of the coupled-axis groupings during NC Start depends on the setting in the machine data:

MD20112 $MC_START_MODE_MASK (definition of initial control settings with NC-START)

| Bit | Value | Description                       |
|-----|-------|-----------------------------------|
| 0   | 0     | Coupled-axis groupings are maintained in NC Start. |
| 0   | 1     | Coupled-axis groupings are phased out in NC Start. |

Response to RESET/part program end

The behavior of the coupled-axis groupings during RESET/part program end depends on the setting in the machine data:

MD20110 $MC_RESET_MODE_MASK (definition of initial control settings after RESET/TP-End)

| Bit | Value | Description                                           |
|-----|-------|-------------------------------------------------------|
| 0   | 0     | Coupled-axis groupings are invalidated on RESET / part program end. |
| 0   | 1     | Coupled-axis groupings remain active even beyond RESET/part program end. |

Note

If with NC RESET or end of part program in a channel, the leading axis is not stopped as well (cross-channel coupling, command axis, PLC axis, etc.), the requested RESET cannot be completed.

Because of traversing of the leading axis, the coupled-motion axis is still active for the channel in which the RESET is requested. With suitable actions (NC RESET in the channel of the leading axis, stopping of the command or PLC axis), the leading axis must also be stopped in parallel to the coupled-motion axis.
4.1.3 Programming

4.1.3.1 Definition and switch on of a coupled axis grouping (TRAILON)

Definition and switch on of a coupled axis grouping take place simultaneously with the TRAILON part program command.

Programming

Syntax: \texttt{TRAILON(<coupled motion axis>, <leading axis>, [<coupling factor>])}

Effective: modal

Parameters:

- Coupled motion axis: Type: AXIS
  Range of values: All defined axis and spindle identifiers in the channel

- Leading axis: Type: AXIS
  Range of values: All defined axis and spindle identifiers in the channel

- Coupling factor: The ratio of the traversing movement of the coupled motion axis to the leading axis is specified via the optional coupling factor:
  Coupling factor = \text{Path of the coupled-motion axis}/\text{path of the leading axis}
  A negative coupling factor results in motion in opposite directions for the leading and coupled motion axis.
  Type: REAL
  Range of values: \((2,2 \times 10^{-308} \ldots 1,8 \times 10^{+308})\)
  Default value: +1.0

Example:

\texttt{TRAILON(V,Y,2)} ; Definition and switch on of the coupling of the coupled-motion axis V with leading axis Y. The coupling factor is 2.
4.1.3.2 Switch off (TRAILOF)

Switch off of the coupling of a coupled-motion axis with a leading axis takes place through the TRAILOF part program command.

Programming

**Syntax:**

TRAILON(<coupled motion axis>, <leading axis>)

or (in abbreviated form):

TRAILOF(<coupled-motion axis>)

**Effective:** modal

**Parameters:**

- **Coupled motion axis:** Type: AXIS
  Range of values: All defined axis and spindle identifiers in the channel

- **Leading axis:** Type: AXIS
  Range of values: All defined axis and spindle identifiers in the channel

**Example**

TRAILOF(V,Y) ; Switch off of the coupling between the coupled-motion axis V and leading axis Y.

4.1.4 Effectiveness of PLC interface signals

**Independent coupled motion axis**

All the associated channel and axis specific interface signals of the coupled-motion axis are effective for the independent motion of a coupled-motion axis, e.g.:

- DB21, ... DBX0.3 (Activate DRF)
- DB31, ... DBX0.0 - 0.7 (feed offset)
- DB31, ... DBX1.3 (axis blocking)
- DB31, ... DBX2.1 (control system enable)
- DB31, ... DBX4.0 - 4.2 (activate handwheel)
- DB31, ... DBX4.3 (feed stop)
- ...

Special functions

200 Function Manual, 07/2012, 6FC5397-2BP40-3BA0
This allows the speed to be changed for the independent motion of a coupled motion axis using a feed override or a DRF offset to be defined using the handwheel in AUTOMATIC and MDA modes.

Dependent coupled motion axis

With respect to the motion of a coupled motion axis, which is dependent on the leading axis, only the coupled-motion axis interface signals that effect termination of the motion (e.g. axis-specific feed stop, axis inhibit, control system enable, etc.) are effective.

Leading axis

When a coupled axis grouping is active, the interface signals (IS) of the leading axis are applied to the appropriate coupled motion axis via the axis coupling, i.e.

- A position offset or feed control action of the leading axis is applied via the coupling factor to effect an appropriate position offset or feed control action in the coupled motion axis.
- Shutdown of the leading axis as the result of an interface signal (e.g. axis-specific feed stop, axis inhibit, servo enable, etc.) causes the corresponding coupled motion axis to shut down.

Position measuring system 1/2 (DB31, ... DBX1.5/1.6))

Switch-over of the position measuring system for the leading and coupled motion axes is not inhibited for an active coupled axis grouping. The coupling is not canceled.

**Recommendation:** Switch-over the measuring system when the coupling is deactivated.

Tracking (DB31, ... DBX1.4)

Activation of tracking for an axis is done via the PLC program by setting the following NC/PLC interface signals:

- DB31, ... DBX2.1 = 0 (control system enable)
- DB31, ... DBX1.4 == 1 (tracking mode)

When activating tracking mode for a coupled axis grouping, the specified NC/PLC interface signals must be set simultaneously for all axes (master and slave axes) of the coupled axis group.

If tracking mode is activated for the master axis only, a permanent offset results within the coupling.

Whether and which axis is a leading or a following axis can be seen from the following NC/PLC interface signals and system variables:

- DB31, ... DBX99.0 (leading axis/spindle active)
- DB31, ... DBX99.1 (following axis/spindle active)
- $AA_COUP_ACT[axis identifier] (see: Status of coupling)
4.1.5 Status of coupling

The coupling status of an axis can be determined using the following system variables:

\[ \text{Value} \quad \text{Description} \]
\[
0 \quad \text{No coupling active} \\
1, 2, 3 \quad \text{Tangential tracking} \\
4 \quad \text{Synchronous spindle coupling} \\
8 \quad \text{Coupled motion active} \\
16 \quad \text{Master value coupling} \\
32 \quad \text{Following axis of electronic gearbox} \\
\]

Note

Only one coupling mode may be active at any given time.

4.1.6 Dynamics limit

The dynamics limit is dependent on the type of activation of the coupled axis grouping:

- Activation in part program

  If activation is performed in the part program and all leading axes are active as program axes in the activating channel, the dynamic response of all coupled-motion axes is taken into account during traversing of the leading axes to avoid overloading the coupled-motion axes.

  If activation in part program takes place with leading axes that are not active as program axes in the activating channel ($\text{AA_TYP} \neq 1$), then the dynamics of the coupled motion axes is not considered while traversing the leading axes. This can result in an overload for coupled motion axes with a dynamic response which is less than that required for the coupling.

- Activation in synchronized action

  If activation is performed in a synchronous action, dynamics of the coupled motion axes is not taken into account during traversing of the leading axis. This can result in an overload for coupled motion axes with a dynamic response which is less than that required for the coupling.

  \[ \text{CAUTION} \]
  
  If a coupled motion grouping
  
  - in synchronized actions
  - is activated in the part program with leading axes, that are not program axes in the channel of the coupled motion axes,

  it is the special responsibility of the user/machine manufacturer to provide suitable measures to ensure that an overload of the coupled motion axes does not occur through traversing of the leading axis.
4.1.7 Supplementary conditions

Control system dynamics

It is recommended to align the position control parameters of the leading axis and the coupled motion axis within a coupled axis group.

Note
Alignment of the position control parameters of the leading axis and the coupled motion axis can be performed via a parameter set changeover.

4.1.8 Examples

Application Example: Two-sided machining
Example 1

Example of an NC part program for the axis constellation shown in Fig.:

| Program code       | Comment                                                                 |
|--------------------|-------------------------------------------------------------------------|
| TRAILON(V,Y,1)     | Activation of 1st coupled axis group                                    |
| TRAILON(W,Z,-1)    | Activation of 2nd coupled axis group                                    |
| G0 Z10             | Infeed Z and W axes in opposite axial directions                        |
| G0 Y20             | Infeed of Y and V axes in same axis direction                           |
| G1 Y22 V25         | Superimpose dependent and independent movement of coupled motion axis "V"|
| TRAILOF(V,Y)       | Deactivate 1st coupled axis group                                       |
| TRAILOF(W,Z)       | Deactivate 2nd coupled axis group                                       |

Example 2

The dependent and independent movement components of a coupled motion axis are added together for the coupled motion. The dependent component can be regarded as a co-ordinate offset with reference to the coupled motion axis.

| Program code       | Comment                                                                 |
|--------------------|-------------------------------------------------------------------------|
| N01 G90 G0 X100 U100 | ; Activation of coupled axis group                                      |
| N02 TRAILON(U,X,1) | ; Activation of coupled axis group                                      |
| N03 G1 F2000 X200  | ; Dependent movement of U, Upos=200, UTraill=100                         |
| N04 U201           | ; Independent movement, Upos=U201+UTrail=301                             |
| N05 X250           | ; Dependent movement of U, UTraill=UTrail(100)+50=150, Upos=351          |
| N06 G91 U100       | ; Independent movement of Upos(351)+U100=451                             |
| N07 G90 X0         | ; Dependent movement of U, Upos=Upos(451)-UTrail(250)=201               |
| N10 TRAILOF(U,X)   |                                                          |
4.2 Curve tables - 840D sl only

4.2.1 Product brief

4.2.1.1 Function

The "curve tables" function can be used to define the complex sequence of motions of an axis in a curve table.

Any axis can be defined as a leading axis and a following axis can be traversed by taking a curve table into account.

The command variable in these motion sequences is an abstract master value, which is generated by the control or derived from an external variable (e.g. simulated position of an axis).

Creating curve tables is performed via part program sequences.

The curve tables in the static NC memory remain valid after the part program has been been closed or after POWER DOWN.

Curve tables can be saved in dynamic NC memory for faster access. Please note that tables need to be reloaded after run-up.

Axis groupings with curve tables must be reactivated independently of the storage location of the curve table after POWER ON.

Linear curve table segments are stored in separate areas to save memory space.

4.2.1.2 Preconditions

Memory configuration

Static NC memory

Memory space for curve tables in static NC memory is defined by machine data:

MD18400 $MN_MM_NUM_CURVE_TABS (number of curve tables)
MD18402 $MN_MM_NUM_CURVE_SEGMENTS (number of curve segments)
MD18403 $MN_MM_NUM_CURVE_SEG_LIN (number of linear curve segments)
MD18404 $MN_MM_NUM_CURVE_POLYNOMS (number of curve table polynomials)
**4.2 Curve tables - 840D sl only**

**Dynamic NC memory**
Memory space for curve tables in dynamic NC memory is defined by machine data:
MD18406 $MN_MM_NUM_CURVE_TABS_DRAM (number of curve tables)
MD18408 $MN_MM_NUM_CURVE_SEGMENTS_DRAM (number of curve segments)
MD18409 $MN_MM_NUM_CURVE_SEG_LIN_DRAM (number of linear curve segments)
MD18410 $MN_MM_NUM_CURVE_POLYNOMS_DRAM (number of curve table polynomials)

**4.2.2 General functionality**

**Curve table**
A functional relation between a command variable "master value" and an abstract following value is described in the curve table.
A following variable can be assigned uniquely to each master value within a defined master value range.

**Curve segment**
The functional relation can be subdivided into separate sections of the master value axes, called curve segments.
Within a curve segment, the relation between the master value and following value is generally described by a polynomial up to the third order. Polynomials up to the 5th degree are also permissible.

**Reference:**
Programming Manual, Production Planning

Curve segments are used if:
- Polynomes or circles are programmed
- Spline is active
- Compressor is active
- Polynomials or circles are generated internally (chamfer/rounding, approximate positioning with G643, WRK etc.)

**Tool radius compensation**
Curve tables are available in which it is possible to specify the tool radius compensation in the table definition even if polynomial blocks or blocks with no motion for an axis, or jumps for the following axis, occur in the curve table (G41/G42/G40 in the table definition).
The equidistant curve (tool center point path of tool radius compensation) of a curve consisting of polynomials can no longer be displayed exactly using polynomials. The associated curve tables must be approximated stepwise, using polynomials in this case. This means that the number of segments in the curve table no longer matches the number of programmed segments. The number of segments required for the curve table is defined by the bend of the curve. The larger the curvature for the programmed curve, the more segments are required for the curve table.

On account of tool radius compensation for curve tables, more memory may be required. Selection option of the memory type should not produce shortage of static NC memory.

### Selection of memory type

While defining a curve table, it can be defined whether the curve table is created in the static or dynamic NC memory.

---

**Note**

Table definitions in the static NC memory are available even after control system run-up. Curve tables of the dynamic NC memory must be redefined after every control system run-up.

### 4.2.3 Memory organization

**Memory configuration**

The storage place available for the curve table in the static and dynamic NC memory is defined during memory configuration (see Section "Memory configuration (Page 209)").

**Memory optimization**

In a curve table with linear segments, the linear segments can be stored efficiently in the memory only if the two following machine data items are > 0:

- MD18403 $MC_MM_NUM_CURVE_SEG_LIN (number of linear curve segments in the static NC memory)
- MD18409 $MC_MM_NUM_CURVE_SEG_LIN_DRAM (number of linear curve segments in the dynamic NC memory)

If no memory areas are created with this machine data, then the linear segments are stored as polynomial segments.
Alarm in case of insufficient memory

If memory has been configured for tables with linear and polynomial segments via machine data and memory for linear segments runs out when generating a linear table, the memory for polynomial segments is used for the linear segments (if available). In this case, memory is "wasted", as a polynomial segment requires an unnecessary amount of memory to store a linear segment. This circumstance is conveyed through an alarm, which also discloses the number of unnecessarily used polynomial segments. The alarm only displays a warning and does not result in the interruption of the program or the generation of the curve table.

If a curve table consists of linear segments and polynomials of a high degree, a memory area for linear segments and a memory area for polynomial segments is required for the storage of the curve table. An alarm is output if insufficient memory is available in the relevant areas. The alarm parameters can be used to detect the resources that are insufficient.

Insufficient memory

If a curve table cannot be created, because sufficient memory is not available, then the newly created table is deleted immediately after the alarm.

If insufficient is available, then one or more table(s) that is/are no longer required can be deleted with CTABDEL or, alternatively, memory can be reconfigured via machine data.

Temporary curve table

When a curve table is created, a temporary curve table is set up first in memory, which is then extended block by block. Finally (CTABEND), the table is checked for consistency. The temporary table is converted to a table that can be used in a coupling only if it is found to be consistent.

Same table number

A new curve table may have the same number as an existing table. The new curve table then overwrites the existing table with the same number. This is done only if the new curve table does not contain any errors. If an error is detected in the new table, the old table is not overwritten.

If the user wishes to have the option of overwriting an existing curve table without deleting it first, then he will need to dimension the table memory so that there is always enough extra memory to accommodate the table to be overwritten.

Overwriting curve tables

Curve tables that are not active in a master value coupling and are locked with CTABLOCK() may be overwritten.

Deleting curve tables

Curve tables that are not active in a master value coupling and are locked with CTABLOCK() may be overwritten.
4.2.4 Commissioning

4.2.4.1 Memory configuration

A defined storage space is available for the curve tables in the static and dynamic NC memory, which is defined through the following machine data:

### Static NC memory

| Machine Data | Description |
|--------------|-------------|
| MD18400 $MN.MM_NUM_CURVE_TABS | Defines the number of curve tables that can be stored in the static NC memory. |
| MD18402 $MN.MM_NUM_CURVE_SEGMENTS | Defines the number of curve table segments that can be stored in the static NC memory. |
| MD18403 $MN.MM_NUM_CURVE_SEG_LIN | Defines the maximum number of linear segments in the static NC memory. |
| MD18404 $MN.MM_NUM_CURVE_POLYNOMS | Defines the number of curve table polynomials that can be stored in the static NC memory. |

### Dynamic NC memory

| Machine Data | Description |
|--------------|-------------|
| MD18406 $MN.MM_NUM_CURVE_TABS_DRAM | Defines the number of curve tables that can be stored in the dynamic NC memory. |
| MD18408 $MN.MM_NUM_CURVE_SEGMENTS_DRAM | Defines the number of curve table segments that can be stored in the dynamic NC memory. |
| MD18409 $MN.MM_NUM_CURVE_SEG_LIN_DRAM | Defines the maximum number of linear segments in the dynamic NC memory. |
| MD18410 $MN.MM_NUM_CURVE_POLYNOMS_DRAM | Defines the number of curve table polynomials that can be stored in the dynamic NC memory. |

**Note**

A curve table with linear segments can be stored efficiently in the memory only if:

MD18403 > 0 or MD18409 > 0

If no memory areas are created with this machine data, then the linear segments are stored as polynomial segments.
4.2.4.2 Tool radius compensation

MD20900

Tool radius compensation can produce segments for which the following axis or leading axis have no movement. A missing movement of the following axis does not normally represent any problem. As against this, a missing movement of the leading axis requests a specification as to how such discontinuities are to be handled, i.e., whether or not a curve table should be generated in these cases. This specification is done in the machine data settings:

MD20900 $MC_CTAB_ENABLE_NO_LEADMOTION (curve tables with discontinuity of the following axis)

| Value | Description |
|-------|-------------|
| 0     | No curve tables that contain a discontinuity in the following axis are produced. Alarm 10949 is output and program processing is aborted. |
| 1     | Curve tables with a discontinuity in the following axis can be generated. If a segment contains a discontinuity in the following axis, Alarm 10955 is output but program processing is continued. |
| 2     | Curve tables with a discontinuity in the following axis can be created without an alarm being output. |

**Note**

In the case of a curve table that contains segments without leading axis movement (this means that the following axis jumps at this point), the following axis can only make a jump within its dynamic limits (max. velocity and max. acceleration). This means that there is always a deviation from the programmed curve.

4.2.4.3 Specification of memory type

MD20905

If there is no memory specification while defining or deleting a curve table, the memory type can be determined through the following machine data:

MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE (default memory type for curve tables)

| Value | Description |
|-------|-------------|
| 0     | Curve tables are normally created in the static NC memory. |
| 1     | Curve tables are normally created in the dynamic NC memory. |
4.2 Curve tables - 840D sl only

4.2.5 Programming

Definition

The following modal language commands work with curve tables: (The parameters are explained at the end of the list of functions.)

- **Beginning of definition of a curve table:**
  
  ```
  CTABDEF(following axis, leading axis, n, applim, memType)
  ```

- **End of definition of a curve table:**
  
  ```
  CTABEND()
  ```

- **Deleting curve table(s):**
  
  ```
  CTABDEL(n)
  ; curve table n
  CTABDEL(n, m)
  ; [n < m], more than one in the range of numbers
  ; it is deleted in static "SRAM" and in dynamic "DRAM" of NC memory.
  CTABDEL(n, m, memType)
  ; Delete with memory specification:
  Those curve tables with the numbers in the range, which are in the specified memory type, will be deleted. All other curve tables are retained.
  Delete all tables in a particular memory type:
  CTABDEL(, "DRAM")
  CTABDEL(, "SRAM")
  CTABDEL()
  ; all in DRAM or
  ; all in SRAM:
  ; all, irrespective of memory type
  ```

- **Read the following value for a master value**
  
  ```
  CTAB(master value, n, degrees, [following axis, lead axis])
  ```

- **Read the master value for a following value**
  
  ```
  CTABINV(following value, approx. master value, n, degrees, [following axis, leading axis])
  ```
Access to curve table segments

- Read start value (following axis value) of a table segment
  \(\text{CTABSSV} \text{(leading value, n, degrees, [following axis, leading axis])}\)
- Read end value (following axis value) of a table segment
  \(\text{CTABSEV} \text{(master value, n, degrees, [following axis, master axis])}\)

**Note**

If curve table functions such as \(\text{CTAB}()\), \(\text{CTABINV}()\), \(\text{CTABSSV}()\) etc., in synchronous actions are used, only main traverse variable, e.g. \(\$\text{AC\_PARAM}[...]\) or \(\$\text{R}[...]\) is permissible for the return value and the argument “degrees” of the function.

Example:

\[\text{ID=1 WHEN TRUE DO } \$\text{R1} = \text{CTABSSV}(10, 1, \$\text{R2})\]
or

\[\text{ID=1 WHEN TRUE DO } \$\text{AC\_PARAM}[1] = \text{CTABSSV}(10, 1, \$\text{AC\_PARAM}[2])\]

Enable/cancel blocking

The following functions can be used to enable or cancel deletion and overwrite blocks for parts programs.

- **Enable** deletion and overwrite block.
  
  General form: \(\text{CTABLOCK}(n, m, \text{memType})\)

- **Cancel** deletion and overwrite block.
  
  \(\text{CTABUNLOCK}(n, m, \text{memType})\) releases the tables locked with \(\text{CTABLOCK}\). Tables involved in an active coupling remain locked, i.e. they cannot be deleted. However, the \(\text{CTABLOCK}\) command is cancelled, i.e. the table can be deleted as soon as the coupling is deactivated. It is not necessary to call \(\text{CTABUNLOCK}\) again.

  General form: \(\text{CTABUNLOCK}(n, m, \text{memType})\)

Applications of the forms:

- Curve table with number \(n\)
  
  \(\text{CTABLOCK}(n)\)

- Curve tables in the number range \(n\) to \(m\).
  
  \(\text{CTABLOCK}(n, m)\)

- All curve tables, irrespective of memory type
  
  \(\text{CTABLOCK}()\)

- All curve tables in the specified memory type
  
  \(\text{CTABLOCK}(), \text{memType}\)

- Curve table with number \(n\)
  
  \(\text{CTABUNLOCK}(n)\)
Curve tables in the number range n to m.

\texttt{CTABUNLOCK(n, m)}

All curve tables, irrespective of memory type

\texttt{CTABUNLOCK()}

All curve tables in the specified memory type

\texttt{CTABUNLOCK(n, , memType)}

Other commands for calculating and differentiating between curve tables for applications for diagnosing and optimizing the use of resources:

- Number of defined tables irrespective of memory type
  \texttt{CTABNO()}

- Number of defined tables in SRAM or DRAM of NC memory
  \texttt{CTABNOMEM(memType)}

- Number of possible curve tables in memory memType.
  \texttt{CTABFNO(memType)}

- Table number of nth curve table.
  
  General form: \texttt{CTABID(n, memType)}

  Generates the table number of the nth curve table with memory type memType. \texttt{CTABID(1, memType)} is used to read out the highest curve number (105) of the memory type specified.

  \texttt{CTABID(n)}

  Generates the table number of the nth curve table in the memory specified using the following machine data:

  MD20905 $MC\_CTAB\_DEFAULT\_MEMORY\_TYPE$ (default memory type for curve tables)

  \texttt{CTABID(p)}

  Generates the ID (table number) of the curve table entered in the memory as the pth curve table.

\underline{Note}

If for example, the sequence is changed between consecutive calls of \texttt{CTABID()}, then \texttt{CTABID(n, ...)} can be used to provide a different curve table than the one provided before the change.

- Indicates the block state of curve table number n.
  \texttt{CTABISLOCK(n)}

- Checks curve table number n.
  \texttt{CTABEXISTS(n)}
M3: Coupled axes

4.2 Curve tables - 840D sl only

- Returns the memory in which curve table number n is stored.
  \texttt{CTABMEMTYPE(n)}

- Returns the table periodicity.
  \texttt{CTABPERIOD(n)}

- Number of curve segments already used in memory memType.
  \texttt{CTABSEG(memType, segType)}

- Number of curve segments used in curve table number n
  \texttt{CTABSEGID(n, segType)}

- Number of still possible curve segments in memory memType.
  \texttt{CTABFSEG(memType, segType)}

- Maximum number of possible curve segments in memory memType.
  \texttt{CTABMSEG(memType, segType)}

- Number of polynomials already used in memory memType.
  \texttt{CTABPOL(memType)}

- Number of curve polynomials used by curve table number n.
  \texttt{CTABPOLID(n)}

- Number of still possible polynomials in memory memType.
  \texttt{CTABFPOL(n)}

- Maximum number of possible polynomials in memory memType.
  \texttt{CTABMPOL(n)}

**Boundary values of curve tables**

Behavior of the leading axis/following axes on the edges of the curve table:

- The value at the beginning of the curve table is read by a following axis.
  \texttt{CTABTSV(n, degrees, FAxis)}, following value at the beginning of the curve table

- The value at the end of the curve table is read by a following axis.
  \texttt{CTABTEV(n, degrees, FAxis)}, following value at the end of the curve table

- The value at the beginning of the curve table is read by the leading axis.
  \texttt{CTABTSP(n, degrees, FAxis)}, master value at the beginning of the curve table

- The value at the end of the curve table is read by the leading axis.
  \texttt{CTABTEP(n, degrees, FAxis)}, master value at the end of the curve table

- Determine the value range of the following value.
  \texttt{CTABTMIN(n, FAxis)}, minimum following value of curve table
  \texttt{CTABTMAX(n, FAxis)}, maximum following value of the curve table
4.2 Curve tables - 840D sl only

**Parameter**

- **Following axis:**
  Identifier of axis via which the following axis is programmed in the definition.

- **Leading axis:**
  Identifier of axis via which the leading axis is programmed.

- **n, m**
  Numbers for curve tables.
  Curve table numbers can be freely assigned. They are used exclusively to uniquely identify a curve table.
  In order to delete a curve table area using the command `CTABDEL(n, m)` must be greater than n.

- **p**
  Entry location (in memType memory area)

- **applim:**
  Behavior at the curve table edges.
  - 0 non-periodic (table is processed only once, even for rotary axes).
  - 1 periodic, modulo (the modulo value corresponds to the LA table values).
  - 2 periodic, modulo (LA and FA are periodic).

- **Master value**
  Position value for which a following value is to be determined.

- **Slave value**
  Position value for which a master value is to be calculated.

- **aproxmastervalue**
  Position value that can be used to determine a unique master value in the case of an ambiguous reversing function of the curve table.

- **degrees**
  Parameter in which the pitch of the table function is returned.
4.2 Curve tables - 840D sl only

- memType
  Optional parameter for specifying memory type to be used in curve tables.
  Possible values:
  "SRAM" curve table is created in static NC memory.
  "DSRAM" curve table is created in dynamic NC memory.
  If an invalid type is entered, the value 2 is returned.
  If the parameter is omitted, then the memory type set via the following machine data takes effect:
  MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE (default memory type for curve tables)

- segType
  Optional parameter for entry of segment type
  Possible values:
  segType "L" linear segments
  segType "P" Polynomial segments

Reference:
Programming Manual, Production Planning; Axis Couplings, Section: Curve tables (CTAB)

Restrictions

The following restrictions apply when programming:

- The NC block must not generate a preprocessing stop.
- No discontinuities may occur in leading axis motion.
- Any block that contains a traverse instruction for the following axis must also include a traverse for the leading axis.
- The direction of motion of the leading axis must not reverse at any point in the rule of motion, i.e. the position of the leading axis must always be unique within the sequence of motions.
  The programmed contour may not move perpendicular to the leading axis.
- Axis names from gantry axis groups cannot be used to define a table (only leading axis are possible).
- Depending on the following machine data, jumps in the following axis may be tolerated if a movement is missing in the leading axis.
  MD20900 $MC_CTAB_ENABLE_NO_LEADMOTION (curve tables with discontinuity of the following axis)
  The other restrictions listed above still apply.
**Axis assignment**

Does not take effect until coupling is activated with curve table.

---

**Note**

The dynamic limit values of the motion commands for a curve table are not checked until activation or interpolation.

---

**Starting value**

The first motion command in the definition of a curve table defines the starting value for the leading and following value.

All instructions that cause a preprocessing stop must be removed.

---

**Example 1**

Without tool radius compensation, without memory type

| Program code     | Comment |
|------------------|---------|
| N100 CTABDEF(AX2, AX1, 3,0) | ; Start of the definition for the non-periodic curve table no. 3 |
| N110 AX1=0 AX2=0   | ; 1. Motion instruction specifies the start value |
|                  | ; Master value: 0, Following value: 0 |
| N110 AX1=20 AX2=0 | ; 1. Curve segment: Master value: 0...20, |
|                  | ; Slave value: Starting value ..0 |
| N120 AX1=100 AX2=6 | ; 2. Curve segment: Master value: 20...100, |
|                  | ; Slave value: 0...6 |
| N130 AX1=150 AX2=6 | ; 3. Curve segment: Master value: 100...150, |
|                  | ; Slave value 6 |
| N130 AX1=180 AX2=0 | ; 4. Curve segment: Master value: 150...180, |
|                  | ; Slave value: 6...0 |
| N200 CTABEND       | ; End of the definition, the curve table |
|                  | ; is generated in its internal representation. |
|                  | ; The advance is reorganized to the status |
|                  | ; at the start of N100 |
Example 2

Example of a curve table with active tool radius compensation:

Prior to definition of a curve table with \texttt{CTABDEF()}, tool radius compensation must not be active; otherwise alarm 10942 is generated. This means that tool radius compensation must be activated within the definition of the curve table. Similarly, it must be deactivated again before the end of the curve table definition, using \texttt{CTABEND}.

| Program code | Comment |
|--------------|---------|
| N10 CTABDEF(Y, X, 1, 0) ; Start of the definition for the non- period curve table no. 1 |
| N20 X0 Y0 |
| N30 G41 X10 Y0 ; WZR compensation on |
| N40 X20 Y20 |
| N50 X40 Y0 |
| N60 X60 Y20 |
| N70 X80 Y0 |
| N80 G40 X90 Y0 ; WZR compensation off |
| N90 CTABEND |

Tool radius compensation is activated in block N30; this causes the approach movement for radius compensation to be made in this block. Similarly, the approach movement for deactivation of the radius compensation is made in block N80.

Note

The value pairs between \texttt{CTABDEF} and \texttt{CTABEND} must be specified for precisely the axis identifiers that have been programmed in \texttt{CTABDEF} as the leading axis and following axis identifiers. In the case of programming errors, alarms or incorrect contours may be generated.

4.2.6 Access to table positions and table segments

Reading table positions

With the program commands \texttt{CTAB} and \texttt{CTABINV} the following value for a master value (\texttt{CTAB}) can be read from the part program and from synchronous actions, or alternatively the master value can be read off for a following value. The pitch value can be used to calculate the speed of the following axis or leading axis at any position in the table.

Reading segment positions

Segment positions of a curve table for the value for the following axis can be read using the \texttt{CTABSSV} and \texttt{CTABSEV} calls.
The language commands `CTABSSV` and `CTABSEV` generally provide the start and end values of the internal segments of the curve tables for the following axis. These values only agree with the programmed values of the curve tables if the programmed segments can be converted 1:1 to the internal segments of the curve table. This is always the case if only `G1` blocks or axis polynomials are used to define the curve tables and no other functions are active.

Programmed sections may under certain circumstances not be transformed unchanged into internal curve segments if:

1. Circles or involutes are programmed
2. Chamfer or rounding is active (`CHF, RND`)
3. Smoothing with `G643` is active
4. Compressor is active (`COMPON, COMPCURV, COMPCAD`)
5. Tool radius compensation is active for polynomial interpolation.

In these cases, the language commands `CTABSSV` and `CTABSEV` may not be used to query the start and end points of the programmed segments.

**CTABINV**

When using the inversion function for the curve tables `CTABINV`, it must be noted that the following value mapped to the leading value may not be unique.

Within a curve table, the following value can assume the same value for any number of master value positions. In order to resolve this ambiguity, the program command `CTABINV` requires a further parameter, in addition to the following value, which it uses to select the ‘correct’ master value. `CTABINV` always returns the master value that is closest to this auxiliary parameter. This auxiliary value can, for example, be the master value from the previous interpolation operation.

**Note**

Although the auxiliary parameter permits calculation of a unique result for the reversal function of the curve table, it should be noted that numerical inaccuracies may give rise to contours, which can cause the reversal function to produce results that deviate from those that would be obtained in a calculation where the accuracy is unrestricted.

**Optional parameters**

The functions `CTAB`, `CTABINV`, `CTABSSV` and `CTABSEV` have optional parameters for the leading and following axes. If one of these parameters is programmed, the master value and following value are modified using the scaling factors of the relevant axes.

This is particularly important if axes have been configured with different length units (inch/metric). If no optional parameters are programmed, the master value and following value are treated as path positions in the conversion from external to internal representation. This means that the values are multiplied according to the configured resolution (decimal places) and the remaining decimal places are truncated.
Identifying the segment associated with master value X

Example of reading the segment starting and end values for determining the curve segment associated with master value X = 30 using \texttt{CTABSSV} and \texttt{CTABSEV}:

| Program code | Comment |
|--------------|---------|
| N10 DEF REAL STARTPOS | ; Start of the definition for the start position of the curve table |
| N20 DEF REAL ENDPOS | |
| N30 DEF REAL GRADIENT | |
| N100 CTABDEF(Y,X,1,0) | ; Beginning of table definition |
| N110 X0 Y0 | ; Start position, 1st table segment |
| N120 X20 Y10 | ; End position, 1st table segment = Start position, 2nd table segment |
| N130 X40 Y40 | |
| N140 X60 Y10 | ; 3. Curve segment |
| N150 X80 Y0 | ; 4. Curve segment |
| N160 CTABEND | ; End of table definition |
| N200 STARTPOS=CTABSSV(30.0,1,GRADIENT) | ; Start position Y in segment 2 = 10. |
| N210 ENDPOS=CTABSEV(30.0,1,GRADIENT) | ; Segment 2 is associated with LW X = 30.0 |

Figure 4-2 Determining the curve segment associated with master value X = 30
Reading values at start and end

The values of the following axes and of the master axis at the start and end of a curve table can be read with the following calls:

R10 = CTABTSV(n, degrees, F axis), following value at the beginning of the curve table
R10 = CTABTEV(n, degrees, F axis), following value at the beginning of the curve table
R10 = CTABTSP(n, degrees, F axis), following value at the beginning of the curve table
R10 = CTABTEP(n, degrees, F axis), following value at the beginning of the curve table

Value range of the following value

The following example illustrates how the minimum and maximum values of the table are determined using CTABTMIN and CTABTMAX:

| Program code | Comment |
|--------------|---------|
| N10 DEF REAL STARTVAL ; Beginning of definition of start and |
| N20 DEF REAL ENDVAL ; Initial values of curve table |
| N30 DEF REAL STARTPARA |
| N40 DEF REAL ENDPARA |
| N50 DEF REAL U_MINVAL |
| N60 DEF REAL U_MAXVAL |
| N70 DEF REAL GRADIENT ... |
| N100 CTABDEF(Y,X,1,0) ; Begin of table definition |
| N110 X0 Y10 ; Start value of the 1st table segment |
| N120 X30 Y40 ; End position 1st table segment = start position 2nd table segment |
| N130 X60 Y5 ; End position of the 2nd table segment ... |
| N140 X70 Y30 ; End position of the 3rd table segment ... |
| N150 X80 Y20 ; End position of the 4th table segment ... |
| N160 CTABEND ; End of table definition ... |
| N200 STARTPOS=CTABTSV(1,GRADIENT) ; STARTPOS = 10 ; Start position of table as well as |
| N210 ENDPoS=CTABTEV(1,GRADIENT) ; ENDPoS = 20 ; End position of table |
| N220 STARTPARA=CTABTSP(1,GRADIENT) ; STARTPARA = 0, ; Master value at beginning of curve table |
| N230 ENDPARA=CTABTEP(1,GRADIENT) ; ENDPARA = 80 ; Master value at end of curve table ... |
| N240 U_MINVAL=CTABTMIN(1) ; Minimum value when Y = 5 and |
| N250 U_MAXVAL=CTABTMAX(1) ; Maximum value when Y = 40 |
4.2.7 Activation/deactivation

Activation

The coupling of real axes to a curve table is activated through this command:

\[ \text{LEADON} \text{ (<Following axis>, <Leading axis>, <n>)} \]

with \(<n>\) = Number of the curve table

Activation is possible:
- In the part program
- in the definition of a synchronous action

Example:

\[
\begin{verbatim}
N1000 LEADON(A,X,3) ; Axis A follows the master value X according to the rule of motions defined in Curve Table No. 3
\end{verbatim}
\]
Deactivation

The switch off of the coupling to a curve table takes place through the following command:

\text{LEADON} (<\text{Following axis}>, <\text{Leading axis}>)

Deactivation is possible:

- In the part program
- In synchronized actions

Note

While programming \text{LEADOF}, the abbreviated form is also possible without specification of the leading axis.

Example:

\begin{verbatim}
... \\
N1010 LEADOF(A,X) ; The coupling of Axis A with its leading axis is canceled \\
... 
\end{verbatim}

Multiple use

A curve table can be used several times in a single part program to couple different channel axes.

4.2.8 Modulo-leading axis special case

Position is absolute

When an axial master value coupling is active, the position of the following axis via a curve table is unique, i.e. an absolute assignment to the master axis exists.

This means that, when a modulo rotary axis is used as the master axis, the position of the master axis is absolute. In other words, the position of the modulo rotary axis entered in the curve table is absolute, and not modulo-reduced.

Example

Let the position of a modulo rotary axis with \text{LEADON} be 210°. The position 210° degrees is used as the starting value in the curve table. After one rotation of the modulo axis, the axis position is again displayed as 210°. The absolute position 570° is however taken as the input value in the curve table:

\[ 210° + 1 \text{ round (360°)} = 570° \]
4.2.9 Behavior in AUTOMATIC, MDA and JOG modes

**Activation**

An activated curve table is functional in the AUTOMATIC, MDA and JOG modes.

**Basic setting after run-up**

No curve tables are active after run-up.

4.2.10 Effectiveness of PLC interface signals

**Dependent following axis**

With respect to the motion of a following axis that is dependent on the leading axis, only the following axis interface signals that effect termination of the motion (e.g. axis-specific feed stop, axis inhibit, servo enable, etc.) are effective.

**Leading axis**

In an activated axis group, the interface signals move the leading axis through the axis coupling to the associated following axis, i.e.:

- feed control of the leading axis causes a corresponding feed control of the following axis.
- A shutdown of the leading axis through interface signals (e.g., axis-specific feed stop, axis inhibit, servo enable etc.) causes the corresponding following axis to shut down.

The effect of the axis inhibit of the leading axis on the following axis can be prevented through the following MD setting:

MD37160 $MA_LEAD_FUNCTION_MASK, Bit 1 = 1

**Position measuring system 1/2 (DB31, ... DBX1.5/1.6))**

Switch-over of the position measuring system for the leading and following axes is not inhibited for an active coupled axis group. The coupling is not canceled.

**Recommendation:** Switch the measuring system over when the coupling is deactivated.
4.2.11 Diagnosing and optimizing utilization of resources

The following functions allow parts programs to get information on the current utilization of curve tables, table segments and polynomials.

One result of the diagnostic functions is that resources still available can be used dynamically with the functions, without necessarily having to increase memory usage. The description of the parameters in Chapter "Programming Curve Tables" also applies to the following functions.

a) Curve tables

- Determine total number of defined tables.

  The definition applies to all memory types (see also CTABNOMEM)
  
  CTABNO()

- Number of defined tables in SRAM or DRAM of NC memory.

  CTABNOMEM (memType)

  If memType is not specified, the memory type specified in the following machine data:
  
  MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE (default memory type for curve tables)

  Result:
  
  >= 0: Number of defined curve tables
  -2: Invalid memory type

- Determine number of curve tables still possible in memory.

  CTABFNO(memType)

  If memType is not specified, the memory type specified in the following machine data:
  
  MD220905 $MC_CTAB_DEFAULT_MEMORY_TYPE

  Result:
  
  >= 0: Number of possible tables
  -2: Invalid memory type
• Determine the table number of the pth table in the memory type specified optionally

\[ \text{CTABID}(p, \text{memType}) \]

If \( \text{memType} \) is not specified, the memory type specified in the following machine data:

\[ \text{MD20905 } $MC\_CTAB\_DEFAULT\_MEMORY\_TYPE \]

Result:

Table number or

Alarm for invalid \( p \) or \( \text{memType} \)

When using the \( \text{CTABID}(p, \text{memType}) \) function, no assumptions should be made regarding the sequence of the curve tables in the memory. The \( \text{CTABID}(p, ...) \) function supplies the ID (table number) of the curve table entered in memory as the pth curve table.

If the sequence of curve tables in memory changes between consecutive calls of \( \text{CTABID}() \), \( \text{CTABID}() \), e.g. due to the deletion of curve tables with \( \text{CTABDEL}() \), the \( \text{CTABID}(p, ...) \) function can supply a different curve table with the same number.

To prevent this from happening, the curve tables concerned can be locked, using the \( \text{CTABLOCK}(...) \) language command. In this case, it should be noted that the curve tables concerned are then unlocked with \( \text{CTABUNLOCK}() \).

• Determine block condition

Table \( n \)

\[ \text{CTABISLOCK}(n) \]

Result:

\( > 0 \): Table is blocked

Reason for block:

1: by \( \text{CTABLOCK}() \)

2: by an active coupling

3: by \( \text{CTABLOCK}() \) and by an active coupling

\( = 0 \): Table is not blocked

-1: Table does not exist

• Check whether the curve table exists

\[ \text{CTABEXISTS}(n) \]

Result:

1: Table exists

0: Table does not exist
**M3: Coupled axes**

4.2 Curve tables - 840D sl only

- **Determine memory type** of a curve table
  
  CTABMEMTYP(n)
  
  Result:
  0: Table in static SRAM NC memory
  1: Table in dynamic "DRAM" NC memory
  -1: Table does not exist

- **by an active coupling periodic**
  
  CTABPERIOD(n)
  
  Result:
  0: Table is not periodic
  1: Table is periodic in the leading axis
  2: Table is periodic in the leading and following axes
  -1: Table does not exist

**b) Curve table segments**

- **Determine number of used** curve segments of the type memType in the memory range.
  
  CTABSEG(memType, segType)
  
  If memType is not specified, the memory type specified in the following machine data:
  
  MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE
  
  Result:
  >= 0: Number of curve segments
  -2: Invalid memory type
  
  If segType is not specified, the sum is produced via linear and polynomial segments in
  
  the memory type.
  
  -2: segType not equal "L" or "P"

- **Determine number of used curve segments** of the type memType in the memory range
  
  CTABSEGRID(n, segType)
  
  Result:
  >= 0: Number of curve segments
  -1: Curve table with number n does not exist
  -2: segType not equal "L" or "P"
M3: Coupled axes

4.2 Curve tables - 840D sl only

- Determine number of **free** curve segments of the type memType in the memory range
  
  \[
  \text{CTABFSEG}(\text{memType}, \text{segType})
  \]
  
  If memType is not specified, the memory type specified in the following machine data:
  
  MD20905 $MC\_CTAB\_DEFAULT\_MEMORY\_TYPE
  
  Result:
  
  >= 0: Number of free curve segments
  
  -2: Invalid memory type, segType not equal "L" or "P"

- Determine **maximum** number of possible curve segments of the type segType in the memory
  
  \[
  \text{CTABMSEG}(\text{memType}, \text{segType})
  \]
  
  If memType is not specified, the memory type specified in the following machine data:
  
  MD20905 $MC\_CTAB\_DEFAULT\_MEMORY\_TYPE
  
  Result:
  
  >= 0: Maximum number of possible curve segments
  
  -2: Invalid memory type, segType not equal "L" or "P"

c) Polynomials

- Determine the number of **used** polynomials of the memory type
  
  \[
  \text{CTABPOL}(\text{memType})
  \]
  
  If memType is not specified, the memory type specified in the following machine data:
  
  MD20905 $MC\_CTAB\_DEFAULT\_MEMORY\_TYPE
  
  Result:
  
  >= 0: Number of polynomials already used in the memory type
  
  -2: Invalid memory type

- Determine the number of curve polynomials used by a curve table
  
  \[
  \text{CTABPOLID}(n)
  \]
  
  Result:
  
  >=0: Number of used curve polynomials
  
  -1: Curve table with number n does not exist
- Determine the number of free polynomials of the memory type

\[ \text{CTABFPOL}(\text{memType}) \]

If memType is not specified, the memory type specified in the following machine data:

\[ \text{MD20905 } \$\text{MC\_CTAB\_DEFAULT\_MEMORY\_TYPE} \]

Result:

\[ \geq 0: \text{Number of free curve polynomials} \]

\[ -2: \text{Invalid memory type} \]

- Determine the maximum number of polynomials of the memory type

\[ \text{CTABMPOL}(\text{memType}) \]

If memType is not specified, the memory type specified in the following machine data:

\[ \text{MD20905 } \$\text{MC\_CTAB\_DEFAULT\_MEMORY\_TYPE} \]

Result:

\[ \geq 0: \text{Maximum number of possible curve polynomials} \]

\[ -2: \text{Invalid memory type} \]

### 4.2.12 Supplementary conditions

#### Transformations

Transformations are not permissible in curve tables. **TRAANG** is an exception.

#### TRAANG

If **TRAANG** is programmed, the rule of motion programmed in the basic co-ordinate system is transformed to the associated machine co-ordinate system. In this way it is possible to program a curve table as Cartesian co-ordinates for a machine with inclined linear axes.

The condition that stipulates that "the direction of motion of the leading axis must not reverse at any point of the rule of motion" must then be met in the machine co-ordinate system. Please note that this condition in the basic co-ordinate system does not have the same meaning as in the machine co-ordinate system, since the contour tangents are changed by the transformation.
4.2.13 Examples

Definition of a curve table with linear sets

```plaintext
%_N_TAB_1_NOTPERI_MPF
;SPATH=/N_WKS_DIR/_N_KURVENTABELLEN_WPD
; Def.TAB1 0-100mm Kue1/1 notperio.
N10 CTABDEF(YGEO,XGEO,1,0) ; FA=Y LA=X Curve No.=1 Not period.
N1000 XGEO=0 YGEO=0 ; Start values
N1010 XGEO=100 YGEO=100
CTABEND
M30
```

Definition of a curve table with polynomial sets

```plaintext
%_N_TAB_1_NOTPERI_MPF
;SPATH=/N_WKS_DIR/_N_KURVENTABELLEN_WPD
; Def.TAB1 0-100mm Kue1/1 notperio.
N10 CTABDEF(Y,X,1,0) ; FA=Y LA=X Curve No.=1 Not period.
N16 G1 X0.000 Y0.000
N17 POLY PO[X]=(31.734,0.352,-0.412) PO[Y]=(3.200,2.383,0.401)
N18 PO[X]=(49.711,-0.297,0.169) PO[Y]=(7.457,1.202,-0.643)
N19 PO[X]=(105.941,1.961,-0.938) PO[Y]=(11.708,-6.820,-1.718)
N20 PO[X]=(132.644,-0.196,-0.053) PO[Y]=(6.815,-2.743,0.724)
N21 PO[X]=(147.754,-0.116,0.103) PO[Y]=(3.359,-0.188,0.277)
N22 PO[X]=(174.441,0.578,-0.206) PO[Y]=(0.123,1.925,0.188)
N23 PO[X]=(185.598,-0.007,0.005) PO[Y]=(-0.123,0.430,-0.287)
N24 PO[X]=(212.285,0.040,-0.206) PO[Y]=(-3.362,-2.491,0.190)
N25 PO[X]=(227.395,-0.193,0.103) PO[Y]=(-6.818,-0.641,0.276)
N26 PO[X]=(254.098,0.355,-0.053) PO[Y]=(-11.710,0.573,0.723)
N27 PO[X]=(310.324,0.852,-0.937) PO[Y]=(-7.454,11.975,-1.720)
N28 PO[X]=(328.299,-0.209,0.169) PO[Y]=(-3.197,0.726,-0.643)
N29 PO[X]=(360.031,0.885,-0.413) PO[Y]=(0.000,-3.588,0.403)
CTABEND
N30 M30
```
Definition of a periodic curve table

Table No: 2
Master value range: 0 - 360

The following axis traverses from N70 to N90, a movement from 0 to 45 and back to 0.

| N10  | DEF REAL DEPPOS |
|------|-----------------|
| N20  | DEF REAL GRADIENT |
| N30  | CTABDEF(Y,X,2,1) |
| N40  | G1 X=0 Y=0 |
| N50  | POLY |
| N60  | PO[X]=(45.0) |
| N70  | PO[X]=(90.0) PO[Y]=(45.0,135.0,-90) |
| N80  | PO[X]=(270.0) |
| N90  | PO[X]=(315.0) PO[Y]=(0.0,-135.0,90) |
| N100 | PO[X]=(360.0) |
| N110 | CTABEND |
| N130 | G1 F1000 X0 ; Testing the curve by coupling Y to X |
| N140 | LEADON(Y,X,2) |
| N150 | X360 |
| N160 | X0 |
| N170 | LEADOFF(Y,X) |
| N180 | DEPPOS = CTAB(75.0,2,GRADIENT) ; Reading the table position at master value 75.0 from the curve table with Table No. 2 |
| N190 | G0 X75 Y=DEPPOS ; Positions of leading and following axis |
| N200 | LEADON(Y,X,2) ; No synchronization of the following axis is required after the coupling is activated |
| N210 | G1 X110 F1000 |
| N220 | LEADOFF(Y,X) |
| N230 | M30 |
4.3 Master value coupling - 840D sl only

4.3.1 Product brief

4.3.1.1 Function

The "axial master value coupling" function can be used to process short programs cyclically with close coupling of the axes to one another and a master value that is either generated internally or input from an external source.

The master value can, for example, be derived from a conveyor belt or a vertical shaft.

The axial master value coupling can be switched on and off in the NC part program or via synchronized action.

4.3.1.2 Preconditions

The option "Master Value Coupling and Curve Tables Interpolation" or the relevant option of generic coupling (refer to "Preconditions (Page 274)" in the "Brief description" for generic coupling) is a prerequisite for utilization of the function.

4.3.2 General functionality

Curve table

With the axial master value coupling, a leading and a following axis are moved in synchronism. It is possible to assign the position of the following axis via a curve table or the resulting polynomial uniquely to a position of the leading axis - simulated if necessary.

Master value object

The master value object is the input variable for the curve table.

The following can be defined as the position of the master value object:

- The axis actual position (actual value measured by transmitter)
  or
- The setpoint position (calculated from interpolator) (default)

If the leading axis is interpolated by the same NCU, the setpoint coupling delivers a better follow-up response than is possible for actual value coupling (in the same IPO cycle).
Virtual leading axis / simulated master value

If the leading axis is not interpolated by the same NCU, the interpolator that is implemented in the NCU for this particular leading axis can be used for master value simulation. The following machine data settings must be defined for this:

MD30132 $MA_IS_VIRTUAL_AX[n] = 1 (axis is virtual axis)
MD30130 $MA_CTRLOUT_TYPE[n] = 0 (simulation as output type of setpoint)

Properties of master value simulation:
- Separation of IPO and servo.
- Actual values of the axis are recorded.
- Setpoint values are produced by IPO but not passed on to the servo motor.
- When switching over to master value coupling, the simulation can be programmed with the last actual value read, whereas the path of the actual value is generally outside the control of the NCU.
- If, for purposes of master value simulation, the master value object is switched from actual value coupling to setpoint value coupling and a traversing command is issued for the leading axis in the same interpolator cycle, the interpolator for the axis is initialized by the NCK so that the master value produces a constant path in the first derivation.

Note
Virtual axes assigned to a real drive must remain unblocked.

Offset and scaling

The setpoint value for the following axis can be offset and scaled. The following setting data is used for this:

SD43102 $SA_LEAD_OFFSET_IN_POS (offset of master value with coupling to this axis)
SD43104 $SA_LEAD_SCALE_IN_POS (scaling of master value with coupling to this axis)
SD43106 $SA_LEAD_OFFSET_OUT_POS (offset of function value of the curve table)
SD43108 $SA_LEAD_SCALE_OUT_POS (scaling of function value of the curve table)

If \( x \) is a periodic curve table and this is interpreted as oscillation, the offset and scaling can also be interpreted as follows:

SD43102 $SA_LEAD_OFFSET_IN_POS[Y] offsets the phase of the oscillation.
SD43106 $SA_LEAD_SCALE_OUT_POS[Y] affects the amplitude.
SD43108 $SA_LEAD_OFFSET_OUT_POS[Y] offsets the center of the oscillation.

If the coupling is activated and synchronous, the new set position is approached as soon as values are written to this setting data.
**M3: Coupled axes**

### 4.3 Master value coupling - 840D sl only

**Special functions**

234 Function Manual, 07/2012, 6FC5397-2BP40-3BA0

---

**Figure 4-4** Master value coupling offset and scaling (multiplied)

**Table definition for example:**

CTABDEF(FA, LA, CTABNR, 1)

LA = 0 FA = 0
LA = 100 FA = 100
CTABEND

With IN, SCALE is not active outside TAB!

---

**Figure 4-5** Master value coupling offset and scaling (with increment offset)

**SD43104 $SSA_LEAD_SCALE_IN_POS[FA]**

The input value is multiplied.

---

**SD43106 $SSA_LEAD_OFFSET_OUT_POS[FA]**

The calculated position of the following axis is offset with the specified increments.

---

With OUT, OFFSET active outside TAB!
Reaction to Stop

All master value coupled following axes react to channel stop and MODE GROUP stop. Master value coupled following axes react to a stop due to end of program (M30, M02) if they have not been activated by static synchronized actions (IDS=...). The following machine data is to be observed in this connection:

MD20110 $MC_RESET_MODE_MASK (definition of initial control settings after RESET / TP End)

MD20112 $MC_START_MODE_MASK (definition of initial control system settings for NC START)

Leading axis and following axis must always interpolate in the same channel. A following axis located in a different channel cannot be coupled (axis replacement).

START and mode change enable a following axis in the master value coupling that has been stopped.

RESET also enables a stopped following axis in master value coupling. If enabling by RESET is not desired, or if it is dangerous (e.g. because the following axis is coupled to an external master value not controlled by the NC), then MD20110 must be programmed such that master value couplings are switched off with RESET (=2001H, i.e. bit 13 set to 1).

Axial functions

Actual value coupling causes a position offset between the leading and following axis. The cause of this is the IPO cycle-based dead time in the position controller, which lies between the actual value of the leading axis and the following axis.

Normally, the position offset and the following error is compensated by a linear extrapolation of the master value to the extent of this dead time, i.e. dead time compensation is active in master value coupling. The following MD setting is defined to deactivate the dead time compensation:

MD37160 $MA_LEAD_FUNCTION_MASK Bit 0 = 0

Interface to axis exchange

A master value coupled following axis receives its setpoint values from curve tables. Overlaid programming of this axis is not possible in the part program. Therefore, the master value coupled following axis is removed from the channel in the same way as for axis exchange. This is carried out automatically when the coupling is activated in the part program.

If the coupling is to be activated with synchronized action, then it must be prepared for it in advance with RELEASE.

After a master value coupling has been deactivated, the former following axis can be programmed again in the part program.
M3: Coupled axes

4.3 Master value coupling - 840D sl only

Spindles in master value coupling

A spindle can only be used as the master value coupled following axis if it has been switched to axis mode beforehand. The machine data parameter block of the axis drive then applies.

Example: Activation from synchronized action

| Program code                   | Comment                                           |
|-------------------------------|--------------------------------------------------|
| SPOS=0                        |                                                  |
| B=IC(0)                       | ; Switch spindle to axis operation.               |
| RELEASE(Y)                    | ; Release for synchronized action.                |
| ID=1 WHEN ($AA_IM[X]<-50) DO LEADON(B,X,2) | ; Y is coupled to X via curve table No. 2.        |

4.3.3 Programming

Definition and activation

An axis master value coupling is defined and activated simultaneously with the modal effective language command LEADON.

Syntax:

LEADON(<FA>,<LA>,<CTABn>)

Meaning:

<FA> Following axis as geometry-, channel- or machine axis name (X, Y, Z,...)

<LA> Leading axis as geometry-, channel- or machine axis name (X, Y, Z,...)

Software axis is also possible:

MD30130 $MA_CTRLOUT_TYPE=0 (setpoint output type)

<CTABn> Number of curve table

Range of values: 1 to 999

Example:

| Program code           | Comment                                                                 |
|------------------------|-------------------------------------------------------------------------|
| LEADON(Y,X,1)          | ; Definition and activation of a master value coupling between the leading axis X and the following axis Y. Curve Table No. 1 should be used to calculate the following value. |
Boundary conditions:

- No reference point is required to activate the coupling.
- A defined following axis cannot be traversed in the JOG mode (not even if the "Synchronized run fine" or "synchronized run coarse" interface signal is not there).
- An activated coupling must first be deactivated with LEADOF before it can be activated again with LEADON. The settings in the following machine data are to be considered in this connection:

  MD20112 MC_START_MODE_MASK (definition of initial control system settings with NC-START)
  MD20110 $MC_RESET_MODE_MASK (definition of initial control settings after RESET/TP-End)

![Diagram](image-url)  
**Figure 4-6 Activating master value coupling**
An axis master value coupling is deactivated with the model language command `LEADOF`.

When the axis master value coupling is deactivated, the following axis becomes the command axis and a stop command is generated implicitly for the following axis. The stop command can be overwitten by another command with a synchronous action.

**Syntax:**

```
LEADOF(<FA>,<LA>)
```

**Meaning:**

- `<FA>`: Following axis as geometry-, channel- or machine axis name (X, Y, Z,...)
- `<LA>`: Leading axis as geometry-, channel- or machine axis name (X, Y, Z,...)

Software axis is also possible:

```
MD30130 $MA_CTRLOUT_TYPE=0 (setpoint output type)
```

**Example:**

| Program code       | Comment                                      |
|--------------------|----------------------------------------------|
| LEADOF(Y,X,1)      | ; Switching off of master value coupling      |
|                    | between the leading axis X and the following  |
|                    | axis Y.                                        |

**Note**

Activating / deactivating the axis master value coupling with `LEADON / LEADOF` is permissible both in the part program and in synchronous actions.

**References:**

Function Manual, Synchronized Actions

### Coupling type

The coupling type is defined by the following axis-specific setting data:

```
SD43100 $SA_LEAD_TYPE[<LA>] (type of master value)
```

- `<LA>`: Leading axis as geometry axis name, channel axis name or machine axis name (X, Y, Z,...)

| Value | Meaning                                                                 |
|-------|-------------------------------------------------------------------------|
| 0     | Actual value coupling (this type of coupling must be used for external  |
|       | leading axes)                                                          |
| 1     | Setpoint coupling (default setting)                                     |
| 2     | Simulated master value (note virtual axis, not evaluated for FA)        |

Switch-over between actual and setpoint value coupling is possible at any time (preferably in the idle phase).
System variables of the master value

The following master value system variables can only be read from part program and from synchronous actions:

| System variable      | Meaning                                                        |
|----------------------|----------------------------------------------------------------|
| $AA_LEAD_V[ax]       | Velocity of the leading axis                                   |
| $AA_LEAD_P[ax]       | Position of the leading axis                                   |
| $AA_LEAD_P_TURN      | Master value position                                          |
|                      | Portion that is deducted during modulo reaction.              |
|                      | The actual (not modulo-reduced) position of the leading axis  |
|                      | $AA_LEAD_P_TURN + $AA_LEAD_P                                   |

The speeds and positions of simulated master values (when $SA_LEAD_TYPE[ax]=2) can be written in and read from the part program and synchronous actions.

| System variable      | Meaning                                                        |
|----------------------|----------------------------------------------------------------|
| $AA_LEAD_SV[ax]      | Simulated master value velocity per IPO cycle                  |
| $AA_LEAD_SP[ax]      | Simulated position in MCS                                      |

System variables of the following axis

For the following axis, the following system variables can be read in the part program and synchronized action:

| System variable      | Meaning                                                        |
|----------------------|----------------------------------------------------------------|
| $AA_SYNC[ax]         | Condition of the coupling between following and leading axes  |
| Value                | Meaning                                                        |
| 0                    | not synchronized                                              |
| 1                    | "Coarse synchronous operation" 1)                             |
| 2                    | "Fine synchronous operation" 2)                                |
| 3                    | "Coarse synchronous operation" 1) AND "Fine synchronous      |
|                      | operation" 2)                                                   |

Corresponds to:

1) • MD37200 $MA_COUPLE_POS_TOL_COARSE
   • DB31, ... DBX98.1 (coarse synchronous operation)

2) • MD37210 $MA_COUPLE_POS_TOL_FINE
   • DB31, ... DBX98.0 (fine synchronous operation)

| $AA_IN_SYNC[ax]      | Condition of the synchronization between following and leading axes |
| Value                | Meaning                                                        |
| 0                    | Synchronization not started or ended.                           |
| 1                    | Synchronization is running, i.e., the following axis is         |
|                      | synchronized.                                                   |

$AA_INSYNC[ax] corresponds to the NC/PLC interface signal: DB31, ... DBX99.4 (synchronization running)
Note
If the following axis is not enabled for travel, it is stopped and is no longer synchronous.

4.3.4 Behavior in AUTOMATIC, MDA and JOG modes

Efficiency
A master value coupling is active depending on the settings in the part program and in the following machine data:
MD20110 $MC_RESET_MODE_MASK (definition of initial control settings after RESET / TP End)
MD20112 $MC_START_MODE_MASK (definition of initial control system settings with NC-START)

Manual mode
Once a master axis coupling has been activated, traversal of the master axis (e.g. with rapid traverse or incremental dimension INC1 ... INC10000) results in a movement of the slave axis, allowing for the curve table definition.

Referencing
A master value coupled following axis is to be referenced prior to activation of the coupling. A following axis cannot be referenced when the coupling is activated.

Deletion of distance-to-go
When deletion of distance-to-go is performed for a leading axis, all axes in the associated, activated master value coupling are shut down.

Basic setting after POWER ON
No master value couplings are active after POWER ON. (Options with ASUB).

Behavior after NC start/RESET
The following behavior results, depending on the settings of the machine data:
MD20110 $MC_RESET_MODE_MASK (bit 13) (definition of initial control settings after RESET / TP End)
MD20112 $MC_START_MODE_MASK (bit 13) (definition of initial control system settings with NC-START)
4.3 Master value coupling - 840D sl only

- MD20110 $MC_RESET_MODE_MASK=2001H
  &&
MD20112 $MC_START_MODE_MASK=0H
→ Master value coupling remains valid after \texttt{RESET} and \texttt{START}.
- MD20110 $MC_RESET_MODE_MASK=2001H
  &&
MD20112 $MC_START_MODE_MASK=2000H
→ Master value coupling remains valid after \texttt{RESET} and is canceled with \texttt{START}. However, master value coupling activated via IDS=... remains valid.
- MD20110 $MC_RESET_MODE_MASK=1H
  → Master value coupling is canceled with \texttt{RESET} irrespective of machine data:
MD20112 $MC_START_MODE_MASK
Master value coupling activated via IDS=... can only be deactivated via an operator front panel reset and remains valid after program end/reset (M30, M02).
- MD20110 $MC_RESET_MODE_MASK=0H
  → Master value coupling remains valid after \texttt{RESET} and is canceled with \texttt{START}, irrespective of machine data:
MD20112 $MC_START_MODE_MASK
However, master value coupling activated via IDS=... remains valid.

\textbf{Reference:}
Function Manual, Basic Functions; Coordinate Systems, Axis Types, Axis Configurations, ... (K2)

\textbf{Activating, deactivating}

Master value couplings activated via a static synchronous action (IDS=...) are:
- not deactivated during program start, regardless of the value of machine data:
MD20110 $MC_RESET_MODE_MASK \text{(definition of initial control settings after RESET / TP End)}
and
MD20112 $MC_START_MODE_MASK \text{(definition of initial control system settings with NC-START)}
- not deactivated during program end reset (M30, M02), regardless of the value of machine data:
MD20110 $MC_RESET_MODE_MASK
4.3.5 Effectiveness of PLC interface signals

Leading axis

When a coupled axis group is active, the interface signals (IS) of the leading axis are applied to the appropriate following axis via axis coupling. i.e.:

- a feed control action of the leading axis is applied via the master value coupling to effect an appropriate feed control action in the following axis.
- shutdown of the leading axis as the result of an IS (e.g. axis-specific feed stop, axis inhibit, servo enable, etc.) causes the corresponding coupled motion axis to shut down.

Position measuring system 1/2 (DB31, ... DBX1.5/1.6))

Switch-over of the position measuring system for the leading and following axes is not inhibited for an active coupled axis group. The coupling is not canceled.

**Recommendation:** Switch the measuring system over when the coupling is deactivated.

4.3.6 Special characteristics of the axis master value coupling function

Control dynamics

Depending on the application in question, it may be advisable to match the position controller parameter settings (e.g. servo gain factor) of the leading axis and following axis in an axis grouping. It may be necessary to activate other parameter sets for the following axis. The dynamics of the following axis should be the same or better than those of the leading axis.

Status of coupling

see Section "Status of coupling (Page 202)"

Actual value display

The display of the actual value is updated for all axes of in a master value coupled axis grouping (only real axes) coupled via a master value.

Interpolation

When the movement defined in the curve table is interpolated, axis position and axis speed are calculated for a master value and its speed.
Logging

The curve tables generated by the definition of motion sequences are stored in the battery-backed memory.

The curve tables are not lost when the control system is switched off.

These functions have no effect on cyclic machines because they are performed without operator actions. Nor does it make sense to perform automatic (re-)positioning via the NC with external master values.

4.3.7 Supplementary conditions

External master value axes

When using the \texttt{REPOS} or \texttt{REPOSA} parts program instructions in conjunction with external master value axes, it should be ensured that these are released by the channel or switched to a "neutral state" using the \texttt{RELEASE} instruction.

When attempting to reposition without release of the axis, the message "Wait: Feed stop active" is displayed and the processing of the part program is not continued.
4.4 Electronic gear (EG) - 840D sl only

4.4.1 Product brief

4.4.1.1 Function

General

The "electronic gear" function makes it possible to control the movement of a following axis, depending on up to five master axes. The relationship between each leading axis and the following axis is defined by the coupling factor. Following axis motion components derived from the individual leading axis motion components have an additive effect.

The coupling can be based on:
- Actual value of the leading axis
- Setpoint of the leading axis

The following functions of a gear grouping can be programmed using part program instructions:
- Defining
- Switch on
- Switch off
- Delete

Curve tables

Non-linear relationships between lead and following axes can also be implemented using curve tables.

Cascading

Electronic gearboxes can be cascaded, i.e. the following axis of an electronic gearbox can be the leading axis for a subsequent electronic gearbox.

Synchronous position

An additional function for synchronizing the following axis allows a synchronous position to be selected:
- Approach next division (tooth gap) time-optimized
- Approach next division (tooth gap) path-optimized
- Approach in positive direction of axis rotation, absolute
4.4 Electronic gear (EG) - 840D sl only

- Approach in negative direction of axis rotation, absolute
- Traverse time-optimized with respect to programmed synchronized position
- Traverse path-optimized with respect to programmed synchronized position

**Application Examples:**
- Machine tools for gear cutting
- Gear trains for production machines

### 4.4.1.2 Preconditions

The "Electronic Gearbox" option or the relevant option of generic coupling (refer to "Preconditions (Page 274)" in the "Brief description" of Generic Coupling) is required for the usage of function.

**Function**

With the aid of the "Electronic gearbox" the movement of a **following axis FA** can be interpolated dependent of up to five **leading axes LA**. The relationship between each leading axis and the following axis is defined by a coupling factor. The following axis motion components derived in this manner from the individual leading axis motion components have an additive effect.

\[
FA_{set} = SynPosFA + (LA_1 \cdot SynPosLA_1) \cdot CF_1 + ... + (LA_5 \cdot SynPosLA_5) \cdot CF_5
\]

with:

- \(SynPosFA, SynPosLA_i\): from call EGONSYN (see below)
- \(FA_{setpoint}\): Partial setpoint of the following axis.
- \(LA_i\): Setpoint or actual value of the ith leading axis (depending on the type of coupling - see below)
- \(KF_i\): Coupling factor of the ith leading axis (see below)

All paths are referred to the basic co-ordinate system **BCS**.

When an EG axis group is activated, it is possible to synchronize the leading axes and following axis in relation to a defined starting position.

From the part program a gearbox group can be:
- defined,
- activated,
- deactivated,
- deleted.
Extensions

The influence of each of the 5 leading axes can be specified using a curve table as an alternative to a transmission ratio (KF=numerator/denominator).

It is thus possible for each curve (except for the special case of a straight line) for the leading axis to influence the following axis in a non-linear manner. The function can only be used with EGONSYN.

The function EG can be activated with curve tables with EGON.

The function EGONSYNE is available for approaching the synchronized position of the following axis with a specified approach mode.

For special applications, it may be advisable configure the position controller as a PI controller.

⚠️ CAUTION

Knowledge of the control technology and measurements with servo trace are an absolute prerequisite for using this function.

References:

- CNC Commissioning Manual: NCK, PLC, drive
- Function Manual, Basic Functions; Velocities, Setpoint-Actual Value Systems, Closed-Loop Control (G2)

Coupling type

The following axis motion can be derived from either of the following:

- Setpoints of leading axes
- Actual values of leading axes

The reference is set in the definition call for the EG axis group:

\[
\text{EGDEF }
\]

(see Section "Definition of an EG axis group (Page 253)")

Coupling factor

The coupling factor must be programmed for each leading axis in the group. It is defined by numerator/denominator.

Coupling factor values numerator and denominator are entered per leading axis with the following activation calls:

\[
\text{EGON}
\]
\[
\text{EGONSYN}
\]
\[
\text{EGONSYNE}
\]

(see Section "Activating an EG axis group (Page 254)"

Number of EG axis groups

Several EG axis groups can be defined at the same time. The maximum possible number of EG axis groupings is set in the following machine data:

MD11660 $MN_NUM_EG

The maximum permissible number of EG axis groups is 31.

Note

The option must be enabled.

EG cascading

The following axis of an EG can be the leading axis of another EG. For a sample configuration file, see Section "Examples".

Synchronous positions

To start up the EG axis group, an approach to defined positions for the following axis can first be requested.

Synchronous positions are specified with:

EGONSYN (see below for details)

EGONSYNE (extended EGONSYN call).
Synchronization

If a gear is started with EGON(), EGONSYN() or EGONSYNE() see below, the actual position of the following axis is only identical to the setpoint position defined by the rule of motion of the gear specified by the positions of the leading axes at this time if the part program developer makes sure that it is. The control then uses the motion of the following axis to ensure that the setpoint and actual positions of the following axes correspond as quickly as possible if the leading axes are moved further. This procedure is called synchronization. After synchronization of the following axis, the term *synchronous* gearing is used.

Activation response

An electronic gearbox can be activated in two different ways:

1. On the basis of the axis positions that have been reached up to now in the course of processing the command to activate the EG axis group is issued without specifying the synchronizing positions for each individual axis.

   EGON (see Section "Activating an EG axis group (Page 254)"")

2. The command to activate the EG axis group specifies the synchronized positions for each axis. From the point in time when these positions are reached, the EG should be synchronized.

   EGONSYN (see Section "Activating an EG axis group (Page 254)"")

3. The command to activate the EG axis group specifies the synchronized positions and approach mode for each axis. From the point in time when these positions are reached, the EG should be synchronized.

   EGONSYNE (see Section "Activating an EG axis group (Page 254)"")

Synchronization with EGON

With EGON(), no specifications are made for the positions at which the following axis is to be synchronized. The control system activates the EG and issues the signal "Synchronized position reached".

Synchronization for EGONSYN

1. With EGONSYN(), the positions of the leading axes and the synchronization position for the following axis are specified by the command.
   
   - The control then traverses the following axis with just the right acceleration and velocity to the specified synchronization position so that the following axis is in position with the leading axes at its synchronization position.
   
   - If the following axis is *stationary*: If the "Feed stop/spindle stop" DB 31, ... DBX 4.3 is set for the following axis, the following axis is not set in motion by EGON or EGONSYN. A traverse command is issued and block changing is blocked until the axis-specific feed is enabled. EGOSYN is topped by RESET transformed into EGON. The programmed synchronized positions are deleted.
• If the following axis is **not stationary**: The NST "Feed Stop/Spindle Stop" DB31, ... DBX4.3 has no direct influence on the electronic gearbox. As before, it does have an indirect effect on the leading axes, if these are located in the same channel.

• For **channel specific** feed enabling and for override nothing is implemented. Override still has no direct influence on the electronic gearbox. The **axis-specific** feed enable is set, depending on the current override setting.

**Synchronization for EGONSYNE**

With EGONSYN(), the positions of the leading axes and the synchronization position for the following axis are specified by the command.

The control system moves the following axis to the synchronized position according to the program approach mode.

**Synchronization abort with EGONSYN and EGONSYNE**

1. The EGONSYN/EGONSYNE command is aborted under the following conditions and changed to an EGON command:
   • RESET
   • Axis switches to tracking

   The defined synchronization positions are ignored. Synchronous traverse monitoring still takes synchronized positions into account.

   Aborting position synchronization generates alarm 16774.

   The alarm may be suppressed with the following machine data:

   MD11410 $MN_SUPPRESS_ALARM_MASK Bit31 = 1

**Synchronous monitoring**

The synchronism of the gearbox is monitored in each interpolator cycle on the basis of the actual values of the following and leading axes. For this purpose, the actual values of the axes are computed according to the rule of motion of the coupling. The **difference in synchronism** is the difference between the actual value of the following axis and the value calculated from the leading axis actual values according to the rule of motion. The synchronous operation difference can be queried from the part program (see below).
Changes in the difference in synchronous traverse

The mass inertia of the axis systems during acceleration can cause dynamic fluctuations in the difference in synchronous traverse. The difference in synchronous traverse is checked continuously and the tolerance values in the machine data used to produce interface signals.

The difference in synchronous traverse is compared with the following machine data:

MD37200 $MA\_COUPLE\_POS\_TOL\_COARSE
and
MD37210 $MA\_COUPLE\_POS\_TOL\_FINE

Depending on the result of this comparison, the following signals are set:
NST "Synchronous travel fine" DB31, ... DBX98.0
and
NST "synchronous traverse coarse" DB31, ... DBX98.1

Difference > .. TOL_COARSE

As long as the synchronous traverse difference is greater than the following machine data, the gearbox is not synchronized and neither IS "Coarse synchronism" DB 31, ... DBX 98.1 nor IS "Fine synchronism" DB 31, ... DBX 98.0 is active.

MD37200 $MN\_COUPLE\_POS\_TOL\_COARSE

Instead, the following interface signal is displayed:
NST "synchronization running" DB31, ... DBX99.4

Difference < .. TOL_COARSE

As long as the synchronous traverse difference is smaller than the following machine data, IS "Coarse synchronism" DB 31, ... DBX 98.1 is at the interface and IS "Fine synchronism" DB31, ... DBX99.4 is deleted.

MD37200 $MN\_COUPLE\_POS\_TOL\_COARSE

Difference > .. TOL_FINE

If synchronous traverse difference is smaller than the following machine data, then NST "synchronous traverse fine" DB31, ... DBX98.0 is at the interface:

MD37210 $MA\_COUPLE\_POS\_TOL\_FINE
Difference in synchronism for EG cascades

Deviation in synchronism for EG cascades is the deviation of the actual position of the following axis from setpoint position that results from the rule of motion for the real axes involved.

Example:

According to the definition given, the difference in synchronism of following axis FA3 in the example below is determined by the value of following axis FA3\text{Act} and the value of leading axis FA2\text{Act} and LA2\text{Act}, but not by LA1\text{Act} and FA1\text{Act}.

If FA2 is not a real axis, the actual value FA2\text{Act} is not available. In this case, the setpoint of the axis derived solely from the leading axis value FA1\text{Act} must be used instead of the actual value of the setpoint of the axis.

Other signals

If an EGON(), EGONSYN() or EGONSYNE() block is encountered in the main run, the signal "Coupling active" is set for the following axis. If the following axis is only overlaid, the signals "Coupling active" and "Axis override" are set. If EGON(), EGONSYN() or EGONSYNE() is active and the following axis is also overlaid, the signals "Coupling active" and "Axis override" are also set.

IS "following spindle active" DB31, ... DBX 99.1: Coupling active,
IS "overlaid movement" DB31, ... DBX98.4: axis is overlaid,
IS "Enable following axis override" DB31, ... DBX26.4

In the case of the commands EGON() and EGONSYNE(), the "Enable following axis override" signal must be present for the gear to synchronize to the specified synchronization position for the following axis. If it is not present, alarm 16771 "Override movement not enabled" is issued. If the signal is present, the following axis travels to the synchronized position with the calculated acceleration and at the velocity set for the approach mode.
Further monitoring signals

Machine data MD37550 $MA_EG_VEL_WARNING allows a percentage of the speeds and accelerations to be specified in the following machine data MD32000 $MA_MAX_AX_VELO and MD32300 $MA_MAX_AX_ACCEL, with reference to the following axis, which results in the generation of the following interface signals:

IS "Speed warning threshold" DB31, ... DBX98.5
IS "Acceleration warning threshold" DB31, ... DBX98.6

The monitoring signals can be used as trigger criteria for emergency retraction (see Section "Trigger sources (Page 376)").

Machine data MD37560 $MA_EG_ACC_TOL allows a percentage with reference to machine data MD32300 $MA_MAX_AX_ACCEL of the following axis to be defined, and the IS signal "axis accelerates" DB31, ... DBX99.3 to be generated.

Request synchronous traverse difference

1. The result of the synchronism difference calculation can be read as an amount in the part program with system variable $VA_EG_SYNCDIFF. The relevant value with sign is available in the system variables $VA_EG_SYNCDIFF_S. The following meanings apply:
   - Negative value (in positive traverse direction for lead and following axis): The following axis lags behind its calculated setpoint position.
   - Positive value (in positive traverse direction for lead and following axis): The following axis leads before its calculated setpoint position (overswing).

   The amount of the synchronization difference with sign corresponds to the system variables without sign from $VA_EG_SYNCDIFF.

   $VA_EG_SYNCDIFF[ax] = ABS($VA_EG_SYNCDIFF_S[ax])

Block change mode

1. When an EG axis group is activated, it is possible to specify the conditions under which a part program block change is to be executed:

2. The specification is made with a string parameter with the following meaning:

3. "NOC": Immediate block change

4. "FINE": Block change if "Fine synchronism" is present

5. "COARSE": Block change if "Coarse synchronism" is present

6. "IPOSTOP": Block change if "Setpoint synchronism" is present

Note

When programmed in activation calls EGON, EGONSYN, EGONSYNE, each of the above strings can be abbreviated to the first two characters.

If no block change has been defined for the EG axis group and none is currently specified, "FINE" applies.
4.4.2 Definition of an EG axis group

Note
The following definition commands and switch instructions of the electronic gearbox must all be contained in only one block of a parts program.

All commands of the electronic gearbox result in a preprocessing stop, except for the activation commands:
- EGON
- EGONSYN
- EGONSYNE

Definition and activation
The definition described below and activation are separate processes. An activation is not possible unless it has been defined previously.

Definition of an EG axis group
An EG axis group is defined through the input of the following axis and at least one, but not more than five, leading axis, each with the relevant coupling type:

`EGDEF(following axis, leading axis1, coupling type1, leading axis2, coupling type2,...)`

The coupling type does not need to be the same for all leading axes and must be programmed separately for each individual leading axis.

Coupling type:
Evaluate actual value of leading axis: 0
Evaluate setpoint of leading axis: 1

The coupling factors are preset to zero when the EG axis group is defined. As such, the group has no effect on the following axis until it is activated. (See EGON, EGONSYN, EGONSYNE).

Preconditions for defining an EG axis group:
- No existing axis coupling may already be defined for the following axis. (If necessary, an existing axis must be deleted with EGDEL.)
- EGDEF triggers preprocessing stop with alarm.

For an example of how to use the EG gearbox for gear hobbing, please see Chapter "Examples", "Electronic Gearbox for Gear Hobbing".

EGDEF
The gearbox definition with EGDEF should also be used unaltered when one or more leading axes affect the following axis via a curve table.

The variant extended with the addition of non-linear coupling via curve tables is illustrated in an extended example in Chapter "Extended Example with non-linear Components".
4.4.3 Activating an EG axis group

Without synchronization

The EG axis group is activated without synchronization selection with:

\[
\text{EGON}(\text{FA}, \text{block change mode}, \text{LA}1, \text{Z}1, \text{N}1, \text{LA}2, \text{Z}2, \text{N}2, \ldots \text{LA}5, \text{Z}5, \text{N}5.)
\]

The coupling is activated immediately.

With:

FA: Following axis

Depending on block change mode, the next block will be activated:

- "NOC": Block change takes place immediately
- "FINE": Block change is performed in "Fine synchronism"
- "COARSE": Block change is performed in "Coarse synchronism"
- "IPOSTOP": Block change is performed for setpoint-based synchronism

LA\textsubscript{i}: Axis identifier of the leading axis \textsubscript{i}

Z\textsubscript{i}: Counter for coupling factor of leading axis \textsubscript{i}

N\textsubscript{i}: Denominator for coupling factor of leading axis \textsubscript{i}

Only the leading axes previously specified with the EGDEF command may be programmed in the activation line. At least one leading axis must be programmed.

The positions of the leading axes and following axis at the instant the grouping is switched on are stored as "Synchronized positions". The "Synchronized positions" can be read with the system variable $AA\_EG\_SYN.

With synchronization

The EG axis group is activated with synchronization selective:

1. EGONSYN

\[
\text{EGONSYN}(\text{FA}, \text{block change mode}, \text{SynPosFA}, \text{LA}i, \text{SynPosLA}i, \text{Z\_LA}i, \text{N\_LA}i)
\]

With:

FA: Following axis

Block change mode:

- "NOC": Block change takes place immediately
- "FINE": Block change is performed in "Fine synchronism"
- "COARSE": Block change is performed in "Coarse synchronism"
- "IPOSTOP": Block change is performed for setpoint-based synchronism
SynPosFA: Synchronized position of the following axis
LAi: Axis identifier of the leading axis i
SynPosLAi: Synchronized position of leading axis i
Zi: Counter for coupling factor of leading axis i
Ni: Denominator for coupling factor of leading axis i

**Note**
The parameters indexed with i must be programmed for at least one leading axis, but for no more than five.

Only leading axes previously specified with the EGDEF command may be programmed in the activation line.

Through the programmed "Synchronized positions" for the following axis (SynPosFA) and for the leading axes (SynPosLA), positions are defined for which the axis grouping is interpreted as synchronous. If the electronic gear is not in the synchronized state when the grouping is switched on, the following axis traverses to its defined synchronized position.

The position specification of the synchronized positions is specified in the configured basic system independently of the programmable dimensions (G70/G71).

If the axis grouping includes modulo axes, their position values are reduced in the modulo, thereby ensuring that they approach the fastest possible synchronized position. (So-called relative synchronization.)

If the following interface signal has not been set for the following axis, the axis will not travel to the synchronization position.

DB31, ... DBX26.4 (Enable following axis override)

Instead the program is stopped at the EGONSYN block and the self-clearing alarm 16771 is issued until the above mentioned signal is set.

**2. EGONSYN**

```plaintext
EGONSYN(FA, block change mode, SynPosFA, Approach mode, LAi, SynPosLAi, Z_LAi, N_LAi)
```

with:

"FA": Following axis

**Block change mode:**

"NOC": Block change takes place immediately

"FINE": Block change is performed in "Fine synchronization"

"COARSE": Block change is performed in "Coarse synchronization"
"IPOSTOP": Block change is performed for setpoint-based synchronism

SynPosFA: Synchronized position of the following axis

Approach mode:

"NTGT": NextToothGapTime-optimized, the next tooth gap is approached time-optimized (preset is used if no setting is applied).

"NTGP": The next tooth gap is approached time-optimized.

"ACN": AbsoluteCo-ordinateNegative, Absolute measurement specification, rotary axis traverses in negative rotation direction

"ACP": AbsoluteCo-ordinatePositive, Absolute measurement specification, rotary axis traverses in positive rotation direction

"DCT": DirectCo-ordinateTime-optimized, Absolute measurement specification, rotary axis traverses time-optimized to programmed synchronized position

"DCP": DirectCo-ordinatePath-optimized, Absolute measurement specification, rotary axis traverses path-optimized to programmed synchronized position

LAi: Axis identifier of the leading axis i

SynPosLAi: Synchronized position of leading axis i

Zi: Counter for coupling factor of leading axis i

Ni: Denominator for coupling factor of leading axis i

Note

The parameters indexed with i must be programmed for at least one leading axis, but for no more than five.

The function is active only for modulo following axes that are coupled to modulo leading axes.

Tooth gap

The tooth gap is defined as 360 degrees * Zi / Ni

Example:

EGONSYNE(A, "FINE", FASysPos, "Traversing mode", B, LASynPos, 2, 10)

Tooth gap: 360*2/10 = 72 (degrees)
Approach response with FA at standstill

In this case, the time-optimized and path-optimized traversing modes are identical.

The table below shows the target positions and traversed paths with direction marker (in brackets) for the particular approach modes:

| Programmed synchronized position FaSysPos | Position of the following axis before EGONSYNE | Traversing mode NTGT/NTGP | Traversing mode DCT/DCP | Traversing mode ACP | Traversing mode ACN |
|------------------------------------------|-----------------------------------------------|---------------------------|--------------------------|---------------------|---------------------|
| 110                                      | 150                                           | 182 (+32)                 | 110 (-40)                | 110 (+320)          | 110 (-40)          |
| 110                                      | 350                                           | 326 (-24)                 | 110 (+120)               | 110 (+120)          | 110 (-240)         |
| 130                                      | 0                                             | 346 (-14)                 | 130 (+130)               | 130 (+130)          | 130 (-230)         |
| 130                                      | 30                                            | 58 (+28)                  | 130 (+100)               | 130 (+100)          | 130 (-260)         |
| 130                                      | 190                                           | 202 (+12)                 | 130 (-60)                | 130 (+300)          | 130 (-60)          |
| 190                                      | 0                                             | 334 (-26)                 | 190 (-170)               | 190 (+190)          | 190 (-170)         |
| 230                                      | 0                                             | 14 (+14)                  | 230 (-130)               | 230 (+230)          | 230 (-130)         |

Approach response for moving FA

The following axis moves at almost maximum velocity in the positive direction when the coupling is activated by EGONSYNE. The programmed synchronized position of the following axis is 110, the current position 150. This produces the two alternative synchronized positions 110 and 182 (see table above).

In the case of traversing mode NTGP (path-optimized), synchronized position 182 is selected independent of the current velocity. This has the shortest distance from the current position of the following axis. Traversing mode NTGT (time-optimized) considers the current speed of the following axis and produces a deceleration on account of the limit for the maximum axis speed to reach synchronism in the shortest possible time (see Figure).

Figure 4-9  Reaching the next tooth gap, FA path-optimized (top) vs. time-optimized (bottom)
M3: Coupled axes

4.4 Electronic gear (EG) - 840D sl only

Sample notations

\[
\text{EGONSYNE}(A, \text{"FINE"}, 110, \text{"NTGT"}, B, 0, 2, 10)
\]

couple A to B, synchronized position A = 110, B = 0, coupling factor 2/10, approach mode = NTGT

\[
\text{EGONSYNE}(A, \text{"FINE"}, 110, \text{"DCT"}, B, 0, 2, 10)
\]

couple A to B, synchronized position A = 110, B = 0, coupling factor 2/10, approach mode = DCT

\[
\text{EGONSYNE}(A, \text{"FINE"}, 110, \text{"NTGT"}, B, 0, 2, 10, Y, 15, 1, 3)
\]

couple A to B, synchronized position A = 110, B = 0, Y = 15, Coupling factor to B = 2/10, Coupling factor to Y = 1/3, approach mode = NTGT

With synchronization

The syntax specified above applies with the following different meanings:

If a curve table is used for a leading axis, then the denominator of the linear coupling of the coupling factor (Ni) must be set to 0 (denominator 0 would be impermissible for linear couplings).

For control, denominator zero is the indicator that the counter of the coupling factor (Zi) is to be interpreted as the number of the curve table to be used. The curve table with the specified number must already be defined at power on (in accordance with Chapter "Curve Tables").

The leading axis specified (LAi) corresponds to the one specified for coupling via coupling factor (linear coupling).

4.4.4 Deactivating an EG axis group

Variant 1

There are different ways to deactivate an active EG axis grouping.

\[
\text{EGOFS}(\text{following axis})
\]

The electronic gearbox is deactivated. The following axis is braked to a standstill. This call triggers a preprocessing stop.
Variant 2

The following parameterization of the command makes it possible to **selectively** control the influence of individual leading axes on the motion of the following axis.

**EGOFS**(following axis, leading axis 1, ... leading axis 5)

---

**Note**

At least one leading axis must be specified.

The influence of the specified leading axes on the slave is selectively inhibited. This call triggers a preprocessing stop.

If the call still includes active leading axes, then the slave continues to operate under their influence. If the influence of all leading axes is excluded by this method, then the following axis is braked to a standstill.

If the command EGONSYN is deactivated selectively, no axis movement is performed.

Variant 3

**EGOFC**(following spindle)

The electronic gear is deactivated. The following spindle continues to traverse at the speed/velocity that applied at the instant of deactivation. This call triggers a preprocessing stop.

---

**Note**

Call for following spindles available. For EGOFC a spindle identifier must be programmed.

4.4.5 Deleting an EG axis group

An EG axis grouping must be switched off, as described in Chapter "Switching off a EG Axis Group", before its definition can be deleted.

**EGDEL**(following axis)

The defined coupling of the axis grouping is deleted. Additional axis groups can be defined by means of EGDEF until the maximum number of simultaneously activated axis groups is reached.

This call triggers a preprocessing stop.
4.4.6 Interaction between rotation feedrate (G95) and electronic gearbox

The FPR( ) part program command can be used to specify the following axis of an electronic gear as the axis, which determines the rotational feedrate. The following behavior is applicable in this case:

- The feedrate is determined by the setpoint velocity of the following axis of the electronic gear.
- The setpoint velocity is calculated from the speeds of the leading spindles and modulo axes (which are not path axes) and from their associated coupling factors.
- Velocity components from other leading axes and overlaid motions of the following axis are not taken into account.

References: /V1/, Feeds

4.4.7 Response to POWER ON, RESET, operating mode change, block search

| Function                  | Coupling state | Configuration       |
|---------------------------|----------------|---------------------|
| Mode change               | Is retained    | Is retained         |
| End of part program       | Is retained    | Is retained         |
| Reset                     | Is retained    | Is retained         |
| Power On ¹)               | Is not retained| Is not retained     |

¹) No coupling is active after Power On.

Block search

A search during active coupling (EG) is possible under the following supplementary conditions:

- Simulation is exclusively with setpoint coupling.
- No cross-channel leading axes may be disabled.
- Axis movements for which all real positions are known to the NC.

If it is technically not sensible or not possible to permit the target block for a block search with calculation or SERUPRO within a part program section with active coupling, this section can be locked for "continue machining at the contour".

Programming

- IPTRLOCK()
- IPTRUNLOCK()

References

(K1) Mode group, channel, program operation, reset response, Section "Block search type 5 SERUPRO" > "Lock program section for continue machining at the contour"
4.4.8 System variables for electronic gearbox

Application

The following system variables can be used in the part program to scan the current states of an EG axis group and to initiate appropriate reactions if necessary:

Table 4-1 System variables, R means: Read access possible

| name        | Type   | Access | Preprocessing stop | Meaning, value                                                                 | Cond. Index                          |
|-------------|--------|--------|--------------------|--------------------------------------------------------------------------------|---------------------------------------|
|             |        |        | part progr        |                                                                                 |                                       |
|             |        |        | Sync act.          |                                                                                 |                                       |
|             |        |        | part progr        |                                                                                 |                                       |
|             |        |        | Sync act.          |                                                                                 |                                       |
| $AA_EG_TYPE[a,b] | INT    | R      | R                  | Type of coupling: 0: actual value                                               | Axis identifier a: Following axis    |
|             |        |        |                    | 1: Setpoint value coupling                                                        | b: Leading axis                       |
| $AA_EG_NUMERA[a,b] | REAL  | R      | R                  | Numerator of coupl. factor                                                       | Axis identifier a: Following axis    |
|             |        |        |                    | KF                                                                               | b: Leading axis                       |
|             |        |        |                    | = numerator/denominator                                                          |                                       |
|             |        |        |                    | preset: 0                                                                         |                                       |
|             |        |        |                    | Number of curve table                                                            |                                       |
|             |        |        |                    | when $AA_EG_DENOM[a,b] is 0.                                                      |                                       |
| $AA_EG_DENOM[a,b] | REAL  | R      | R                  | Denominator of coupl. factor                                                     | Axis identifier a: Following axis    |
|             |        |        |                    | KF                                                                               | b: Leading axis                       |
|             |        |        |                    | = numerator/denominator                                                          |                                       |
|             |        |        |                    | preset: 1                                                                         |                                       |
|             |        |        |                    | Denominator must be positive.                                                    |                                       |
|             |        |        |                    | Denominator is 0, if instead of the numerator $AA_EG_NUMERA[a,b]                  |                                       |
|             |        |        |                    | the number of a curve table is specified.                                        |                                       |
| $AA_EG_SYN[a,b] | REAL  | R      | R                  | Synchronized position for specified leading axis                                 | Axis identifier a: Following axis    |
|             |        |        |                    | Default: 0                                                                        | b: Leading axis                       |
| $AA_EG_SYNFA[a] | REAL  | R      | R                  | Synchronized position for specified following axis                               | Axis identifier a: Following axis    |

Special functions

Function Manual, 07/2012, 6FC5397-2BP40-3BA0 261
## M3: Coupled axes

### 4.4 Electronic gear (EG) - 840D sl only

#### Special functions

| name             | Type    | Access | Preprocessing stop | Meaning, value                                                                                                                                                                                                 | Cond. Index |
|------------------|---------|--------|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| $P_{EG\_BC[a]}$  | STRING  | R      | R                  | Block change criterion for EG activation calls: EGON, EGONSYN:  
  "NOC": immediately  
  "FINE": Synchronous traverse fine  
  "COARSE": Synchronous traverse coarse  
  "IPOSTOP": Setpoint-based synchronism                                                                                                           | Axis identifier  
  a: Following axis |
| $AA_{EG\_NUM\_LA[a]}$ | INT     | R      | R                  | Number of leading axes defined with EGDEF. 0 if no axis has been defined as a following axis with EGDEF.                                                                                                      | Axis identifier  
  a: Following axis |
| $AA_{EG\_AX[n,a]}$ | AXIS    | R      | R                  | Axis identifier of leading axis whose index n has been specified.                                                                                                                                           | Axis identifier  
  n: Index of leading axis in the EG coupling 0 ... 4  
  a: Following axis |
| $AA_{EG\_ACTIVE[a,b]}$ | BOOL    | R      | R                  | Determine power-on state of a leading axis: 0: Switched off 1: activated                                                                                                                                   | Axis identifier  
  a: Following axis  
  b: Leading axis |
| $VA_{EG\_SYNCDIFF[a]}$ | REAL    | R      | R                  | Actual value of synchronism difference. A comparison of machine data MD37200 $MA_{COUPLE\_POS\_TOL\_COARSE}$ and MD37210 $MA_{COUPLE\_POS\_TOL\_FINE}$ produces interface signals. | Axis identifier  
  a: Following axis |
4.4.9 Examples

4.4.9.1 Example using linear couplings

Use of axes

The following diagram shows the configuration of a typical gear hobbing machine. The machine comprises five numerically closed loop controlled axes and an open loop controlled main spindle. These are:

- The rotary motion of the workpiece table (C) and hobbing cutter (B).
- The axial axis (Z) for producing the feed motion over the entire workpiece width.
- The tangential axis (Y) for moving the hobbing cutter along its axis.
- The radial axis (X) for infeeding the cutter to depth of tooth.
- The cutter swivel axis (A) for setting the hobbing cutter in relation to the workpiece as a function of cutter lead angle and angle of inclination of tooth.

![Diagram of gear hobbing machine axes](image)

**Legend:**
- X = Radial axis
- Y = Tangential axis (master drive 3)
- Y = Tangential axis (master drive 2)
- A = Milling swivel axis
- C = Workpiece rotational axis (following drive)
- B = Milling rotational axis, Main spindle (Main drive 1)

Figure 4-10 Definition of axes on a gear hobbing machine (example)
The functional interrelationships on the gear hobbing machine are as follows:

\[
\begin{align*}
\text{B} & \quad \text{Z} & \quad \text{Y} \\
\text{LA 1} & \quad \text{LA 2} & \quad \text{LA 3} \\
\text{Workpiece width} & \quad \text{Longitudinal axis of cutter} & \\
\text{Rotation of cutter} & & \text{Rotation of cutter} \\
\text{C} & & \text{C} \\
\text{FA} & & \text{FA} \\
\text{Rotary axis of workpiece (gear tooth)} & & \text{Rotary axis of workpiece (gear tooth)}
\end{align*}
\]

In this case, the workpiece table axis (C) is the following axis which is influenced by three master drives.

The setpoint of the following axis is calculated cyclically with the following logic equation:

\[
n_c = n_b \cdot \left( \frac{z_0}{z_2} \right) + v_z \cdot \left( \frac{u_{dz}}{z_2} \right) + v_y \cdot \left( \frac{u_{dy}}{z_2} \right)
\]

with:
- \( n_c \) = Rotational speed of workpiece axis (C)
- \( n_b \) = Rotational speed of milling spindle (B)
- \( z_0 \) = Number of gears of the hobbing machine
- \( z_2 \) = Number of teeth of the workpiece
- \( v_z \) = Feed velocity of axial axis (Z)
- \( v_y \) = Feed velocity of tangential axis (Y)
- \( u_{dz} \) = Axial differential constant
- \( u_{dy} \) = Tangential differential constant

**Quantities which influence the setpoint of workpiece axis C**

The first addend of the above equation determines the speed ratio between workpiece table and cutter, and thus the number of teeth of the workpiece.

The second addend effects the necessary additional rotation of the C axis as a function of the axial feed motion of the cutter to produce the tooth inclination on helical teeth.

The third component also makes allowance for additional rotation of the C axis to compensate for the tangential movement of the cutter in relation to the workpiece, thus ensuring that the tool is equally stressed over its entire length.
Workpiece/tool parameter

The values $z_0$, $z_2$, $u_{dz}$, and $u_{dy}$ are workpiece- or tool-dependent and are specified by the NC operator or in the part program.

Differential constants

Differential constants $u_{dz}$ and $u_{dy}$ make allowance for the angle of workpiece teeth and for cutter geometry. These differential constants can be determined in user-specific cycles.

\[
\begin{align*}
  u_{dz} &= \left(\sin \beta^\circ \div (m_n \times \pi)\right) \times 360 \quad \text{[degrees/mm]} \\
  u_{dy} &= \left(\cos \gamma^\circ \div (m_n \times \pi)\right) \times 360 \quad \text{[degrees/mm]}
\end{align*}
\]

with:
- $m_n$ = Normal module (in mm)
- $\beta^\circ$ = Incline angle of gear wheel
- $\gamma^\circ$ = Pitch angle of hobbing machine

Extract from part program:

| Program code | Comment |
|--------------|---------|
| EGDEF(C,B,1,Z,1,Y,1) | ; Definition of EG axis grouping with setpoint coupling (1) from B, Z, Y to C (following axis). |
| EGON(C,"FINE",B,z0,z2,Z,u_{dz},z2,Y,u_{dy},z2) | ; Activate coupling. |
| ... | |

4.4.9.2 Extended example with non-linear components

Introduction

The following example extends the example (see "Figure 4-10 Definition of axes on a gear hobbing machine (example) (Page 263)") with the following:

- Machine error compensations which are not linearly dependent on the Z axis, and
- A Z-axis dependent component with tooth geometry.

This can be used to create a slightly convex tooth surface, so that the centre of the tooth is stressed more than the edges during operation.

Figure 4-11 Extended example with non-linear machine fault compensation and non-linear components on the tooth geometry
The following section of a part program is intended to illustrate the general concept; supplementary curve tables and gear wheel/machine parameters are still to be added. Components to be added are marked with <...> . Stated parameters may also have to be modified, e.g. coupling factors.

| Program code | Comment |
|--------------|---------|
| N100         | CTABDEF(X, Z, 1, 0) ; declaration and specification of non-periodic curve table C1 |
| N110         | < ... > ; Preset of curve table: Curve points or polynomial blocks |
| N190         | CTABEND |
| N200         | CTABDEF(Y, Z, 2, 0) ; declaration and specification of non-periodic curve table C2 |
| N210         | < ... > ; Preset of curve table: Curve points or polynomial blocks |
| N290         | CTABEND |
| N300         | CTABDEF(A, Z, 3, 0) ; declaration and specification of non-periodic curve table C3 |
| N310         | < ... > ; Preset of curve table: Curve points or polynomial blocks |
| N390         | CTABEND |
| N400         | CTABDEF(C, Z, 4, 0) ; declaration and specification of non-periodic curve table C4 |
| N410         | < ... > ; Preset of curve table: Curve points or polynomial blocks |
| N490         | CTABEND |
| N500         | EGDEF(X, Z, 1) ; Declaration of path via C1, setpoint coupling |
| N510         | G1 F1000 X10 ; declaration of command component of X |
| N520         | EGONSYN(X, "NOC", <SynPosX>, Z, <SynPosX_Z>, 1, 0) ; Path switch-on via C1 |
| N600         | EGDEF(Y, Z, 1) ; Declaration of path via C2, setpoint coupling |
| N610         | G1 F1000 Y10 ; declaration of command component of Y |
| N620         | EGONSYN(Y, "COARSE", <SynPosY>, Z, <SynPosY_Z>, 2, 0) ; Path switch-on via C2 |
| N700         | EGDEF(A, Z, 1) ; Declaration of path via C3, setpoint coupling |
| N710         | G1 F1000 A10 ; declaration of command component of A |
| N720         | EGONSYN(A, "FINE", <SynPosA>, Z, <SynPosA_Z>, 3, 0) ; Path switch-on via C3 |
| N800         | EGDEF(C99, Y, 1, Z, 1, B, 1 ) ; 1. Gear stage, C99 is the software axis between the two electronic gears |
| N810         | EGONSYN(C99, "NOC", ; Switch-on of leading axis B |
### Program code

| Program code | Comment |
|--------------|---------|
| `<SynPosC99>`, B,  
  `<SynPosC99_B>`, 18,  
  2, &  
  Y, `<SynPosC99_Y>`,  
  R1 * n, 1, &  
  Z, `<SynPosC99_Z>`,  
  10, 1) | ; Switch-on of leading axis Y  
  ; Switch-on of leading axis Z  
  ; "&" character means: command continued in next line, no LF nor comment permissible in program |
| N900  
  EGDEF(C, C99, 1, Z,  
  1) | ; declaration of following C99 of step 1 as leading axis of step 2,  
  ; Setpoint value coupling |
| N910 | ; Declaration of path via C4, setpoint coupling |
| N92D  
  EGONSYN(C, "NOC",  
  `<SynPosC>`, C99,  
  `<SynPosC_C99>`, 1,  
  1, &  
  Z, `<SynPosC_Z>`, 4,  
  0) | ; Switch-on of software axis C99  
  ; and of leading axis Z via C4 |
| N999 | M30 |

### Machine data

Only one section is specified, which extends beyond the necessary geometry/channel configuration and machine axis parameters.

```
$MN_NUM_EG = 5 ; Maximum number of gearboxes
$MN_MM_NUM_CURVE_TABS = 5 ; Maximum number of curve tables
$MN_MM_NUM_CURVE_SEGMENTS = 50 ; Maximum number of curve segments
$MN_MM_NUM_CURVE_POLYNOMS = 100 ; Maximum number of curve polynomials
```

### Setting data

If the scaling described in Section "Electronic gear (EG) - 840D sl only (Page 244) is used, the function value from the following machine data changes according to the displacement:

```
MD43108 $SD_LEAD_SCALE_OUT_POS[4] = 1.2 ; scaling for table C4
```
System variables

In accordance with the above definitions, the following values are entered in the associated system variables by the control.

Reference:
Parameter Manual, System Variables

The system variables listed below are only used for explanatory purposes!

; ************** Gear X (G1)
$AA_EG_TYPE[X, Z] = 1 ; Setpoint value coupling
$AA_EG_NUMERA[X, Z] = 1 ; curve table No. = 1
$AA_EG_DENOM[X, Z] = 0 ; nominator = 0 → curve table applies
&P_EG_BC[X] = "NOC" ; Block change criterion
$AA_EG_NUM_LA[X] = 1 ; Number of leading axes
$AA_EG_AX[0, X] = Z ; leading axis identifier
$AA_EG_SYN[X, Z] = <SynPosX_Z> ; Synchronized position of leading axis Z
$AA_EG_SYNFA[X] = <SynPosX> ; Synchronized position of the following axis

; ************** Gear Y (G2)
$AA_EG_TYPE[Y, Z] = 1 ; Setpoint value coupling
$AA_EG_NUMERA[Y, Z] = 2 ; curve table No. = 2
$AA_EG_DENOM[Y, Z] = 0 ; nominator = 0 → curve table applies
&P_EG_BC[Y10] = "COARSE" ; Block change criterion
$AA_EG_NUM_LA[Y] = 1 ; Number of leading axes
$AA_EG_AX[0, Y] = Z ; leading axis identifier
$AA_EG_SYN[Y, Z] = <SynPosY_Z> ; Synchronized position of leading axis Z
$AA_EG_SYNFA[Y] = <SynPosY> ; Synchronized position of the following axis

; ************** Gear A (G3)
$AA_EG_TYPE[A, Z] = 1 ; Setpoint value coupling
$AA_EG_NUMERA[A, Z] = 3 ; curve table No. = 3
$AA_EG_DENOM[A, Z] = 0 ; nominator = 0 → curve table applies
&P_EG_BC[A10] = "FINE" ; Block change criterion
$AA_EG_NUM_LA[A] = 1 ; Number of leading axes
$AA_EG_AX[0, A] = Z ; leading axis identifier
$AA_EG_SYN[A, Z] = <SynPosA_Z> ; Synchronized position of leading axis Z
$AA_EG_SYNFA[A] = <SynPosA> ; Synchronized position of the following axis

; ************** Gear C99 (G4)
$AA_EG_TYPE[C99, Y] = 1 ; Setpoint value coupling
$AA_EG_NUMERA[C99, Y] = 18 ; numerator for coupling factor_y
$AA_EG_DENOM[C99, Y] = 2 ; denominator for coupling factor_y
$AA_EG_TYPE[C99, Z] = 1 ; Setpoint value coupling
$AA_EG_NUMERA[C99, Z] = R1 * π ; numerator for coupling factor_z
$AA_EG_DENOM[C99, Z] = 1 ; denominator for coupling factor_z
M3: Coupled axes

4.4 Electronic gear (EG) - 840D sl only

$AA_EG_TYPE[C99, B] = 1 ; Setpoint value coupling
$AA_EG_NUMERA[C99, B] = 10 ; numerator for coupling factor b
$AA_EG_DENOM[C99, B] = 1 ; denominator for coupling factor b
$P_EG_BC[C99] = "NOC" ; Block change criterion
$AA_EG_NUM_LA[C99] = 3 ; Number of leading axes
$AA_EG_AX[0, C99] = Y ; leading axis Y identifier
$AA_EG_AX[1, C99] = Z ; leading axis Z identifier
$AA_EG_AX[2, C99] = B ; leading axis B identifier
$AA_EG_SYN[C99, Y] = <SynPosC99_Y> ; Synchronized position of leading axis Y
$AA_EG_SYN[C99, Z] = <SynPosC99_Z> ; Synchronized position of leading axis Z
$AA_EG_SYN[C99, B] = <SynPosC99_B> ; Synchronized position of leading axis B
$AA_EG_SYNFA[C99] = <SynPosC99> ; Synchronized position of the following axis

; *************** Gear C (G5)
$AA_EG_TYPE[C, Z] = 1 ; Setpoint value coupling
$AA_EG_NUMERA[C, Z] = 4 ; curve table No. = 4
$AA_EG_DENOM[C, Z] = 0 ; nominator = 0 -> curve table applies
$AA_EG_TYPE[C, C99] = 1 ; Setpoint value coupling
$AA_EG_NUMERA[C, C99] = 1 ; numerator for coupling factor C99
$AA_EG_DENOM[C, C99] = 1 ; denominator for coupling factor C99
$P_EG_BC[C] = "NOC" ; Block change criterion
$AA_EG_NUM_LA[C] = 2 ; Number of leading axes
$AA_EG_AX[0, C] = Z ; leading axis Z identifier
$AA_EG_AX[1, C] = C99 ; leading axis C99 identifier
$AA_EG_SYN[C, Z] = <SynPosC_Z> ; Synchronized position of leading axis Z
$AA_EG_SYN[C, C99] = <SynPosC_C99> ; Synchronized position of leading axis C99
$AA_EG_SYNFA[C] = <SynPosC> ; Synchronized position of leading axis C
Machine data

Extract from MD:

; *************** Channel 1
CHANDATA(1)
; *************** Axis 1, "X"
$MC_AXCONF_GEOAX_NAME_TAB[0] = "X"
$MC_AXCONF_CHANAX_NAME_TAB[0] = "X"
$MC_AXCONF_MACHAX_USED[0]=1
$MN_AXCONF_MACHAX_NAME_TAB[0] = "X1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX1] = 0
$MA_IS_ROT_AX[AX1] = FALSE

; *************** Axis 2, "Y"
$MC_AXCONF_GEOAX_NAME_TAB[1]="Y"
$MC_AXCONF_CHANAX_NAME_TAB[1] = "Y"
$MC_AXCONF_MACHAX_USED[1] = 2
$MN_AXCONF_MACHAX_NAME_TAB[1] = "Y1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX2] = 0
$MA_IS_ROT_AX[AX2] = FALSE

; *************** Axis 3, "Z"
$MC_AXCONF_GEOAX_NAME_TAB[2] = "Z"
$MC_AXCONF_CHANAX_NAME_TAB[2] = "Z"
$MC_AXCONF_MACHAX_USED[2] = 3
$MN_AXCONF_MACHAX_NAME_TAB[2] = "Z1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX3] = 0
$MA_IS_ROT_AX[AX3] = FALSE

; *************** Axis 4, "A"
$MC_AXCONF_CHANAX_NAME_TAB[3] = "A"
$MC_AXCONF_MACHAX_USED[3]=4
$MN_AXCONF_MACHAX_NAME_TAB[3] = "A1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX4]=0
$MA_IS_ROT_AX[AX4] = TRUE
$MA_ROT_IS_MODULO[AX4] = TRUE

; *************** Axis 5, "B"
$MC_AXCONF_CHANAX_NAME_TAB[4] = "B"
$MC_AXCONF_MACHAX_USED[4]=5
$MC_SPIND_DEF_MASTER_SPIND = 1
$MN_AXCONF_MACHAX_NAME_TAB[4] = "B1"
$MA_SPIND_ASSGN_TO_MACHAX[AX5] = 1
$MA_IS_ROT_AX[AX5] = TRUE
$MA_ROT_IS_MODULO[AX5] = TRUE
; ************** Axis 6, "C"
$MC_AXCONF_CHANAX_NAME_TAB[5] = "C"
$MC_AXCONF_MACHAX_USED[5]=6
$MN_AXCONF_MACHAX_NAME_TAB[5] = "C1"
$MA_SPIND_ASSGN_TO_MACHAX[AX6] = 0
$MA_IS_ROT_AX[AX6] = TRUE
$MA_ROT_IS_MODULO[AX6] = TRUE
; ************** Axis 10, "C99"
$MC_AXCONF_CHANAX_NAME_TAB[9] = "C99"
$MC_AXCONF_MACHAX_USED[9]=10
$MA_SPIND_ASSGN_TO_MACHAX[AX10] = 0
$MA_IS_ROT_AX[AX10] = TRUE
$MA_ROT_IS_MODULO[AX10] = TRUE
4.5 Generic coupling

4.5.1 Brief description

4.5.1.1 Function

Function

"Generic Coupling" is a general coupling function, combining all coupling characteristics of existing coupling types (coupled motion, master value coupling, electronic gearbox and synchronous spindle).

The function allows flexible programming:

- Users can select the coupling properties required for their applications (building block principle).
- Each coupling property can be programmed individually.
- The coupling properties of a defined coupling (e.g. coupling factor) can be changed.
- Later use of additional coupling properties is possible.
- The coordinate reference system of the following axis (Base co-ordinate system or Machine co-ordinate system) is programmable.
- Certain coupling properties can also be programmed with synchronous actions.

References:
Function Manual, Synchronized Actions

Adaptive cycles

Previous coupling calls for coupled motion (TRAIL*), Master value coupling (LEAD*), Electronic Gearbox (EG*) and Synchronous spindle (COUP*) are still supported via adaptive cycles (see Section "Adaptive cycles (Page 330)").
4.5.1.2 Requirements

CP Versions

Generic coupling is available in a basic version as part of the NCK software and the four optional versions CP_STATIC, CP-BASIC, CP-COMFORT and CP-EXPERT.

This structure is based on the following considerations:

- Functional scope and required application knowledge increase from the basic version to the optional CP_EXPERT version.
- The number of required couplings (following axes, following spindles) and their properties are decisive in the selection of versions.
  - Example of simultaneous operation:
    If sequential operation of 1 x synchronous spindle pair for part transfer from the main to the supplementary spindle and the 1 x multi-edge turning is required, then the CP-BASIC option is suitable and sufficient. However, if it cannot be excluded that both operations can overlap (multi-edge turning running when part transfer is started), the CP-COMFORT option would be required.
  - Example of property:
    If a coupled axis grouping with a leading axis is required, the base version is sufficient. For coupled motion groups with two leading axes, one of the optional versions is required.
- Individual versions are independent of each other. They can be combined and can be activated simultaneously.

Table 4-2 Scaling of the number of simultaneously permitted coupling modules

| Type | CP versions allow one or more different CPSETTYPE coupling objects simultaneously: | Base version | CP_STATIC | CP_BASIC | CP_COMFORT | CP_EXPERT |
|------|----------------------------------------------------------------------------------|--------------|-----------|-----------|------------|-----------|
| A    | Coupled motion                                                                   |              |           |           |            | 4         |
| B    | Synchronous spindle with |1|:1 coupling                                           | -           | 1         | -         | -         | -         |
| C    | o./u. Synchronous spindle/multi-edge turning                                      | -           | -         | 1         | 4          | 8         |
|      | o./u. Master value coupling / curve tables interpolation                           |              |           |           |            |           |
|      | o./u. MCS coupling                                                                |              |           |           |            |           |
| D    | o./u. Electronic gearbox "plain"                                                  | -           | -         | -         | 1          | 8         |
|      | o./u. Free generic coupling "plain"                                               |              |           |           |            |           |
| E    | o./u. Electronic gearbox                                                          | -           | -         | -         | -          | 5         |
|      | o./u. Free generic coupling                                                       |              |           |           |            |           |

o./u. stands for over/under
Table 4-3  Scaling of availability of coupling properties

| Type A | Type B | Type C | Type D | Type E |
|--------|--------|--------|--------|--------|
| 20     | 1      | 13     | 9      | 5      | Maximum number of CPSETTYPE-related functionalities (per type) |

TRAIL - Coupled motion

| Type A | Type B | Type C | Type D | Type E |
|--------|--------|--------|--------|--------|
| 20     | -      | 13     | 9      | 5      | Maximum number of coupled motion groups with the following properties: → refer to CPSETTYPE="TRAIL"¹ |
| 1      | 2      | 2      | 2      |        | Maximum number of master values |
| -      | +      | +      | +      | From part program and synchronous actions |
| +      | +      | +      | +      | Superimposition / speed difference permitted |
| -      | -      | -      | +      | Cascading permitted |
| BCS    | BCS / MCS | BCS / MCS | BCS / MCS | Co-ordinate reference (default): CPFRS="BCS" |

Synchronous spindle with [1]:1 coupling

| Type A | Type B | Type C | Type D | Type E |
|--------|--------|--------|--------|--------|
| -      | 1      | -      | -      | -      | Maximum number of synchronous spindles / multi-edge turning with the following properties: → refer to CPSETTYPE="COUP"¹ |
| 1      |        |        |        | Maximum number of master values |
| -      |        |        |        | From part program and synchronous actions |
| -      |        |        |        | Superimposition / speed difference permitted |
| -      |        |        |        | Cascading permitted |
| MCS    |        |        |        | Co-ordinate reference fix (CPFRS="MCS") |

COUP - Synchronous spindle/multi-edge turning

| Type A | Type B | Type C | Type D | Type E |
|--------|--------|--------|--------|--------|
| -      | -      | 13     | 9      | 5      | Maximum number of synchronous spindles / multi-edge turning with the following properties: → refer to CPSETTYPE="COUP"¹ |
| 1      | 1      | 1      | 1      | Maximum number of master values |
| -      | -      | -      | -      | From part program and synchronous actions |
| +      | +      | +      | +      | Superimposition / speed difference permitted |
| -      | -      | -      | -      | Cascading permitted |
| MCS    | MCS    | MCS    |        | Co-ordinate reference fix (CPFRS="MCS") |

LEAD - Master value coupling / curve tables interpolation

| Type A | Type B | Type C | Type D | Type E |
|--------|--------|--------|--------|--------|
| -      | -      | 13     | 9      | 5      | Maximum number of master value couplings / curve tables interpolation with the following properties: → refer to CPSETTYPE="LEAD"¹ |
| 1      | 1      | 1      | 1      | Maximum number of master values |
| +      | +      | +      | +      | From part program and synchronous actions |
| +      | +      | +      | +      | Superimposition / speed difference permitted |
| -      | -      | 1      | +      | Cascading permitted |
### M3: Coupled axes

#### 4.5 Generic coupling

| Type A | Type B | Type C | Type D | Type E |
|--------|--------|--------|--------|--------|
| BCS / MCS | BCS / MCS | BCS / MCS | Co-ordinate reference (default): CPFRS="BCS" |

**EG - Electronic gearbox**

| - | - | - | 9 | 5 |
|---|---|---|---|---|
| Maximum number of electronic gearboxes with the following properties: | → refer to CPSETTYPE="EG"¹ |

| 3 | 5 |
|---|---|
| Maximum number of master values |

| + | + |
|---|---|
| Superimposition / speed difference permitted |

| - | + |
|---|---|
| Cascading permitted |

| BCS / MCS | BCS / MCS |
|---|---|
| Co-ordinate reference (default): CPFRS="BCS" |

| - | + (max. 2x) |
|---|---|
| Non-linear coupling law (CPLCTID) permitted |

**CP - Free generic coupling**

| - | - | - | 9 | 5 |
|---|---|---|---|---|
| Maximum number of free generic couplings with the following properties: | Default (corresponds to CPSETTYPE="CP"¹) |

| 3 | 5 |
|---|---|
| Maximum number of master values |

| + | + |
|---|---|
| From part program and synchronous actions |

| + | + |
|---|---|
| Superimposition / speed difference permitted |

| - | + |
|---|---|
| Cascading permitted |

| BCS / MCS | BCS / MCS |
|---|---|
| Co-ordinate reference (default): CPFRS="BCS" |

| - | + |
|---|---|
| Non-linear coupling law (CPLCTID) permitted |

¹Refer to "Coupling types (CPSETTYPE) (Page 331)."

**Note**

Existing coupling options (Master value coupling, Electronic gearbox and Synchronous spindle) are not taken into consideration by generic coupling. Simultaneous operation of existing coupling options and generic coupling is only possible if the couplings refer to different axes/spindles.
Memory configuration

Memory space reserved in dynamic NC memory for generic coupling is defined in machine data:

- MD18450 $MN_MM_NUM_CP_MODULES (maximum allowed number of CP coupling modules)
- MD18452 $MN_MM_NUM_CP_MODUL_LEAD (maximum allowed number of CP master values)

**Note**

Recommendation: Expected maximum values, which can be expected simultaneously for this machine in its maximum configuration, should already be set during commissioning.

Hardware requirements

Utilization of the "CP-EXPERT" option requires the application of:

- Systems with more than 6 axes
- NCUs ≥ NCU572 or NCU720

4.5.2 Fundamentals

4.5.2.1 Coupling module

With the aid of a coupling module, the motion of one axis, (→ following axis), can be interpolated depending on other (→ leading) axes.

Coupling rule

The relationships between leading axis/values and a following axis are defined by a coupling rule (coupling factor or curve table). The individual motion components from the individual leading axes/values have an additive effect.
The relationship is demonstrated by the following example (following axis with two leading axes):

\[ FA_{Total} = FACmd + FA_{Dep1} + FA_{Dep2} \]

\[ \text{Dependent motion component of leading axis 1: } \quad FA_{Dep1} = (LA_1 - \text{SynPosLA}_1) \times KF_1 \]

\[ \text{Dependent motion component of leading axis 2: } \quad FA_{Dep2} = (LA_2 - \text{SynPosLA}_2) \times KF_2 \]

\[ \text{Independent motion component of the following axis: } \quad FACmd \]

\[ \begin{align*}
\text{FA}_{Total} & \quad \text{Total setpoint value of the following axis} \\
\text{FACmd} & \quad \text{Setpoint value set in part program} \\
& \quad = \text{independent motion component of the following axis} \\
\text{FA}_{Dep1} & \quad \text{Dependent motion component of leading axis 1} \\
\text{FA}_{Dep2} & \quad \text{Dependent motion component of leading axis 2} \\
\text{LA}_1 & \quad \text{Setpoint or actual value of the 1st leading axis} \\
\text{LA}_2 & \quad \text{Setpoint or actual value of the 2nd leading axis} \\
\text{SynPosLA}_1 & \quad \text{Synchronized position of the 1st leading axis} \\
\text{SynPosLA}_2 & \quad \text{Synchronized position of the 2nd leading axis} \\
\text{KF}_1 & \quad \text{Coupling factor of the 1st leading axis} \\
\text{KF}_2 & \quad \text{Coupling factor of the 2nd leading axis}
\end{align*} \]
Following axis overlay

The overlay of dependent and independent motion components of the following axis is called following axis overlay.

The independent motion component of the following axis can be programmed with the full range of available motion commands.

4.5.2.2 Keywords and coupling characteristics

Keywords

Programming is done via language commands, e.g. coupled motion with TRAILON(X,Y,2).

Keywords replace the language commands in generic coupling.

This has the following advantages:

- Coupling characteristics can be programmed individually (see following example).
- Programming of multiple couplings can be done in one block (since keywords do not require their own block).

Advantage: Reduction of work off time

Example:

The properties set with the existing coupling call TRAILON(X,Y,2)(following axis, leading axis and coupling factor) are defined in the generic coupling with the following keywords:

\[
\begin{align*}
\text{CPON} &= (X1) \\
\text{CPLA}[X1] &= (X2) \\
\text{CPLNUM}[X1,X2] &= 2
\end{align*}
\]

- **CPON** = (X1) Switch on coupling to following axis X1.
- **CPLA**[X1] = (X2) Define axis X2 as leading axis.
- **CPLNUM**[X1,X2] = 2 Set numerator of the coupling factor to 2.

Notation

In order to be uniquely assigned, keywords are furnished with the prefix “CP”, for Coupling). Depending on meaning and application position, a third letter is used:

| Keyword prefix | Meaning | Example |
|---------------|---------|---------|
| CP*           | Describes the characteristics of the entire coupling. | CPON\(^1\) |
| CPF*          | Describes the characteristics of the following axis (Following axis). | CPFPOS\(^1\) |
| CPL*          | Describes a property referring to the leading axis (Leading axis) or the coupling rule. | CPLON\(^1\), CPLNUM\(^1\) |
| CPM*          | Describes a property of the entire coupling for special states. | CPMRESET\(^1\) |

\(^1\) For keyword meaning, please refer to the following table "Overview of all keywords and coupling characteristics".
Overview of all keywords and coupling characteristics

The following table gives an overview of all keywords of the generic coupling and the programmable coupling characteristics:

| Keyword   | Coupling characteristics / meaning                                                                 | Default setting (CPSETTYPE="CP") |
|-----------|------------------------------------------------------------------------------------------------------|-----------------------------------|
| CPDEF     | Creating a coupling module                                                                         |                                   |
| CPDEL     | Deletion of a coupling module                                                                       |                                   |
| CPLDEF    | Definition of a leading axis and creation of a coupling module                                      |                                   |
| CPLDEL    | Deleting a leading axis of a coupling module                                                        |                                   |
| CPON      | Switching on a coupling module                                                                      |                                   |
| CPOF      | Switching off a coupling module                                                                     |                                   |
| CPLON     | Switching on a leading axis of a coupling module                                                    |                                   |
| CPLOF     | Switching off a leading axis of a coupling module                                                   |                                   |
| CPLNUM    | Numerator of the coupling factor                                                                   | 1.0                               |
| CPLDEN    | Denominator of the coupling factor                                                                  | 1.0                               |
| CPLCTID   | Number of curve table                                                                               | Not set                           |
| CPLSETVAL | Coupling reference                                                                                  | CMDPOS                            |
| CPFRS     | Co-ordinate reference system                                                                        | BCS                               |
| CPBC      | Block change criterion                                                                              | NOC                               |
| CPFPOS + CPON | Synchronized position of the following axis when switching on                                      | Not set                           |
| CPLPOS + CPON | Synchronized position of the leading axis when switching on                                       | Not set                           |
| CPFMON    | Synchronization mode                                                                                | CFAST                             |
| CPFMON    | Behavior of the following axis at switching on                                                      | STOP                              |
| CPFMOF    | Behavior of the following axis at complete switch-off                                               | STOP                              |
| CPFPOS + CPOF | Switch-off position of the following axis when switching off                                     | Not set                           |
| CPMRESET  | Coupling response to RESET                                                                         | NONE                              |
| CPMSTART  | Coupling behavior at part program start                                                              | NONE                              |
| CPMRPT    | Coupling response at part program start under search run via program test                          | NONE                              |
| CPLINTR   | Offset value of the input value of a leading axis                                                  | 0.0                               |
| CPLINSC   | Scaling factor of the input value of a leading axis                                                 | 1.0                               |
| CPLOUTTR  | Offset value for the output value of a coupling                                                     | 0.0                               |
4.5 Generic coupling

| Keyword        | Coupling characteristics / meaning                                                                 | Default setting (CPSETTYPE=CP) |
|----------------|-----------------------------------------------------------------------------------------------------|--------------------------------|
| CPLOUTSC       | Scaling factor for the output value of a coupling                                                  | 1.0                            |
| CPSYNCOF       | Threshold value of position synchronism "Coarse"                                                   | MD37200                        |
| CPSYNFIP       | Threshold value of position synchronism "Fine"                                                    | MD37210                        |
| CPSYNCOF2      | Second threshold value for the "Coarse" position synchronism                                        | MD37202                        |
| CPSYNFIP2      | Second threshold value for the "Fine" position synchronism                                         | MD37212                        |
| CPSYNCOV       | Threshold value of velocity synchronism "Coarse"                                                  | MD37220                        |
| CPSYNFIV       | Threshold value of velocity synchronism "Fine"                                                    | MD37230                        |
| CPMBRAKE       | Response of the following axis to certain stop signals and stop commands                           | 1                              |
| CPMVDI         | Response of the following axis to certain NC/PLC interface signals                                | 0 (for bits 0 to 3, 5)         |
|                |                                                                                                   | 1 (for bits 4, 6)              |
| CPMAALARM      | Suppression of special coupling-related alarm outputs                                              | MD11410                        |
|                |                                                                                                   | MD11415                        |
| CPSETTYPE      | Coupling type                                                                                      | CP                              |

**Note**

Coupling characteristics, which are not explicitly programmed (in part program of synchronous actions), become effective with their default settings (see right hand column of the table).

Depending on the settings of the keyword CPSETTYPE instead of the default settings (CPSETTYPE=CP) preset coupling characteristics can become effective (see Section "Coupling types (Page 331)").

### 4.5.2.3 System variables

The current state of a coupling characteristic set with a keyword, can be read and written to with the relevant system variable.

**Note**

When writing in the part program, PREPROCESSING STOP is generated.

**Notation**

The names of system variables are normally derived from the relevant keywords and a corresponding prefix.
The first letter of the prefix defines the access location when reading:

| System variable prefix | Access location during read | Features                                                                 |
|------------------------|-----------------------------|--------------------------------------------------------------------------|
| $\text{PA}_{\text{CP}}$ | Reading of channel referenced axis specific coupling characteristics in block preparation (Preparation) | Use in synchronous actions is not possible. Does not generate an implicit preprocessing stop. |
| $\text{AA}_{\text{CP}}$ | Reading of the current state of the coupling module (across channels). | Use in the part program and in synchronous actions is possible. Generates an implicit preprocessing stop when used in the part program. |

**Note**

The preprocessing value of a $\text{PA}_{\text{CP}}.. \text{CP}$ system variable only differs from the values of the corresponding $\text{AA}_{\text{CP}}.. \text{CP}$ system variable during active part program processing. At the end of the program or with abort, there is an appropriate synchronization of the preprocessing with the main run states.

**System variable list**

A list of all system variables which can be used in a generic coupling is contained in the data lists (see Section "System variables (Page 360)").

For a detailed description of system variables, refer to:

**Reference:**

Parameter Manual, System Variables
4.5.3 Creating/deleting coupling modules

4.5.3.1 Creating a coupling module (CPDEF)

An axial coupling module is created through the definition of the following axis.

**Programming**

Syntax: \[\text{CPDEF=} (\text{<following axis/spindle>})\]

Designation: Coupling Definition

Functionality: Definition of a coupling module The coupling is not activated.

Following axis/spindle: Type: AXIS

Range of values: All defined axis and spindle identifiers in the channel

Example:

| Programming       | Comment                                                             |
|-------------------|---------------------------------------------------------------------|
| CPDEF=(X2)        | ; A coupling module is created with axis X2 as following axis.     |

**Supplementary conditions**

- The maximum number of coupling modules is limited (see Section "Requirements (Page 274)").
- The application of CPDEF to an already created coupling module is possible and will not result in an alarm being generated.
4.5.3.2 Delete coupling module (CPDEL)

A coupling module created with CPDEF can be deleted with CPDEL.

Programming

Syntax: \[ CPDEL=(<\text{following axis/spindle}>) \]

Identifiers: Coupling Delete

Functionality: Deletion of a coupling module. All leading axis modules are deleted with the coupling module and reserved memory is released.

Following axis/spindle: Type: AXIS

Range of values: All defined axis and spindle identifiers in the channel

Example:

| Programming | Comment |
|-------------|---------|
| CPDEL=(X2)  | ; Deletion of the coupling module with following axis X2. |

Constraints

- The switch command CPDEL results in a preprocessing stop with active coupling.
  
  Exception: No preprocessing stop occurs in CPSETTYPE="COUP".

- Applying CPDEL to a coupling module active in the block preparation results in implicit deactivation of this coupling.

- Applying CPDEL to an undefined coupling module does not result in any action.
4.5.3.3 Defining leading axes (CPLDEF or CPDEF+CPLA)

The leading axes/spindles defined for a coupling can be programmed/created with the keyword CPLDEF or with the keyword CPLA in conjunction with CPDEF.

Programming with CPLDEF

Syntax: \[ \text{CPLDEF}[\text{FAx}] = (<\text{leading axis/spindle}>) \]

Designation: Coupling Lead Axis Definition

Functionality: Definition of leading axis/spindle for following axis/spindle FAx. A leading axis/spindle module is created in the coupling module. If the coupling module of the following axis/spindle has not yet been created, the coupling module will be created implicitly.

Leading axis/spindle: Type: AXIS

Range of values: All defined axis and spindle identifiers in the channel

Example:

| Programming | Comment |
|-------------|---------|
| CPLDEF[X2]=(X1) | ; Definition of leading axis X1 for following axis X2. |

Programming with CPLA and CPDEF

Syntax: \[ \text{CPLA}[\text{FAx}] = (<\text{leading axis/spindle}>) \]

Designation: Coupling Lead Axis

Functionality: Definition of leading axis/spindle for following axis/spindle FAx.

Leading axis/spindle: Type: AXIS

Range of values: All defined axis and spindle identifiers in the channel
## 4.5 Generic coupling

### Example:

| Programming         | Comment                                                                 |
|---------------------|-------------------------------------------------------------------------|
| CPDEF=(X2) CPLA[X2]=(X1) | ; Definition of leading axis X1 for following axis X2.                 |

### Supplementary conditions

- **CPLDEF is only allowed in blocks without CPDEF/CPON/CPGF/CPDEL.**
  
  (This limitation applies to the case where the keywords refer to the same coupling module.)

- The maximum number of leading axis modules per coupling module is limited (see Section "Requirements (Page 274)").

- Definition of leading axes on an already defined or active coupling module is possible. Any newly defined leading axes and their properties (e.g. coupling factor) are not active immediately. A corresponding switch-on command like (CPON or CPLON) is required.

### 4.5.3.4 Delete leading axes (CPLDEL or CPDEL+CPLA)

Defined leading axes can be deleted with CPLDEL or with CPLA in conjunction with CPDEL, i.e. removed from the coupling module.

#### Programming with CPLDEL

**Syntax:**

\[\text{CPLDEL}[\text{FAx}]=\langle\text{leading axis/spindle}\rangle\]

**Identifiers:**  
Coupling Lead Axis Delete

**Functionality:**  
Deletion of leading axis/spindle of following axis/spindle FAx. The leading axis/spindle module will be deleted and the corresponding memory will be released. If the coupling module does not have a leading axis/spindle any more, the coupling module will be deleted and the memory will be released.

**Leading axis/spindle:**  
Type: AXIS

Range of values: All defined axis and spindle identifiers in the channel
Example:

| Programming | Comment |
|-------------|---------|
| CPLDEL[X2]=(X1) ; Deletion of leading axis X1 of the coupling to following axis X2. |

Programming with CPLA and CPDEL

Syntax: CPLA[FAx]=<leading axis/spindle>

Identifiers: Coupling Lead Axis

Functionality: Deleting a leading axis/spindle: The leading axis/spindle module will be deleted and the corresponding memory will be released. If the coupling module does not have a leading axis/spindle any more, the coupling module will be deleted and the memory will be released.

Leading axis/spindle: Type: AXIS

Range of values: All defined axis and spindle identifiers in the channel

Example:

| Programming | Comment |
|-------------|---------|
| CPDEL=(X2) CPLA[X2]=(X1) ; Deletion of leading axis X1 of the coupling to following axis X2. |

Constraints

- CPLDEF is only allowed in blocks without CPDEF/CPON/CPOF/CPDEL.
  (This limitation applies to the case where the keywords refer to the same coupling module.)
- If an active leading axis is deleted, the coupling to this leading axis is implicitly deactivated.
- Deletion of the last leading axis results in the entire coupling module to be deleted.
4.5.4 Switching coupling on/off

4.5.4.1 Switching on a coupling module (CPON)

A defined coupling module is switched on with the switch command CPON.

Coupling characteristics like coupling reference can be programmed together with the switch on command (see Section "Programming coupling characteristics (Page 292)").

Without programming, a coupled motion group or a synchronous spindle pair becomes effective based on a setpoint coupling (default setting for CPLSETVAL) with the coupling rule 1:1 (default setting for CPLNUM/CPLDEN).

Programming

Syntax: CPON= (<following axis/spindle>)

Designation: Coupling On

Functionality: Activate the coupling of the following axis to all defined leading axes.

Following axis/spindle: Type: AXIS

Range of values: All defined axis and spindle identifiers in the channel

Example:

| Programming | Comment |
|-------------|---------|
| CPON=(X2)   | ; Activation of coupling of the following axis X2. |

Supplementary conditions

Application of CPON to an already active coupling results in a resynchronization. If applicable, changed coupling properties become effective as a result. Any lost synchronization (for example, following axis was in tracking mode) is restored.
4.5.4.2 Switch off coupling module (CPOF)

An activated coupling can be deactivated with the CPOF switching command. The deactivation, i.e. the switching off of the coupling to the leading axis, is performed in accordance with the set switch-off properties (see CPFMOF).

Programming

Syntax: 

\[\text{CPOF} = \text{<Following axis / spindle>}\]

Identifiers: 

Coupling Off

Functionality: 

Deactivate the coupling of the following axis to all defined leading axes.

Following axis/spindle:

Type: AXIS

Range of values: All defined axis and spindle identifiers in the channel

Example:

| Programming   | Comment                      |
|---------------|------------------------------|
| CPOF=(X2)     | ; Deactivation of coupling of following axis X2. |

Constraints

- The switch command CPOF results in a preprocessing stop with active coupling.
  
  Exception: CPSETTYPE="COUP" does not result in a preprocessing stop

- A CPOF switching command on an already deactivated or deleted coupling module has no effect and is not executed.

- CPOF can be programmed in synchronous actions.
Switching on leading axes of a coupling module (CPLON)

CPLON activates the coupling of a leading axis to a following axis. If several leading axes are defined for a coupling module, they can be activated and deactivated separately with CPLON.

Programming

Syntax: CPLON[FAx] = (<leading axis/spindle>)

Identifiers: Coupling Lead Axis On

Functionality: Activates the coupling of a leading axis/spindle to following axis/spindle FAx.

Leading axis/spindle: Type: AXIS

Range of values: All defined axis and spindle identifiers in the channel

Example:

| Programming | Comment |
|-------------|---------|
| CPLON[X2]=(X1) | ; The coupling of leading axis X1 to following axis X2 is activated. |

Constraints

CPON can be programmed in synchronous actions.
4.5.4.4 Switching off leading axes of a coupling module (CPLOF)

CPLOF deactivates the coupling of a leading axis to a following axis. If several leading axes are defined for a coupling module, they can be deactivated separately with CPLOF.

Programming

Syntax: \[ CPLOF[L]= (<leading axis/spindle>) \]

Identifiers: Coupling Lead Axis Off

Functionality: Deactivates the coupling of a leading axis/spindle to the following axis/spindle FAx.

Leading axis/spindle:

Type: AXIS

Range of values: Axes of the channel

Example:

| Programming      | Comment                                                                 |
|------------------|-------------------------------------------------------------------------|
| CPLOF[X2]=(X1)   | ; The coupling of leading axis X1 to following axis X2 is deactivated.  |

Constraints

CPLOF can be programmed in synchronous actions.
4.5 Generic coupling

4.5.4.5 Implicit creation and deletion of coupling modules

Switch-on commands may also be used to create coupling modules (without prior definition with CPDEF).

Example

| Programming | Comment |
|-------------|---------|
| CPON=(X2) CPLA[X2]=(X1) or CPLON[X2]=(X1) | ; Creates a coupling module for following axis X2 with leading axis X1 and activates the coupling module. |
| CPOF=(X2) | ; After its deactivation, the implicitly created coupling module is deleted. |

Constraints

- Implicitly created coupling modules (via switch-on commands) are deleted once they are completely deactivated (CPOF).
  Advantage: Deleting them with CPDEL/CPLDEL is not necessary.
  Disadvantage (possibly): All coupling properties which were set with CPOF are lost.
- Implicitly created coupling modules can be transformed into explicit coupling modules with the following instruction CPDEF/CPLDEF. In this case CPOF does not delete the coupling module and the data is retained.

4.5.5 Programming coupling characteristics

4.5.5.1 Coupling rule (CPLNUM, CPLDEN, CPLCTID)

The functional relationship between the leading value and the following value is specified by a coupling rule for each leading axis. This functional relationship can be defined linear via a coupling factor or non-linear via a curve table. The following axis components calculated in this way from the individual leading values have an additive effect.

Programming: Coupling factor

When programming a coupling factor, a previously activated non-linear coupling relationship (curve table) is deactivated.

The coupling factor is programmed with numerator and denominator.

In the default state, i.e. without explicit programming after creation of a new coupling module, the numerator and denominator are each preset with 1.

If only the numerator is programmed, this is applied as the factor as the denominator is 1.

More exact linear relationships can be defined by programming numerators and denominators.
Numerator of the coupling factor

Syntax: \[ \text{CPLNUM}[\text{FAx,LAx}] = <\text{value}> \]

Designation: Coupling Lead Numerator

Functionality: Defines the numerator of the coupling factor for the coupling rule of the following axis/spindle FAx to the leading axis/spindle LAx.

Value: Type: REAL

Range of values: \(-2^{31}\) to \(+2^{31}\)

Default value: +1.0

Example:

| Programming | Comment |
|-------------|---------|
| CPLNUM[X2,X1]=1.3 | ; The numerator of the coupling factor of the coupling of the following axis X2 to the leading axis X1 must be 1.3. |

Denominator of the coupling factor

Syntax: \[ \text{CPLDEN}[\text{FAx,LAx}] = <\text{value}> \]

Designation: Coupling Lead Denominator

Functionality: Defines the denominator of the coupling factor for the coupling rule of the following axis/spindle FAx to the leading axis/spindle LAx.

Value: Type: REAL

Range of values: \(-2^{31}\) to \(+2^{31}\)

Default value: +1.0

Example:

| Programming | Comment |
|-------------|---------|
| CPLDEN[X2,X1]=2 | ; The denominator of the coupling factor of the coupling of the following axis X2 to the leading axis X1 must be 2. |
Programming: Curve table

When programming a table number, a previously activated non-linear coupling relationship (coupling factor) is deactivated.

The leading axis specific coupling component for the leading value of the leading axis is calculated using the specified curve table.

Syntax:
\[
\text{CPLCTID}[\text{FAx,LAx}]=<\text{value}>
\]

Designation: Coupling Lead Curve Table Id

Functionality: Specifies the number of the curve table to be used to calculate how the leading axis/spindle must act on the following axis/spindle.

Value: Type: INT

Range of values: 
\(-2^{15} \text{ to } +2^{15}\)

Example:

| Programming          | Comment                                                                 |
|----------------------|-------------------------------------------------------------------------|
| CPLCTID[X2,X1]=5     | ; The leading axis specific coupling component of the coupling of the   |
|                      | following axis X2 to the leading axis X1 is calculated with curve table |
|                      | No. 5.                                                                  |

Supplementary conditions

- A coupling factor of zero (CPLNUM=0) is a permissible value. In this case, the leading axis/spindle does not provide a path component for the following axis/spindle, however, it remains a part of the coupling. Contrary to the switched-off state, the leading axis/spindle still has an influence on the following axis/spindle. This affects, for example, reactions to errors, limit switches and NC/PLC interface signals.
- CPLDEN=0 is not a valid value and is rejected with alarm.
- CPLNUM, CPLDEN and CPLCTID can be programmed in synchronous actions.
- Availability of non-linear coupling relationships (CPLCTID) depends on selected options (see Section "Requirements (Page 274)").
4.5.5.2 Coupling relationship (CPLSETVAL)

The following value can be derived from either of the following:

- position setpoint of the leading axis
- speed setpoint of the leading axis
- position actual value of the leading axis

The following couplings can be programmed accordingly:

- Setpoint value coupling
- Speed coupling
- Actual value coupling

Programming

Syntax:

\[
\text{CPLSETVAL}[\text{FAx}, \text{LAx}] = \langle \text{value} \rangle
\]

Identifiers: Coupling Lead Set Value

Functionality: Defines tapping of the leading axis/spindle LAx and the reaction point on the following axis/spindle FAx.

Coupling reference: Type: STRING

Range of values:

- "CMDPOS" Commanded Position Setpoint value coupling
- "CMDVEL" Commanded Velocity Speed coupling
- "ACTPOS" Actual Value Actual value coupling

Default value: "CMDPOS"

Example:

| Programming | Comment |
|-------------|---------|
| CPLSETVAL[X2,X1]="CMDPOS" | The coupling of following axis X2 to leading axis X1 is deducted from the setpoint. |

Constraints

For a coupling module speed coupling cannot be activated simultaneously with setpoint or actual value coupling of another leading axis.
4.5.3 Co-ordinate reference (CPFRS):

The co-ordinate reference of the following axis/spindle specifies in which co-ordinate reference system the coupling component resulting from the coupling is applied. In the base co-ordinate system or in the machine co-ordinate system.

It is further specified which co-ordinate reference the leading values of the leading axis spindle must have. When a transformation is active and when the machine co-ordinate system is specified as co-ordinate reference (CPFRS="MCS"), the initial transformation values are taken as leading values.

Programming

Syntax:  

| CPFRS[Fax] = (co-ordinate reference) |

Identifiers: Coupling Following Relation System

Functionality: Defines the co-ordinate reference system for the coupling module of the following axis/spindle FAx.

Co-ordinate reference:

Type: STRING

Range of values:

| "BCS" | Basis Co-ordinate System | Basic Coordinate System |
| "MCS" | Machine Co-ordinate System | Machine coordinate system |

Default value: "BCS"

Example:

| Programming | Comment |
|-------------|---------|
| CPFRS[X2]="BCS" | ; The base co-ordinate system is the co-ordinate reference for the coupling module with following axis X2. |

Constraints

- Co-ordinate reference has to be specified when creating a coupling module, else the default value is used. It is not possible to effect subsequent changes.
- Simultaneous active transformation and coupling via RESET is not supported.
  Solution: Switching off the coupling with CPMRESET="OF" with RESET switching it on again with CPMSTART="ON" in the part program.
- Simultaneous operation of the previous function "Axial Coupling in the Machine Co-ordinate System (MCS Coupling)" and the generic coupling is not supported.
- CPFRS is not available in the main run.
4.5.5.4 Block change behavior (CPBC)

The block change criterion can be used to specify under which conditions the block change with activated coupling is to be permitted in the processing of the part program. The status of the coupling influences the block change behavior. If the specified condition is not fulfilled, the block change is disabled. The block change criterion is only evaluated with an active coupling.

The block change criterion can be defined with the keyword CPBC or with the programming command WAITC. The instruction programmed last is valid.

Programming with PCBC

Syntax:

\[ \text{CPBC}[\text{FAx}] = "<block change criterion>" \]

Identifiers:

Coupling Block Change Criterion

Functionality:

Defines block change criterion with active coupling.

Block change criterion:

Type: STRING

Range of values:

"NOC" Block change is performed irrespective of the coupling status.

"IPOSTOP" Block change is performed with setpoint synchronism.

"COARSE" Block change is performed with actual value synchronism “coarse”.

"FINE" Block change is performed with actual value synchronism “fine”.

Default value: "NOC"

Example:

| Programming       | Comment                                                                 |
|-------------------|-------------------------------------------------------------------------|
| CPBC[X2]="IPOSTOP"| ; Block change during processing of the part program is done with setpoint synchronism (with active coupling to following axis X2). |
**Programming with WATC**

Syntax: \[ \text{WAITC}(\text{FAx}, \text{BC}) \]

Identifiers: **Wait for Coupling Condition**

Functionality: Defines block change criterion with active coupling.

Parameter: 
- **Fax**: Designates the following axis and therefore the coupling module.
- **BC**: Defines the desired block change criterion.

**Fax**: Type: STRING

Range of values: Axes of the channel

**BC**: Type: STRING

Range of values:
- "NOC" Block change is performed irrespective of the coupling status.
- "IPOSTOP" Block change is performed with setpoint synchronism.
- "COARSE" Block change is performed with actual value synchronism "coarse".
- "FINE" Block change is performed with actual value synchronism "fine".

Default value: "NOC"

Example:

|     |     |
|-----|-----|
| **Programming** | **Comment** |
| \text{WAITC}(X2,"IPOSTOP") | ; Block change during processing of the part program is done with setpoint synchronism (with active coupling to following axis X2). |

**Constraints**

WAITC can only occur singularly in a block, contrary to the keyword CPBC.
4.5.5.5 Synchronized position of the following axis when switching on (CPFPOS+CPON)

When switching on the coupling (CPON) approach of the following axis can be programmed for a specified synchronized position.

The synchronized position takes immediate effect at switch on. The total position, resulting from the synchronized position and the coupling rule, is approached according to the specified synchronization mode (CPFMS). The total position is approached according to the dynamic response limits.

Programming

Syntax: CPON=FAx CPFPOS[FAx]=<value>

Designation: Coupling Following Position

Functionality: Defines the synchronized position of the following axis when switching on. AC, IC and GP are possible in position specification.

Value: Type: REAL

Range of values: All positions within the traverse range boundaries

Example:

| Programming        | Comment                                                                 |
|--------------------|-------------------------------------------------------------------------|
| CPON=X2 CPFPOS[X2]=100 | Activation of coupling to following axis X2. 100 is taken as synchronized position of following axis X2. |

Supplementary conditions

- CPFPOS is only effective as synchronized position with the switch-on command CPON/CPLON.

  The switch-off command CPOF evaluates CPFPOS as switch off position (see Section "Following Axis Position on Switch off (Page 305)").

- CPFPOS without switch-on command results in an alarm.

- If the synchronized position of the following axis is not set during switch on, then the current position of the following axis takes effect as synchronized position.

  The program instruction IC can be used to move the current position.

- The position specification is specified in the configured basic system independently of the programmable dimensions (G70/G71).
**M3: Coupled axes**

### 4.5 Generic coupling

#### Part program section (Example)

| Programming | Comment |
|-------------|---------|
| CPON=(X2) CPFPOS[X2]=100 | ; Activation of coupling to following axis X2. 100 is taken as synchronized position of the following axis. |
| ... | |
| G00 X2=123 | ; Following axis X2 is traversed to position 123. |
| CPON=(X2) | ; The current position (=123) is taken as synchronized position of the following axis. The previously active synchronized position 100 becomes effective. |

#### 4.5.5.6 Synchronized position of the leading axis when switching on (CPLPOS)

The current leading axis position, taken as leading value, can be offset. The synchronized position of the leading axis therefore defines the zero point of the input variable.

**Programming**

Syntax:  
CPLPOS[FAx,LAx]=<value>

Identifiers:  
Coupling Lead Position

Functionality:  
Defines the synchronized position of the following axis at switch on. Only AC is possible in the position specification.

Value:  
Type: REAL

Range of values: All position within the traverse range boundaries

Example:

| Programming | Comment |
|-------------|---------|
| CPLPOS[X2,X1]=200 | ; 200 is taken as synchronized position of the leading axis X1 of the coupling to following axis X2. |

**Constraints**

- CPFPOS can only set with the switch-on command CPON / CPLON. CPFPOS without switch-on command results in an alarm.
- If the synchronized position of the leading axis is not set with the switch on command (CPON), then the current position of the leading axis takes effect as synchronized position and therefore as zero point of the input variable.
- The position specification is specified in the configured basic system independently of the programmable dimensions(G70 / G71).
Part program section (Example)

| Programming                  | Comment                                                                 |
|------------------------------|-------------------------------------------------------------------------|
| CPON=(X2) CFPPOS[X2]=100 CPLPOS[X2,X1]=200 | ; Activation of coupling to following axis X2. 100 is taken as synchronized position of following axis and 200 for leading axis X1. |
|                             |                                                                         |
| ...                         |                                                                         |
| N20 X1=280 F1000            | ; Leading axis X1 is traversed to position 280.                        |
| CPON=(X2)                   | ; The current position X1=280 is taken as synchronized position of the leading axis. The previously active synchronized position of the leading axis (200) becomes ineffective. |

4.5.5.7 Synchronization mode (CPFMSON)

Synchronization mode determines synchronization behavior during switch-on of the coupling.

Programming

Syntax: CPFMSON[FAx] = "<synchronization mode>"

Identifiers: Coupling Following Mode Strategy On

Functionality: Determines the synchronization mode during coupling.

Synchronization mode:

Type: STRING

Range of values:

- **"CFAST"** Closed Coupling Fast
  - The coupling is closed time-optimized.

- **"CCOARSE"** Closed If Gab Coarse
  - The coupling is only closed when the following axis position, required according to the coupling rule, is in the range of the current following axis position.

- **"NTGT"** Next Tooth Gap Time Optimized
  - The next tooth gap is approached time-optimized.

- **"NTGP"** Next Tooth Gap Path Optimized
  - The next tooth gap is approached path-optimized.
**NRGT** | **Next Ratio Gap Time Optimized** | The next segment is approached in a time-optimized manner, in accordance with the ratio of the number of gears to the number of teeth.

**NRGP** | **Next Ratio Gap Path Optimized** | The next segment is approached in a path-optimized manner, in accordance with the ratio of the number of gears to the number of teeth.

**ACN** | **Absolute Co-ordinate Negative** | For rotary axes only! The rotary axis traverses towards the synchronized position in the negative axis direction. Synchronization is effected immediately.

**ACP** | **Absolute Coordinate Positive** | For rotary axes only! The rotary axis traverses to the synchronized position in the positive axis direction. Synchronization is effected immediately.

**DCT** | **Direct Co-ordinate Time Optimized** | For rotary axes only! The rotary axis traverses to the programmed synchronized position in time-optimized fashion. Synchronization is effected immediately.

**DCP** | **Direct Co-ordinate Path Optimized** | For rotary axes only! The rotary axis traverses to the programmed synchronized position in path-optimized fashion. Synchronization is effected immediately.

Default value: "CFAST"

Example:

| Programming | Comment |
|-------------|---------|
| CPMSON[X2]="CFAST" | ; CFAST is taken as synchronization mode of the coupling to following axis X2. |
4.5.5.8 Behavior of the following axis at switch-on (CPFMON)

The behavior of the following axis/spindle during switch-on of the coupling can be programmed with the keyword CPFMON.

Programming

Syntax: CPFMON[Fax]="<block change criterion>"

Identifiers: Coupling Following Mode On

Functionality: Defines the behavior of the following axis/spindle during switch-on of the coupling.

Poweron response: Type: STRING

Range of values:

| "STOP"   | Stop   | For spindles only! An active motion of the following spindle is stopped before switch-on. |
| "CONT"   | Continue | For spindles and main traverse axes only! The current motion of the following axis/spindle is taken over into the coupling as start motion. |
| "ADD"    | Additional | For spindles only! The motion components of the coupling operate in addition to the currently overlaid motion, i.e. the current motion of the following axis/spindle is retained as overlaid motion. |

Default value: "STOP"

Example:

| Programming | Comment |
|-------------|---------|
| CPFMON[X2]="CONT" | ; The current motion of following axis X2 is taken over as start motion. |
4.5.5.9 Behavior of the following axis at switch-off (CPFMOF)

The behavior of the following axis/spindle during complete switch-off of an active coupling can be programmed with the keyword CPFMOF.

Programming

Syntax: \( \text{CPFMOF}[^{FAx}] = "<\text{switch-off behavior}>" \)

Identifiers: Coupling Following Mode Off

Functionality: Defines the behavior of the following axis/spindle during complete switch-off of the coupling.

Switch-off response:

- **Type:** STRING
- **Range of values:**
  - "STOP" Stop Stop of a following axis/spindle. An active overlaid motion is also braked to standstill. Then the coupling is opened, (deactivated).
  - "CONT" Continue For spindles and main traverse axes only! The following spindle continues to traverse at the speed/velocity that applied at the instant of deactivation.

Default value: "STOP"

Example:

| Programming       | Comment                                                                 |
|-------------------|-------------------------------------------------------------------------|
| CPFMOF[S2]="CONT" | ; The following spindle S2 continues to traverse at the speed that was applied at the instant of deactivation. |
4.5.5.10 Position of the following axis when switching off (CPFPOS+CPOF)

When switching off a coupling (CPOF) traversing to a certain position can be requested for the following axis.

Programming

Syntax: \[ \text{CPOF}=(\text{FAx}) \quad \text{CPFPOS}[\text{FAx}]=<\text{value}> \]

Functionality: Defines the switch-off position of the following axis FAx.

Value: Type: REAL

Range of values: All positions within the traverse range boundaries

Example:

| Programming                  | Comment                                                                 |
|------------------------------|-------------------------------------------------------------------------|
| \text{CPOF}=(\text{X2}) \text{CPFPOS[X2]}=100 | ; Deactivation of coupling to following axis X2. 100 is approached as switch-off position of the following axis. |

Supplementary conditions

- CPFPOS is only effective as switch-off position with the switch-off command CPOF.
- The switch command CPON evaluates CPFPOS as switch-on position (see Section "Synchronized Position of the Following Axis on Switch-on (Page 299)").
- The setting of a switch-off position is only permitted with the switch-off mode: CPFMOF=STOP
- Switch-off position is approached with maximum dynamics.
- Block change behavior depends on parameterization of the keyword CPBC.
### 4.5.5.11 Condition at RESET (CPMRESET)

With RESET, the coupling can be activated, deactivated or the current status can be retained. The behavior can be set separately for each coupling module.

**Programming**

**Syntax:**

```
CPMRESET[FAx] = "<Reset behavior>"
```

**Identifiers:**

- **Coupling Mode RESET**

**Functionality:**

Defines the behavior of a coupling at RESET.

**Reset response:**

- **Type:** STRING

**Range of values:**

- "NONE" The current state of the coupling is retained.
- "ON" When the appropriate coupling module is created, the coupling is switched on. All defined leading axis relationships are activated. This is also performed when all or parts of these leading axis relationships are active, i.e. resynchronization is performed even with a completely activated coupling.
- "OF" An active overlaid motion is also braked to standstill. The coupling is then deactivated. When the relevant coupling module was created without an explicit definition (`CPDEF`), the coupling module is deleted. Otherwise it is retained, i.e. it can still be used.
- "OFC" Possible only in spindles! The following spindle continues to traverse at the speed/velocity that applied at the instant of deactivation. The coupling is switched off. When the relevant coupling module was created without an explicit definition (`CPDEF`), the coupling module is deleted. Otherwise it is retained, i.e. it can still be used.
- "DEL" An active overlaid motion is also braked to standstill. The coupling is then deactivated and then deleted.
- "DELC" Possible only in spindles! The following spindle continues to traverse at the speed/velocity that applied at the instant of deactivation. The coupling is deactivated and then deleted.

**Default value:** "NONE"
Example:

| Programming         | Comment                                                                 |
|---------------------|-------------------------------------------------------------------------|
| CPMRESET[X2]="DEL"  | On RESET the coupling to following axis X2 is deactivated and then deleted. |

Constraints

- The coupling characteristics set with CPMRESET is retained until the coupling module is deleted with (CPDEL).
- For the coupling type (CPSETTYPE="TRAIL", "LEAD", "EG" or "COUP") the response is defined by the following machine data during RESET:

  MD20110 $MC_RESET_MODE_MASK (definition of initial control system settings after RESET/TP-End)

  → See Section "Defaults" in "Coupling Types (CPSETTYPE) (Page 331)".

4.5.5.12 Condition at parts program start (CPMSTART)

At part program start the coupling can be activated, deactivated or the current status can be retained. The behavior can be set separately for each coupling module.

Programming

Syntax:

```
CPMSTART[FAx]=<value>
```

Identifiers: Coupling Mode Start

Functionality: Defines the behavior of a coupling at part program start.

Value: Type: STRING

Range of values:

- "NONE" The current state of the coupling is retained.
- "ON" When the appropriate coupling module is created, the coupling is switched on. All defined leading axis relationships are activated. This is also performed when all or parts of these leading axis relationships are active, i.e. resynchronization is performed even with a completely activated coupling.
- "OF" The coupling is switched off. When the relevant coupling module was created without an explicit definition (CPDEF), the coupling module is deleted. Otherwise it is retained, i.e. it can still be used.
- "DEL" The coupling is deactivated and then deleted.

Default value: "NONE"
Example:

| Programming          | Comment                                                                 |
|----------------------|-------------------------------------------------------------------------|
| CPMSTART[X2]="ON"    | ; At part program start, coupling to following axis X2 is switched on. |

Constraints

- The coupling characteristics set with CPMSTART are retained until the coupling module is deleted with (CPDEL).
- For the set coupling type (CPSETTYPE="TRAIL", "LEAD", "EG" or "COUP"), the response is defined by the following machine data during part program start:
  MD20112 $MC_START_MODE_MASK (Definition of the control default settings in case of NC START)
  → See Section "Defaults" in " Coupling Types (CPSETTYPE) (Page 331) ".

4.5.5.13 Status during part program start in search run via program test (CPMPRT)

At part program start during search run via program test (SERUPRO), the coupling can be activated, deactivated or the current status can be retained. The behavior can be set separately for each coupling module.

Programming

Syntax: CPMPRT[FAx] = <value>

Identifiers: Coupling Mode Program Test

Functionality: Defines the behavior of a coupling at part program start during search run via program test.
Value: STRING

Range of values:

"NONE" The current state of the coupling is retained.

"ON" When the appropriate coupling module is created, the coupling is switched on. All defined leading axis relationships are activated. This is also performed when all or parts of these leading axis relationships are active, i.e. resynchronization is performed even with a completely activated coupling.

"OF" The coupling is switched off. When the relevant coupling module was created without an explicit definition (CPDEF), the coupling module is deleted. Otherwise it is retained, i.e. it can still be used.

"DEL" The coupling is deactivated and then deleted.

Default value: "NONE"

Example:

| Programming | Comment |
|-------------|---------|
| CPMPRT[X2]="ON" | At part program start during search run via program test, coupling to following axis X2 is switched on. |

Constraints

- The coupling characteristics set with CPMPRT is retained until the coupling module is deleted with (CPDEL).

- If CPMPRT="NONE" is set, then the response at part program start during search run via program test (SERUPRO) is defined by CPMSTART.

- For the set coupling type (CPSETTYPE="TRAIL", "LEAD", "EG" or "COUP"), the response is defined by the following machine data at part program start during search run via program test:

  MD22620 $MN_START_MODE_MASK_PRT (definition of initial control system settings with special start)
  MD22621 $MC_ENABLE_START_MODE_MASK_PRT (activation of MD22620)
  MD20112 $MC_START_MODE_MASK (Definition of the control default settings in case of NC START)

  → See Section "Defaults" in "Coupling Types (CPSETTYPE) (Page 331) ".

Special functions

Function Manual, 07/2012, 6FC5397-2BP40-3BA0 309
Offset / scaling (CPLINTR, CPLINSC, CPLOUTTR, CPLOUTSC)

An existing coupling relationship between a following axis and a leading axis can be scaled and offset.

The effect of these functions on the total setpoint value of the following axes can be viewed from the following formula:

\[\text{FA}_{\text{Total}} = \text{FA}_{\text{Cmd}} \times \left( \text{transOut} + \text{scaleOut} \times \left( \left( \text{LA} - \text{SynPosLA} \right) \times \text{scaleIn} + \text{transIn} \right) \times \text{KF} \right)\]

- \(\text{FA}_{\text{Total}}\): Total setpoint value of the following axis
- \(\text{FA}_{\text{Cmd}}\): Setpoint set in the part program
- \(\text{LA} / 2\): Setpoint or actual value of the 1st or 2nd leading axis/value
- \(\text{SynPosLA} / 2\): Synchronized position of the 1st or 2nd leading axis/value
- \(\text{scaleIn} / 2\): Scaling factor of the 1st or 2nd lead value
- \(\text{transIn} / 2\): Offset of the 1st or 2nd lead value
- \(\text{KF} / 2\): Coupling factor of the 1st or 2nd leading axis/value
- \(\text{scaleOut} / 2\): Scaling factor of the 1st or 2nd output value
- \(\text{transOut} / 2\): Offset of the 1st or 2nd output value

**Note**
The scaling and offset values can be defined for each leading axis.

**Programming**

**Offset of the input value**

**Syntax:**

\[\text{CPLINTR}[\text{FAx,LAx}] = <\text{value}>\]

**Identifiers:**

- **Coupling Lead In Translation Displacement**

**Functionality:**

Defines the offset value for the input value of the LAx leading axis.

**Value:**

Type: REAL

Default value: 0

**Example:**

| Programming | Comment |
|-------------|---------|
| CPLINTR[X2,X1]=-50 | The input value of the leading axis X1 is moved in the negative direction by the value 50. |
Scaling the input value

Syntax: \( \text{CPLINSC} \{\text{FAx, LAx}\} = \text{<value>} \)

Identifiers: Coupling Lead In Scale Factor

Functionality: Defines the scaling factor for the input value of the LAx leading axis.

Value: Type: REAL

Default value: 1

Example:

| Programming | Comment |
|-------------|---------|
| CPLINSC[X2,X1]=0.5 | ; The input value of the leading axis X1 is multiplied with the factor 0.5. |

Offset of the output value

Syntax: \( \text{CPLOUTTR} \{\text{FAx, LAx}\} = \text{<value>} \)

Identifiers: Coupling Lead Out Translation Displacement

Functionality: Defines the offset value for the output value of coupling the following axis FAx to leading axis LAx.

Value: Type: REAL

Default value: 0

Example:

| Programming | Comment |
|-------------|---------|
| CPLOUTTR[X2,X1]=100 | ; The output value of the coupling of the following axis X2 with leading axis X1 is displaced by the value 100 in the positive direction. |
Scaling of the output value

Syntax: \[\text{CPLOUTSC}[\text{FAx}, \text{LAx}]=<\text{value}>\]

Identifiers: Coupling Lead Out Scale Factor

Functionality: Defines the scaling factor for the output value of coupling the following axis FAx with leading axis LAx.

Value: Type: REAL

Default value: 1

Example:

| Programming      | Comment                                                                 |
|------------------|-------------------------------------------------------------------------|
| CPLOUTSC[X2,X1]=3 | ; The output value of the coupling of the following axis X2 with leading axis X1 is multiplied with factor 3. |

NOTICE

The following setting data used in the existing coupling type "Master value coupling" is considered in generic coupling independently of the set coupling type (CPSETTYPE):

SD43102 $SA\_LEAD\_OFFSET\_IN\_POS[FAx]$ (offset of master value)
SD43104 $SA\_LEAD\_SCALE\_IN\_POS[FAx]$ (scaling of master value)
SD43106 $SA\_LEAD\_OFFSET\_OUT\_POS[FAx]$ (offset of function value of the curve table)
SD43108 $SA\_LEAD\_SCALE\_OUT\_POS[FAx]$ (scaling of function value of the curve table)

These setting data have the following effect:

- on all leading axes that are coupled with the following axis via a curve table. This must be taken into account for couplings with more than one leading axis!
- in addition to the CP key words CPLINTR, CPLINSC, CPLOUTTR and CPLOUTSC.
4.5.5.15 Synchronism monitoring stage 1 (CPSYNCOP, CPSYNFIP, CPSYNCOV, CPSYNFIV)

Synchronism monitoring stage 1

In each interpolation cycle, the synchronous operation of the coupling group is monitored - both on the setpoint and actual value sides. The synchronous operation monitoring responds as soon as the **synchronous operation difference** (the difference between the setpoint or actual value of the following axis and the value calculated from the setpoints or actual values of the leading axes according to the coupling rule) reaches one of the following programmed threshold values:

- At setpoint / actual value coupling (see "Coupling relationship (CPLSETVAL) (Page 295)"):
  - "Coarse" position synchronous operation threshold value
  - "Fine" position synchronous operation threshold value
- For speed coupling (see "Coupling relationship (CPLSETVAL) (Page 295)"):
  - Threshold value of "Coarse" speed synchronous operation
  - Threshold value of "Fine" speed synchronous operation

The actual synchronous operation difference can be read out with the following CP system variables.

| System variable       | Meaning                                    |
|-----------------------|--------------------------------------------|
| $AA_SYNCDIFF [FAx]   | Synchronous operation difference of the setpoint |
| $VA_SYNCDIFF [FAx]   | Synchronous operation difference of the actual value |

**Note**

Synchronous operation differences are signed and enable the advance or delay of the following axis to be determined.
Status of the coupling during synchronous operation

| State                                 | Description                                                                                                                                 |
|---------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Not synchronized                      | Provided the synchronous operation difference is greater than the threshold value for position "coarse" synchronous operation or "coarse" speed synchronous operation, the coupled group is designated as non-synchronous. |
| "Coarse" synchronous operation reached | The synchronous operation difference has reached the threshold value for "coarse" position synchronous operation or "coarse" speed synchronous operation.  |
|                                      | In case the actual value-based synchronous operation difference, the following NC/PLC-interface signal is set: DB31, ... DBX98.1 (coarse synchronous operation) |
| "Fine" synchronous operation reached  | The synchronous operation difference has reached the threshold value for "fine" position synchronous operation or "fine" speed synchronous operation. |
|                                      | In case the actual value-based synchronous operation difference, the following NC/PLC-interface signal is set: DB31, ... DBX98.0 (fine synchronous operation) |

The status of the coupling for the synchronous operation can be read with the following system variables:

| System variable | Meaning                     | Value | State                          |
|-----------------|-----------------------------|-------|--------------------------------|
| $AA_SYNC [Fax]  | State of the coupling       | 0     | Not synchronized               |
|                 |                             | 1     | "Coarse" synchronous operation reached |
|                 |                             | 2     | "Fine" synchronous operation reached |

Signal reaction

Synchronous operation signals themselves do not stop the involved axes, but can release them via synchronized action or NC/PLC interface signals (see Section "R3: Extended stop and retract (Page 365)").

Configuration

The threshold values for the first stage of the synchronous operation monitoring will be adjusted:

- For setpoint / actual value coupling in the machine data:
  - MD37200 $MA_COUPLE_POS_TOL_COARSE (threshold value for "coarse synchronism")
  - MD37210 $MA_COUPLE_POS_TOL_FINE (threshold value for "fine synchronism")

- For speed coupling in the machine data:
  - MD37220 $MA_COUPLE_VELO_TOL_COARSE ("coarse" speed tolerance)
  - MD37230 $MA_COUPLE_VELO_TOL_FINE ("fine" speed tolerance)
Programming

CP keywords can also be used to program the threshold values for the first stage of the synchronous operation monitoring:

"Coarse" position synchronous operation threshold value

Syntax: \[ \text{CPSYN COP}[\text{FAx}] = <\text{value}> \]

Designation: Coupling Synchronous Difference Coarse Position

Functionality: Defines the threshold value for the "Coarse" position synchronous operation.

Value: Type: REAL

The default value corresponds to the setting in the machine data: MD37200 $MA\_COUPLE\_POS\_TOL\_COARSE [FAx]

"Fine" position synchronous operation threshold value

Syntax: \[ \text{CPSYNFIP}[\text{FAx}] = <\text{value}> \]

Designation: Coupling Synchronous Difference Fine Position

Functionality: Defines the threshold value for the "Fine" position synchronous operation.

Value: Type: REAL

The default value corresponds to the setting in the machine data: MD37210 $MA\_COUPLE\_POS\_TOL\_FINE [FAx]

Threshold value of "Coarse" speed synchronous operation

Syntax: \[ \text{CPSYNCOV}[\text{FAx}] = <\text{value}> \]

Designation: Coupling Synchronous Difference Coarse Velocity

Functionality: Defines the threshold value for the "Coarse" speed synchronous operation.

Value: Type: REAL

The default value corresponds to the setting in the machine data: MD37220 $MA\_COUPLE\_VELO\_TOL\_COARSE [FAx]
Threshold value of "Fine" speed synchronous operation

Syntax: \[ CPSYNFIV[F\alpha]=<value> \]

Designation: Coupling Synchronous Difference Fine Velocity

Functionality: Defines the threshold value for the "Fine" velocity synchronous operation.

Value: Type: REAL

The default value corresponds to the setting in the machine data:
MD37230 $MA_COUPLE_VELO_TOL_FINE [F\alpha]$

Example

| Program code | Comment |
|--------------|---------|
| CPDEP=(S2) CPLA[S2]=(S1) | ; Definition of a spindle coupling: Leading spindle S1 with following spindle S2 |
| CPON=(S2) CPSYNCP[S2]=0.5 CPSYNFIP[S2]=0.25 | ; Activation of the coupling with following spindle S2. The threshold values for the position synchronous operation are set to 0.5 ("coarse") and 0.25 ("fine"). |
| ... |

Supplementary conditions

- When considering the synchronous operation difference, an active coupling cascade is not taken into account. This means: if in the considered coupling module, the leading axis is a following axis in another coupling module, the current actual or setpoint position is still used as input variable for the calculation of the synchronous operation difference. This synchronous operation difference therefore does not show the total synchronous operation error of the cascade.

- If the leading axis is not a real axis, but a simulated axis, then the actual value is not available for the synchronous operation monitoring of the actual value. In this case, the modeled actual values (according to the machine data setting) are used.
4.5.5.16 Synchronous operation monitoring stage 2 (CPSYNOP2, CPSYNFIP2)

Synchronism monitoring stage 2

For active CP position coupling (setpoint or actual value coupling, see "Coupling relationship (CPLSETVAL) (Page 295)"), after reaching the "COARSE"/"FINE" block change criterion (see "Block change behavior (CPBC) (Page 297)"), the second stage of the synchronous operation monitoring can be used to monitor for compliance on the actual value side of one of the threshold values of the first stage-independent synchronous operation tolerance.

The following two threshold values must configured or programmed for the second stage of synchronous operation monitoring:

- "Coarse" 2 position synchronous operation threshold value
- "Fine" 2 position synchronous operation threshold value

Technical background

As part of the "synchronous operation reached" and "exit synchronous window" monitoring functions, for a correct assessment of any problems that arise during the multi-edge turning (surface, waste, etc.) and sometimes also for oscillating synchronous spindle it is often very useful to define a tolerance range independent of the coarse/fine synchronous operation tolerances of the first stage of the synchronous operation monitoring (similar to exact stop and standstill monitoring for axes) in order to obtain a configurable error message or warning.

Configuration

The threshold values for the second stage of the synchronous operation monitoring are set in the machine data:

MD37202 $MA_COUPLE_POS_TOL_COARSE_2 (second threshold value for "coarse synchronous operation")

MD37212 $MA_COUPLE_POS_TOL_FINE_2 (second threshold value for "fine synchronous operation")

Note

If the appropriate threshold value = 0, the associated monitoring is inactive. This is also the default value so that the compatibility with older software versions is retained.
4.5 Generic coupling

Programming

CP keywords can also be used to program the threshold values for the second stage of the synchronous operation monitoring:

"Coarse" 2 position synchronous operation threshold value

Syntax: \[ CPSYNCOP2[FAx]=<value> \]

Designation: Coupling Synchronous Difference Coarse Position 2

Functionality: Defines the second threshold value for the "Coarse" position synchronous operation.

Value: Type: REAL

The default value corresponds to the setting in the machine data: MD37202 $MA_COUPLE_POS_TOL_COARSE_2 [FAx]

"Fine" 2 position synchronous operation threshold value

Syntax: \[ CPSYNFIP2[FAx]=<value> \]

Designation: Coupling Synchronous Difference Fine Position 2

Functionality: Defines the second threshold value for the "Fine" position synchronous operation.

Value: Type: REAL

The default value corresponds to the setting in the machine data: MD37212 $MA_COUPLE_POS_TOL_FINE_2 [FAx]

The programming applies similarly to the general CP behavior in the part program only with the next switching command; for synchronized actions, immediately.
**Sequence**

**Starting**

The second stage of the synchronous operation monitoring function starts with active coupling as soon as the following conditions are fulfilled:

- The setpoint synchronous operation is reached:
  
  \[ \text{DB31, ..., DBX99.4 (synchronization running) = 0} \]

- The actual value-related "coarse"/"fine" synchronous operation of the tolerance of the first stage of the synchronous operation monitoring (see "Synchronism monitoring stage 1 (CPSYNCP, CPSYNFIP, CPSYNCOV, CPSYNFIV) (Page 313)") is reached:
  
  \[ \text{DB31, ..., DBX98.1 (coarse synchronous operation) = 1 / DB31, ..., DBX98.0 (fine synchronous operation) = 1} \]

**Monitor**

As long as the synchronous operation difference of the actual the threshold values for "fine" 2 position synchronous operation and "coarse" 2 position synchronous operation are not exceeded, the following NC/PLC interface signals are set:

\[ \text{DB31, ..., DBX103.4 (synchronous operation 2 fine)} \]  
\[ \text{DB31, ..., DBX103.5 (synchronous operation 2 coarse)} \]

If, due to temporary overload in machining process (e.g. infeed feedrate for multi-edge machining too high), the following axis/spindle can no longer follow the specifications of the leading axis(n)/spindle(s) and the deviation is greater than the set tolerance, the violation of the "fine"/"coarse" tolerance displays a display alarm that can be deleted:

- Alarm 22026 "Channel %1 Block %2 Following axis/spindle %3 Synchronism (2): Coarse tolerance exceeded"

- Alarm 22025 "Channel %1 Block %2 Following axis/spindle %3 Synchronism (2): Fine tolerance exceeded"

This does not interrupt the machining.

**Note**

Both alarms can also occur simultaneously.
Exit

The second stage of the synchronous operation monitoring ends in the following cases:

- Deactivation of the coupling (for the power-down command: CPOF, CPDEL, CPOF, CPLDEL)
- New synchronization request by:
  - CPON / CPLON / CPDO / CPOF
  - DB31, ... DBX31.4 (synchronize following spindle) = 1
  - Internal synchronization requests

If, in these cases, the start conditions are satisfied again, the monitoring will be restarted.

- For coupled block changes?? (CPLNUM, CPLDEN, CPLCTID) in synchronized actions
- Resetting the setpoint synchronous operation because of missing enable signals for the following spindle (emergency stop, alarm responses)

The DB31, ... DBX103.4/5 signals are reset when the monitoring is ended.

Supplementary conditions

Exclusion conditions

No monitoring is performed in the following cases:

- MD37202 or MD37212 = 0.
- Speed coupling is active (CPLSETVAL="CMDVEL").
- DB31, ... DBX31.5 (inhibit synchronization of the FS) = 1
- Rapid stop of the following axis/spindle or one of the active leading axes/spindles.
- SERUPRO or block search is active.
- For channel-related run-in when the following axis/spindle or an active leading axis/spindle does not really move.
- DB31, ... DBX63.3 (axis/spindle disable active) = 1 for the following axis/spindle or for one of the active leading axes/spindles.

"Track the synchronous operation deviation" is active

Monitoring is suspended while the "track the synchronous operation deviation" function is active (DB31, ... DBX31.6 = 1; see "Tracking the deviation from synchronism (Page 341)").

Synchronous operation monitoring level 2 for the traditional coupling types

For the traditional coupled motion, master value coupling, electronic gearbox and synchronous spindle coupling types, the "stage 2 synchronous operation monitoring" is available only with the CP adaptive cycles (see "Adaptive cycles (Page 330)").
Example

Example for the synchronous spindle with CP adaptive cycle:

| Program code       | Comment                                                                 |
|--------------------|-------------------------------------------------------------------------|
| G0 Z-300 X50       |                                                                         |
| COUPEFS2,1,1,-6,”FINE”) | ; Gear ratio in accordance with tool.                                 |
| M3 S2000           |                                                                         |
| COUPON(S2,S1,0)    | ; Block change is performed on reaching "fine synchronous operation". Synchronous operation monitoring stage 2 is also activated. |
| X20                |                                                                         |
| G1 Z-200           | ; Machining block, material removal. Synchronous operation determines the quality. |
| COUPOF(S2,S1)      | ; Deselect coupling, disable the monitoring function.                  |

Example for CP programming:

| Program code       | Comment                                                                 |
|--------------------|-------------------------------------------------------------------------|
| G0 Z-300 X50       |                                                                         |
| CPDEF=(S2) CPLA=(S1) CPLNUM=2 CPLDEN=1 CPBC="FINE" | ; Gear ratio in accordance with tool.                                 |
| M3 S2000           |                                                                         |
| CPON=(S2) CPLA=(S1) CPSYNCP2[X]=1.6 CPSYNFIP2[X]=0.8 | ; Block change is performed on reaching "fine synchronous operation". Synchronous operation monitoring stage 2 is also activated (tolerances: coarse 1.6, fine 0.8). |
| X20                |                                                                         |
| G1 Z-200           | ; Machining block, material removal. Synchronous operation determines the quality. |
| CPOF=(S2)          | ; Deselect coupling, disable the monitoring function.                  |
4.5.5.17 Reaction to stop signals and commands (CPMBRAKE)

The response of the following axis to certain stop signals and commands can be defined with the CP keyword CPMBRAKE.

Programming

Syntax: \[ CPMBRAKE[F_{ax}] = \text{<value>} \]

Identifiers: Coupling Mode Brake

Functionality: CPMBRAKE is a bit-coded CP keyword, which defines the braking behavior of the following axis FAx with reference to the following events:

| Bit | Event                                                                 | Value | Meaning                                                                 |
|-----|-----------------------------------------------------------------------|-------|-------------------------------------------------------------------------|
| 0   | Approaching the NC/PLC interface signal: DB31, ..., DBX4.3 (feed stop / spindle stop) | 0     | The NC/PLC interface signal: DB31, ..., DBX4.3 (feed stop / spindle stop) has no influence on the coupling. |
| 1   | The NC/PLC interface signal: DB31, ..., DBX4.3 (feed stop / spindle stop) brakes the coupling group. | 1 - 31 Reserved | The NC/PLC interface signal: DB31, ..., DBX4.3 (feed stop / spindle stop) brakes the coupling group. |

The following rules apply for the programming:

- CPMBRAKE must be programmed in one block with CPDEF or CPON (⇒ can only be programmed for an inactive coupling).
- While defining a coupling, the following values are captured without the explicit programming of CPMBRAKE: Bit 0=1

Examples

Example 1:

| Programming | Comment |
|-------------|---------|
| CPDEF=(AX5) CPLA[AX5]=(AX4) CPMBRAKE[AX5]=0 ; Defining a coupling (leading axis Ax4 with following axis Ax5). NST "feed stop / spindle stop" should not brake the coupling group. |

...
Example 2:

| Programming | Comment |
|-------------|---------|
| CPDEF=(S2) CPLA[S2]=(S1) | Definition of a spindle coupling: Leading spindle S1 with following spindle S2 |
| CPON=(S2) CPMBRAKE[S2]=1 ; | Activation of the coupling with following spindle S2. NST "feed stop/spindle stop" should brake the coupling group. |

4.5.5.18 Response to certain NC/PLC interface signals (CPMVDI)

The CP keyword CPMVDI can be used to define the coupling module's response to certain NC/PLC interface signals.

Programming

Syntax: 

```
CPMVDI[FAx]= <value>
```

Designation: Coupling Mode VDI Signal

Functionality: CPMVDI is a bit-coded CP keyword, which defines the response of the coupling module of following axis FAx to certain NC/PLC interface signals. The bit combination operators B_OR, B_AND, B_NOT, and B_XOR can be used to set individual bits.

| Bit | Meaning |
|-----|---------|
| 0   | Reserved. |
| 1   | Reserved. |
| 2   | Reserved. |
| 3   | The effect of NC/PLC interface signal DB31, ..., DBX1.3 (axis/spindle disable) on the following axis/spindle can be set via bit 3: |
|     | Bit 3 = 0 DB31, ..., DBX1.3 has no effect on the following axis/spindle. The state of the following axis/spindle with reference to the axis/spindle disable is derived solely from the state of the leading axes/spindles. |
|     | Bit 3 = 1 DB31, ..., DBX1.3 has an effect on the following axis/spindle. The state of the leading axes/spindles with reference to the axis/spindle disable is not imposed on the following axis/spindle. |

Note:

If bit 3 = 1, the axis/spindle disable state of the following axis/spindle only has an effect if the program test or SERUPRO states are not active (see also bit 5).
4 Bit 4 is used to define the enable for the dependent motion components when the NC/PLC interface signal DB31, … DBX1.3 (axis/spindle disable) has an effect on the following axis/spindle:

- **Bit 4 = 0** Dependent motion components of the leading axes/spindles become effective irrespective of the state of the axis/spindle disable of the relevant leading axis/spindle.
- **Bit 4 = 1** Dependent motion components of the leading axes/spindles only become effective if the state of the axis/spindle disable of the leading axis/spindle matches that of the axis/spindle disable of the following axis/spindle. Otherwise, the components are suppressed.

**Note:**
Bit 4 is only of significance if bit 3 is set, i.e. if the NC/PLC interface signal DB31, … DBX1.3 (axis/spindle disable) has an effect on the following axis/spindle (see bits 3 and 5).

5 The effect of NC/PLC interface signal DB21, … DBX25.7 (program test selected) or DB21, … DBX1.7 (activate program test) on the following axis/spindle can be set via bit 5:

- **Bit 5 = 0** DB21, … DBX25.7 or DB21, … DBX1.7 has no effect on the following axis/spindle. The state of the following axis/spindle with reference to the axis/spindle disable is derived solely from the state of the leading axes/spindles.
- **Bit 5 = 1** If the "program test" state is active for an axis of the coupling module, then for the following axis/spindle, its own state is active regarding the axis/spindle disable. The state of the leading axes/spindles with reference to the axis/spindle disable is not imposed on the following axis/spindle.

**Note:**
When bit 5 is set, the program test state still has an effect on the following axis/spindle, even if the leading axes/spindles have a different state.

6 Bit 6 is used to define the enable for the dependent motion components when the NC/PLC interface signal DB21, … DBX25.7 (program test selected) or DB21, … DBX1.7 (activate program test) has an effect on the following axis/spindle:

- **Bit 6 = 0** Dependent motion components of the leading axes/spindles become effective irrespective of the state of the axis/spindle disable of the relevant leading axis/spindle.
Bit 6 = 1  Dependent motion components of the leading axes/spindles only become effective if the state of the axis/spindle disable of the leading axis/spindle matches that of the axis/spindle disable of the following axis/spindle. Otherwise, the components are suppressed.

Note:
Bit 6 is only of significance if bit 5 is set, i.e. if the program test state has an effect on the following axis/spindle (see bits 3 and 5).

7  Reserved.
8  Reserved.

Note
The axis/spindle disable, which is set for the following axis/spindle via the NC/PLC interface signal DB31, ..., DBX1.3, can be overwritten by the program test (DB21, ..., DBX25.7 or DB21, ..., DBX1.7) and SERUPRO states; this is also the case for every other axis/spindle.
Effect of bits 3/5 and 4/6

The effects of the different motion components on the following axis/spindle as a function of the associated axis/spindle disable are illustrated in the table below:

| A/S disable | A/S disable | A/S disable | CPMVDI | CPMVDI | FA<sub>Total</sub> | Meaning  
|-------------|-------------|-------------|--------|--------|------------------|---------
| FA          | LA<sub>1</sub> | LA<sub>2</sub> | Bit 3/5 | Bit 4/6 |                  |         |
| 0           | 0           | 0           | 0      | 0      | FA<sub>Cmd</sub> + FA<sub>DEP</sub><sub>1</sub> + FA<sub>DEP</sub><sub>2</sub> | Real movement. |
| 0           | 0           | 0           | 0      | 1      | FA<sub>Cmd</sub> + FA<sub>DEP</sub><sub>1</sub> + FA<sub>DEP</sub><sub>2</sub> | Real movement. |
| 0           | 1           | 1           | 0      | 0      | FA<sub>Cmd</sub> + FA<sub>DEP</sub><sub>1</sub> + FA<sub>DEP</sub><sub>2</sub> | Simulated movement |
| 0           | 0           | 1           | 0      | 0      | - | Different leading axis states with ref. to the A/S disable |
| 1           | 0           | 0           | 0      | 0      | FA<sub>Cmd</sub> + FA<sub>DEP</sub><sub>1</sub> + FA<sub>DEP</sub><sub>2</sub> | Real movement, FA spindle disable has no effect. |
| 1           | 0           | 1           | 1      | 0      | FA<sub>Cmd</sub> + FA<sub>DEP</sub><sub>1</sub> + FA<sub>DEP</sub><sub>2</sub> | Simulated movement, FA spindle disable has an effect. |
| 1           | 0           | 1           | 1      | 1      | FA<sub>Cmd</sub> + FA<sub>DEP</sub><sub>2</sub> | Simulated movement, as bit 4 is set, FA<sub>DEP</sub><sub>1</sub> is suppressed. |
| 0           | 1           | 1           | 0      | 1      | Alarm 19000, bit 4/6 = 1 is only supported if bit 3/5 = 1. |
| 1           | 0           | 0           | 0      | 1      | Alarm 19000, bit 4/6 = 1 is only supported if bit 3/5 = 1. |
A/S disable: Axis/spindle disable
This refers to the resulting internal state of the axis/spindle disable. The spindle disable, which is set via the NC/PLC interface signal DB31, … DBX1.3 (axis/spindle disable), can be overwritten by states such as program test (DB21, … DBX25.7 or DB21, … DBX1.7) and SERUPRO, thus generating an axis/spindle state which differs from the NC/PLC interface signal.

FA: Following axis
LA1: Leading axis 1
LA2: Leading axis 2
FA\textsubscript{Total}: Total setpoint value of the following axis
FA\textsubscript{Cmd}: Independent motion component of the following axis
FA\textsubscript{DEP1}: Dependent motion component of leading axis 1
FA\textsubscript{DEP2}: Dependent motion component of leading axis 2

Real movement: Real movement means that positioning movements are transferred to the position control.

Simulated movement: Simulated movement means that no positioning movements are transferred to the position control. The real machine axis remains stationary. This corresponds to the state of an activated axis/spindle disable or program test.

---

**Note**

The states in the columns for leading axes 1 and 2 also apply if there are several leading axes/spindles, which have the same state with reference to the axis/spindle disable.
### 4.5.5.19 Alarm suppression (CPMALARM)

The CP keyword `CPMALARM` can be used to suppress coupling-related alarms.

#### Programming

**Syntax:**

```
CPMALARM[FAX] = <value>
```

**Designation:** Coupling Mode Alarm

**Functionality:**

`CPMALARM` is a bit-coded CP keyword for suppressing special coupling-related alarm outputs. The bit combination operators `B_OR`, `B_AND`, `B_NOT`, and `B_XOR` can be used to set individual bits.

| Bit | Value | Meaning |
|-----|-------|---------|
| 0   | 1     | Alarm 16772 is suppressed. |
| 1   | 1     | Alarm 16773 is suppressed. |
| 2   | 1     | Alarm 16774 is suppressed. |
| 3   | 1     | Alarm 22012 is suppressed. |
| 4   | 1     | Alarm 22013 is suppressed. |
| 5   | 1     | Alarm 22014 is suppressed. |
| 6   | 1     | Alarm 22015 is suppressed. |
| 7   | 1     | Alarm 22016 is suppressed. |
| 8   | 1     | Alarm 22025 is suppressed. |
| 9   | 1     | Alarm 22026 is suppressed. |
| 10  | 1     | Alarm 22040 is suppressed for the cyclic check in case of previously activated position control for following and leading spindles. |
| 11  | 1     | Alarm 16771 is suppressed. |
| 12 - 31 | 1 | Reserved. |

The default values correspond to the settings in the machine data:

- MD11410 `$MN_SUPPRESS_ALARM_MASK` (mask for suppressing special alarm outputs)
- MD11415 `$MN_SUPPRESS_ALARM_MASK_2` (suppress alarm outputs)

#### Example

| Program code | Comment |
|--------------|---------|
| CPMALARM[X2]="H300" | The 22025 and 22026 alarms are suppressed for the coupling of the X2 following axis. |
4.5.6 Coupling cascading

Coupling cascades
The coupling modules can be connected in series. The following axis/spindle of a coupling module then becomes the leading axis/spindle of another coupling module. This results in a coupling cascade.

Multiple coupling cascades in series is also possible. The internal computation sequence of the individual coupling modules is performed so that there is no position offset in the coupling relationship. This also applies for a cross-channel cascading.

Example:
Two new coupling modules are created. For the coupling module with following axis X2, the leading axis X1 is defined. For the coupling module with following axis X2, the leading axis X2 and A1 are defined.

| Programming |
|-------------|
| CPDEF=(X2) CPLA=(X1) CPDEF=(A2) CPLA=(X2) CPLA=(A2)=(A1) |

Supplementary conditions
- The availability of cascading is option-based (see Section "Requirements (Page 274)").
- Cascades between couplings of existing coupling functions and couplings of generic couplings are not possible.
- A ring coupling is not permitted. It is rejected with alarm 16778:
  "Ring coupling with following axis FAx and leading axis LAx not allowed"
  (A ring coupling occurs when a following axis is also a leading axis of its own coupling module or a leading axis in a series-connected coupling module).
4.5.7 Compatibility

4.5.7.1 Adaptive cycles

Adaptive cycles
The provision of adaptive cycles as fixed component of the NCK software ensures a syntactic and functional compatibility to coupling calls of existing coupling types (coupled motion, master value coupling, electronic gearbox and synchronous spindle). This means that as long as the manufacturer/user does not need new coupling characteristics, it is not necessary to modify present coupling calls and any dependent application components (e.g. PLC evaluation of coupling signals).

Assignment to existing coupling commands
The number of adaptive cycles corresponds to the number of existing coupling commands. The assignment is as follows:

| Coupling commands | Adaptive cycle |
|-------------------|----------------|
| TRAILON           | cycle700       |
| TRAILOF           | cycle701       |
| LEADON            | cycle702       |
| LEADOF            | cycle703       |
| COUPDEF           | cycle704       |
| COUPON            | cycle705       |
| COUPONC           | cycle706       |
| COUPOF            | cycle707       |
| COUPOFS           | cycle708       |
| COUPOFS           | cycle709       |
| COUPRES           | cycle710       |
| EGDEF             | cycle711       |
| EGON              | cycle712       |
| EGONSYN           | cycle713       |
| EGONSYNE          | cycle714       |
| EGOFCE            | cycle715       |
| EGOFSS            | cycle716       |
| EGDEL             | cycle717       |

Memory location
Adaptive cycles are stored in the directory "CST".
User specific adaptive cycles

If necessary (functional completion) the user can copy an adaptive cycle to the directory "CMA" or "CUS" and apply changes there. When reading adaptive cycles, the sequence CUS → CMA → CST is observed and cycle variants are taken over on a first found basis, i.e. the adaptive cycles copied into the directory CMA/CUS by the user are selected on a priority basis.

Note

When upgrading the NCK software, a log file is saved in the "CST" directory (Changelog), indicating necessary changes of the adaptive cycles.

4.5.7.2 Coupling types (CPSETTYPE)

Coupling types

If presetting of coupling types (coupled motion, master value coupling, electronic gearbox and synchronized spindle) is required, when creating the coupling module (CPON/CPLON or CPDEF/CPLDEF), the keyword CPSETTYPE needs to be used also.

Programming

Syntax: CPSETTYPE[F Ax] = <value>

Designation: Coupling Set Type

Functionality: Defines the presettings of coupling characteristics (coupling type).

Value: Type: STRING

Range of values:
"CP" Freely programmable
"TRAIL" Coupling type "Coupled motion"
"LEAD" Coupling type "Master Value Coupling"
"EG" Coupling type "Electronic gearbox"
"COUP" Coupling type "Synchronized spindle"

Default value: "CP"
Example:

```
CPLON[X2]=(X1) CPSETTYPE[X2]="LEAD" ; Creates a coupling module for
following axis X2 with leading axis X1
and activates the coupling module.
Coupling properties are set such that
they correspond to the existing master
value coupling type.
```

**Default settings**

Presettings of programmable coupling characteristics for various coupling types can be found in the following table:

| Keyword | Coupling type |
|---------|---------------|
|         | Default (CP) | Coupled motion (TRAIL) | Master value coupling (LEAD) | Electronic gear (EG) | Synchronous spindle (COUP) |
| CPDEF   | -            | -                        | -                             | 1.0                  | 1.0                        |
| CPDEL   | -            | -                        | -                             | 1.0                  | 1.0                        |
| CPLDEF  | -            | -                        | -                             | 1.0                  | 1.0                        |
| CPLDEL  | -            | -                        | -                             | 1.0                  | 1.0                        |
| CPON    |              |                          |                               |                      |                            |
| CPOF    |              |                          |                               |                      |                            |
| CPLON   |              |                          |                               |                      |                            |
| CPLOF   |              |                          |                               |                      |                            |
| CPLNUM  | 1.0          | 1.0                      | -                             | 1.0                  | 1.0                        |
| CPLDEN  | 1.0          | 1.0                      | -                             | 1.0                  | 1.0                        |
| CPLCTID | Not set      | -                        | 0                             | Not set              | -                          |
| CPLSETVAL | CMDPOS     | CMDPOS                   | CMDPOS                         | CMDPOS               | CMDPOS                     |
| CPFRS   | BCS          | BCS                      | BCS                           | BCS                  | MCS                        |
| CPBC    | NOC          | NOC                      | NOC                           | FINE                 | IPOSTOP                    |
| CPFPOS + CPON | Not set | -                        | -                             | Not set              | Not set                    |
| CPFPOS + CPOF | Not set | -                        | -                             | -                    | Not set                    |
| CPFMSN0 | CFAST        | CFAST                    | CCOARSE                        | NRGT                 | CFAST                      |
| CPFMON  | STOP         | VL[1]: STOP              | VL[1]: CONT                    | STOP                 | CONT                       |
| CPFMOF  | STOP         | VL[1]: STOP              | VL[1]: STOP                   | STOP                 | CONT                       |
## 4.5 Generic coupling

### Special functions

| Keyword | Default (CP) | Coupled motion (TRAIL) | Master value coupling (LEAD) | Electronic gear (EG) | Synchronous spindle (COUP) |
|---------|--------------|------------------------|------------------------------|----------------------|----------------------------|
| CPLPOS + CPON | Not set | - | - | Not set | - |
| CPMRESET | NONE | MD20110 | MD20110 | MD20110 | MD20110 |
| CPMSTART | NONE | MD20112 | MD20112 | MD20112 | MD20112 |
| CPMMPRT | NONE | MD20112 / MD22620\(^1\) | MD20112 / MD22620\(^3\) | MD20112 / MD22620\(^3\) | MD20112 / MD22620\(^3\) |
| CPLINTR | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| CPLINSC | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| CPOUTTR | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| CPOUTSC | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| CPSYNCOF | MD37200 | MD37200 | MD37200 | MD37200 | MD37200 |
| CPSYNCOF2 | MD37202 | MD37202 | MD37202 | MD37202 | MD37202 |
| CPSYNCOF3 | MD37212 | MD37212 | MD37212 | MD37212 | MD37212 |
| CPSYNCOF4 | MD37220 | MD37220 | MD37220 | MD37220 | MD37220 |
| CPSYNCOF5 | MD37230 | MD37230 | MD37230 | MD37230 | MD37230 |
| CPMVBRK | Bit 0 | 1 | - | - | - |
| CPMVBDI | Bit 3 | 0 | 0 | 0 | 0 |
|           | Bit 4 | 1 | 1 | 1 | 1 |
|           | Bit 5 | 0 | 0 | 0 | 0 |
|           | Bit 6 | 1 | 1 | 1 | 1 |
| CPMALARM | MD11410 | MD11410 | MD11410 | MD11410 | MD11410 |
|           | MD11415 | MD11415 | MD11415 | MD11415 | MD11415 |

### Legend:

1) Pre-processing
2) Main run
3) depends additionally on MD22621
- not relevant or not allowed
### Additional properties

Value ranges or availability of additional properties of a set coupling type (\texttt{CPSETTYPE}) can be found in the following table:

|                             | Default (CP) | Coupled motion (TRAIL) | Master value coupling (LEAD) | Electronic gear (EG) | Synchronous spindle (COUP) |
|-----------------------------|--------------|------------------------|-------------------------------|----------------------|---------------------------|
| Number of leading axes      | ≤ 5          | ≤ 2                    | 1                             | ≤ 5                  | 1                         |
| Following axis type         | Axis/spindle | Axis/spindle           | Axis/spindle                  | Axis/spindle         | Spindle                   |
| Defining/deleting coupling module | CPDEF/CPDEL or CPON/CPOF | CPON/CPOF | CPON/CPOF                     | CPDEF/CPDEL          | CPDEF/CPDEL               |
| Defining/deleting leading axis | CPLDEF/CPLDEL or CPLON/CPOF | CPLON/CPOF | CPLON/CPOF                      | CPLDEF/CPLDEL         | CPLDEF/CPLDEL             |
| Cascading                   | +            | +                      | +                             | +                    | -                         |
| Dynamic observation of the leading spindle | - | - | - | - | + |
| Implicit selection/deselection of state control\(^1\) | - | - | - | - | + |

**Legend:**

\(^1\) also refer to: Function Manual, Extended Functions; Synchronous Spindle (S3)

- not relevant or not allowed

Availability of the specified characteristics depends on the available version (see Section "Requirements (Page 274)").

**Example:**
The coupled motion coupling type (\texttt{CPSETTYPE}="TRAIL") allows a maximum of two leading axes and cascading. However, this is not available in the basic version, but requires the CP-EXPERT option.

### Supplementary conditions

- \texttt{CPSETTYPE} can be programmed in synchronous actions.
- If the coupling type (\texttt{CPSETTYPE}) is set, certain coupling characteristics are preset and cannot be changed. Subsequent change attempts with keywords cause an error and are rejected with an alarm:
| CPSETTYPE= | TRAIL     | LEAD      | EG        | COUP          |
|------------|-----------|-----------|-----------|---------------|
| CPDEF      | Alarm 16686 | Alarm 16686 |           |               |
| CPDEL      | Alarm 16686 | Alarm 16686 |           |               |
| CPLDEF     |            |           |           |               |
| CPLDEL     |            |           |           |               |
| CPON       | Alarm 16686 | Alarm 16686 |           |               |
| CPLON      |            |           |           |               |
| CPOF       | Alarm 16686 | Alarm 16686 |           |               |
| CPLOF      |            |           |           |               |
| CPRES      | Alarm 16686 | Alarm 16686 |           |               |
| CPLNUM     | Alarm 16686 | Alarm 16686 |           |               |
| CPLDEN     | Alarm 16686 | Alarm 16686 |           |               |
| CPLCTID    | Alarm 16686 |           | Alarm 16686 |               |
| CPLSETVAL  | Alarm 16686 | Alarm 16686 | Alarm 16686 |               |
| CPFRS      | Alarm 16686 | Alarm 16686 | Alarm 16686 |               |
| CPBC       | Alarm 16686 | Alarm 16686 |           |               |
| CPFPOS + CPON | Alarm 16686 | Alarm 16686 |           |               |
| CPFPOS + CPOF | Alarm 16686 | Alarm 16686 | Alarm 16686 |               |
| CPFMON     | Alarm 16686 | Alarm 16686 | Alarm 16686 |               |
| CPFMOF     | Alarm 16686 | Alarm 16686 | Alarm 16686 |               |
| CPLPOS + CPON | Alarm 16686 | Alarm 16686 |           |               |
| CPMRESET   | Alarm 16686 | Alarm 16686 | Alarm 16686 |               |
| CPMSTART   | Alarm 16686 | Alarm 16686 | Alarm 16686 |               |
| CPMPRT     | Alarm 16686 | Alarm 16686 | Alarm 16686 |               |
| CPMBRAKE   | Alarm 16686 | Alarm 16686 | Alarm 16686 |               |

Number of leading axes (∑ LA)

| CPSETTYPE= | TRAIL     | LEAD      | EG        | COUP          |
|------------|-----------|-----------|-----------|---------------|
|            | Alarm 16672 | Alarm 16672 | Alarm 16672 | Alarm 16672 |
| Number of leading axes (∑ LA) | Alarm 16672 with ∑ LA > 2 | Alarm 16672 with ∑ LA > 1 | Alarm 16672 with ∑ LA > 5 | Alarm 16672 with ∑ LA > 1 |

Following axis type

| CPSETTYPE= | TRAIL     | LEAD      | EG        | COUP          |
|------------|-----------|-----------|-----------|---------------|
|            |            |           |           |               |
|            | Alarm 14092 |           |           |               |

Alarm 14092 with axis
Projected coupling (CPRES)

If the coupling type "Synchronous spindle" is set, (see CPSETTYPE), the coupling properties contained in machine data can be activated instead of the programmed coupling properties.

References:
Functions Manual Extension Functions; Synchronous spindles (S3); Chapter "Programming of synchronous spindle couplings"

Programming

Syntax: CPRES= (<following spindle>)

Identifiers: Coupling Restore

Functionality: Activates projected data of the synchronous spindle coupling to following spindle FAm.

Following spindle: Type: AXIS

Range of values: All defined spindle identifiers in the channel

Example:

| Programming     | Comment                                                                 |
|-----------------|-------------------------------------------------------------------------|
| CPLON[S2]=(S1)  | ; Creates a coupling module for following spindle S2 with leading spindle S1 and activates the coupling module. Coupling properties are set such that they correspond to the existing synchronous spindle coupling type. |
| ...             |                                                                         |
| CPRES=(S2)      | ; Activates projected data of the synchronous spindle coupling to following spindle S2. |

Constraints

- CPRES is only allowed when the coupling type "Synchronous spindle" (CPSETTYPE="COUP") is set.
- Application of CPRES to an already active coupling results in a new synchronization.
- Applying CPRES to an undefined coupling module does not result in any action.
4.5.8 Cross-channel coupling, axis replacement

The following and leading axes must be known to the calling channel.

Following axis

The following axis is requested for replacement in the channel when programming a CP keyword in the part program, depending on the axis replacement projection (MD30552) with the language command GETD.

Axis change of the following axis after activating the coupling module is only permitted in the channel. Changing from channel to main run and vice versa is still possible, however a change across channel boundaries is not. Supplementary conditions and properties still apply to axis change. The axis replacement via channel axes is released again after deactivating the coupling module.

Reference:
Function Manual, Extended Functions; Mode Groups, Channels, Axis Replacement (K5)

Leading axes

Axis change of leading axes can be performed independently of the state of the coupling.

4.5.9 Behavior with rotary axes

Rotary axes as leading or following axes

It is possible to couple rotary axes to a linear axis and vice versa. Note that a direct assignment of degrees to mm must be performed using the coupling rule.

Example:
A = Rotary axis, X = Linear axis

| Programming       | Comment                                      |
|-------------------|----------------------------------------------|
| N10 G0 A0 X0      | ; Traverse motion: X = 0 mm, A = 0 degrees    |
| N20 CPON=(A) CPLA[A]=(X) CPLNUM[A,X]=2 | ; A coupling module for rotary axis A with linear axis X as leading axis is created and activated. The coupling value is 2. |
| N30 X100          | ; Traverse motion: X = 100 mm, A = 200 degrees (= 100*2) |
Modulo reduced rotary axes as leading axes

With modulo reduced rotary axes as leading axes, the input variable is not reduced during the reduction of the leading axis. The non-reduced position is still taken as the input variable, i.e. the traversed distance is considered.

Example:

\[ A = \text{Modulo reduced rotary axis, } X = \text{Linear axis} \]

```plaintext
| Programming | Comment |
|-------------|---------|
| N10 G0 A0 X0 | ; Traverse motion: X = 0 degrees, X = 0 mm |
| N20 CPON=(X) CPLA[X]=(A) CPLNUM[X,A]=0.5 | ; A coupling module for linear axis X with rotary axis A as leading axis is created and activated. The coupling value is 0.5. |
| N30 A200 | ; Traverse motion: A = 200 degrees, X = 100 mm (= 200*0.5) |
| N40 A=IC(200) | ; A traverses through 200 degrees in a positive direction to 400 degrees, Display A = 40. X traverses through 100 mm to 200. |
| N50 A=IC(100) | ; A traverses from 40 degrees to 140 degrees, X traverses additional 50 mm to 250. |
| N60 A=ACP(80) | ; A traverses in the positive direction to 50 degrees, the traversing path is 300 degrees in the positive direction. X traverses correspondingly by 150 mm in the positive direction. The end position is therefore X = 400. |
```

Figure 4-12 Example: Modulo reduced rotary axis to linear axis

(…) Position indication for X, A
4.5.10 Behavior during POWER ON, ...

Power on

No coupling is active at power ON. Coupling modules are not available.

RESET

The behavior on RESET can be set separately for each coupling module (see \texttt{CPMRESET}). The coupling can be activated, deactivated or the current state can be retained.

Mode change

The coupling remains active during a mode change. The coupling is suppressed (not deselected!) only in JOG-REF mode when referencing a following axis.

Reference point approach

$G74$ of the following axis is not possible with an active coupling. An alarm is output.

If the JOG-REF mode is selected and the following axis is traversed, the coupling is suppressed. The coupling is only performed after JOG-REF mode is cancelled.

SERUPRO

The SERUPRO procedure will simulate the generic coupling and provide values for a restart.

With axial couplings, the simulation always assumes a setpoint coupling, which means, when there is an actual value coupling, this is switched to setpoint coupling during the SERUPRO procedure. This can mean that the simulation is not performed correctly.

Further deviations from the real procedure can occur due to increased simulation speed and canceled axis dynamics limitations.
4.5 Generic coupling

4.5.11 Disturbance characteristic

4.5.11.1 Rapid stop

Function

The rapid stop stops the axis / spindle without ramp, i.e. the velocity setpoint value is specified as zero. This default applies the brakes at the current limit. The servo enable is retained.

The rapid stop is set at:

- Stop A and Stop C (Safety Integrated)
- Alarms with rapid stop as configured braking behavior
- Reaching the hardware limit switch and rapid stop as configured braking behavior:

  MD36600 $MA_BRAKE_MODE_CHOICE = 1

Switchover to actual-value coupling.

The actual values of the leading spindle are used to calculate the setpoint values as soon as the rapid stop of the leading spindle is reported to a generic coupling.

The changeover to actual value coupling takes place smoothly and remains active till the servo enable as well as the pulse enable is available again to the leading spindle and no more position offset takes place. The setpoint value calculation is programmed as with CPLSETVAL only if these conditions are fulfilled.

Note

A rapid stop that was initiated on reaching the hardware limit switch does not changeover the actual value coupling.

Response of the following spindle

If a rapid stop is detected for a leading spindle and the following spindle does not execute any rapid stop by itself, then the following spindle tries to follow the dynamics of the movement of the leading spindle defined within its framework. As position synchronization is generated, there may be oscillations in the following axis in relation to the position to be approached.

The start of a rapid stop for a leading axis/spindle is detected across NCUs.

Note

A simultaneous rapid stop of the leading and following spindle is executed in the synchronized spindle coupling type (CPSETTYPE="COUP") during a servo alarm.
4.5.12 Tracking the deviation from synchronism

4.5.12.1 Fundamentals

Deviation from synchronism

Workpiece machining operations which are to be carried out both on the face front and the face rear require a workpiece transfer to another workpiece receptacle (e.g. a counterspindle chuck).

When workpieces are transferred from front to rear machining, a position offset may result from the closing of the workpiece receptacle. This could be down to square-edged workpieces or due to the generation of an angular momentum when the workpiece receptacle (chuck) is closed quickly during a movement. Depending on the resistance of the workpiece, the tension can be detected by means of an increase in the current consumption of both the motors involved in the coupling and/or by means of the workpiece being subjected to torsion.

This could lead to the following NC/PLC interface signals being reset, according to the synchronism tolerance which has been set, and the magnitude of the offset:

- DB31, ... DBX98.1 (coarse synchronous operation) and/or
- DB31, ... DBX98.0 (fine synchronous operation)

For the setpoint coupling, the position and velocity setpoints are calculated precisely in accordance with the programmed coupling rule and output to the Control Units. If identical drives and a rigid workpiece are used, this will lead to a regulative deviation at the leading and following spindle, half due to a setpoint difference and half due to an actual value difference.

Function

The "track the deviation from synchronism" function serves to detect the position offset which has been imposed on the actual value and to correct the following spindle when calculating the setpoint.

Requirement

A coupling closed via the part/chuck must be in place in order to use this function.

Versions

There are two different options for determining the deviation from synchronism:

1. The deviation from synchronous operation is determined by the NCK (see "Measuring the deviation from synchronism (Page 342)").
2. The deviation value is already known and entered by the user directly (see "Entering the deviation from synchronism directly (Page 345)").

In both cases, the deviation value is then incorporated into the setpoint value calculation for the following spindle, as a correction value.
Availability

The "track the deviation from synchronism" function was developed for machine couplings (CPFRS=“machine”). This means that it is also available for the “synchronous spindle” coupling type (CPSETTYPE=“COUP”).

Like the other higher-level movements, the availability of the function (e.g. speed difference) depends on the option (see “Requirements (Page 274)”).

4.5.12.2 Measuring the deviation from synchronism

The controller measures the difference between the setpoint positions and actual positions when the following spindle is operating in synchronism. This results in a correction value, which is saved in a system variable.

Requirements

The following requirements must be met to enable the controller to calculate the correction value:

- **Requirements if the set coupling type is "synchronous spindle" (CPSETTYPE="COUP"):**
  - The coupling has precisely one leading spindle (requirement is met if CPSETTYPE="COUP").
  - The coupling factor (quotient from CPLNUM and CPLDEN) is 1 or -1.
  - The following value is derived from the setpoint position ("DV") or the actual position ("AV") of the leading spindle.
  - Setpoint synchronism must be achieved: DB31, ... DBX99.4 (synchronization running) = 0
  - Setpoint synchronism must not decline again.
  - No overlaid movement (DB31, ... DBX98.4 = 0) must be present.
  - A dynamics limit is required for the leading spindle, in order to exclude the possibility of the following spindle being subjected to excessive demands.

- **Requirements for a free generic coupling with CPFRS="machine":**
  - The configured spindles are coupled.
  - The coupling has precisely one leading spindle.
  - The coupling factor (quotient from CPLNUM and CPLDEN) is 1 or -1.
  - The following value is derived from the setpoint position (CPLSETVAL="CMDPOS") or the actual position (CPLSETVAL="ACTPOS") of the leading spindle.
  - Setpoint synchronism must be achieved: DB31, ... DBX99.4 (synchronization running) = 0
  - Setpoint synchronism must not decline again.
  - No overlaid movement (DB31, ... DBX98.4 = 0) must be present.
  - A dynamics limit is required for the leading spindle, in order to exclude the possibility of the following spindle being subjected to excessive demands.
Note

Dynamics limit for the leading spindle

The "dynamics limit for the leading spindle" property is specified automatically when the "synchronous spindle" coupling type (CPSETTYPE="COUP") is set. In the case of other coupling types, it is the particular responsibility of the user/machine manufacturer to provide suitable measures to ensure that the following spindle cannot become dynamically overloaded.

---

Activation

Measuring and tracking of the deviation from synchronism are activated by setting the following NC/PLC interface signal to "1":

DB31, ... DBX31.6 (track synchronism)

The signal only has an effect on the following spindle.

---

Note

In the following cases, signal DB31, ... DBX31.6 (track synchronism) is ignored:

- Axis/spindle disable is active (DB31, ... DBX1.3 = 1).
- Program test is selected.
- SERUPRO is active.

If one of these situations arises when the "track the deviation from synchronism" function is already active, the function will be deactivated.

---

Time when measurement is performed

The time when the measurement is performed and the correction value is calculated depends on the bit 7 setting made in the following item of machine data:

MD30455 $MA_MISC_FUNCTION_MASK (axis functions)

| Bit | Value | Meaning |
|-----|-------|---------|
| 7   | 0     | The correction value is calculated continuously, as long as the NC/PLC interface signal DB31, ... DBX31.6 (track synchronism) is set and setpoint synchronism is active (cyclic calculation). |
|     | 1     | The correction value is only calculated at the time when the NC/PLC interface signal DB31, ... DBX31.6 (track synchronism) changes from 0 to 1 (edge evaluation). |
M3: Coupled axes

4.5 Generic coupling

Note
If a temporal extension is taken into account when relieving the tension between the leading and the following spindle, bit 7 should be set to 0. The interface signal is then state-controlled.

The time required to relieve the tension can depend on various factors (e.g. Kv factor of position control, accelerating power of the motors) and must be determined by way of experiment.

Measuring sequence

The NC/PLC interface signal DB31, ... DBX31.6 (track synchronism) can only become effective once setpoint synchronism has been achieved:

DB31, ... DBX99.4 (synchronization running) = 0

The actual values of the following spindle are read within the interpolation cycle and the difference between them and the setpoint position is calculated, either for as long as signal DB31, ... DBX31.6 is activated, or just once on that signal's rising edge (the frequency will depend on the setting of bit 7 in MD30455; see the section titled "Time when measurement is performed").

The correction value is the difference between the setpoint and actual-value synchronism positions. This value is saved for the corresponding following spindle in the following system variable:

$AA_COUP_CORR[S<n>]$ (following spindle: correction value for synchronous spindle coupling)

Note
You must ensure that the velocity of the leading and following axes is kept as constant as possible and that no acceleration jump occurs for the duration of the measurement.

Example

When the coupling of the synchronous spindle [S2] is activated, a position offset of 77 degrees is also programmed:

CPON=S2 ... CPFPOS[S2]=AC(77)

When the workpiece receptacle is closed, this results in a mechanical position offset, which leads to an actual-value position offset of 81 degrees.

When the "track the deviation from synchronism" function is activated (DB31, ... DBX31.6 = 1) and setpoint synchronism has been achieved (DB31, ... DBX99.4 = 0), the actual-value position offset ($VA_COUP_OFFS[S2] = 81$) is compared with the setpoint position offset ($AA_COUP_OFFS[S2] = 77$). This results in a correction value of 4 degrees, which is saved in system variable $AA_COUP_CORR[S2]$. 
4.5.12.3 Entering the deviation from synchronism directly

If the deviation value is known, it can be written directly to the system variable $AA_COUP_CORR for the corresponding following spindle. This is executed via a part program or synchronized action.

Note

Please note that the system variable can only be written once the mechanical coupling has been created. Otherwise a new offset may arise when closing the chuck.

Requirements

To enable the system variable $AA_COUP_CORR to be written from the part program or synchronized actions, a generic machine coupling must have been activated at least once for the corresponding following spindle since the most recent controller power-up was performed.

4.5.12.4 Synchronism correction

If correction value $AA_COUP_CORR[S\langle n\rangle]$ is a value other than zero and a generic machine coupling has been active for following spindle $S\langle n\rangle$ (by means of CPFRS="machine" or CPSETTYPE="COUP"), the following NC/PLC interface signal is set:

$DB31, ... DBX103.0$ (synchronism correction is taken into account)

The correction value is incorporated into the setpoint value calculation for the following spindle, in the coupling module. Resetting the setpoint by the coupling offset relieves the tension between the leading and following spindles.

The synchronism signals are produced by comparing the actual values with the corrected setpoints. Once a correction process has been undertaken, the synchronism signals should be present again:

$DB31, ... DBX98.1$ (coarse synchronism) and/or
$DB31, ... DBX98.0$ (fine synchronism)

The correction value can be implemented and the synchronism signals produced as well, since the whole point of the "tracking the deviation from synchronism" function is to improve synchronism when tension is present. When implementing this value, the accelerating power is restricted to no more than 10% of the maximum acceleration and velocity.

When $AA_COUP_CORR[S\langle n\rangle]$ has been implemented in full, the following NC/PLC interface signal is set:

$DB31, ... DBX99.2$ (synchronism correction implemented)

This still applies even if $AA_COUP_CORR[S\langle n\rangle]$ is zero and no correction needs to be implemented.

When synchronism correction is complete, the NC/PLC interface signal $DB31, ... DBX31.6$ (track synchronism) must be reset to "0" in order to restore the rigidity of the coupling.
The correction value is not changed again once signal DB31, ... DBX31.6 has been reset or once the coupling has been deactivated (with \textit{CPOF}). The system variable $AA\_COUP\_CORR[S<n>]$ then returns a constant value.

The correction value is taken into account until it is reset by setting system variable $AA\_COUP\_CORR[S<n>]$ to "0", which must be done, at the very latest, once the workpiece is removed from the spindle.

\textbf{Note}

The setpoint correction by means of the system variable $AA\_COUP\_CORR[S<n>]$ impacts on all subsequent following spindle programming in the same way as a position offset, similar to a DRF offset in the machine.

\subsection*{4.5.12.5 Diagnostics for synchronism correction}

The current value of $AA\_COUP\_CORR$ (correction value for tracking the deviation in synchronism) is displayed in the "Axis/Spindle Service" window, under the "Position offset for the leading axis/spindle setpoint" line, for the purposes of diagnostics.

System variable $AA\_COUP\_CORR\_DIST$ ($AA\_COUP\_CORR$ distance-to-go) can be used to determine how much of the correction value is still be implemented.

\subsection*{4.5.12.6 Resetting synchronism correction}

\textbf{Versions}

Synchronism correction can be reset in the following ways:

- Writing value "0" to variable $AA\_COUP\_CORR[S<n>].
  Synchronism correction is suppressed via a ramp with reduced accelerating power (just as when a correction value is implemented).

- Resetting synchronism correction via the PLC.
  On the rising edge of the NC/PLC interface signal: DB31, ... DBX31.7 (reset synchronism correction), the variable $AA\_COUP\_CORR[S<n>]$ is set to zero and synchronism correction is reset as follows:
  - If the spindle is in speed control mode, the correction movement is stopped. The existing synchronism correction is then transferred to the setpoint position.
  - In all other cases, the synchronism correction that has already been implemented is reset in exactly the same way as when variable $AA\_COUP\_CORR[S<n>]$ is set to zero.

\textbf{Requirements}

A requirement for resetting synchronism correction is that the correction value is not currently being calculated (see Section "Measuring the deviation from synchronism (Page 342)").
End of the reset procedure

When the reset procedure is complete, the following NC/PLC interface signal is set:
DB31, ... DBX99.2 (synchronism correction implemented)

If the NC/PLC interface signal DB31, ... DBX103.0 (synchronism correction is taken into account) is also reset, so too can the NC/PLC interface signal DB31, ... DBX31.7 (reset synchronism correction) be reset.

![Time diagram for synchronizing and resetting synchronism correction](image)

**Note**

If the correction path has not been traversed in full and the NC/PLC interface signal DB31, ... DBX31.7 (reset synchronism correction) has not been reset, writing to variable $AA_COUP_CORR[S<n>] will not have any effect.
4.5.12.7 Limitations and constraints

Several following spindles

If a leading spindle has several following spindles, each of these following spindles can be processed with the axial NC/PLC interface signal DB31, ... DBX31.6 (track synchronism) separately from one another.

Writing variable $AA_COUP_CORR

System variable $AA_COUP_CORR is only written from the part program or synchronized actions when a generic machine coupling has been activated for the corresponding axis/spindle at least once.

Correction value

If the correction value $AA_COUP_CORR is being written via a part program/synchronized action, as well as being determined due to the "track the deviation from synchronism" function being activated (DB31, ... DBX31.6 = 1), the most recent event to occur is always the one that takes effect.

Resetting synchronism correction

Writing correction value $AA_COUP_CORR (via a part program or synchronized action or when performing a calculation) has no effect when synchronism correction is being reset.

Response to channel/mode group reset

Synchronism correction is not reset in the event of a channel/mode group reset, it is retained instead.

Response to search for reference and zero mark synchronization

If a search for reference or zero mark synchronization procedure is performed for spindles, synchronism correction is reset automatically.

The system variable $AA_COUP_CORR must not be set during the search for reference/zero mark synchronization and, as a result, no measurement may be taken for the deviation from synchronism either.

Furthermore, the synchronism correction must have been implemented in full before the search for reference/zero mark synchronization is started.
Response to an interruption

If an interruption occurs (e.g. emergency stop), synchronism correction is reset automatically, the existing synchronism correction is transferred to the setpoint position, and the NC/PLC interface signal DB31, ... DBX99.2 (synchronism correction implemented) is set.

If, once the interruption has been dealt with, the generic machine coupling remains active and the NC/PLC interface signal DB31, ... DBX31.6 (track synchronism) is set, the following applies:

- If the signal is level-triggered (MD30455 $MA_MISC_FUNCTION_MASK, bit 7 = 0), the deviation from synchronism is measured and written to $AA_COUP_CORR; the setpoints are corrected accordingly.
- If the signal is edge-triggered (MD30455 $MA_MISC_FUNCTION_MASK, bit 7 = 1) and the deviation from synchronism is to be measured, a new rising edge of the NC/PLC interface signal DB31, ... DBX31.6 (track synchronism) needs to be executed.

4.5.13 Examples

4.5.13.1 Programming examples

Direct switch on/off with one leading axis

A coupling module is created and activated with following axis X2 and leading axis X1. The coupling factor is 2.

\[
\text{CPON}=(X2) \quad \text{CPLA}[X2]=(X1) \quad \text{CPLNUM}[X2,X1]=2 \\
\ldots\\
\text{CPOF}=(X2) \quad ; \quad \text{The coupling is deactivated and the created coupling module is deleted with CPOF.}
\]

Direct switch on/off with two leading axes

A coupling module is created and activated with following axis X2 and leading axes X1 and Z. The coupling factor regarding leading axis X1 is 2, the coupling factor regarding leading axis Z is 3.

\[
\text{CPON}=(X2) \quad \text{CPLA}=(X1) \quad \text{CPLNUM}[X2,X1]=2 \quad \text{CPLA}=(Z) \quad \text{CPLNUM}[X2,Z]=3 \\
\ldots\\
\text{CPOF}=(X2) \quad ; \quad \text{The coupling is deactivated and the created coupling module is deleted with CPOF.}
\]
Selective switch-off with two leading axes

A coupling module is created and activated with following axis Y and leading axes X and Z. The coupling factor regarding leading axis X is 2, the coupling factor regarding leading axis Z is 1.2.

\[
\text{CPON=(X2) CPLA}[X2]=(X1) \quad \text{CPLNUM}[X2,X1]=2 \quad \text{CPLA}[X2]=(Z) \quad \text{CPLNUM}[X2,Z]=1.2
\]

...  

\[
\text{CPOF=(X2) CPLA}[X1]=(Z) \quad ; \text{The coupling with leading axis Z is deactivated with CPOF, the coupling with leading axis X1 is retained. The created coupling module remains created.}
\]

Selective switch-on/off with three leading axes

A coupling module is created and activated with following axis X2 and leading axes X1, Z and A.

\[
\text{N10 CPDEF=(X2) CPLA}[X2]=(X1) \quad \text{CPLA}[X2]=(Z) \quad \text{CPLA}[X2]=(A)
\]

\[
\text{N20 CPON=(X2)} \quad ; \text{All leading axes become active, i.e. all contribute a position component according to the coupling rule (coupling component) to axis X2.}
\]

\[
\text{N30 CPOF=(X2)} \quad ; \text{All leading axes are deactivated.}
\]

\[
\text{N40 CPLON}[X2]=(X1) \quad ; \text{Leading axis X1 is activated, only this axis supplies a coupling component. Leading axes Z and A remain deactivated.}
\]

\[
\text{N50 CPLON}[X2]=(A) \quad ; \text{Leading axis X1 remains active, leading axis A is deactivated, X1 and A contribute coupling components (selective switch-on is additive, the condition of the other leading axes is retained).}
\]

\[
\text{N60 CPLON}[X2]=(Z) \quad \text{CPLOF}[X2]=(A) \quad ; \text{Leading axis Z is activated, leading axis A deactivated. Leading axes X and Z are now active.}
\]

\[
\text{N70 CPLOF}[X2]=(X1) \quad ; \text{Leading axis X1 is deactivated. Leading axis Z remains active.}
\]

Definition/deletion of a coupling module

With CPDEF a coupling module is created and activated with following axis X2 and leading axes X1 and Z. The coupling is not activated. Following axis X2 does not follow the coupling rule!

\[
\text{CPDEF=(X2) CPLA}[X2]=(X1) \quad \text{CPLNUM}[X2,X1]=2 \quad \text{CPLA}[X2]=(Z) \quad \text{CPLNUM}[X2,Z]=3
\]

...
Activation can be done with **CPON**, deactivation with **CPOF**.

After the deactivation of the coupling relationship to all leading axes, the coupling object can be deleted. Reserved memory is released:

```
CPDEL=(X2)
```

### 4.5.13.2 Adapt adaptive cycle

**Target**

Coupled motion in the machine co-ordinate system must be possible with the existing coupling command **TRAILON**. The adaptive cycle for **TRAILON** is supplemented with the coupling characteristic "Co-ordinate reference" (**CPFRS**).

**Procedure**

1. Copy adaptive cycle 700 from directory "CST" to directory "CMA".
2. Supplement cycle 700 with the following entry:
   ```
   CPFRS[FA]="MCS"
   ```
3. Comment to cycle changes (e.g. user version number and change date).
4. Save cycle.

```
;CHANGE : 07.01.02.00 Mar 08, 2006
$PATH="/N_CST_DIR\N_CYCLE700_SPF
.USER V1.1 Mar 22, 2006
;classic TRAILON(FA,LA,Factor)
PROC CYCLE700(AXIS _CPF=NO_AXIS, AXIS _CPL=NO_AXIS,REAL _CPLF=1) IPRLOCK SBOF DISPLOF ICYCOF
;IF _CPLF==0 GOTO _CPOF
CPFRS[CPF]=(_CPF) CPSETTYPE[CPF]="TRAIL" CPLNUM[CPF, CPL]=(_CPF) CPFRS[CPF]="MCS"
RET
_CPOF:
IF ($P_TECYCLES==TRUE)
   IF ($AA_CPSETTYPE[CPF]<="TRAIL") GOTO _EXIT
ELSE
   IF ($P_CPSETTYPE[CPF]<="TRAIL") GOTO _EXIT
ENDIF
IF _CPF==NO_AXIS
   CPSETTYPE[CPF]="TRAIL" CPOF=(CPF)
ELSE
   CPSETTYPE[CPF]="TRAIL" CPOF=(CPF)
ENDIF
_EXIT: RET
```

Figure 4-14 Cycle 700 after adaptation. Changes are indicated by a colored bar.
4.6 Dynamic response of following axis

4.6.1 Parameterized dynamic limits

The dynamics of the following axis is limited with the following machine data values:

- MD32000 $MA_MAX_AX_VELO (maximum axis velocity)
- MD32300 $MA_MAX_AX_ACCEL (Maximum axis acceleration)

4.6.2 Programmed dynamic limits

4.6.2.1 Programming (VELOLIMA, ACCLIMA)

Reducing or increasing dynamics limits

The dynamic limits of the following axis (FA) specified through MD32000 and MD32300 can be reduced or increased from the part program:

| Command     | Meaning                                           |
|-------------|---------------------------------------------------|
| VELOLIMA[FA]| Reducing or increasing the maximum Axis velocity  |
| ACCLIMA[FA] | Reducing or increasing the maximum Axis acceleration|

The values specified during the programming of VELOLIMA[FA] and ACCLIMA[FA] are process values. They define the proportion with which the parameterized dynamic limits (MD32000 and MD32300) are to be considered:

| Range of values | Meaning                   |
|-----------------|---------------------------|
| 1 ≤ value < 100 | effects a Reduction in the dynamic limit |
| 100 ≤ value < 200 | effects an increase in the dynamic limit |

The dynamic limits for the velocity and acceleration of the following axis are then calculated as follows:

- Maximum axis velocity = MD32000 $MA_MAX_AX_VELO * VELOLIMA[FA]
- Maximum axis acceleration = MD32300 $MA_MAX_AX_ACCEL * ACCLIMA[FA]

Note

The reduction / increase is effected on the overall dynamics of the axis, i.e., on the sum of the axis component from overlay and coupling.
Programming in synchronized actions

The possibility of programming VELOLIMA[FA] and ACCLIMA[FA] in synchronized actions depends on the coupling type:

| Coupling type          | Parts program | Synchronized actions |
|------------------------|---------------|----------------------|
| Tangential correction  | x             |                      |
| Coupled motion         | x             | x                    |
| Master value coupling  | x             | x                    |
| Electronic gearbox     | x             |                      |
| Synchronous spindle    | x             |                      |
| Generic coupling       | x             | x                    |

Synchronization between following and leading axes

The acceleration characteristics set and the dynamics offsets set change the duration for synchronization between following and leading axes during acceleration operations as follows:

| Dynamic offset       | Activation                                                                 |
|----------------------|-----------------------------------------------------------------------------|
| Dynamic reduction    | Prolongs the synchronized difference. The monitoring from leading to following value may exceed the allowed range for extended periods. |
| Dynamic increase     | Shortens the synchronized difference. The monitoring of leading and following values may exceed the allowed range for short periods. |

Note

The user must restore the technological synchronization between machining and the synchronism difference.

Acceleration mode

Only BRISKA is available for the following axis, i.e., abrupt axis acceleration. Acceleration modes SOFTA and DRIVEA are not available for the following axes described.

Furthermore, it is also possible to configure the positions controller as a PI controller.
4.6 Dynamic response of following axis

⚠️ CAUTION

This option can only be used in conjunction with servo trace and with the appropriate technical knowledge of the control.

References:
- CNC Commissioning Manual: NCK, PLC, drive
- Function Manual, Basic Functions; Velocities, Setpoint-Actual Value Systems, Closed-Loop Control (G2)

POWER ON

During POWER ON the values of \( \text{VELOLIMA} \) and \( \text{ACCLIMA} \) are initialized to 100%.

Mode change

The dynamic offsets remain valid only on transition from AUTO → JOG mode.

RESET

The validities of the \( \text{VELOLIMA} \) and \( \text{ACCLIMA} \) dynamic offsets after RESET depend on the setting in the channel-specific machine data:

\[ \text{MD22410} \text{ $MC_F\_VALUES\_ACTIVE\_AFTER\_RESET} \] \text{(F Function is active even after RESET)}

| Value | Meaning |
|-------|---------|
| 0     | The values of \( \text{VELOLIMA[FA]} \) and \( \text{ACCLIMA[FA]} \) are set to 100% after RESET. |
| 1     | The last programmed values of \( \text{VELOLIMA[FA]} \) and \( \text{ACCLIMA[FA]} \) are also active after RESET. |

This response also applies for dynamics offsets set using static synchronized actions. If this is not the case even when \( \text{MD22410} = 0 \), then the IDS synchronized action should trigger a repeat or continuous writing of the dynamic offset.

References:
- Function Manual, Synchronized Actions
4.6 Dynamic response of following axis

4.6.2.2 Examples

Electronic gearbox

Axis 4 is coupled to X via an electronic gearbox coupling. The acceleration capability of the following axis is limited to 70% of maximum acceleration. The maximum permissible velocity is limited to 50% of maximum velocity. After POWER ON, the maximum permissible velocity is set to 100% again.

```
...  
N120 ACCLIMA[AX4]=70               ; reduced speed
N130 VELOLIMA[AX4]=50
N150 EGON(AX4, "FINE", X, 1, 2)
N200 VELOLIMA[AX4]=100              ; full speed
...  
```

Master value coupling

Axis 4 is coupled to X via a master value coupling. The acceleration capability of the following axis is limited to 80% of maximum acceleration.

```
...  
N120 ACCLIMA[AX4]=80                ; 80%
N130 LEADON(AX4, X, 2)              ; Activate coupling
...  
```

Master value coupling with synchronized action

Axis 4 is coupled to X via a master value coupling. The acceleration response is limited to position 80% by static synchronized action 2 from position 100.

```
...  
N120 IDS=2 WHENEVER $AA_IM[AX4] > 100 DO ACCLIMA[AX4]=80
N130 LEADON(AX4, X, 2)              
...  
```
4.6 Dynamic response of following axis

4.6.2.3 System variables

For geometry axis, channel axis, machine axis and spindle axis, the following readable system variables are available in the part program and synchronous actions:

| Identifier     | Data type | Description                                | Unit |
|----------------|-----------|--------------------------------------------|------|
| $PA_ACCLIMA[n] | REAL      | Acceleration offset set with ACCLIMA[Ax]   | %    |
| $PA_VELOLIMA[n]| REAL      | Velocity offset set with VELOLIMA[Ax]      | %    |
| $AA_ACCLIMA[n] | REAL      | Acceleration offset set with ACCLIMA[Ax]   | %    |
| $AA_VELOLIMA[n]| REAL      | Velocity offset set with VELOLIMA[Ax]      | %    |

Note

Reading the main run variables, implicitly triggers a preprocessing stop.
4.7 General supplementary conditions

**NOTICE**

**Drive optimization**

At a SINAMICS S120 drive unit, a maximum of 3 drives can be optimized or measured at the same time (speed controller optimization/function generator). Therefore, for a coupling with more than 3 coupled drives at the same time, we recommend that these are distributed over several drive units.

**Note**

**Block search with active coupling**

For an active coupling, it is recommended to only use block search type 5, "Block search via program test" (SERUPRO) for a block search.
4.8 Data lists

4.8.1 Machine data

4.8.1.1 NC-specific machine data

| Number  | Identifier: $MN_ | Description                                                   |
|---------|------------------|---------------------------------------------------------------|
| 11410   | SUPPRESS_ALARM_MASK       | Screen form for suppressing special alarm outputs             |
| 11415   | SUPPRESS_ALARM_MASK_2     | Suppress alarm outputs                                        |
| 11660   | NUM_EG               | Number of possible electronic gears                           |
| 11750   | NCK_LEAD_FUNCTION_MASK  | Functions for master value coupling                           |
| 11752   | NCK_TRAIL_FUNCTION_MASK | couple motion functions                                      |
| 18400   | MM_NUM_CURVE_TABS       | Number of curve tables (SRAM)                                 |
| 18402   | MM_NUM_CURVE_SEGMENTS   | Number of curve segments (SRAM)                               |
| 18403   | MM_NUM_CURVE_SEG_LIN    | Number of linear curve segments (SRAM)                        |
| 18404   | MM_NUM_CURVE_POLYNOMS   | Number of curve table polynomials (SRAM)                      |
| 18406   | MM_NUM_CURVE_TABS_DRAM  | Number of curve tables in DRAM                                |
| 18408   | MM_NUM_CURVE_SEGMENTS_DRAM | Number of curve segments in DRAM             |
| 18409   | MM_NUM_CURVE_SEG_LIN_DRAM | Number of linear curve segments (DRAM)                      |
| 18410   | MM_NUM_CURVE_POLYNOMS_DRAM | Number of curve polynomials in DRAM             |
| 18450   | MM_NUM_CP_MODULES       | Maximum number of allowed CP coupling modules                 |
| 18452   | MM_NUM_CP_MODUL_LEAD    | Maximum number of allowed CP master values                  |

4.8.1.2 Channelspecific machine data

| Number  | Identifier: $MC_ | Description                                                      |
|---------|------------------|-----------------------------------------------------------------|
| 20110   | RESET_MODE_MASK  | Definition of control basic setting after run-up and RESET/part program end |
| 20112   | START_MODE_MASK  | Definition of control basic setting after run-up and RESET      |
| 22620   | START_MODE_MASK_PRT | Definition of the control basic settings for special start    |
| 22621   | ENABLE_START_MODE_MASK_PRT | Activation of MD22620                                      |
| 20900   | CTAB_ENABLE_NO_LEADMOTION | Curve tables with jump of following axis             |
| 20905   | CTAB_DEFAULT_MEMORY_TYPE | Default memory type for curve tables               |
| 21300   | COUPLE_AXIS_1     | Projection synchronous spindle pair                             |
4.8 Data lists

4.8.1.3 Axis/spindlespecific machine data

| Number | Identifier: $MA_ | Description |
|--------|------------------|-------------|
| 30130  | CTRLOUT_TYPE     | Setpoint output type |
| 30132  | IS_VIRTUAL_AX    | Axis is virtual axis |
| 30455  | MISC_FUNCTION_MASK | Axis functions |
| 35040  | SPIND_ACTIVE_AFTER_RESET | Own spindle RESET |
| 37160  | LEAD_FUNCTION_MASK | Functions for master value coupling |
| 37200  | COUPLE_POS_TOL_COARSE | Threshold value for "Coarse synchronous operation" |
| 37202  | COUPLE_POS_TOL_COARSE_2 | Second synchronous operation monitoring: Threshold value for "Coarse synchronous operation" |
| 37210  | COUPLE_POS_TOL_FINE | Threshold value for "Fine synchronous operation" |
| 37212  | COUPLE_POS_TOL_FINE_2 | Second synchronous operation monitoring: Threshold value for "Fine synchronous operation" |
| 37220  | COUPLE_VELO_TOL_COARSE | Velocity tolerance "coarse" |
| 37230  | COUPLE_VELO_TOL_FINE | Velocity tolerance "fine" |
| 37500  | ESR_REACTION     | Reaction definition with extended stop and retract |
| 37550  | EG_VEL_WARNING, | Threshold value for speed warning threshold |
| 37560  | EG_ACC_TOL       | Threshold value for the signal "Axis is accelerating" |

4.8.2 Setting data

4.8.2.1 Axis/spindle-specific setting data

| Number | Identifier: $SC_ | Description |
|--------|------------------|-------------|
| 43100  | LEAD_TYPE        | Definition of master value type |
| 43102  | LEAD_OFFSET_IN_POS | Offset of master value with coupling to this axis |
| 43104  | LEAD_SCALE_IN_POS | Scaling of master value with coupling to this axis |
| 43106  | LEAD_OFFSET_OUT_POS | Offset of the function value of the curve table |
| 43108  | LEAD_SCALE_OUT_POS | Scaling of the function value of the curve table |
4.8 Data lists

4.8.3 System variables

Electronic gear (EG) and master value coupling

| Identifier       | Meaning                                                                 |
|------------------|-------------------------------------------------------------------------|
| $AA_EG_ACTIVE    | Coupling for leading axis b is active, i.e. switched on                 |
| $AA_EG_AX        | Identifier for nth leading axis                                         |
| $AA_EG_DENOM     | Numerator of the coupling factor for leading axis b                   |
| $AA_EG_NUMERA    | Numerator of the coupling factor for leading axis b                   |
| $AA_EG_NUMLA     | Number of leading axes defined with EGDEF                               |
| $AA_EG_SYN       | Synchronized position of leading axis b                                |
| $AA_EG_SYNFA     | Synchronous position of following axis a                              |
| $AA_EG_TYPE      | Type of coupling for leading axis b                                    |
| $AA_IN_SYNC[FA]  | Synchronization status of the following axis                          |
| $AA_LEAD_P       | Current leading position value (modulo reduced).                       |
| $AA_LEAD_P_TURN  | current leading value - position component lost as a result of modulo reduction. |
| $AA_LEAD_SP      | simulated master value - position MCS                                  |
| $AA_LEAD_SV      | Simulated master value - velocity                                      |
| $AA_LEAD_V       | Current leading velocity value                                         |
| $AA_SYNC         | Coupling status of the following axis for leading value coupling.      |
| $P_EG_BC         | Block change criterion for EG activation calls: EGON, EGONSYN. WAITC = immediate synchronism fine or coarse and setpoint synchronous operation. |
| $VA_EG_SYNCDIFF  | Synchronization difference                                             |

Generic coupling

| Identifier       | Meaning                                                                 |
|------------------|-------------------------------------------------------------------------|
| $AA_ACCLIMA      | Acceleration correction (main run) set with ACCLIMA                     |
| $AA_COUP_ACT     | Coupling type of a following axis/spindle                               |
| $AA_COUP_CORR    | Following spindle - correction value for tracking the deviation in synchronism |
| $AA_COUP_CORR_DIST | $AA_COUP_CORR distance-to-go                                           |
| $AA_COUP_OFFS    | Setpoint position offset                                               |
| $AA_CPACTFA      | Name of active following axis                                          |
| $AA_CPACTLA      | Name of active leading axis                                            |
| $AA_CPBC         | Block change criterion                                                 |
| $AA_CPDEFLA      | Name of defined leading axis                                           |
| $AA_CPFFACT      | Coupling type of the following axis/spindle.                           |
| $AA_CPFCMDPT     | Axis setpoint position for all coupling components                      |
| $AA_CPFCMDVT     | Axis setpoint speed for all coupling components                         |
| $AACPFMOF        | Behavior of the following axis at switch-off                           |
### Table of Identifiers

| Identifier   | Meaning                                                                 |
|--------------|-------------------------------------------------------------------------|
| $AA\_CPFMON | Behavior of the following axis at switching on                           |
| $AA\_CPFMSON| Synchronization mode                                                    |
| $AA\_CPFRS  | Coupling reference system                                               |
| $AA\_CPLCMDP| Axis setpoint component of the leading axis                             |
| $AA\_CPLCMDV| Axis setpoint speed component of the leading axis                       |
| $AA\_CPLCID | Table number of the active curve table                                  |
| $AA\_CPLDEN | Denominator of the coupling factor                                      |
| $AA\_CPLNUM | Numerator of the coupling factor                                        |
| $AA\_CPLSETVAL| Coupling reference of the leading axis                                 |
| $AA\_CPLSTATE| State of the coupling                                                   |
| $AA\_CPSYNCOP| Threshold value of position synchronism "Coarse" (main run)             |
| $AA\_CPSYNOV| Threshold value of velocity synchronism "Coarse" (main run)             |
| $AA\_CPSYNFINP| Threshold value of position synchronism "Fine" (main run)               |
| $AA\_CPSYNFINV| Threshold value of velocity synchronism "Fine" (main run)               |
| $AA\_CPLINS C | Scaling factor of the input value of a leading axis (main run)          |
| $AA\_CPLIN T | Offset value of the input value of a leading axis (main run)            |
| $AA\_CPLOUTSC| Scaling factor of the output value of the coupling (main run)           |
| $AA\_CPLOUTTR| Offset value of the output value of the coupling (main run)             |
| $AA\_CPLTYPE | Type of coupling                                                        |
| $AA\_CPMRESET| State of the coupling after RESET                                        |
| $AA\_CPMSTART| State of the coupling after program start                               |
| $AA\_CPNACTFA| Number of active following axes                                        |
| $AA\_CPNACTLA| Number of active leading axes                                           |
| $AA\_CPNDEFLA| Number of defined leading axes                                          |
| $AA\_CPSETTYPE| Preset coupling type                                                    |
| $AA\_E G\_ACTIVE | Coupling for leading axis b is active                                  |
| $AA\_EG\_AX | Identifier for nth leading axis                                         |
| $AA\_EG\_BC  | Block change criterion                                                  |
| $AA\_EG\_DENOM| Denominator of the coupling factor                                      |
| $AA\_EG\_NUMERA| Numerator of the coupling factor                                        |
| $AA\_EG\_NUMLA| Number of leading axes defined with EGDEF                                 |
| $AA\_EG\_SYN | Synchronized position of the leading axis                               |
| $AA\_EG\_SYNFA| Synchronized position of the following axis                           |
| $AA\_EG\_TYPE | Type of coupling                                                        |
| $AA\_IN\_SYNC[FA]| Synchronization status of the following axis                         |
| $AA\_JERKLIMA| Jerk correction set with JERKLIMA (main run)                            |
| $AA\_LEAD\_SP  | Simulated master value - position with LEAD                          |
| $AA\_LEAD\_SV  | Simulated master value - speed with LEAD                               |
| $AA\_LEAD\_P\_TURN| Current leading value - position component lost as a result of modulo reduction. |
| $AA\_LEAD\_P  | Current leading value - position (modulo reduced)                      |
4.8 Data lists

### Dynamics of following axis

| Identifier          | Meaning                                                        |
|---------------------|----------------------------------------------------------------|
| $AA_{ACCLIMA}$      | Main run acceleration correction set with ACCLIMA              |
| $AA_{VELOLIMA}$     | Main run speed correction set with VELOLIMA                    |
| $PA_{ACCLIMA}$      | Preprocessing acceleration correction set with ACCLIMA         |
| $PA_{VELOLIMA}$     | Preprocessing speed correction set with VELOLIMA                |

| Identifier          | Meaning                                                        |
|---------------------|----------------------------------------------------------------|
| $AA_{LEAD_V}$       | Current leading velocity value                                  |
| $AA_{SYNC}$         | Coupling status of following axis                              |
| $AA_{SYNCDIFF}[FA]$ | Synchronous operation difference of the setpoint               |
| $AA_{SYNCDIFF_STAT}[FA]$ | Status of the synchronism difference of the setpoint           |
| $AA_{TYP/TYPE}$     | Axis type                                                      |
| $AA_{VELOLIMA}$     | Velocity correction set with VELOLIMA (main run)               |
| $PA_{ACCLIMA}$      | Acceleration correction set with ACCLIMA (pre-processing)      |
| $PA_{CPFACT}$       | Coupling type of the following axis/spindle.                   |
| $PA_{CPFOPOSSTAT}$  | Validity of synchronous and stop position                      |
| $PA_{CPSYNCOCP}$    | Threshold value of position synchronism "Coarse" (pre-processing) |
| $PA_{CPSYNCOV}$     | Threshold value of velocity synchronism "Coarse" (pre-processing) |
| $PA_{CPSYNFIP}$     | Threshold value of position synchronism "Fine" (pre-processing) |
| $PA_{CPSYNFIV}$     | Threshold value of velocity synchronism "Fine" (pre-processing) |
| $PA_{CPLINSC}$      | Scaling factor of the input value of a leading axis (pre-processing) |
| $PA_{CPLINTR}$      | Offset value of the input value of a leading axis (pre-processing) |
| $PA_{CPLOUTSC}$     | Scaling factor of the output value of the coupling (pre-processing) |
| $PA_{CPLOUTTR}$     | Offset value of the output value of the coupling (pre-processing) |
| $PA_{CPSETTYPE}$    | Preset coupling type                                           |
| $PA_{JERKLIMA}$     | Jerk correction set with JERKLIMA (pre-processing)              |
| $PA_{VELOLIMA}$     | Velocity correction set with VELOLIMA (pre-processing)          |
| $VA_{COUP_OFFS}[S2]$ | Actual-value position offset of the synchronous spindle       |
| $VA_{EG_SYNCDIFF}$  | Synchronization difference                                     |
| $VA_{EG_SYNCDIFF}_S$ | Synchronism difference with sign                               |
| $VA_{SYNCDIFF}[FA]$ | Synchronous operation difference of the actual value           |
| $VA_{SYNCDIFF_STAT}[FA]$ | Status of the synchronism difference                           |
| $SP_{COUP_OFFS}[S2]$ | Programmed position offset of the synchronous spindle          |
| $SP_{EG_BC}$        | Block change criterion                                         |
### 4.8.4 Signals

#### 4.8.4.1 Signals to axis/spindle

| Signal name                          | SINUMERIK 840D sl | SINUMERIK 828D |
|--------------------------------------|------------------|----------------|
| Feedrate override                   | DB31, ..., DBX0.0-7 | DB380x.DB80   |
| Axis disable                         | DB31, ..., DBX1.3 | DB380x.DBX1.3 |
| Controller enable                    | DB31, ..., DBX2.1 | DB380x.DBX2.1 |
| Activate handwheel                   | DB31, ..., DBX4.0-2 | DB380x.DBX4.0/1 |
| Feed stop                            | DB31, ..., DBX4.3 | DB380x.DBX4.3 |
| Enable following axis overlay        | DB31, ..., DBX26.4 | -             |
| Synchronize following spindle        | DB31, ..., DBX31.4 | -             |
| Disable synchronization              | DB31, ..., DBX31.6 | -             |
| Track synchronism                    | DB31, ..., DBX31.7 | DB380x.DBX5007.7 |
| Reset synchronism correction         |                  |                |

#### 4.8.4.2 Signals from axis/spindle

| Signal name                          | SINUMERIK 840D sl | SINUMERIK 828D |
|--------------------------------------|------------------|----------------|
| Limiting of differential speed       | DB31, ..., DBX83.1 | DB390x.DBX2001.1 |
| Spindle in setpoint range, differential speed | DB31, ..., DBX83.5 | DB390x.DBX2001.5 |
| Speed limit exceeded, total speed    | DB31, ..., DBX83.6 | DB390x.DBX2001.6 |
| Actual direction of rotation clockwise, total speed | DB31, ..., DBX83.7 | DB390x.DBX2001.7 |
| Synchronous mode                     | DB31, ..., DBX84.4 | DB390x.DBX2002.4 |
| Synchronism fine                     | DB31, ..., DBX98.0 | -             |
| Synchronism coarse                   | DB31, ..., DBX98.1 | -             |
| Actual value coupling                | DB31, ..., DBX98.2 | -             |
| Overlaid movement                    | DB31, ..., DBX98.4 | DB390x.DBX5002.4 |
| Velocity warning threshold           | DB31, ..., DBX98.5 | DB390x.DBX5002.5 |
| Acceleration warning threshold       | DB31, ..., DBX98.6 | DB390x.DBX5002.6 |
| Leading spindle active               | DB31, ..., DBX99.0 | -             |
| Following spindle active             | DB31, ..., DBX99.1 | -             |
| Synchronism correction implemented   | DB31, ..., DBX99.2 | DB390x.DBX5003.2 |
| Following axis accelerated           | DB31, ..., DBX99.3 | DB390x.DBX5003.3 |
| Synchronization in progress          | DB31, ..., DBX99.4 | DB390x.DBX5003.4 |
| Maximum velocity reached             | DB31, ..., DBX99.5 | DB390x.DBX5003.5 |
| Maximum acceleration reached         | DB31, ..., DBX99.6 | DB390x.DBX5003.6 |
| Synchronism correction is taken into account | DB31, ..., DBX103.0 | DB390x.DBX5007.0 |
| Synchronism 2 fine                   | DB31, ..., DBX103.4 |                |
| Synchronism 2 coarse                 | DB31, ..., DBX103.5 |                |
4.8 Data lists
5.1 Brief description

The extended stop and retract function - subsequently called ESR - offers the possibility of flexibly responding when a fault situation occurs as a function of the process:

- **Extended stop**
  Assuming that the specific fault situation permits it, all of the axes, enabled for extended stopping, are stopped in an orderly fashion.

- **Retract**
  The tool currently in use is retracted from the workpiece as quickly as possible.

- **Generator operation (SINAMICS drive function "Vdc control")**
  If a parameterizable value of the DC link voltage is fallen below, e.g. because the line voltage fails, the electrical energy required for retraction is generated by recovering the braking energy of the drive intended for this purpose (generator operation).

ESR and active couplings

While stopping and retracting, active couplings are kept for a parameterizable time.

Retracting along a path

A straight line can be programmed as retraction path, as an alternative to purely axial retraction.
5.2 Control-managed ESR - 840D sl only

5.2.1 Extended stop and retract (ESR)

Using the Extended stop/retract (ESR) function, axes that have been enabled for the function are stopped and retracted in a defined, delayed fashion. This is done to quickly separate the tool and workpiece in certain programmable system states.

In order that the energy required for the retraction motion is available in the drives involved – even when the power fails – then one or several drives can be parameterized as "generators" using the SINAMICS S120 "Vdc control" drive function. Even when the power fails, their kinetic energy is used to maintain the DC link voltage in order to permit NC controlled retraction motion.

Note

Detailed information on the SINAMICS S120 drive function "Vdc control" can be found in:

References

Function Manual SINAMICS S120 Drive Functions

NC-controlled responses

The function provides the following NC-controlled responses:

- "Extended stop"
  Programmable, defined, delayed path-related stopping of traversing motion

- "Retraction"
  Fastest possible retraction away from the machining plane to a safe retraction position to separate the tool and workpiece

The responses are independent of each other. Retraction operations and temporary continuation of axis couplings before stopping can be configured so that they are executed in parallel from a time perspective. In this case, an axis in generator mode (Vdc control) can maintain the DC link voltage.

Interaction of NC-controlled responses

The NC controlled responses are initiated via the channel-specific system variable $AC_ESR_TRIGGER.

Using $AC_ESR_TRIGGER, interpolatory stopping along the path or contour is possible. The NC-controlled retraction is performed in synchronism by the retraction axes in the channel.
For ESR-enabled axes, only precisely one channel may be assigned and it is not possible to switch between channels.

For NC-controlled stopping, an existing traversing motion as well as an active electronic coupling is maintained over an adjustable time (MD21380 $MC_ESR_DELAY_TIME1) even if there is an alarm with motion stop. After the parameterized time has expired, the axis is braked down to standstill along the programmed path.

In order to perform retraction outside the AUTOMATIC mode as well, triggering of this function is linked to the system variable $AC_ESR_TRIGGER. Retraction initiated via $AC_ESR_TRIGGER is locked, in order to prevent multiple retractions.

5.2.2 Drive-independent reactions

Generator operation

Generator operation is a drive function. Using the "Vdc control" function, the SINAMICS S120 drive unit can monitor the DC link group for undervoltage. When an adjustable voltage value is fallen below, then the drive intended for the purpose is switched into generator operation. The kinetic energy of the drive is used to buffer the DC link voltage. This allows the axes that are still moving to be stopped and retracted in an ordering fashion on the NC side.

Note

At the instant that it is switched into generator operation, if an axis is in closed-loop position control, then additional alarms can occur.

References

For detailed information on the SINAMICS S120 drive function "Vdc control", see:

Function Manual SINAMICS S120 Drive Functions
5.2.3 Power failure detection and bridging

DC link voltage limit values

The DC link is monitored against the limit values shown in the following diagram:

![DC Link Voltage Diagram](image)

Figure 5-1 DC link voltage limit values

The drive and DC link pulses are cancelled at specific voltage levels, which means that the drives coast-down. If this behavior is not desired, the excess energy can be discharged using a resistor module. The operating range of the resistor module (shown highlighted in the diagram) lies below the critical voltage level.

**Note**

The pulse power of the resistor module is greater than the infeed power.

Monitoring the intermediate circuit minimum voltage limit

The DC link voltage can be monitored against a limit value that can be parameterized in the drive:

- p1248 (DC link voltage – lower voltage threshold)

When the limit value is fallen below, the following signal is set in the message word (MELDW) of the PROFinet drive telegram:

- MELDW.Bit4 = 1 (VDC_min controller is active (Vdc link < p1248))

The signal can be used in the NC to initiate ESR responses.
5.2.4 NC-controlled extended stop

Parameter assignment

NC-controlled extended stopping is parameterized with:

MD37500 $MA_ESR_REACTION = 22

Definition of the timing behavior

The following machine data is used to define the timing for extended stopping:

MD21380 $MC_ESR_DELAY_TIME1 (delay time, ESR axes)
MD21381 $MC_ESR_DELAY_TIME2 (ESR time for interpolatory braking)

Delay time for ESR axes

For the time set in MD21380 $MC_ESR_DELAY_TIME1, the axes are still traversed as programmed up to this instant in time. After the time expires, traversing movements are braked in an interpolatory fashion along their programmed path.

ESR time for interpolatory braking

The maximum time for interpolatory braking is the time set in MD21380 $MC_ESR_DELAY_TIME2. After this time expires, fast braking with subsequent tracking is initiated.

To illustrate this

T1 = MD21380 $MC_ESR_DELAY_TIME1
T2 = MD21381 $MC_ESR_DELAY_TIME2
**Note**

For safety reasons, the sum of T1 and T2 should not exceed a maximum value of approx. 1 second.

**Precondition**

The precondition in this case is that at least one of the axes involved is parameterized as NC controlled retraction or stopping axis: MD37500 $MA_ESR_REACTION > 20

For axes, which are not parameterized as NC controlled retraction or stopping axis, fast braking with subsequent tracking is realized immediately that extended stopping starts ($AC_ESR_TRIGGER = 1$).

Processing of all commands, especially those that result in an axis stop (e.g. Reset, Stop, Stopall), as well as the standard alarm responses STOPBYALARM and NOREADY, is delayed by the sum of the parameterized times:

\[
\text{Delay time} = \text{MD21380 $MC_ESR_DELAY_TIME1} + \text{MD21381 $MC_ESR_DELAY_TIME2}
\]

An NC controlled stop is also active in conjunction with the "Electronic gearbox" function (see Chapter "Electronic gear (EG) - 840D sl only (Page 244)"). It contains the (selective) switchover of the electronic gearbox to actual value coupling if there is a fault on the leading axes, and also maintains traversing motion and the enable signals during the delay time for ESR axes: MD21380 $MC_ESR_DELAY_TIME1

**NC-controlled extended stopping and path axes**

If NC controlled extended stopping (MD37500 $MA_ESR_REACTION = 22) is parameterized for a path axis, then the behavior is transferred to all path axes of the channel.

**NC-controlled extended stopping and leading axes**

If NC controlled extended stopping (MD37500 $MA_ESR_REACTION = 22) is parameterized for a leading axis, then the behavior is transferred to all following axes of the channel.
Note
A following axis of the electronic gearbox follows the leading axis during both phases of the extended stop according to the motion rule, i.e. no independent braking is possible on transition from machine data phase MD21380 $MC_DELAY_TIME1 to machine data phase MD21381 $MC_ESR_DELAY_TIME2.

In order for ESR to function correctly, the enable signals must be set and remain set.

5.2.5 Retract

Parameterization
NC-controlled retraction is parameterized with:

\[ \text{MD37500 $MA_ESR_REACTION = 21} \]

Behavior
If the channel-specific system variable $AC_ESR_TRIGGER = 1 is set, and if there is a retraction axis in this channel and $AA_ESR_ENABLE=1 is set for this, then \text{LIFTFAST} is activated in this channel.

Requirement
The retraction position must have been programmed in the part program. The enable signals must be set and remain set for retraction motion.

Rapid lift to the position defined with \text{POLF} is triggered using the modal program command \text{LFPOS} (46th G code group).

The retraction movement configured with \text{LFPOS}, \text{POLF} for the axes selected with \text{POLFMLIN} or \text{POLFMASK} replaces the path motion defined for these axes in the part program.

During retraction:
- The axes defined in \text{POLFMASK} independently travel to the positions specified with \text{POLF}.
- The axes defined in \text{POLFMLIN} travel to the positions specified with \text{POLF} in a linear relationship.

The extended retraction (i.e. \text{LIFTFAST/LFPOS} initiated through $AC_ESR_TRIGGER ) cannot be interrupted and can only be terminated prematurely using \text{EMERGENCY OFF}.

Speed and acceleration limits for the axes involved in the retraction are monitored during the retraction motion. The retracting movement takes place with \text{BRISK}, i.e. without jerk limitation.

The maximum time available for retraction is the sum of the following delay times:
- \text{MD21380 $MC_ESR_DELAY_TIME1} (delay time, ESR axes)
- \text{MD21381 $MC_ESR_DELAY_TIME2} (delay time, ESR axes)
After the delay time has expired, fast braking is also initiated for the retraction axis with subsequent tracking.

**Supplementary conditions**

Retraction and/or rapid lift is **not** executed for the following axes:

- Axes, which are not permanently assigned a channel
- Axes, which are in the open-loop speed-controlled mode (spindles)
- Axes, which are interpolated as positioning spindles (SPOS/SPOSA)

**Modulo rotary axes** respond to rapid lift as follows:

- For incremental programming of the target position, the latter is approached without modulo offset.
- With absolute programming, the target position is approached time-optimized with the use of modulo offsets. This is almost identical to positioning via the shortest path.

The retraction movement is interpolated linearly, using the maximum acceleration and speed of the axes involved in POLFLIN.

Only **one linear** retraction is permitted in each **channel**. This means that multiple axis groups, which approach their retraction positions linearly, cannot be created in the channel.

In **parallel** with linear retraction, additional axes can also use POLFMASK for independent axial retraction movement to their programmed retraction positions.

If axes are used in both POLFMASK and in POLFM, it should be noted that the last state programmed is always active for retracting movement. This means that an axis previously activated with POLFM is removed from the linear relationship following programming in POLFMASK and the retracting movement would then take place as an independent movement (see the examples in Section "Lift fast with linear relation of axes (Page 404)").

The parameters valid at the **triggering time** are decisive for the retraction movement. If one of these parameters (POLF, POLFMASK, POLFM, Frame, etc.) changes during the retracting movement (e.g. due to a block change), this change does not affect the retracting movement that has already started.

**Supplementary conditions, path axes**

If, for a path axis ESR_REACTION=21 (NC-controlled retraction) is configured and enabled with $AC_ESR_ENABLE=1, then ESR_REACTION=22 (extended stopping) is active for all path axes for which no ESR_REACTION=21 is configured or enabled.

**Rapid lift without enabled retraction movement**

If, for an axis ESR_REACTION=21 is configured and enabled with $AC_ESR_ENABLE=1, but e.g. no retraction motion is enabled with POLFMASK, then for this axis ESR_REACTION=22 is active (extended stopping).

**Examples for the behavior of path axes for different enable signals**

"NC-controlled retraction" is configured for path axes X and Y:

- ESR_REACTION[X] =21
- ESR_REACTION[Y] =21
1. "Extended stop and retract" and retraction motion is enabled for both path axes:
   - $AA_ESR_ENABLE(X)$=1
   - $AA_ESR_ENABLE(Y)$=1
   - POLFMASK(X,Y)
   For rapid lift, X and Y execute the programmed retraction motion.

2. "Extended stop and retract" is only enabled for one path axis, but retraction motion is enabled for both path axes:
   - $AA_ESR_ENABLE(X)$=0
   - $AA_ESR_ENABLE(Y)$=1
   - POLFMASK(X,Y)
   For rapid lift, for axis X, due to the path interrelationship, in spite of the fact that it is not enabled, "extended stopping" corresponding to ESR_REACTION = 22 is executed.
   Axis Y executes the programmed retraction motion.

3. However, for both path axes, "Extended stop and retract" is enabled – but retraction motion only for one path axis:
   - $AA_ESR_ENABLE(X)$=1
   - $AA_ESR_ENABLE(Y)$=1
   - POLFMASK(Y)
   For rapid lift, for axis X, due to the path interrelationship, in spite of the missing enable signal, retraction motion "extended stopping" corresponding to ESR_REACTION = 22 is executed.
   Axis Y executes the programmed retraction motion.

Responses to stop and axis enable signals

Stop behavior for the retracting movement in response to "Axial feed stop" and "Feed disable" signals are defined with the following channel-specific machine data:

MD21204 $MC_LIFTFAST_STOP_COND

| Bit | Value | Description |
|-----|-------|-------------|
| 0   | 0     | Retraction motion stop for axial feed stop or context-sensitive interpolator stop |
|     | 1     | No retraction motion stop for axial feed stop or context-sensitive interpolator stop |
| 1   | 0     | Retraction motion stop for feed disable in the channel |
|     | 1     | No retraction motion stop for feed disable in the channel |
R3: Extended stop and retract

5.2 Control-managed ESR - 840D sl only

---

**Note**

**Influence of the interface signals**

- The axial NC/PLC interface signal DB31 DBB4.3 (feed stop) influences the complete retraction motion. All axis motion defined using `POLFMASK` and `POLFMLIN` are stopped.
- The NC/PLC interface signal, DB21 DBB7.3 (NC stop) has no effect on the retraction movement.

---

**Programming**

The destination position for the retraction axis is programmed with the language command:

**Retraction position `POLF`**

The retraction position for geometry and channel axes is programmed using the `POLF` program command. The retraction position can be specified either absolutely or incrementally. When the retraction position is incrementally specified, the effective retraction position is obtained from the actual position of the axis at the time of rapid lift plus the programmed retraction distance.

- **Absolute**: `POLF[axis name] = <retraction position>`
- **Incremental**: `POLF[axis name] = IC(<retraction distance>)`

`POLF` is modal.

**Geometry axes**

The retraction positions of geometry and channel axes are programmed in the workpiece coordinate system WCS. The frame valid in the channel at the time of rapid lift is taken into account for the retraction motion.

| NOTICE |
| --- |
| **Frames with rotation** |
| Frames with rotation influence the retraction position of geometry axes. |

**Machine axes**

The retraction positions of machine axes are programmed in the machine coordinate system MCS. The frame valid in the channel at the time of rapid lift is not taken into account for the retraction motion.

| NOTICE |
| --- |
| **Frames with rotation** |
| Frames with rotation do **not** influence the retraction position of machine axes. |

For the same axis names of a channel and machine axis, retraction motion is executed in the workpiece coordinate system.
Enable for retraction without geometrical interrelationship \texttt{POLFMASK}

The program command \texttt{POLFMASK} allows axes to be selected, which, when rapid lift is activated, traverse \textit{independently} to their position defined with \texttt{POLF}. A variable parameter list can be used to select any number of axes for rapid lift; however, all axes must be located in the \textit{same coordinate system}.

Enable for retraction with linear relationship \texttt{POLFMLIN}

The language command \texttt{POLFMLIN} allows axes to be selected that when rapid lift is activated, should travel to their position defined with \texttt{POLF} in a linear relationship. A variable parameter list can be used to select any number of axes for rapid lift; however, all axes must be located in the \textit{same coordinate system}, (i.e. only geo axes). Between initiating retraction and the linear relationship being reached, a path is traveled, which, although deviating from the programmed path, does not quite reach the linear relation. If the constellation is unsuitable, this transition may last as far as the end points programmed with \texttt{POLF}. The control optimizes to the shortest possible transition.

General

The following sections are equally valid for \texttt{POLFMASK} and \texttt{POLFMLIN}.

The parameters valid at triggering time are decisive for the retraction movement. If one of these parameters (G code, \texttt{POLF}, \texttt{POLFMASK}, \texttt{POLFMLIN}, Frame, etc.) changes during retraction (block change), this change does not affect the retracting movement that has already been started.

Before rapid lift to a fixed position can be enabled via \texttt{POLFMASK} or \texttt{POLFMLIN} a position must have been programmed with \texttt{POLF} for the selected axes. There is no machine data for pre-assigning the values of \texttt{POLF}.

During interpretation of \texttt{POLFMASK} or \texttt{POLFMLIN}, alarm 16016 is issued if \texttt{POLF} has not been programmed.

If retraction is activated, the position for retraction can still be changed. However, it is no longer possible to change the coordinate system and an attempt is rejected with an alarm 16015.

If \texttt{POLF} is programmed again after enabling retraction, the position at which this axis was first programmed must be specified in the coordinate system, in which this axis was first programmed.

Change coordinate system

If the coordinate system is to be changed, rapid lift must first be deactivated using \texttt{POLFMASK} or \texttt{POLFMLIN}, and only then can programming commence with \texttt{POLF} in the new coordinate system.

Deactivate rapid lift

\texttt{POLFMASK} or \texttt{POLFMLIN} without specifying an axis \texttt{deactivates} rapid lift for all axes activated in the enable call.
Interactions, POLFMASK/POLFMLIN

The last data entered for a specific axis in one of the two instructions applies. For example:

| Program code | Description |
|--------------|-------------|
| N200 POLFLIN(X,Y,Z) ; | Linear retraction motion for X, Y and Z |
| ...           |             |
| N300 POLFMASK(Z) ; | Independent retraction for Z |
|               ; | (no longer any linear interpolation with X and Y) |
| N500 POLFMLIN(X,Z) ; | Linear retraction motion for X and Z |
|               ; | The independent retraction of Z, activated with POLFMASK, is |
|               ; | deleted. The retraction of Y, activated with POLFMLIN, is |
|               ; | also deleted. |
|               ; | Only X and Z carry out a retraction movement |

Part program start

The retraction positions (POLF) and axial enable signals (POLFMASK, POLFMLIN) are deleted when the part program is started. This means that the user must reprogram the retraction positions and the axial enable signals (POLFMASK, POLFMLIN) in every part program.

5.2.6 Trigger sources

A distinction must be made between the ESR trigger sources on a user-for-user basis by evaluating system variables. This is the reason that all system variables are available, which can be read in synchronized actions.

General trigger sources

- Digital inputs (NCU module) or the internal control image of digital outputs that can be read back: $A_IN, $A_OUT
- Channel status: $AC_STAT
- VDI signals
- System variables written from the PLC: $A_DBB, $A_DBW, $A_DBD

It is not recommended to use these system variables, written by the PLC for time-critical signals, as in this case, the PLC cycle time is included in the total response time. Nevertheless, it is an appropriate way for the PLC to influence the sequence or release of the extended stop and retract function. However, linking-in PLC states in such a fashion does make sense if these are exclusively input from the PLC (e.g. Emergency-Stop, reset button, stop button)

- Group signals from alarms: $AC_ALARM_STAT
Axial trigger sources

- Emergency retraction threshold of the following axis (synchronism difference of the electronic coupling: $VA_{EG\_SYNCDIFF}[\text{following axis}]

Alarm responses

When ESR is active, the alarm responses NOREADY and STOPBYALARM are delayed by one IPO cycle. The following self-clearing alarm is displayed to indicate this delay: Alarm 21600 "Monitoring for ESR active".

Note

The display of this alarm can be suppressed by:

MD11410 $MN\_SUPPRESS\_ALARM\_MASK$, bit 16 = 1

5.2.7 Logic gating functions: Source and reaction linking

The flexible logic operation possibilities of the static synchronized actions can be used to trigger specific reactions based on sources. Linking all relevant sources using static synchronized actions is the responsibility of the user/machine manufacturer. They can selectively evaluate the source system variables as a whole or by means of bit masks, and then make a logic operation with their desired reactions. The static synchronous actions are effective in all operating modes. A detailed description of how to use synchronous actions can be found in:

References:

- Function Manual, Synchronized Actions
- Programming Manual Work Preparation (Synchronized Actions, System Variables)

Linking of axial sources with global or channel specific sources can be configured variably with the aid of $AA\_TYP$ (axis type).

5.2.8 Activation

Option

The function "Extended stop and retract" is an option.

Axis-specific function enable ($AA\_ESR\_ENABLE$)

The axis-specific function enable is realized using the system variable:

$AA\_ESR\_ENABLE[\text{<axis>}] = 1$
Axis-specific enable for extended stopping

An axis is enabled for extended stopping with:

$$\text{MD37500 } $\text{MA_ESR_REACTION[axis]} = 22$$

Axis-specific enable for retraction

An axis is enabled for retraction using:

$$\text{MD37500 } $\text{MA_ESR_REACTION[axis]} = 21$$

Channel-specific trigger ($\text{AC_ESR_TRIGGER}$)

ESR is triggered on a channel-for-channel basis by setting the following system variable:

$$\text{AC_ESR_TRIGGER} = 1$$

ESR is then realized in all channel axes for which the following applies:

$$\text{AA_ESR_ENABLE[<axis>]} == 1 \text{ AND } \text{MD37500 } $\text{MA_ESR_REACTION[axis]} == 21 \text{ OR } 22$$

5.2.9 Configuring aids for ESR

Voltage failure

For the following modules, when the line voltage fails, the power supply must be maintained using suitable measures at least until it is ensured that the axes have come to a stop:

- SINUMERIK 840D sl NCU 7x0
- SINUMERIK NCU I/O modules
- SIMATIC PLC I/O modules
- SINAMICS drive system S120 (booksize)

DC link energy

The energy available in the DC link of the drive units when the line supply fails is calculated as follows:

$$E = \frac{1}{2} \times C \times (\text{VDC link}_{\text{alarm}}^2 - \text{VDC link}_{\text{min}}^2)$$

- $E$: Energy in Wattseconds [Ws]
- $C$: Total capacity of intermediate circuit in Farad [F]

$\text{VDC link}_{\text{alarm}}$: DC link voltage below which an undervoltage is identified.

$\text{VDC link}_{\text{min}}$: Lower limit of the DC link voltage for safe and reliable operation, taking into account the motor-specific EMF.

Note $\text{VDC link}_{\text{min}} > \text{VDC link}_{\text{off}}$
This energy is available for a minimum time of $t_{\text{min}}$:

$$t_{\text{P}} = \frac{E}{P_n \cdot \eta}$$

- $E$: Energy in Wattseconds [Ws]
- $t_{\text{P}}$: Buffer time in milliseconds [ms]
- $P_n$: Power in kilowatt [kW]
- $\eta$: Efficiency of the drive unit

**Example**

**Assumptions:**
- $P_n = 16$ kW, infeed (ALM) with 3-ph. 380 VAC
- $C = C_{\text{act}} - 20\% = 20,000 \, \mu\text{F} - 20\% = 16,000 \, \mu\text{F} = 16 \times 10^{-3} \, \text{F}$, for safety, a 20\% lower effectively available capacity is assumed.
- $V_{\text{DC link}} = 550$ V
- $V_{\text{DC link}} = 350$ V

$$E = \frac{1}{2} \cdot 16 \times 10^{-3} \, \text{F} \cdot ((550 \, \text{V})^2 - (350 \, \text{V})^2) = 1440 \, \text{Ws}$$

This energy is available for a time $t_{\text{min}}$, e.g. for a retraction:
- $E = 1440$ Ws
- $P_n = 16$ kW
- $\eta = 0.9$ (assumption)

$$t_{\text{min}} = \frac{1440 \, \text{Ws}}{16 \, \text{kW} \times 0.9} = 81 \, \text{ms}$$

The following table shows a summary of the values for various SINAMICS infeeds (ALMs). Whereby, the nominal ($C_{\text{max}}$) and minimum ($C_{\text{min}}$) capacitance are taken into account.

The total capacitance of the DC link available is made up, taking into account the respective maximum capacitance $C_{\text{max}}$ (charge limit), from the capacitances of the infeed (ALM), the motor modules as well as any capacitor modules.

The capacitance specified in the table for the minimum energy content ($E_{\text{min}}$ for $C_{\text{min}}$) takes into account a component tolerance of -20\% (worst case).

**Note**

It is recommended that the SIZER configuration tool be used to determine the total capacitance of the DC link available.
R3: Extended stop and retract

5.2 Control-managed ESR - 840D sl only

Table 5-1 SINAMICS infeeds (ALMs): Nominal and minimum buffer times

| ALM | Max. capacitance Cmax[μF] | Energy content Emax for Cmax [Ws] | Energy content Emin for Cmin [Ws] | Backup time tPmax for Pmax [ms] | Buffer time tPmin for Pmax [ms] |
|-----|---------------------------|----------------------------------|----------------------------------|---------------------------------|---------------------------------|
| 16  | 20000                     | 1800                             | 1440                             | 101.25                          | 81.00                           |
| 36  | 20000                     | 1800                             | 1440                             | 45.00                           | 36.00                           |
| 55  | 20000                     | 1800                             | 1440                             | 29.46                           | 23.56                           |
| 80  | 20000                     | 1800                             | 1440                             | 20.25                           | 16.20                           |
| 120 | 20000                     | 1800                             | 1440                             | 13.50                           | 10.80                           |

Energy balance

When configuring retraction motion, it is always necessary to consider the energy flow (balance) to find out whether an additional capacitor module or a generator axis (with correspondingly dimensioned flywheel effect) is required - or not.

Stopping as energy supply

From approx. the third interpolation cycle, the speed setpoints of the configured stopping or retraction axes change. After this time the braking phase starts (if no drive independent shutdown is projected for this axis).

As soon as the braking process is initiated, the energy released in this manner is available for retraction motion. An energy balance must be used to ensure that the kinetic energy of the braking axes is sufficient for retraction.

The energy balance shows the maximum setting for the interpolator cycle time, which will allow a safe emergency retraction to be executed.

Example

With a 16 kW unit under maximum load and minimum intermediate circuit capacity it should be possible to execute an emergency retraction without generator operation. Interpolation cycle time may theoretically not be more than 4.86 ms, i.e. in this case a maximum of 4 ms could be set.

If necessary, a more powerful NCU must be chosen in order to achieve optimal conditions.

Generator operation

For cases in which the DC link energy is not sufficient for safe and reliable retraction (minimum 3 IPO cycles), generator operation can be configured for a drive using the SINAMICS "Vdc control" function. In this case, the mechanical energy of an axis is fed back into the DC link.

The energy stored in an axis can be calculated as follows:

\[ E = \frac{1}{2} \cdot J_{\text{tot}} \cdot \omega^2 \]

- \( J_{\text{tot}} \): Total moment of inertia [kg*m²]
- \( \omega \): Angular velocity at the time of switchover to generator operation [s⁻¹]
This energy is fed back into the DC link with an efficiency of approx. 90%:

When using infeeds with a high power rating (55, 80, 120 kW) for generator operation, it is recommended to use a dedicated axis with additional flywheel effect. For this axis, after the drive has accelerated to the rated speed, energy is only required to compensate for friction losses.

However, in principle, any axis can be used for generator operation, if this is not directly involved in extended stopping and retraction or in couplings that must be specifically maintained.

In order to prevent that the DC link voltage does not become too high when using generator operation (limit value: pulse cancelation, drive) and as a consequence that the pulses are canceled for the axes, adequately dimensioned pulsed resistor modules must be used or the SINAMICS function "Vdc control, overvoltage monitoring".

### 5.2.10 Control system response

#### 5.2.10.1 Axis behavior depending on the enable signals

**Systematic**

If an axis is configured as NC-controlled retraction axis:

MD37500, $MA_ESR_REACTION[<axis>]= 21

The following enable signals must be available to execute retraction motion in the case of ESR:

- Axial ESR enable using system variable: $AA_ESR_ENABLE[<axis>]= 1
- Axial retraction enable using a part program command: POLFMASK(<axis>)

If one of the two enable signals is not available, then in the case of ESR, the axis responds corresponding to the following table:

| Enable signals                           | Response in the case of ESR                                                                 |
|------------------------------------------|------------------------------------------------------------------------------------------|
| $AA_ESR_ENABLE[<axis>]= 1                | For the axis, implicit NC-controlled stopping becomes active, corresponding to MD37500, $MA_ESR_REACTION[<axis>]= 22. |
| No enable via POLFMASK                   |                                                                                          |
| $AA_ESR_ENABLE[<axis>]= 0                | Axis is a path axis: For the axis, the ESR response is implicitly changed to NC-controlled stopping. The response corresponds then to a configuration of MD37500, $MA_ESR_REACTION[<axis>]= 22. |
| Enable via POLFMASK(<axis>)             | Axis is a special axis: The axis is stopped with rapid stop.                               |

Special functions

Function Manual, 07/2012, 6FC5397-2BP40-3BA0 381
Examples for axes with path relationship

For the subsequent examples, two path axes X and Y are assumed, which are configured as NC-controlled retraction axes and are programmed for the retraction positions:

- MD37500, $MA_ESR_REACTION[X] = 21
- MD37500, $MA_ESR_REACTION[Y] = 21
- $POLF[X] = \text{<retraction position> LFPOS}
- $POLF[Y] = \text{<retraction position> LFPOS}

### Initial situation
- $AA_ESR_ENABLE[X] = 1$
- $AA_ESR_ENABLE[Y] = 1$
- $POLFMASK(X,Y)$
- Program traversing motion: Y

### Response in the case of ESR
- Retraction motion in X and Y.

### Examples for axes with path relationship

| Initial situation                                      | Response in the case of ESR                                                                 |
|--------------------------------------------------------|-------------------------------------------------------------------------------------------|
| - $AA_ESR_ENABLE[X] = 0                               | X does not have ESR enable. As X is a path axis, for X, the ESR response is implicitly changed to 22 (stopping). This is the reason that X continues to traverse corresponding to the programmed traversing motion. |
| - $AA_ESR_ENABLE[Y] = 1                               | Y executes retraction motion. After the delay time for ESR axes expires, MD21380, $MC_ESR_DELAY_TIME1, X is stopped with the path acceleration. |
| - $POLFMASK(X,Y)$                                     |                                                                                           |
| - Program traversing motion: X and Y                  |                                                                                           |

### Initial situation
- $AA_ESR_ENABLE[X] = 1
- $AA_ESR_ENABLE[Y] = 1
- $POLFMASK(X)$
- Program traversing motion: X

### Response in the case of ESR
- X has no retraction enable ($POLFMASK$). X does not execute any retraction motion. As X is a path axis, for X, the ESR response is implicitly changed to 22 (stopping). This is the reason that X continues to traverse corresponding to the programmed traversing motion.
- Y executes retraction motion. After the delay time for ESR axes expires, MD21380, $MC_ESR_DELAY_TIME1, X is stopped with the path acceleration.

### Initial situation
- $AA_ESR_ENABLE[X] = 1
- $AA_ESR_ENABLE[Y] = 1
- $POLFMASK(Y)$
- Program traversing motion: X

### Response in the case of ESR
- Y has not been enabled for retraction ($POLFMASK$). see lines above, syntax Y does not execute retraction motion. As Y is a path axis, for Y, the ESR response is implicitly changed to 22 (stopping). This is the reason that Y continues to traverse corresponding to the programmed traversing motion.
- X executes retraction motion. After the delay time for ESR axes expires, MD21380, $MC_ESR_DELAY_TIME1, Y is stopped with the path acceleration.

### Initial situation
- $AA_ESR_ENABLE[X] = 1
- $AA_ESR_ENABLE[Y] = 1
- $POLFMASK(X)$
- Program traversing motion: X and Y

### Response in the case of ESR
- Y has not been enabled for retraction ($POLFMASK$). see lines above, syntax Y does not execute retraction motion. As Y is a path axis, for Y, the ESR response is implicitly changed to 22 (stopping). This is the reason that Y continues to traverse corresponding to the programmed traversing motion.
- X executes retraction motion. After the delay time for ESR axes expires, MD21380, $MC_ESR_DELAY_TIME1, Y is stopped with the path acceleration.

Examples for axes without path relationship

For the subsequent examples, path axes X and a command axis B are assumed, which are configured as NC-controlled retraction axis:

- MD37500, $MA_ESR_REACTION[X] = 21
- MD37500, $MA_ESR_REACTION[B] = 21
5.2 Control-managed ESR - 840D sl only

### Initial situation
- $AA_ESR_ENABLE[X] = 1$
- $AA_ESR_ENABLE[B] = 0$
- POLFMASK (X)
- Program traversing motion: X and B

#### Response in the case of ESR
- B is not enabled for ESR. As B is not a path axis, B is immediately stopped with a rapid stop.
- X executes retraction motion.

- $AA_ESR_ENABLE[X] = 1$
- $AA_ESR_ENABLE[B] = 1$
- POLFMASK (X)
- Program traversing motion: X and B

#### Response in the case of ESR
- B is not enabled for retraction (POLFMASK). However, as B is enabled for ESR, for B, the ESR response is implicitly changed to 22 (stopping). This is the reason that B continues to traverse corresponding to the programmed traversing motion.
- Y executes retraction motion.
- After the delay time for ESR axes expires, MD21380, $MC_ESR_DELAY_TIME1$, B is stopped with axial acceleration.

---

#### 5.2.10.2 POWER OFF/POWER ON

If the user-specific retraction logic is programmed within motion synchronized actions, then it must be observed that this is still not active when the control boots (POWER ON). A retraction logic, which should be active after POWER ON, must therefore lie in an ASUB started from the PLC or must be called event-controlled.

**References**

For detailed information about event-driven program calls see:

Function Manual, Basic Functions; Mode Group, Channel, Program Operation, Reset Response (K1), Section: Event controlled program calls

---

#### 5.2.10.3 Operating mode change, NC Stop, reset

Static synchronized actions (IDS keyword) can be used for user-specific retraction logic as the following system states or state change have no influence on this:

- Operating mode change, e.g. DB11, DBX0.0 (AUTOMATIC)
- NC stop, e.g. DB21, ... DBX7.3 (NC stop)
- Reset, e.g. DB21, ... DBX7.7 (reset)

**Note**

**NC stop and command axes**

Traversing of command axes is interrupted for an NC stop.

---

**Note**

**POLF/POLFMASK reset and program commands**

For reset, the programmed absolute retraction positions (POLF) and the enable signals for the retraction axes (POLFMASK) are not deleted.
5.2.10.4 Part program start, NC start

In order that a defined initial state is available when a part program starts, the programmed absolute retraction positions and the enable signals of the retraction axes are deleted when the part program starts.

**NOTICE**

Retraction positions and enable signals

When a part program starts, the absolute retraction positions (POLF) and the enable signals of the retraction axes (POLFMASK) must be reprogrammed corresponding to the actual requirements.

5.2.10.5 Alarm behavior

The alarm response depends on the axis in which a fault occurs:

- Fault in an axis outside the axis grouping of an electronic gear:
  
  This axis switches off "normally". Extended stopping and retraction continue to operate as before - or are initiated by such a fault.

- Error in a leading axis (LA):

  selective switchover to actual value linkage already during stop, otherwise as previously.

- Error in a following axis (FA):
  
    - Carry out retract: Retraction axis may not be a following axis, that is, no conflict.
    
    - Carry out stop: The following axis may react with uncontrollable behavior. Saving the workpiece/tool must be left to the retraction; however, the stop should not disrupt the process any further.

- Error in the retraction axis: There is no retraction.

- Emergency Stop

  An Emergency Stop is not a fault from a control system point of view, rather the response is the same as for any other control signal. For safety reasons, Emergency Stop interrupts the interpolation and all traversing motion, and also cancels the electronic coupling by withdrawing the controller enable signals.

  In applications where the coupling and traversing movements must remain valid after Emergency Stop, the PLC must delay the Emergency Stop long enough for the required NC or drive-end reactions to terminate.

  The following interface signal is available as feedback signal to the PLC:

  IS "ESR reaction not initiated" DB31, ... DBX98.7

  If an alarm with cross-channel NOREADY reaction is issued during the active phase of the ESR (i.e. NOREADY | NCKREACTIONVIEW | BAGREACTIONVIEW), then ESR is triggered in all channels.
5.2.10.6 Block search, REPOS

Extended stop and retract does not affect block search or REPOS motions.

5.3 ESR executed autonomously in the drive

5.3.1 Fundamentals

Function

Drive-autonomous extended stop and retract (ESR) enables the fast separation of workpiece and tool independent of the higher-level control (NC).

For this purpose, the following axial functions can be configured in the drive:

- Generator operation
- Extended stop
- Retract

The drive-autonomous responses are automatically initiated in error situations. The triggering of the drive-autonomous responses can also be realized user-specific via the part programs or synchronized actions from the higher-level control system.

As the stopping and retraction motion of the drive-autonomous ESR are purely axial, in contrast to control-managed ESR, couplings are not taken into account.

ESR executed autonomously in the drive can also be used in combination with control-managed ESR. This also allows stopping and retraction motion in the drives if they are no longer specified from the control, e.g. as a result of a communication failure.

Note

DRIVE-CLiQ

For possibly occurring DRIVE-CLiQ errors, in order to keep the reaction to the functions of the drive-autonomous ESR as low as possible, within the scope of ESR, we recommend that the motor modules involved are not connected through a line-type topology, but instead are directly connected to the Control Unit (CU).

Link axes

The "Drive-autonomous extended stop and retract (ESR)" function cannot be used in conjunction with link axes.
R3: Extended stop and retract

5.3 ESR executed autonomously in the drive

Requirements

The following conditions must be fulfilled:

- SINAMICS and SINUMERIK software version: ≥ V4.4
- Servo drive object, PROFIdrive telegram: 102 - 199
- Servo drive object, function module "ESR" active

The function module "Extended stop and retract (ESR)" must be activated via the drive wizard of SINUMERIK Operate. For a description of the drive wizard, see: SINUMERIK Operate online help.

- Servo drive object, generator operation (Vdc control) active in a drive
- Control unit drive object, PROFIdrive telegram: 390, 391
- The 24 V power supply for the modules is buffered via CSM or UPS

5.3.2 Configuring stopping in the drive

Drive integrated shutdown is configured via the following drive parameters:

| Parameter | Description          |
|-----------|----------------------|
| p0888     | ESR: Configuration   |
| Value     | Meaning              |
| 1         | Extended stopping (function integrated in the drive) |

| Parameter | Description          |
|-----------|----------------------|
| p0891     | ESR: Off ramp        |
| Value     | Meaning              |
| 0         | OFF3 (default)       |
|           | The drive is braked along the OFF3 down ramp by immediately entering n_set = 0 (p1135: OFF3 ramp-down time). |
| 1         | OFF1                 |
|           | By immediately entering n_set = 0, the drive is braked along the ramp generator down ramp (p1121: ramp-function generator ramp-down time). |

| Parameter | Description          |
|-----------|----------------------|
| p0892     | ESR: Timer           |
|           | The drive travels at a constant velocity for the configured time with the speed setpoint that was present when the fault occurred. |
5.3 ESR executed autonomously in the drive

Feedback signal

The stop status is returned to the control (see Section "Feedback of the ESR status (Page 391)").

References

For a detailed description of drive parameters, refer to:
SINAMICS S120/S150 Parameter Manual

5.3.3 Configuring retraction in the drive

Drive integrated retraction is configured using the following drive parameters:

| Parameter | Description |
|-----------|-------------|
| p0888     | ESR: Configuration |
| Value     | Meaning     |
| 2         | Extended retraction (function integrated in the drive) |

| Parameter | Description |
|-----------|-------------|
| p0891     | ESR: Off ramp |
| Value     | Meaning     |
| 0         | OFF3 (default) |
| n_set = 0 (p1135: OFF3 ramp-down time). |
| 1         | OFF1         |
| By immediately entering n_set = 0, the drive is braked along the ramp generator down ramp (p1121: ramp-function generator ramp-down time). |
### 5.3 ESR executed autonomously in the drive

**Parameter** | **Description**
--- | ---
p0892 | ESR: Timer

The parameter specifies the total time that elapses to reach the speed specified in p0893 followed by constant velocity travel. This is followed by an OFF1 or OFF3 ramp depending on the parameterization in p0891.

**Parameter** | **Description**
p0893 | ESR: Speed

The parameter specifies the speed (retraction speed) that is reached when initiating retraction via an OFF3 ramp.

---

**Feedback signal**

The retract status is returned to the control (see Section "Feedback of the ESR status (Page 391)").

**References**

For a detailed description of drive parameters, refer to:
SINAMICS S120/S150 Parameter Manual
5.3.4 Configuring generator operation in the drive

The generator mode for the drive-autonomous ESR is configured using the following drive parameters:

| Parameter | Description |
|-----------|-------------|
| p0888     | ESR: Configuration          |
|           | **Value** | **Meaning**          |
|           | 3      | Generator operation (Vdc controller) |

| Parameter | Description                              |
|-----------|------------------------------------------|
| p1240     | Vdc controller or Vdc monitoring configuration |
|           | **Value** | **Meaning**                              |
|           | 2      | When reaching the lower DC link voltage threshold (p1248) - for ESR - the motor is braked in order to utilize its kinetic energy to buffer the DC link. |

| Parameter | Description |
|-----------|-------------|
| p1248     | Lower DC link voltage threshold |
|           | Setting the lower threshold for the DC link voltage. For p1240 = 2, this threshold is used as setpoint limit for the Vdc_min controller. |

![Figure 5-6 Configuring drive integrated generator operation](image-url)
Generator minimum speed

The lower limit of the motor speed of the generator axis is configured using drive parameter p2161:

| Parameter | Description |
|-----------|-------------|
| p2161     | Speed threshold value 3 |
|           | Speed threshold value for the message: $n_{act} < $speed threshold value 3 |

Feedback signal

The generator operation status is returned to the control (see Section "Feedback of the ESR status (Page 391)").

References

A detailed description of drive parameters and the Vdc control can be found in:

- SINAMICS S120/S150 List Manual
- SINAMICS S120 Function Manual; Section "Servo control" > "Vdc control"

5.3.5 ESR is enabled via system variable

The ESR response of an axis, configured via drive parameter, must be programmed on a user-specific basis in a part program/synchronized action using the following axis-specific system variable:

| Value | Meaning                      |
|-------|------------------------------|
| 1     | ESR responses enabled in the drive |
| 0     | ESR responses inhibited in the drive |
### 5.3.6 Triggering ESR via system variable

Triggering ESR responses, configured via drive parameter and enabled via 
$AA\_ESR\_ENABLE[<axis>]$, must be programmed on a user-specific basis in a part program/synchronized action via the following NC-specific system variable:

| $AN\_ESR\_TRIGGER | Value | Meaning                                      |
|-------------------|-------|----------------------------------------------|
|                   | 1     | trigger drive-integrated ESR reactions that have been released |
|                   | 0     | do not trigger drive-integrated ESR reactions that have been released |

**Note**

The trigger signal must be active in the part program/synchronized action for at least 2 IPO cycles to ensure that the PROFIdrive telegram is transferred to the drive.

**Mapping in the drive unit**

System variable $AN\_ESR\_TRIGGER$ is mapped to drive parameter p0890.0. Interconnecting drive parameter p0890.0 with the corresponding information in telegram 390 or 391 to the Control Unit is realized automatically when the drive unit commissions the ESR reactions.

**References**

For a detailed description of drive parameters, refer to:

SINAMICS S120/S150 List Manual

---

### 5.3.7 Feedback of the ESR status

The drive signals back the actual ESR status to the control using the message word (MELDW) of the cyclic PROFIdrive telegram. In the PLC, the status can be read via system variables and NC/PLC interface signals.

**System variable**

The system variables can be evaluated in the part program / synchronized action, for example, in order to trigger the drive-autonomous ESR (see Section "Triggering ESR via system variable (Page 391)") or to acknowledge the ESR reactions triggered in the drive (see Section "Acknowledge ESR reactions (Page 392)").
Interrelationship between signals

| MELDW | System variable | NC/PLC interface signal | Meaning |
|-------|-----------------|-------------------------|---------|
| Bit 9 | $AA_ESR_STAT.Bit 2 | DB31, ... DBX95.2 | ESR: Reaction triggered or generator operation active (r0887.12) |
| Bit 4 | $AA_ESR_STAT.Bit 3 | DB31, ... DBX95.1 | ESR: DC link undervoltage (p1248) |
| Bit 2 | $AA_ESR_STAT.Bit 4 | DB31, ... DBX95.3 | ESR, generator operation: Speed lower than minimum (p2161) |

5.3.8 Acknowledge ESR reactions

Acknowledging axis-specific drive-independent ESR reactions must be programmed on a user-specific basis in a part program/synchronized action. After the ESR reaction in the drive has been completed, an edge change must be generated in the system variables $AA_ESR_ENABLE: 1→0→1. In order that the drive-integrated ESR reactions can be retriggered, the system variable $AN_ESR_TRIGGER must also be reset to a value of 0.

System variable and drive parameters

The following diagram shows the relationship between system variables and drive parameters when triggering and acknowledging ESR reactions.
R3: Extended stop and retract

5.3 ESR executed autonomously in the drive

1. NC: Enabling the ESR reaction via $AA_ESR_ENABLE = 1 (axis-specific)
2. NC: Triggering ESR responses via $AN_ESR_TRIGGER = 1
3. Drive: The triggering of the ESR reaction has been detected (CU_STW.2 == 1) ⇒ r0887.12 = 1 AND r0887.13 = 1
   Feedback signal to the NC via $AA_ESR_STAT.BIT2 = 1 (axis-specific)
4. NC: Acknowledging the ESR reaction via $AA_ESR_ENABLE = 0 (axis-specific)

Black: At instant in time ④, the ESR reaction no longer runs (r0887.13 == 0) ⇒
5. Drive: Reset "ESR triggered": r0887.12 = 0
   Feedback signal to the NC via $AA_ESR_STAT.BIT2 = 0 (axis-specific)
6. NC: The ESR reaction can be re-enabled ⇒ $AA_ESR_ENABLE = 1 AND $AN_ESR_TRIGGER = 0

Red: At instant in time ④, the ESR reaction still runs (r0887.13 == 1) ⇒
5. Drive: "ESR initiated" is only reset (r0887.12 = 0), if the ESR response in the drive has been completed: r0887.13 == 0
   Feedback signal to the NC via $AA_ESR_STAT.BIT2 = 0 (axis-specific)
6. See ⑥ above

Figure 5-7 Sequence: Trigger and acknowledge drive-integrated ESR reactions

5.3.9 Configuring ESR in the part program

Configuring drive-autonomous ESR functions "stop" and "retract" can be changed using the commands from the part program described in the following.

Function: $ESRS(...):(stopping)$

Syntax

$ESRS(<access_1>,<stopping time_1>[,...,<axis_n>,<stopping time_n>])$

Meaning

| Element | Description |
|---------|-------------|
| $ESRS$: | Using the function, drive parameters can be changed regarding the drive-autonomous "stop" ESR function. Special situations:  
• Must be alone in the block  
• Triggers a preprocessing stop  
• Cannot be used in synchronized actions  
• A maximum of five axes can be programmed per function call; n = 5 |
| $<axis_n>$: | For the specified axis, writes to drive parameter p0888 (configuration): p0888 = 1 |

Data type: AXIS
Range of values: Channel axis identifier
## R3: Extended stop and retract

### 5.3 ESR executed autonomously in the drive

**Element** | **Description**
---|---
<stopping time_n>: | For the axis specified under <Axis_n>, writes to drive parameter p0892 (timer): P0892 = <stopping time_n>
  
  **Unit:** s
  
  **Data type:** REAL
  
  **Range of values:** 0.00 - 20.00

---

**Function: ESRR(...):(retraction)**

**Syntax**

ESRR(<axis_1>,<retraction distance_1>,<retraction velocity_1>[,...,<axis_n>,<retraction distance_n>,<retraction velocity_n>])

**Meaning**

**Element** | **Description**
---|---
ESRR: | Using the function, drive parameters can be changed regarding the drive-autonomous "retract" ESR function.
  
  **Special situations:**
  
  - Must be alone in the block
  - Triggers a preprocessing stop
  - Cannot be used in synchronized actions
  - A maximum of five axes can be programmed per function call; n = 5

<axis_n>: | For the specified axis, writes to drive parameter p0888 (configuration):
  
  p0888 = 2

  **Data type:** AXIS
  
  **Range of values:** Channel axis identifier

<retraction velocity_n>: | For the axis specified under <Axis_n>, writes to drive parameter p0893 (retraction speed):
  
  p0893 = (<retraction velocity> based on the currently effective total transmission ratio converted into the corresponding retraction speed)

  **Unit:** mm/min, inch/min, degrees/min (depending on the unit of the axis)
  
  **Data type:** REAL
  
  **Range of values:** 0.00 - MAX

<retraction distance_n>: | For the axis specified under <Axis_n>, writes to drive parameter p0892 (retraction time):
  
  p0892 = <retraction distance_n> / <retraction velocity _n>

  **Unit:** mm, inches, degrees (depending on the unit of the axis)
  
  **Data type:** REAL
  
  **Range of values:** MIN - MAX
Note

Retraction speed

It should be ensured that the drive can reach the programmed retraction speed (see parameters <retraction velocity_n>) within the retraction time (see parameter <retraction distance_n>). The settings in the following drive parameters should be checked:

- p1082[0...n] (maximum speed)
- p1135[0...n] (OFF3 ramp-down time)

Retraction path

The conversion of the retraction path into a retraction duration for drive parameter p0892 is performed under the assumption that the retraction motion is triggered at axis standstill (see "Figure 5-5 Behavior for drive-integrated retraction (Page 388)").

If retraction motion is triggered while the axis is traversing, the actually traversed retraction distance differs from the programmed retraction distance due to the fact that traversing motion is superimposed on the retraction motion.

Dependencies

The programmed values for the retraction path and the retraction velocity refer to the load side. Before writing to the drive parameters these are converted over to the motor side. The transmission ratio effective in the NC at the execution time is applicable for the conversion. If the drive-autonomous ESR for an axis is enabled ($AA_ESR_ENABLE[<axis>] == 1), then it is not permissible to change the axial transmission ratio.

| NOTICE |
| Changing the transmission ratio |
| If the drive-autonomous ESR is not enabled for an axis ($AA_ESR_ENABLE[<axis>] == 0), then the axial transmission ratio can be changed. After the change, it is the sole responsibility of the user/machine manufacturer to appropriately adapt the retraction distance and the retraction velocity. |

Supplementary conditions

Consistent parameter change

In order that the drive parameters are consistently effective, in the part program the drive-autonomous ESR enable signals must be withdrawn before the functions ESR(... and ESRR(...).
Example:

| Program code | Comment |
|--------------|---------|
| N100 $AA_ESR_ENABLE[X1]=0 | Withdraw ESR enable signals |
| N110 $AA_ESR_ENABLE[Z1]=0 |
| N120 $AA_ESR_ENABLE[Y1]=0 |
| N130 ESRR(X1,20.0,10000.0,Z1,-15.0,20000.0) | Change ESR parameters |
| N140 ESRS(Y1,0.5,B1,0.5) |
| N150 $AA_ESR_ENABLE[X1]=1 | Set ESR enable signals |
| N160 $AA_ESR_ENABLE[Z1]=1 |
| N170 $AA_ESR_ENABLE[Y1]=1 |

When executing the functions ESRS(...) and ESRR(...) further execution of the part program is stopped until writing to the drive parameters has been acknowledged. During this time, the following message is used as feedback signal for the user: "Wait for drive parameters from <Function>" is displayed.

**Block search with calculation**

During the block search with calculation, the functions ESRS(...) and ESRR(...) are collected and executed in the action block.

**Block search with calculation in "Program test" (SERUPRO) mode**

During SERUPRO, the functions ESRS(...) and ESRR(...) are executed immediately.

**Reset behavior**

The parameter values written using the functions ESRS(...) and ESRR(...) are overwritten with the parameter values saved in the drive when the drive runs up or for a drive warm restart.

Drive parameters can be saved from the HMI user interface or in the drive. From the HMI user interface:

- Operating area "Commissioning" > Machine data > Drive MD > Save/Reset > Save > Drive Object or drive unit or drive system

**Control run-up (POWER ON)**

If the drive-autonomous ESR should already be active after the control has run up (POWER ON), then the programming and enable must be realized within one of the ASUBs started from the PLC - or using the PROGEVENT mechanism.

**References**

For detailed information about event-controlled program calls, refer to:

Basic Functions Function Manual, Mode Group, Channel, Program Operation, Reset Response (K1);
Section: Event-driven program calls
5.3.10 ESR and Safety Integrated (840D sl)

Delay times

In order that drive-integrated ESR reactions in conjunction with Safety Integrated are executed before the safety reactions associated with STOP E, STOP F or communication failure, then the corresponding delay times regarding safety reactions must be configured in the control and in the drive.

Delay times are configured in the controller using the following machine data:

- **MD36954 $MA_SAFE_STOP_SWITCH_TIME_E** (transition time, STOP E to safe standstill)
- **MD36955 $MA_SAFE_STOP_SWITCH_TIME_F** (transition time, STOP F to STOP B)
- **MD36961 $MA_SAFE_VELO_STOP_MODE** (stop response, safely-reduced speed)
- **MD36963 $MA_SAFE_VELO_STOP_REACTION** (stop response, safely reduced speed)
- **MD10089 $MN_SAFE_PULSE_DIS_TIME_BUSFAIL** (wait time pulse cancellation when the bus fails)

Delay times in the drive are configured using the following drive parameters:

- **p9554** (transition time from STOP E to "Safe Operating Stop" (SOS))
- **p9555** (transition time STOP F to STOP B)
- **p9561** (SG stop response)
- **p9563** (SG-specific stop response)
- **p9580** (wait time after which the pulses are safely cancelled after bus failure)
- **p9697** (delay time for the pulse suppression after bus failure)
- **p9897** (delay time for the pulse suppression after bus failure)

**Note**

The drive parameters are automatically aligned with the corresponding NC machine data using the Safety Integrated commissioning function "Copy SI data".

Trigger drive-integrated ESR reactions

The feedback signal of the safety stop reaction – currently active in the drive – sent to the control is realized using the following axis-specific system variable:

- **$VA_STOPSI[<axis>]** (actual stop reaction)

By evaluating the system variable in a part program/synchronized action, a user-specific response can be realized to the particular stop reaction of the axis, e.g. by triggering the drive-integrated and/or control-managed ESR reactions using the system variable $AN_ESR_TRIGGER and/or $AC_ESR_TRIGGER or $AA_ESR_TRIGGER.
5.3 ESR executed autonomously in the drive

References

A detailed description for configuring and the interaction of drive-integrated ESR responses and Safety Integrated functions is given in:

References
Function Manual SINUMERIK Safety Integrated

5.3.11 ESR and Safety Integrated (828D)

No feedback signal of the safety stop reaction

Within the scope of SINUMERIK 828D, the safety stop reaction currently active in the drive is not signaled back to the control. Therefore, in conjunction with Safety Integrated, only drive-autonomous ESR functions can be used.

The ESR reactions triggered from the drive can be determined on the control side using the axis-specific system variable $AA_ESR_STAT (see Chapter "Feedback of the ESR status (Page 391)" and additional ESR reactions can be triggered on a user-specific basis via $AN_ESR_TRIGGER.

ESR reactions that are not influenced by safety reactions

In order that drive-autonomous ESR reactions in conjunction with Safety Integrated are not influenced by safety reactions as a result of STOP E, STOP F executed in parallel or communication failure, then the appropriate delay times must be configured in the drive regarding safety reactions and the corresponding safety trigger sources to trigger the drive-autonomous ESR reactions.

Delay times

Delay times in the drive are configured using the following drive parameters:

- p9554 (transition time from STOP E to "Safe Operating Stop" (SOS))
- p9555 (transition time STOP F to STOP B)
- p9580 (wait time after which the pulses are safely cancelled after bus failure)
- p9697 (delay time for the pulse suppression after bus failure)
**Safety trigger sources**

Using a BICO interconnection of drive parameter p0890 (BI: ESR trigger) ESR reactions can also be triggered with the safety reactions STOP E, STOP F and communication failure:

| Parameter | BICO interconnection with: | Effect: Trigger ESR reactions for |
|-----------|----------------------------|----------------------------------|
| p0890[1]  | r9721[15]                  | Safety STOP E                    |
| p0890[2]  | r9723[1]                   | Safety STOP F                    |
| p0890[3]  | r9723[2]                   | Safety communication failure     |

**References**

A detailed description of the drive parameters and the BICO technology can be found in:

- SINAMICS S120/S150 List Manual
- SINAMICS S120 Function Manual; Section "Fundamentals of drive technology" > "BICO technology: Interconnecting signals"
5.4 Boundary conditions

Operational performance of the components

The drive components involved in the extended stopping and retraction must be capable of functioning. If one of these components fails, the full scope of the described reaction no longer applies. Axis-specific servo or drive alarms indicating the failure of one of these components are also implicitly signaling that the configured stop or retract reaction of the axis involved is no longer available or only partially.

Motion synchronous actions

Motion synchronous actions are executed in the interpolator cycle. Increasing the interpolation cycle, e.g. due to a high number of active motion synchronized actions, results in a coarser time grid for evaluating trigger conditions and triggering responses for extended stopping and retraction.
5.5 Examples

5.5.1 NC-controlled reactions

Example using NC-controlled reactions. The important details are specified.

Exercise

The A axis is to operate as the generator drive, while the X axis should retract 10 mm at maximum speed in the event of a fault, and axes Y and Z should stop after a delay of 100 ms so that the retraction axis has time to cancel the mechanical coupling.

Requirements

The "Extended stop and retract", "Static synchronized actions" and "ASUB" options must be available.

Parameter assignment

Parameterization or programming required for the example:

| Parameter Assignment                                                                 | Description |
|-------------------------------------------------------------------------------------|-------------|
| $MN_ASUP.START_MASK = 5                                                             | MD11602     |
| $MC_PROG_EVENT.IGN_REFP_LOCK = 'H3F'                                               | MD20105     |
| $MC_IGNORE_REFP_LOCK_ASUP = 'HFFFFFFFF'                                            | MD20115     |
| $MA_ESR_REACTION[X]=21                                                              | MD37500     |
| $MA_ESR_REACTION[Y]=22                                                              |             |
| $MA_ESR_REACTION[Z]=22                                                              |             |
| LFPOS                                                                               |             |
| POLF[X]=IC(10)                                                                      |             |
| POLFMASK(X)                                                                         |             |
| $MC_ESR_DELAY_TIME1=0.1                                                              | MD21380, duration of |
| $MC_ESR_DELAY.TIME2=0.04                                                             | MD21381, braking duration in seconds |
| $AA_ESR_ENABLE[X] = 1                                                                | Set system variables |
| $AA_ESR_ENABLE[Y]=1                                                                  |             |
| $AA_ESR_ENABLE[Z]=1                                                                  |             |
| $AA_ESR_ENABLE[A]=1                                                                  |             |
Trigger conditions and static synchronized actions

Example 1: Trigger condition is the occurrence of alarms, which activate the follow-up (tracking) mode:

Program code

```
IDS=02 WHENEVER ($AC_ALARM_STAT B_AND 'H2000')>0 DO $AC_ESR_TRIGGER=1
```

Example 2: Trigger condition is when the ELG synchronous monitoring responds, if, e.g. Y is defined as ELG following axis and the max. permissible synchronism difference is 100 mm:

Program code

```
IDS=03 WHENEVER $VA_EG_SYNCDIFF[Y]>0.1 DO $AC_ESR_TRIGGER=1
```

5.5.2 Retraction while thread cutting

During thread cutting, for a fault/interruption, axis X should be retracted to the position specified under POLF as response. Axis Z continues to traverse until it is stopped as normal.

| Program code | Comment |
|--------------|---------|
| N10 G0 G90 X200 Z0 S200 M3 | ; |
| N20 G0 G90 X170 | ; |
| N30 POLF[X]=210 | ; Retraction position axis X, absolute |
| N40 LFPOS | ; Retraction via POLF/POLFMASK ON |
| N50 POLFMASK(X) | ; Enable retraction, axis X |
| N60 LFON | ; Fast retraction for thread cutting ON |
| | ; Thread cutting |
| N70 G33 X100 I10 | ; Retraction response, axial: X (abs.) |
| N80 X130 Z-45 K10 | ; dto. |
| N90 X155 Z-128 K10 | ; dto. |
| N100 X145 Z-168 K10 | ; dto. |
| N110 X120 I10 | ; dto. |
| N120 G0 Z0 LFON | ; Fast retraction for thread cutting OFF |
| N130 POLFMASK{} | ; Disable retraction of all axes |
5.5 Examples

5.5.3 Rapid lift using ASUB and fast input

Activating ASUB using the LIFTFAST program command (rapid lift) via fast input 1:

| Program code | Comment |
|--------------|---------|
| N10 SETINT (1) PRIO=1 ABHEB_Y LIFTFAST | ASUB activation, fast input 1 |
| N20 LFPOS | Retraction via POLF/POLFMASK ON |
| N30 POLF[X]=19.5 POLF[Y]=33.3 | Retraction positions axis X Y, absolute |
| N40 POLF[Z]=100 | Retraction position axis Z, absolute |
| N50 X0 Y0 G0 | |
| N60 POLFMASK(X, Y) | Enable retraction, axis X Y |
| N70 Z100 G1 F1000 | Retraction response, axial: X, Y (abs.) |
| N80 POLFMASK(Z) | Enable retraction, axis Z |
| N90 Y10 | Retraction response, axial: Z (abs.) |
| N100 POLFMASK() | Disable retraction of all axes |

5.5.4 Rapid lift, absolute and incremental

Retraction to absolute positions and through an incremental distance:

| Program code | Comment |
|--------------|---------|
| N10 $AA_ESR_ENABLE[X]=1 | Enable retraction, axis X |
| N20 $AA_ESR_ENABLE[Z]=1 | Enable retraction, axis Z |
| N30 $AA_ESR_ENABLE[Y]=1 | Enable retraction, axis Y |
| N40 LFPOS | Retraction via POLF/POLFMASK ON |
| N50 POLF[X]=IC(3.0) POLF[Y]=-4.0 | Retraction position axis X, incremental |
| | Retraction position axis Y, absolute |
| N60 POLF[Z]=100 | Retraction position axis Z, absolute |
| N70 G0 X0 Y0 Z0 | |
| N80 POLFMASK(X, Y) | Enable retraction, axis X Y |
| N90 Z100 G1 F1000 | Retraction response, axial: X (inc.), Y (abs.) |
| N100 POLF[X]=10 | Retraction position axis X, absolute |
| N110 Y0 G1 F1000 | Retraction response, axial: X, Y (abs.) |
| N120 POLFMASK(Z) | Disable retraction, axis X Y and |
| | Enable retraction, axis Z |
| N130 Y10 | Retraction response, axial: Z (abs.) |
| N140 POLFMASK() | Disable retraction of all axes |
### 5.5.5 Lift fast with linear relation of axes

Retracting in a linear relationship, absolute and incremental

| Program code | Comment |
|--------------|---------|
| N10 $AA_ESR_ENABLE[X]=1 | ; Enable retraction, axis X |
| N20 $AA_ESR_ENABLE[Y]=1 | ; Enable retraction, axis Y |
| N30 $AA_ESR_ENABLE[Z]=1 | ; Enable retraction, axis Z |
| N40 LFPOS | ; Retraction via POLF/POLFMASK ON |
| N50 POLF[X]=IC(3.0) POLF[Z]=−4.0 | ; Retraction position axis X, incremental; Retraction position axis Z, absolute |
| N60 POLF[Y]=100 | ; Retraction position axis Z, absolute |
| N70 X0 Y0 Z0 G0 | ; --- |
| N80 POLFMLIN(X, Y) | ; Enable retraction, axis X Y, linear; Relationship |
| N85 POLFMASK(Z) | ; Enable retraction, axis Z |
| N90 Z100 G1 F1000 | ; Retraction response:; — linear relation: X (inc.), Y (abs.); — axial: Z (abs.) |
| N95 POLF[X]=10 | ; Retraction position axis X, absolute |
| N100 Y0 G1 F1000 | ; Retraction response:; — linear relation: X (inc.), Y (abs.); — axial: Z (abs.) |
| N110 POLFMLIN{} | ; Disable retraction, axis X Y |
| N120 Y10 | ; Retraction response, axial: Z (abs.) |
| N130 POLFMASK{} | ; Disable retraction of all axes |
5.6 Data lists

5.6.1 Machine data

5.6.1.1 Channelspecific machine data

| Number | Identifier: $MC_ | Description |
|--------|------------------|-------------|
| 21204  | LIFTFAST_STOP_COND | Stop characteristics for rapid lift |
| 21380  | ESR_DELAY_TIME1   | Delay time (STOPBYALARM, NOREAD) for ESR axes |
| 21381  | ESR_DELAY_TIME2   | Time for interpolatory braking of ESR axes |

5.6.1.2 Axis/spindlespecific machine data

| Number | Identifier: $MA_ | Description |
|--------|------------------|-------------|
| 37500  | ESR_REACTION     | Reaction definition with extended stop and retract |

5.6.2 System variables

| Identifier          | Meaning                                      |
|---------------------|----------------------------------------------|
| $A_DBB              | Read/write data byte from/to PLC             |
| $A_IN               | Digital input                                |
| $A_OUT              | Digital output                               |
| $AA_ESR_ENABLE[<axis>] | Enable "Extended stop and retract"        |
| $AA_TYP[<axis>]     | Axis type                                    |
| $AA_ESR_TRIGGER[<axis>] | Initiate ESR for PLC controlled axis (single axis) |
| $AC_ALARM_STAT      | Alarm status in the channel                  |
| $AC_ESR_TRIGGER     | Trigger ESR, channel-specific               |
| $AC_STAT            | Channel status                               |
## 5.6 Data lists

### 5.6.3 Signals

#### 5.6.3.1 Signals to channel

| Signal name                        | SINUMERIK 840D si | SINUMERIK 828D |
|-----------------------------------|-------------------|----------------|
| Feedrate disable                  | DB21, ... DBX6.0  |                |

#### 5.6.3.2 Signals to axis/spindle

| Signal name                        | SINUMERIK 840D si | SINUMERIK 828D |
|-----------------------------------|-------------------|----------------|
| Feed stop                         | DB31, ... DBX4.3  |                |

#### 5.6.3.3 Signals from axis/spindle

| Signal name                                      | SINUMERIK 840D si | SINUMERIK 828D |
|-------------------------------------------------|-------------------|----------------|
| ESR: DC link undervoltage (p1248)               | DB31, ... DBX95.1 | DB390x.DBX4003.1 |
| ESR: Reaction triggered / generator operation active (r0887.12) | DB31, ... DBX95.2 | DB390x.DBX4003.2 |
| ESR, generator operation: Speed lower than minimum (p2161) | DB31, ... DBX95.3 | DB390x.DBX4003.3 |
6.1 Brief description

Function

The "setpoint exchange" function is used in applications in which the same motor is used to traverse different machine axes.

Replacing the technology function "setpoint changeover" (TE5)

The "setpoint changeover" function replaces the technology function "setpoint changeover" (TE5). Migration of the technology function to the functionality of the "setpoint changeover" function means that you must make changes in the machine data and in the PLC user program:

- Machine data MD63750 is no longer required.
- The meanings of various NC/PLC interface signals have changed.
- Alarms 70451 and 70452 are no longer applicable.
- A setpoint changeover with simulated axes (MD30130 = 0) is no longer supported.
- The known restrictions relating to the technology card function are no longer applicable.
6.2 Startup

The "setpoint exchange" function is required in applications in which a single motor needs to drive a number of axes/spindles such as, for example, on milling machines with special millheads. The spindle motor is operated as both a tool drive and a millhead orienting mechanism.

**Figure 6-1** Example 1: 1 motor encoder, extra encoder for millhead

**Figure 6-2** Example 2: 1 motor encoder, separate millhead encoder and spindle encoder

**Configuration**

Setpoint exchange enables a number of axes to use the same drive.

The same setpoint channel on this drive is assigned a number of times to define the axes participating in setpoint exchange. For this, the following machine data must be pre-assigned with the same logical drive number for every axis:

MD30110 $MA_CTRLOUT_MODULE_NR (setpoint assignment: module number)

**Note**

Alarm 26018 is output if the option is missing.
The encoder assignment is programmed in machine data:

MD30230 $MA_ENC_INPUT_NR

(actual-value assignment: Input on drive module/measuring circuit module)

**Activation**

The setpoint is exchanged and the corresponding interface signals are evaluated in the PLC user program.

**Note**

An existing PLC user program may need to be modified due to changes in the meaning of interface signals in comparison with the technology card solution.

Only one of the machine axes with the appropriate logical drive number may have control via the setpoint channel of the drive at any one time.

Requests to transfer drive control are sent using NC/PLC interface signal:

DB31, ... DBX24.5 (change setpoint output assignment)

The current status of drive control is displayed in the NC/PLC interface signal:

DB31, ... DBX96.5 (setpoint output assignment)

Access rights to the shared drive must be managed in the PLC user program.
Transfer conditions

- Axis standstill of all axes involved.
- Special functions such as reference point approach, measuring, travel to fixed stop, function generator, star/delta changeover, drive parameter set changeover are not active in the axis with drive control.
- No sign-of-life error and no faults pending on PROFIBUS drive.

The PLC interface provides constant information about the current state of the exchange. At any one time, only one axis has drive control DB31, ... DBX96.5=1.

During exchange, servo enables on all axes involved are automatically withdrawn by the control.

Axes without drive control are not in closed-loop control. Therefore, a brake control must be set up for vertical axes.

Special cases

If a number of transfer requests are made simultaneously, exchange will not take place. The last axis used remains in control of the drive. This is also the case if there are no transfer requests pending.

If there are no transfer requests during machine power-up, drive control is assigned to the first machine axis located with the same logical drive number. The interface signal DB31, ... DBX24.5 (change setpoint output assignment) must be set explicitly in advance to be able to traverse the axis. Logical drive numbers are scanned in ascending order.

```
MD30110 $MA_CTRLOUT_MODULE_NR[0,AX1] = 1
MD30110 $MA_CTRLOUT_MODULE_NR[0,AX2] = 2
MD30110 $MA_CTRLOUT_MODULE_NR[0,AX3] = 3
MD30110 $MA_CTRLOUT_MODULE_NR[0,AX4] = 4 ; Drive control during power-up
MD30110 $MA_CTRLOUT_MODULE_NR[0,AX5] = 4
```
6.3 Interface signals

Axis specific signals

Despite assignment of a drive to several machine axes, the use of NC/PLC interface signals remains unchanged. This requires an explicit coordination of access operations to the NC/PLC interface signals in the PLC user program.

Status signals

The status signals contained in the following bytes are always displayed in the same way for all of the machine axes involved in the changeover:

DB31, ... DBB92 to DBB95

Control signals

The control signals contained in the following bytes are only active in the machine axes, which are currently assigned to the drive:

DB31, ... DBB20 to DBB21

Controller enable

The NC/PLC interface signal:

DB31, ... DBX2.1 (controller enable)

is only effective if the machine axis currently has control of the drive:

DB31, ... DBX96.5 == 1 (status setpoint changeover)

Axes without drive control are subject to functional restrictions and are therefore operated in the follow-up mode. To do this, the control automatically withdraws the controller enable signals for these machine axes.
S9: Setpoint exchange - 840D sl only

6.3 Interface signals

Schematic sequence of a setpoint changeover

Initial state: Machine axis AX1 currently controls the drive.
Objective: Machine axis AX2 should assume control.

AX2: Traversing command is present
DB32.DBX61.0 == 1

AX2: Control via the drive active?
DB32.DBX96.5
== 0 (yes)
== 1 (no)

AX1: Close brake

AX1: Relinquish drive control
DB31.DBX24.5 = 0

AX1, AX2: Tracking active?
DB31.DBX61.3

DB32.DBX61.3
== 0 (no)
== 1 (yes)

Change over mechanical system

AX2: Request drive control
DB32.DBX24.5 = 1

AX2: control accepted
DB32.DBX96.5
== 0 (no)
== 1 (yes)

AX2: set controller enable
DB32.DBX2.1 = 1

AX2: Open brake

Figure 6-4  Schematic setpoint changeover from machine axes AX1 to AX2
6.4 Interrupts

Drive alarms are only displayed for axes with drive control.

6.5 Position control loop

During setpoint exchange, the drive train and therefore the position control loop are isolated. In order to avoid instabilities, exchange only takes place at standstill and once all servo enables have been deleted.

The use of a single drive means that only one of the control loops can be closed at any one time. Axes without drive control are operated with open position controller and following positions.

6.6 Reference points

The use of load-side encoders does not affect the axial reference points of a setpoint exchange.

However, the mechanical reference to the load can be lost following setpoint exchange for a load-side position derived solely from the motor encoder. These types of axis must be referenced again after every setpoint exchange.
6.7 Supplementary conditions

Parameter sets
A setpoint exchange is not performed if one of the following states is present:

- A parameter set changeover in one of the two machine axes has not completed.
- The parameter sets of the two machine axes are not the same.

Note
No message is displayed.

"Parking" operating status
The "parking" operating state can only be exited using the axis with the drive checking function.

Service display drive
The "Drive Service Display" HMI diagnostics screen does not take into account the changes in assignments between machine axes and the drive.

Commissioning via SinuCom NC
The SinuCom NC commissioning tool can be used to commission the setpoint exchange only via the expert list.

Safety Integrated (only 840D sl)
A detailed description of the supplementary conditions for setpoint changeover in conjunction with Safety Integrated is available in:

References:
Manual SINUMERIK Safety Integrated
## 6.8 Data lists

### 6.8.1 Machine data

#### 6.8.1.1 Axis/spindle-specific machine data

| Number | Identifier: $MA_\_ | Description                                           |
|--------|---------------------|-------------------------------------------------------|
| 30130  | CTRLOUT_TYPE        | Output type of setpoint                                |
| 30200  | NUM_ENCS            | Number of encoders                                     |
| 30220  | ENC_MODULE_NR       | Actual-value assignment: Drive number / measurement circuit number |
| 30230  | ENC_INPUT_NR        | Actual-value assignment: Input on drive module/measuring circuit module |
S9: Setpoint exchange - 840D sl only

6.8 Data lists
7.1 Brief description

Tangential control

The tangential control function belongs to the category of NC functions with coupled axes. It is characterized by the following features:

● There are two leading axes which are moved independently by means of normal traversing instructions (leading axes). In addition there is a following axis whose position is determined as a function of the status of these leading axes (position, tangent).

● The leading axes and following axis are only coupled at certain times, i.e. the coupling can be switched on and off by program instructions.

● Tangential control is defined for the basic coordinate system/workpiece coordinate system.

● The leading axes are defined as geometry axes and the following axis as a rotary axis.

● The coupled axes are assigned to the same channel.

● The position of the following axis can be the input value for a transformation.

● Tangential control is only active in AUTOMATIC and MDA modes.

Corners in the path contour

If the contour defined by the leading axes contains a corner the following points must noted with respect to the following rotary axis. You can select one of two different types of response:

● The path velocity is reduced to such an extent that the following axis reaches its target position synchronously with the other axes.

● If TLIFT has been programmed, an intermediate block is inserted at any corner whose angle is greater than the following machine data:

   MD37400 $MA_EPS_TLIFT_TANG_STEP (Tangent angle for corner recognition)

   In this inserted intermediate block, the rotary axis is moved as fast as possible to the position corresponding to the tangent after the corner. The limit values set for this axis are not violated.
Canceling of follow-up grouping

The definition of a follow-up grouping can be canceled in order to track new leading axes with the following axis.

Applications

The tangential control function can be used for example for the following applications:

- Tangential positioning of a rotatable tool for nibbling operations.
- Follow-up control of tool alignment for a bandsaw.
- Positioning a dressing tool on a grinding wheel.
- Positioning of a gear shaping cutter in glass or paper processing applications.
- Tangential feed of a wire for 5-axis welding.
### 7.2 Characteristics of tangential follow-up control

#### Task specification

Follow-up control for the rotary axis must be implemented so that the axis is always positioned at a specified angle on the programmed path of the two leading axes.

![Diagram of tangential control](image)

**Figure 7-1** Tangential control, offset angle of zero degrees to path tangent

In the diagram, X and Y are the leading axes in which the path is programmed; C is the following axis whose position is determined by the control as a function of the leading axis values and of the desired offset angle between tangent and alignment in C.

The tangential control will function only if the leading axes are used as path axes. A leading axis which is programmed as a positioning axis (POS or POSA) does not specify values required for the follow-up control function.

#### Response on follow-up

A difference must be made between the following cases:

- **Without intermediate block (TLIFT)**
  
  The path velocity of the leading axes is reduced to such an extent that the following axis reaches its target position synchronously with the other axes.

- **With intermediate block (TLIFT), rounding off without G641**
  
  The intermediate block generates the required turn of the tangentially following axis. It is interpolated in such a way that the following axis travels at its limit velocity. The intermediate block is not rounded. At the beginning of the intermediate block, the path velocity of the leading axes is zero.
7.2 Characteristics of tangential follow-up control

Special cases

- **G641 rounding** is possible between two blocks, both of which move at least one of the two leading axes of the tangentially following axis.

- **G641 rounding** is possible between two blocks, both of which do not move either of the leading axes of the tangentially following axis.

In both cases, an intermediate block for the tangentially following axis is not created. An intermediate block is not required because in the preprocessing run the rounded contour is detected and the limit values for the following axis are calculated.

- **Hidden corner in area**

  A corner relevant for the tangential follow-up control can be hidden in space. (The projection of the contour on the plane defined by the two leading axes is relevant).

  If there is a hidden corner in space, an intermediate block is inserted before the block (here N6) causing the tangential jump. This intermediate block moves the following axis to the new position. The block transition is not rounded.

| Programming | Comment |
|-------------|---------|
| N1 TANG (C, X, Y, 1) | |
| N2 TLIFT (C) | |
| N3 G1 G641 X0 Y0 F1000 | |
| N4 TANGON (C) | |
| N5 X10 | |
| N6 Y10 | ; The rotary axis is repositioned before working off this block. |
| N7 M3001.97 | |
7.3 Using tangential follow-up control

**Activation**

The following axis can only be aligned if:

- The assignment between the leading and following axes is declared to the system (TANG)
- Follow-up control is activated explicitly (TANGON)
- The response at corners is specified, if required (TLIFT).

**Additional functions**

Further functions are provided in order to:

- Terminate follow-up control of the following axis (TANGOF)
- Deactivate the special behavior at path corners (TANG without subsequent TLIFT)
- Cancel the definition of a follow-up grouping (TANGDEL).

**Effect on transformation**

The position of the rotary axis to which follow-up control is applied can act as the input value for a transformation.

**References:**

Function Manual Extended Functions; Kinematic Transformation (M1)

---

**Note**

The user is recommended to program TLIFT if tangential control is used together with a transformation. TLIFT prevents the follow-up axis from overtraveling and protects against excessive compensating movements.

---

**Explicit programming of the follow-up axis**

If a following axis, which is being made to follow its leading axes, is positioned explicitly, then the position specification is added to the offset angle programmed in the activation instruction TANGON (see Section "Activation of follow-up control (Page 423)"). Motion commands (AC, IC, DC, POS) are permissible.

---

**Reference point approach**

Follow-up control is deactivated while the following axis executes a reference point approach.
Cross-channel block search

The cross-channel block search in the program test mode (SERUPRO "Search-Run by Program test") can be used to simulate the tangential tracking of axes.

Further information about the multi-channel block search function SERUPRO, see:

**References:**
Function Manual, Basic Functions; Mode Group, Channel, Program Mode, Reset Response (K1),
Section: "Program test"

### 7.3.1 Assignment between leading axes and following axis

**Programming**

The programming is carried out using the pre-defined sub-program TANG. The following parameters are transferred to the control:

- Following axis (additional rotary axis)  
  - The appropriate axis identifiers are used to specify the axes.
- Leading axis 1 (geometrical axis)  
- Leading axis 2 (geometrical axis)  
- Coupling factor  
  - The coupling factor is generally 1 (default setting).
  - The coupling factor can be omitted.
- Identifier of the coordinate system  
  - "B" = Basic coordinate system (default setting)
  - "W" = Workpiece coordinate system (not available)

**Example**

TANG(C, X, Y)

**References**

Programming Manual, Job Planning
7.3.2 Activation of follow-up control

Programming

The programming is performed using the pre-defined TANGON subprogram. When the tangential control is activated, the name of the following axis which must be made to follow is transferred to the control. This specification refers to the assignment between master and following axes made beforehand with TANG (see Section "Assignment between leading axes and following axis (Page 422)"). An angle between the tangent and the position of the following axis can be specified optionally when follow-up is activated. This angle is maintained by the control for as long as the following axis is made to follow. The angle is added to the angle stored in the following machine data:

MD37402 $MA_TANG_OFFSET (preselection angle for tangential follow-up)

If the angle is zero both in TANGON and in the machine date, the following axis takes the direction of the tangent.

Figure 7-2 Tangential control, offset angle of 90 degrees to path tangent

Activation is programmed as follows for the above example and an offset angle of 90 degrees:

TANGON(C, 90)

In response to every motion in path axes X and Y, following axis C is rotated to an angle of 90 degrees in the relation to the path tangent.
7.3.3 Switching on corner response

After axis assignment with TANG(), the TLIFT() instruction must be written if the corner response is to be contained in an intermediate block.

TLIFT (C)

The control reads the following machine data for the tangential following axis C:

MD37400 $MA_EPS_TLIFT_TANG_STEP (Tangent angle for corner recognition)

If the tangential angle jump exceeds the angle (absolute value) of the angle set in the machine data, the control recognizes a "corner" and approaches the new position of the follow-up axis via an intermediate block.

System variable $AC_BLOCKTYPE

The system variable $AC_BLOCKTYPE indicates whether the current block is an intermediate block generated by TLIFT. If the value of the system variable is 6, TLIFT inserted the current block as an intermediate block.

7.3.4 Termination of follow-up control

Programming

The programming is carried out using the pre-defined sub-program TANGOF. The name of the following axis to be decoupled from its leading axes for the remainder of the machining operation must be transferred to the control in conjunction with the sub-program name TANGOF.

With reference to the example in Section "Assignment between leading axes and following axis (Page 422)", the termination is as follows:

TANGOF(C)

The follow-up control process initiated with TANGON is terminated.

Termination of follow-up control initiates a preprocessing stop.

RESET/end of part program

An activated tangential control can remain active for further machining operations. For further details, see:

References:

Function Manual, Basic Functions; Coordinate Systems, Axis Types, Axis Configuration (K2)

Section: Actual-value system for workpiece
Section: External zero offset
7.3 Using tangential follow-up control

Note
The assignment between 2 master axes and a slave axis programmed with TANG(...)
is not canceled by TANGOF (see Section "Canceling the definition of a follow-up axis assignment."
(Page 425)).

7.3.5 Switching off intermediate block generation
In order to stop generating the intermediate block at corners during program execution with
active tangential follow-up control, the TANG() block must be repeated without following
TLIFT().

7.3.6 Canceling the definition of a follow-up axis assignment.
A follow-up axis assignment specified by TANG() remains active after TANGOF. This inhibits a
plane change or geometry axis switchover.

The predefined subprogram TANGDEL is used to cancel the definition of a follow-up axis
assignment so that the follow-up axis can be operated dependent on new leading axes when
a new follow-up axis assignment is defined.

TANGDEL(C)

The existing definition in the example of TANG(A, X, Y) is canceled.

Example for plane change

| Program code         | Comment                                      |
|----------------------|----------------------------------------------|
| N10 TANG(A, X, Y, 1) |                                              |
| N20 TANGON(A)        |                                              |
| N30 X10 Y20          |                                              |
| ...                  |                                              |
| N80 TANGOF(A)        | ; Defined coupling from A to X and Y as      |
| N90 TANGDEL(A)       | ; Delete leading axes                        |
| ...                  |                                              |
| N120 TANG(A, X, Z)   | ; A can be coupled to new leading axes       |
| N130 TANGON(A)       |                                              |
| ...                  |                                              |
| N200 M30             |                                              |
Example for geometry axis switchover

If the definition of the follow-up axis assignment is not canceled, an attempt to execute a geometry axis switchover is suppressed and an alarm is output.

| Program code  | Comment                                           |
|---------------|---------------------------------------------------|
| N10 GEOAX(2, Y1) |                                                   |
| N20 TANG(A, X, Y) |                                                   |
| N30 TANGON(A, 90) |                                                   |
| N40 G2 F8000 X0 Y0 I0 J50 |                                                   |
| N50 GEOAX(2, Y2) ; Alarm 14415, geometry axis to be deleted is still leading axis of follow-up axis assignment |

Geometry axis switchover with TANGDEL

The following example shows how TANGDEL is used correctly in association with an axis switchover.

| Program code  | Comment                                           |
|---------------|---------------------------------------------------|
| N10 GEOAX(2, Y1) ; Geometry axis group is determined |
| N20 TANG(A, X, Y) ; Channel axis Y1 is being assigned |
| N30 TANGON(A, 90) ; Follow-up group with Y1 is being activated |
| N40 G2 F8000 X0 Y0 I0 J50 ; Movement block for the leading axes |
| N50 TANGOF(A) ; Activation of follow-up control |
| N60 TANGDEL(A) ; Deletion of the definition of follow-up axis assignment |
| N70 GEOAX(2, Y2) ; Geometry axis switchover permitted |
| N80 TANG(A, X, Y) ; Redef. Follow-up axis group |
| N90 TANGON(A, 90) ; Follow-up group with Y2 is being activated |
| ...           |                                                   |
7.4 Limit angle

Description of problem

When the axis moves backwards and forwards along the path, the tangent turns abruptly through 180 degrees at the path reversal point. This response is not generally desirable for this type of machining operation (e.g. grinding of a contour). It is far better for the reverse motion to be executed at the same offset angle (negative) as the forward motion.

Programming

A minimum and a maximum value for the position of the axis made to follow ("C" in example) referred to the base coordinate system are transferred to the control with \texttt{G25} and \texttt{G26}. These working area limitations are activated with \texttt{WALIMON} and deactivated again with \texttt{WALIMOF}. The working area limitation must be active at the instant of path reversal.

References:
Programming Manual Fundamentals
Activation

If the current offset angle is outside the active working area limit for the following axis, an attempt is made to return to within the permissible working area by means of the negative offset angle. This response corresponds to that shown in the lower diagram of the above Fig.

7.5 Supplementary conditions

Block search with active coupling

Note

For an active coupling, it is recommended to only use block search type 5, "Block search via program test" (SERUPRO) for a block search.
7.6 Examples

Positioning of workpiece

Figure 7-4  Tangential positioning of a workpiece on a bandsaw

Positioning of tool

Figure 7-5  Positioning of a dressing tool on a grinding wheel
Example Corner in area

| Programming |
|-------------|
| TANG(A,X,Y,1.0,“B”) |
| TLIFT(A) |
| G1 G641 X0 Y0 Z0 A0 |
| TANGON(A,0) |
| N4 X10 |
| N5 Z10 |
| N6 Y10 |
| M30 |

Here, a corner is hidden in the area between N4 and N6. N6 causes a tangent jump. That is why there is no rounding between N5 and N6 and an intermediate block is inserted.

In the case of a hidden corner in area, an intermediate block is inserted before the block that has caused the tangent jump. The intermediate block moves the following axis to the new tangent position.
7.7 Data lists

7.7.1 Machine data

7.7.1.1 Axis/spindlespecific machine data

| Number | Identifier: $MA_ | Description                          |
|--------|------------------|--------------------------------------|
| 37400  | EPS_TLIFT_TANG_STEP | Tangential angle for corner recognition |
| 37402  | TANG_OFFSET      | Default angle for tangential follow-up control |

7.7.2 System variables

| Identifier         | Description                                      |
|--------------------|--------------------------------------------------|
| $AC_BLOCKTYPE      | Current block is an intermediate block generated by TLIFT |
T3: Tangential control - 840D sl only

7.7 Data lists
TE01: Installation and activation of loadable compile cycles

Contents

The following sections describe how technology and special functions in the form of individual loadable compile cycle files (*.ELF) are installed and activated in the control.

Technology functions available from Siemens in the form of compile cycles

- **1D/3D clearance control in position control cycle**
  Compile cycle: CCCLC.ELF
  References: Function Manual Special Functions; Clearance Control (TE1)

- **Handling transformation package**
  Compile cycle: CCRCTRA.ELF
  References: Function Manual Special Functions; Handling Transformation Package (TE4)

- **Setpoint exchange**
  Compile cycle: CCSETP.ELF
  References: Function Manual Special Functions; Setpoint Exchange (S9)

- **Axial coupling in the machine coordinate system (MCS coupling)**
  Compile cycle: CCMCSC.ELF
  References: Function Manual Special Functions; MCS Coupling (TE6)

- **Continue machining at the contour (retrace support)**
  Compile cycle: CCRESU
  References: Function Manual Special Functions; Retrace Support (TE7)

- **Fast laser switching signal**
  Compile cycle: CCHSLC.ELF
  References: Function Manual Special Functions; Cycle-Independent Path-Synchronous Signal Output (TE8)

- **Axis pair collision protection**
  Compile cycle: CCPROT.ELF
  References: Function Manual Special Functions; Axis Pair Collision Protection (TE9)
Compile cycle files (*.ELF) and CNC software versions

Compile cycle files (*.ELF) that have been generated for CNC software versions up to SW 4.4 cannot run on systems with CNC software versions as of SW 4.5.

Technology and special functions provided by Siemens in the form of compile cycles

New compile cycle files (*.ELF) for Siemens technology and special functions for CNC software versions as of SW 4.5 can be obtained from your regional Siemens sales office.

Technology and special functions provided by third parties in the form of compile cycles

Compile cycle files (*.ELF) that have been generated for CNC software versions up to SW 4.4 must be adapted and recompiled with the generation environment for CNC software versions as of SW 4.5. Details on the adaptation of the user-specific technology and special functions can be found in the the Open Architecture (OA) documentation.

Compile cycles

Compile cycles are functional expansions of the NCK system software that can be created by the machine manufacturer and/or by Siemens and then imported in the control later.

As part of the open NCK system architecture, compile cycles have comprehensive access to data and functions of the NCK system level via defined software interfaces. In this way, compile cycles significantly extend the functionality of the NCK.

Including a compile cycle in the NCK system software is performed by loading the compile cycle into the file system of the NCK. The compile cycle can be loaded at any time.

Siemens compile cycles

Siemens compile cycles are the technological functions supplied by Siemens.

When you order one of these technological functions, you get only the corresponding software license number. To obtain the compile cycle in the form of a loadable file (*.ELF" extension for "executable and linking format") please contact your regional Siemens sales partner.

Note

Compile cycles created by Siemens are options that require explicit activation and licensing.

References:
Ordering information in Catalog NC 60/61
8.1 Loading compile cycles

8.1.1 Loading a compile cycle with SINUMERIK Operate

Requirement

- The compile cycle to be transferred to the control must be saved on a storage medium which can be directly connected to the control, such as a USB-FlashDrive.
- Access authorization of protection level 1 "Machine manufacturer" (default password "SUNRISE") must be available.
- If there are no cursor keys on the operator panel, a mouse or keyboard must be connected in order to perform the operator actions.

Execution

Perform the following steps to load a compile cycle from a USB-FlashDrive, for example:

1. Insert the USB-FlashDrive in the USB interface on the front of the operator panel.
2. At the user interface, change into the "System data" area:
   Horizontal softkey bar: "Operating area switchover" > "Commissioning" > "System data"
3. Open the USB-FlashDrive: Directory "USB" (keyboard: "Enter" key).
4. Select the compile cycle and copy it:
   Vertical softkey bar: "Copy"
5. Within the "System data" area, switch into the directory "NC data" > "Compile cycles" and insert the compile cycle there:
   Vertical softkey bar: "Insert"
6. Initiate an NCK reset to activate the compile cycle:
   Horizontal softkey bar: "Operating area switchover" > "Commissioning" > "Machine data"
   Vertical softkey bar: "Reset (po)"
8.1.2  Loading a compile cycle with HMI Advanced

Precondition

To transfer a compile cycle to the control, the following requirements must be met:

A storage medium (e.g. USB FlashDrive), which stores the compile cycle, is connected to the PCU.

Execution

Perform the following operation to load a compile cycle from a USB FlashDrive in the NCK:

1. Insert the USB FlashDrive into the PCU 50/70.
2. Open the USB FlashDrive as the local drive:
   Operating area switchover > Services > Data admin > Local USB
   If the "Local USB" is displayed as disabled, it implies the USB FlashDrive was not detected. Wait until HMI Advanced has detected the USB FlashDrive.
3. Select the compile cycle and copy it:
   Vertical "Copy" softkey
4. Go to the OEM directory of the NC card:
   etc Key "">> NC Card > Directory: "cc" or "Loadable Compile Cycles"
5. Add the compile cycle to the NC card's OEM directory:
   Vertical "Add" softkey
6. Initiate an NCK reset.

8.1.3  Loading a compile cycle from an external computer with WinSCP3

Precondition

To transfer a compile cycle to the control, the following requirements must be met:

- The external computer (programming device / PC) which the compile cycle is loaded onto is linked to the PCU via a network (TCP / IP).
- The program "WinSCP3" is installed on the external computer.
- The host name or the IP address of the PCU is known.
- The user name and the password to log onto the PCU are known.
**Execution**

Perform the following operation to load a compile cycle from an external computer into the NCK:

1. Start the "WinSCP3" program on the external computer (programming device / PC)
2. Establish a connection to the PCU by selecting an appropriate profile or by entering the host name or IP address, the user name and the password.
3. Copy the compile cycle from the external computer into the following directory on the PCU:
   /card/oem/sinumerik/data/cc
   The file name of the compile cycle should have any capital letters in the directory of the PCU.
4. Initiate an NCK reset.
8.2 Interface version compatibility

The compile cycle and the NCK system software communicate via a SINUMERIK-specific interface. The interface version used by the loaded compile cycle must be compatible with the interface version of the NCK system software.

Interface versions

The relevant interface versions are displayed under:

- Interface version of the NCK system software
  HMI Advanced:
  Diagnosis > Service Display > Version > NCU Version
  Display (excerpt)
  -------------------------------------------
  CC Interface Version:
  @NCKOPI . . . @Interfaces=<1st digit>.<2nd digit> . . . .
  Loaded Compile Cycles:
  . . .
  -------------------------------------------

- Interface version of a compile cycle that has not yet been loaded
  HMI Advanced (excerpt):
  Services > <Medium> > Softkey: "Properties"
  Display:
  -------------------------------------------
  Contents: Loadable compile cycle
  Interface: . . . @Interfaces=<1st digit>.<2nd digit> . . . .
  -------------------------------------------

- Interface version of a loaded compile cycle
  HMI Advanced:
  Diagnosis > Service Display > Version > NCU Version
  Display (excerpt)
  -------------------------------------------
CC Interface Version:
@NCKOPI....

Loaded Compile Cycles:
<Identifier> <Version> <Generation Data>

CC start address
_N_<Identifier><Version>IF<1st digit><2nd digit>_ELF....

Example:
_N_CLC407IF003001_ELF corresponds to interface version: 3.1

Dependencies

The following dependencies exist between the interface versions of a compile cycle and the NCK system software:

- 1. Position of the interface version number
   The 1st digit of the interface version number of a compile cycle and the NCK system software must be the same.

- 2. Position of the interface version number
   The 2nd digit of the interface version number of a compile cycle must be less than or equal to the 2nd digit of the NCK system software.

CAUTION
If alarm 7200 is displayed after start-up, this means no compile cycle has been loaded!
8.3 Software version of a compile cycle

The SW version of a compile cycle is displayed under:

HMI Advanced:

Diagnosis > Service Display > Version > NCU Version

Display (excerpt)

-------------------------------------------

CC Interface Version:
@NCKOPI . . .

Loaded Compile Cycles:
</Identifier> <Version> <Generation Data>

CC start address

_N_<Identifier><Version>IF<1st digit><2nd digit>_ELF . .

Code=<Address> Data=<Address> . .

-------------------------------------------

Example:

_N_CLC407IF003001_ELF corresponds to software version 4.7

Note

The display of code and data range start addresses of a compile cycle are provided for diagnostics purposes only and have no significance in normal operation.
8.4 Activating the technological functions in the NCK

Requirement

The corresponding option must be enabled before activating a technology function as described below.

If the option data has not been set, the following alarm appears every time the NCK boots and the technology function will not be activated:

Alarm 7202 "XXX_ELF_option_bit_missing: <Bit number>"

Function-specific machine data

Each technology function loaded by compile cycle creates a function-specific global NCK machine data within the range of numbers from 60900 to 60999 in accordance with the following pattern:

$MN_CC_ACTIVE_IN_CHAN_\langle identifier\rangle[n], \text{ with } n = 0, 1

Example:

$MN_CC_ACTIVE_IN_CHAN_MCSC[0]

$MN_CC_ACTIVE_IN_CHAN_MCSC[1]

Activation for 1st NC channel

The technology functions are activated in the first NC channel via:

$MN_CC_ACTIVE_IN_CHAN_\langle identifier\rangle[0], \text{ bit}0 = 1

The meaning of machine data bits Bit1 - Bit31 are described in the relevant function description (TE1 - REn).

After the NCK is booted up next, the activated technology functions are integrated into the system software.

⚠️ CAUTION

SINUMERIK 840D sl

The following alarm is displayed when a bit is set for the first time in the function-specific NCK machine data:

$MN_CC_ACTIVE_IN_CHAN_\langle identifier\rangle[0], 

Alarm 4400 "MD modification causes reorganization of the buffered memory (data loss)"

And you are warned that all user data (part programs, tool data, etc.) will be deleted on the next run-up. If necessary, an archive should be created after setting the date and before triggering NCK RESET.
8.5 Function-specific startup

Further function-specific installation routines are described in the corresponding function description (TE1 - TEn).
8.6 Creating alarm texts

The alarm texts of the technology functions are stored in the system. If a new alarm is required, then the alarm texts can be supplemented accordingly. The general process depends on the existing interface.

8.6.1 Creating alarm texts with SINUMERIK Operate

The following alarms should be added to the alarm texts of the technology functions:

075999 0 0 "Channel %1 Sentence %2 Call parameter is invalid"

Proceed as follows

1. Please copy the "oem_alarms_deu.ts" file from the "/siemens/sinumerik/hmi/template/Ing" directory to the "/oem/sinumerik/hmi/Ing" directory.
2. Rename the file ("xxx_deu.ts").
3. Open the file in the editor and add the new alarm number and the new alarm text in German:

   <message>
   <source>075999/NCK</source>
   <translation>
   Channel %1 block %2 call parameter is invalid
   </translation>
   </message>

   Note
   Each alarm starts with the <message> tag and ends with the </message> tag. The <source> tag contains the alarm number and the source URL. The <translation> tag contains the alarm text.

4. To create an alarm text file in a foreign language, copy the just changed file and modify the language code in the filename (e.g., "xxx_eng.ts for English").
5. Open the foreign language alarm text file in the editor and record the translated alarm text in the <translation> tag.
6. Please copy the "oem_slaesvcadapconf.xml" file from the "/siemens/sinumerik/hmi/base" directory to the "/oem/sinumerik/hmi/fg" directory.
7. Rename the file to "slaesvcadapconf.xml".
8.6 Creating alarm texts

8. Open the "slaesvcadapconf.xml" file in the editor and record the new base name (filename of the newly created alarm text file without language code and postfix), e.g.:

```xml
<BaseNames>
  <BaseName_02 type="QString" value="xxx"/>
</BaseNames>
```

9. Restart SINUMERIK Operate.

More information about creating alarm texts with SINUMERIK Operate can be found in:

References:
SINUMERIK 840D sl Base Software and Operating Software Commissioning Manual;
SINUMERIK Operate (IM9) Commissioning Manual, Section: Configuring machine data/alarms

8.6.2 Creating alarm texts with HMI Advanced

The following alarms should be added to the alarm texts of the technology functions:

075999 0 0 "Channel %1 Sentence %2 Call parameter is invalid"

Proceed as follows

1. Copy the "mbdde.ini" file from the "F:\mmc2" directory to the "F:\oem" directory.
2. Add the following two lines in the "mbdde.ini" file:

```ini
[TextFiles]
UserCZYK=F:\oem\alc_
```

3. Create the language-dependent "alc_XX.com" text files in the "F:\oem" directory, e.g.:

"alc_GR.com" for German
"alc_UK.com" for English

4. Insert the new alarm text in the language-dependent text files, e.g.:

075999 0 0 "Channel %1 Sentence %2 Call parameter is invalid"

in the text file in German.

5. Restart HMI Advanced.

For more information about creating alarm texts with HMI Advanced, please refer to:

References:
SINUMERIK 840Di sl/840D sl/840D Base Software and HMI Advanced Commissioning Manual; HMI-Advanced (IM4) Commissioning Manual, Section: Configuring user alarms

Note

HMI reinstallation

Retain the added alarm texts in the text files of the F:\oem even after a reinstallation of HMI.
8.7 Upgrading a compile cycle

To upgrade a compile cycle installed in the control, under no circumstances is it sufficient to just exchange the corresponding ELF file. If only the ELF file is replaced, then this can result in undefined behavior of the NCK software due to inconsistent data of the memory and data management.

Execution

Perform the following actions to upgrade a compile cycle:

1. Create an NC archive without compile cycles:
   Operating area: "Commissioning" > "ETC" key > "Commissioning archive" > "Creating a commissioning archive" > "OK"
   - Selection: "NC data" selected
   - Selection: "with Compile cycles" not selected!

2. Delete the old compile cycle:
   Operating area: "Commissioning" > "System data" in the directory: NC data > "Compile cycles" > file: <Identifier>.ELF

3. Download the new compile cycle (see Section "Loading compile cycles (Page 435)"

4. Initiate a power on reset with an NCK general reset (NCK commissioning switch: position "1"). See:
   References: Commissioning Manual IBN CNC: NCK, PLC, Drive; Section: General tips > Separate NCK and PLC general reset > NCK general reset

5. Download the NC archive created under Point 1 into the NCU again.
   Operating area: "Commissioning" > "ETC" key > "Commissioning archive" > "Read-in commissioning archive" > "OK"

Note

Version check

To check the version of the newly loaded compile cycle, see Section "Software version of a compile cycle (Page 440)"

Several loaded compile cycles

If several compile cycles are loaded in the control, when upgrading a compile cycle according to the process described above, the other compile cycles remain unchanged.

Newly created data

If the new compile cycle creates new data, which were still not available in the previous version, then after subsequently booting the control, these are preassigned default values.
8.8 Deleting a compile cycle

If a loaded compile cycle is to be completely deleted in the control, it is not enough to only delete the corresponding ELF file. With this procedure, the following data is kept in the retentive memory of the control:

- Activation data: $MN_CC_ACTIVE_IN_CHAN_<identifier>[n]
- Function-specific machine data

Execution

Perform the following actions to delete a compile cycle:

1. Create an NC archive without compile cycles:
   Operating area: "Commissioning" > "ETC" key > "Commissioning archive" > "Creating a commissioning archive" > "OK"
   - Selection: "NC data" selected
   - Selection: "with Compile cycles" not selected!

2. Delete the compile cycle:
   Operating area: "Commissioning" > "System data" in the directory: NC data > "Compile cycles" > File: Select "<identifier>.ELF" > "Delete"

3. Initiate a power on reset with an NCK general reset (NCK commissioning switch: position "1"). See:
   References:
   Commissioning Manual IBN CNC: NCK, PLC, drive; Chapter: General tips > Separate NCK and PLC general reset > NCK general reset

4. Download the NC archive created under Point 1 into the NCU again.
   Operating area: "Commissioning" > "ETC" key > "Commissioning archive" > "Read-in commissioning archive" > "OK"

Note

Several loaded compile cycles

If several compile cycles are loaded in the control, when deleting a compile cycle according to the process described above, the other compile cycles remain unchanged.

Efficient memory usage

To use the memory resources in the NCK as efficiently as possible, you should only load the compile cycles (ELF files) that are actually required into the machine.
8.9   Data lists

8.9.1    Machine data

8.9.1.1   NC-specific machine data

| Number  | Identifier: $MN_\ldots$ | Description                                      |
|----------|--------------------------|--------------------------------------------------|
| 60900 + i| CC_ACTIV_IN_CHAN_XXXX[n] | n = 0: Activating the technology function in NC channels |
|          | with:                    |                                                  |
|          | i = 0, 1,               | n = 1: Additional functions within the technology function |
|          | XXXX = function code    |                                                  |
|          | n = 0 or 1              |                                                  |
9.1 Brief description

9.1.1 Function

If part programs, which use compile cycles, are simulated on the SINUMERIK user interface (e.g. HMI Advanced) simulation is aborted and corresponding error messages are issued. The reason is that compile cycle support has not yet been implemented on the HMI.

The measures described below show how to set up the simulation runtime environment to enable the simulation of part programs, which use compile cycles, without error messages.
9.2 OEM transformations

When using OEM transformations, the simulation runtime environment has to be set.

Proceed as follows

1. Create a new directory: "<installation path>/OEM" in addition to the standard directory: "<installation path>/MMC2" in the directory structure of the HMI application on the computer on which the HMI application (e.g. HMI Advanced) is installed.

2. In the "OEM" directory, create the file "DPSIM.TEA" with the following contents:
   
   $MN_NC_USER_CODE_CONF_NAME_TAB[196]="TRAORI"
   $MN_NC_USER_CODE_CONF_NAME_TAB[197]="_TRAORI"
   $MN_NC_USER_CODE_CONF_NAME_TAB[198]="TRACON"
   $MN_NC_USER_CODE_CONF_NAME_TAB[199]="_TRACON"
   CHANDATA(1)
   $MC_AXCONF_GEOAX_ASSIGN_TAB[0]=1
   $MC_AXCONF_GEOAX_ASSIGN_TAB[1]=2
   $MC_AXCONF_GEOAX_ASSIGN_TAB[2]=3
   $MC_TRAFO_RESET_VALUE=0
   ; Make sure that transformation types 4096 - 4101 are deleted
   $MC_TRAFO_TYPE_1=0
   $MC_TRAFO_TYPE_2=0
   $MC_TRAFO_TYPE_3=0
   ; Delete transformation chains with OEM transformations
   $MC_TRACON_CHAIN_1[0]=0
   $MC_TRACON_CHAIN_1[1]=0
   ; NOTICE! No spaces after M30
   M30

3. In the "OEM" directory, create the file "DPSIM.INI" with the following contents:
   
   [PRELOAD]
   CYCLES=1
   CYCLEINTERFACE=0

4. Close the HMI application.

5. Launch the HMI application.
6. In the directory for the manufacturer cycles, create the file "TRAORI.SPF" with the following contents:

   PROC TRAORI(INT II)
   RET

7. In the directory for the manufacturer cycles, create the file "TRACON.SPF" with the following contents:

   PROC TRACON(INT II)
   RET

   **Note**
   The "TRAORI.SPF" and "TRACON.SPF" manufacturer cycles created under 6. and 7. must not be loaded onto the NC.

8. Start the simulation.

9. Run a data comparison for the cycles after the simulation has started up:
   
   - HMI Advanced: **Data comparison > Compare cycles**

   **Note**
   At least the password for protection level 3 "End user: Service" is needed for the data comparison.
10.1 Brief description

10.1.1 General

Function description
The "clearance control" technological function is used to maintain a one-dimensional (1D) or three-dimensional (3D) clearance required for technological reasons during a defined machining process. The clearance to be maintained may be e.g. the distance of a tool from the workpiece surface to be machined.

Function code
The code for the "clearance control" technological function for function-specific identifiers of program commands, machine data, etc. is:

CLC = Clearance Control

Function restriction
The "clearance control" technological function is only available in the first NC channel, even on controls with more than one NC channel. This means:

- Only channel axes in the first NC channel may be used as clearance-controlled axes.
- CLC part program commands for the clearance control may only be used in part programs processed in the first NC channel.

Note
The "clearance control" technological function is only available in the first NC channel!

Availability
The "clearance control" technological function is available in SINUMERIK 840D sl.
Compile cycle

The “clearance control” technological function is a compile cycle.

For a description of the system-specific availability and use of compile cycles (see Section “TE01: Installation and activation of loadable compile cycles (Page 433)”).

10.1.2 Function description

Laser cutting technology is used as an example for the detailed description of the "clearance control" functionality.

Laser cutting

During laser cutting, a divergent parallel laser beam is directed across a fiber-optic cable or via a mirror to a light-collecting lens mounted on the laser machining head. The collecting lens focuses the laser beam at its focal point. Typical focal lengths are from 5 to 20 cm.

The position of the focal point in relation to the workpiece is an extremely critical process parameter in laser cutting operations and must be kept constant within a tolerance of ≤100 µm.

The distance between the focal point and the workpiece, which is also a key process variable, is usually measured by means of a high-speed capacitive clearance sensor. The analog output voltage of the clearance sensor is approximately proportional to the distance between the sensor and the workpiece surface.

The output voltage of the clearance sensor is transmitted as a digital input value via an analog I/O module to the control where, in the event of deviations from the setpoint clearance, it generates an additional velocity setpoint for the machining head motion axes.
10.1 Brief description

System overview (840D sl)

An overview of the system components required for clearance control in conjunction with SINUMERIK 840D sl is provided in the following diagram.

![Diagram of system components for clearance control with SINUMERIK 840D sl](image)

1D/3D machining

Clearance control can be used for 1D and 3D machining with up to five interpolatory axes.

- **1D machining**
  
  For the 1D machining, only one axis is affected by the clearance control. For example, axis Z, as shown in the machine configuration described in the system overview (see previous figure). Clearance control acts only in the direction of the Z axis.

- **3D machining**
  
  3 linear axes are used to position the tool. One or two rotary axes are used for the orientation of the tool vector (5-axis machining). Up to 3 linear axes are controlled by the clearance control. The direction of the compensation movement can be defined either in the direction:
  
  - of the tool orientation vector (normal case)
  - of the programmable compensation vector
10.2 Clearance control

10.2.1 Control dynamics

Closed-loop control gain $K_v$

The dynamic response of the closed control loop (sensor - open-loop control - axis) is determined by the maximum closed-loop control gain $K_v$.

The closed-loop control gain $K_v$ is defined as:

$$K_v = \frac{\text{Velocity [m/min]}}{\text{following error [mm]}} \quad \text{in [} \frac{\text{m/min}}{\text{mm}}\text{]}$$

Clearance control characteristics

Clearance control is based on the two characteristics shown in the following diagram:

- Clearance sensor characteristic (sensor property)
- Clearance control characteristic (can be parameterized via machine data)

Figure 10-2 Correlation between characteristics: Clearance sensor and clearance control
The clearance sensor measures the actual distance from the workpiece surface and returns as its output variable a voltage in [V], which is almost directly proportional to the distance.

The clearance control function uses the parameterized voltage/velocity characteristic from the voltage provided by the clearance sensor to calculate a compensatory velocity for the clearance-controlled axes that is appropriate for the clearance.

From the point of view of the control, the unit for the closed-loop control gain is \([(\text{mm/min})/\text{V}]\). In the same way as the setpoint clearance in standardized in [mm], values can only be standardized in \([(\text{mm/min})/\text{mm}]\) by using the sensor electronics.

Max. closed-loop control gain

The maximum achievable closed-loop control gain is determined by the following delay and reaction times of the overall system:

1. Reaction time of sensor
2. Delay time of A/D converter
3. Signal processing delay times/deadtimes
4. Reaction time of position controller
5. Reaction times of speed and current controllers
6. Time constants of motor and mechanical components

In practice, only items 3 and 4 are relevant.

The influencing variables together produce an effective time constant. A closed-loop control gain set too high based on this time constant will induce natural oscillations in the range of several hertz in the axis/axes to be controlled.

The objective when starting up the clearance control is to minimize important time constants so that the closed-loop control gain required by the process can be set without inducing natural oscillation of this type.

Dead time

In order to maximize the dynamics of the control response, clearance control takes place on the highest priority position controller level of the NCK.

SINUMERIK 840D sl with I/O modules and drives connected via PROFIBUS DP produces a deadtime \(T_{\text{dead}}\) of:

\[
T_{\text{dead}} = 2 \times \text{position controller cycle} + 2 \times \text{speed controller cycle} + \text{input lead time } T_i
\]
10.2.2 Velocity feedforward control

Eliminating the delay time

The closed-loop control gain, $K_v$, set for the position controller corresponds to a delay time $\Delta t$. The display time, $\Delta t$, is the time which elapses until the actual position of the axis to be controlled reaches the set position at a prescribed velocity of $v$.

\[
\text{With } \Delta t = \frac{1}{K_v}
\]

and a closed-loop control gain $K_v$ in seconds:

\[
K_v \text{ in } \frac{\text{m/min}}{\text{mm}} = \left[ \frac{1000 \text{ mm/s}}{60 \text{ s}} \right] = 16.67 \left[ \frac{1}{\text{s}} \right]
\]

for an assumed closed-loop control gain $K_v = 4$, the corresponding delay time $\Delta t$ is:

\[
\Delta t = \frac{1}{4} \cdot 16.67 \left[ \frac{1}{\text{s}} \right] = 14.999 \text{ ms}
\]

Optimizing the control response

If the control response of the axis is too rigid due to the velocity feedforward control, the control response can be optimized with the following axis-specific NC machine data:

- MD32410 $MA\_AX\_JERK\_TIME$ (time constant for the axial jerk filter)
- MD32610 $MA\_VELO\_FFW\_WEIGHT$ (feedforward control factor for speed)

The speed filters of the SINAMICS S120 drive provide additional damping capabilities:

- Parameter 1414 and following: (time constant for speed setpoint filter 1, 2)

⚠️ CAUTION

Every damping measure implemented contributes to increasing the overall time constant of the control loop!

You will find a complete description of the velocity feedforward control in:

References:

Function Manual, Extended Functions; Compensation (K3),
Section: Following error compensation (feedforward control)
### 10.2.3 Control loop structure

The figures below provide an overview of how the clearance control function is embedded in the control loop structure of the NC position controller and the internal structure of the function.

![Control structure, position controller with clearance control (principle)](image)

**Figure 10-3** Control structure, position controller with clearance control (principle)

![Control structure, clearance control (principle)](image)

**Figure 10-4** Control structure, clearance control (principle)
10.2.4 Compensation vector

**Standard compensation vector**

The compensation vector of the clearance control and the tool orientation vector are normally identical. Consequently, the compensation movement of the clearance control is normally always in the direction of the tool orientation.

![Diagram](image)

*Figure 10-5  Clearance control with standard compensation vector*

**Note**

In all the figures in this section, the traversing movement of the machining head needed in order to machine the workpiece is in the direction of the Y coordinate, i.e. perpendicular to the drawing plane.

As long as the tool orientation, and hence the compensation vector, is perpendicular to the workpiece surface, no disadvantage for the machining process results from the compensation movements of the clearance control.

If a tool setting angle is needed for technological reasons, with the result that the tool orientation is no longer perpendicular to the workpiece surface, the machining point on the workpiece surface is shifted during compensation movements of the clearance control along the standard compensation vector.
The reason for the shift in the machining point is the X component (KX) of the compensation vector parallel to the workpiece surface. The TCP of the tool, and thus the machining point B, is shifted by this amount.

**Programmable compensation vector**

When using the programmable compensation vector, the compensation movements of the clearance control are in the direction of the programmed vector, and not in the direction of the tool orientation.

The X component specified above (KX) is omitted because the programmable compensation vector is defined perpendicular to the workpiece surface. This does not cause the machining point (B) to be shifted as a result of the compensation movement of the clearance control.
**Changes in orientation**

Based on the above observations, a different behavior also results when the orientation of the machining head is changed while the clearance control is active.

In the following diagram the normal case is shown on the left (compensation vector == tool orientation vector); and the case with the programmed compensation vector is shown on the right.

![Diagram of Changes in Orientation](image)

**Figure 10-8 Change in orientation of the machining head**

The meaning of the individual positions of the machining head is as follows:

1. Programmed position of the machining head
2. Actual position of the machining head with clearance control active before the orientation change
3. Programmed position of the machining head after the change in orientation
4. Actual position of the machining head with clearance control active after the orientation change

The machining head movement visible on the machine when the change in orientation takes place is direct from position 2 to position 4.
10.3 Technological features of clearance control

Clearance control is characterized by the following technological features:

- **Dynamic Response**
  The overlaid sensor motion uses the current residual dynamic response that is still in reserve after the programmed axis motion (velocity and acceleration). The proportion of residual acceleration that must be used can be set as a percentage in a machine data.

- **Sensor characteristic**
  The gain characteristic of a sensor can be defined with up to 10 interpolation points.

- **Sensors**
  Two sensors with different gain characteristics (e.g. a mechanical and a capacitive sensor) can be used simultaneously. The active sensor characteristic can be switched over block-synchronously by means of a language command in the part program.

- **Closed-loop control gain of clearance control**
  The closed-loop control gain configured in the NC machine data for clearance control can be changed block-synchronously by means of a language command in the part program.

- **Motion limitation**
  The lower and upper limits configured in the NC machine data for the axis movements induced by clearance control can be changed block-synchronously by means of a language command in the part program.

  An alarm appears when a limit is reached. The alarm response (stop all traversing movements or display only) can be configured. The current position offset can be frozen by means of a PLC signal.

- **Response on deactivation**
  The deactivation response of the clearance control function can be programmed either for synchronization with the current axis positions (no compensating movement) or for compensating axis movements to the last programmed axis positions (axis positions without clearance control).

- **Programmable clearance setpoint**
  An additional voltage value can be programmed in order to alter the setpoint distance set in the sensor electronics on a block-related basis.
**Control options via the PLC interface**

The following signals are available at the PLC interface:

- **Status signals:**
  - Closed-loop control active
  - Overlaying movement at standstill
  - Lower limit reached
  - Upper limit reached.

- **Control signals:**
  - Path override for sensor movement active

**Status data of clearance control**

Both the current values and the min/max values of the sensor signal and of the position offset are available as GUD and/or OPI variables.

**Sensor signal**

The sensor signal can be smoothed via a PT1 filter with adjustable time constant.
10.4 Sensor collision monitoring

Sensor signal

If the clearance sensor used has an additional "sensor collision" signal for detecting a collision between the sensor and the workpiece being machined, this signal can be made available to the clearance control function via a digital NCK peripheral input.

In response to this signal, the clearance control function applies a retraction motion in all clearance-controlled axes. The retraction motion is executed independently of the current value of the velocity override with maximum traversing velocity in a positive control direction until the currently valid upper limit of the control range is reached. Path motion is stopped simultaneously.

Once all traversing movements have come to a standstill, part program processing can be resumed with NC START.

Parameterization

The digital peripheral input, which the "sensor collision" signal is wired to, is assigned to the clearance control function via the following machine data:

MD62504 $MC_CLCSENSOR_TOUCHED_INPUT (digital peripheral input for "sensor collision" signal)

The digital peripheral input is specified by entering the input number in the same way as $A_IN/$A_OUT digital I/O peripheral system variables are specified ($A_IN[input number]$).

If a negative input number is entered, the "sensor collision" signal will be processed with internal inversion by the clearance control function (fail safe method).
10.5 Startup

Compile cycle

Before commissioning the technological function, ensure that the corresponding compile cycle has been loaded and activated (see Section "TE01: Installation and activation of loadable compile cycles (Page 433)").

10.5.1 Activating the technological function

The technological function is activated via the machine data:

MD60940 $MN_CC_ACTIVE_IN_CHAN_CLC[0], bit n = 1

n = channel number - 1; bit0 = channel 1, bit1 = channel 2, etc.

Note

The technological function can be activated for several channels simultaneously.

10.5.2 Configuring the memory

The technological function requires additional data in the NCK-internal block memory. The values must be increased for the following memory configuring channel-specific machine data:

- MD28090 $MC_MM_NUM_CC_BLOCK_ELEMENTS += 4 (number of block elements for compile cycles)
- MD28100 $MN_MM_NUM_CC_BLOCK_USER_MEM += 20 (size of block memory for compile cycles (DRAM) in KB)

10.5.3 Parameter settings for input signals (840D sl)

The following input signals must be parameterized in the machine data:

- Clearance sensor input voltage
  - 1 analog input
- "Sensor collision" input signal (optional)
  - 1 digital input
10.5 Startup

Analog input

The following machine data must be parameterized for the analog input:

- MD10300 $MN_FASTIO_ANA_NUM_INPUTS (number of active analog NCK inputs)
- MD10362 $MN_HW_ASSIGN_ANA_FASTIN (per analog module) (hardware assignment for the fast analog NCK inputs)

Specifying the physical address activates the analog input module.

Digital input

The following machine data must be parameterized for the digital input:

- MD10350 $MN_FASTIO_DIG_NUM_INPUTS (number of active digital NCK input bytes)
- MD10366 $MN_HW_ASSIGN_DIG_FASTIN (per digital module) (hardware assignment for the external digital NCK inputs)

Specifying the physical address activates the digital input module.

A complete description of the analog and digital inputs appears in:

References:

Function Manual, Extended Functions, Digital and Analog NCK I/Os (A4)

10.5.4 Parameters of the programmable compensation vector

Reference coordinate system

The programmable compensation vector specifies the direction in which the compensation movement of the clearance control takes place. The compensation vector always refers to the basic coordinate system (machine coordinate system).

The start coordinates \([X_a, Y_a, Z_a]\) of the compensation vector coincide with the origin of the basic coordinate system and are thus always \([0, 0, 0]\).

The end coordinates \([X_e, Y_e, Z_e]\) of the compensation vector are determined by the actual positions of 3 channel axes, known as the direction axes.
**Direction axes**

The direction axes must meet the following conditions:

1. The direction axes must be channel axes of the channel in which the clearance control is activated.

2. The direction axes must be linear axes.

   **Note:**

   Since the direction axes are only used to interpolate the direction components, they do not need mechanical axes and can therefore be configured as simulation axes.

3. [mm] or [inch] must be selected as the unit of measurement for the direction axes.

4. The direction axes may not participate in an axis coupling, e.g. transformation, electronic gear, etc.

5. To ensure that the dynamic response of the path is not limited by the dynamic response of the direction axes, the following machine data for the direction axes must be set equal to or more than the corresponding values of the geometry axes of the channel:
   - \( MD32000 \ $MA\_MAX\_AX\_VELO[x] \) (maximum axle velocity)
   - \( MD32200 \ $MA\_POSCTRL\_GAIN[x] \) (Kv factor)
   - \( MD32230 \ $MA\_MAX\_AX\_ACCEL[x] \) (position controller structure configuration)

   \( x = \text{axle number} \)

The following machine data is used to specify which channel axis is the direction axis:

- \( MD62528 \ $MC\_CLC\_PROG\_ORI\_AX\_MASK \) (progr. orientation vector: axis mask)

Each machine data bit corresponds to a channel axis.

| Format: | Bit31 | Bit30 | Bit29 | Bit28 | Bit27 | Bit26 | Bit25 | Bit24 | Bit23 | Bit22 | Bit21 | Bit20 | Bit19 | Bit18 | Bit17 | Bit16 | Bit15 | Bit14 | Bit13 | Bit12 | Bit11 | Bit10 | Bit9 | Bit8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
|         |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |

| Channel axis 1 |
|----------------|
| :             |
| :             |
| :             |
| :             |
| :             |
| :             |

- Coordinate X = channel axis corresponding to bit a
- Coordinate Y = channel axis corresponding to bit b
- Coordinate Z = channel axis corresponding to bit c

with \( a < b < c \)
Current difference angle

The difference angle is the angle between the tool orientation vector and the compensation vector. If the current difference angle of the clearance control is to be output in a system variable $AC_PARAM[n]$, index n of the system variable should be entered in the following machine data:

MD65530 $MC_CLC_PROG_ORI_ANGLE_AC_PARAM ()

Permissible limit angle

The permissible limit angle specifies the maximum difference angle allowed between the tool orientation vector and the compensation vector. The limit angle is configured via the following machine data:

MD65520 $MC_CLC_PROG_ORI_MAX_ANGLE ()

10.5.5 Parameter settings for clearance control

Part program identifiers

The following machine data must be parameterized for the declaration of the function-specific part program identifiers CLC_GAIN and CLC_VOFF:

- MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[0] = "OMA1" (list of re-configured NC codes)
- MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[1] = "CLC_GAIN"
- MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[2] = "OMA2"
- MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[3] = "CLC_VOFF"

1D/ 3D clearance control

The following machine data is used to select 1D or 3D clearance control:

- MD62500 $MC_CLC_AXNO = <n> (axis assignment for clearance control)
  - <n> > 0: 1D clearance control where <n> = axis number of the clearance-controlled channel axis
  - <n> = -1: 1. 5-axis transformation configured in channel
  - <n> = -2: 2. 5-axis transformation configured in channel
10.5 Startup

Input signals

The clearance sensor input signals parameterized above are made known to the clearance control function with the following machine data (see also Section "Parameter settings for input signals (840D sl) (Page 466)"):

- MD62502 $MN_CLC_ANALOG_IN = <n> (analog input for the clearance control)
  <n= input number, analog to the addressing of the system variables $A_INA[<n>]
- MD62504 $MN_CLC_SENSOR_TOUCHED_INPUT = <n> (assignment of an input bit for the "sensor collision" signal)
  <n= input number, analog to the addressing of the system variables $A_IN[<n>]

Exact stop

In order to be able to meet a programmed "Exact stop coarse/fine reached" block change condition (G601/G602), the traversing velocity induced by the clearance control function in the clearance-controlled axes must be lower than the standstill velocity tolerance at least for the duration of the standstill delay time.

The following machine data must be modified to optimize the block change time:

- MD36000 $MA_STOP_LIMIT_COARSE[<x>] (exact stop coarse)
- MD36010 $MA_STOP_LIMIT_FINE[<x>] (exact stop fine)
- MD36020 $MA_POSITIONING_TIME[<x>] (delay time exact stop fine)
- MD36040 $MA_STANDSTILL_DELAY_TIME[<x>] (standstill monitoring time delay)
- MD36060 $MA_STANDSTILL_VELO_TOL[<x>] ("axis/spindle stopped" velocity/speed threshold)
  <x> = axis number of clearance-controlled machine axis

10.5.6 Starting up clearance control

Clearance sensor

The clearance sensor outputs should be connected to the I/O modules that were activated using the following machine data:

- MD10362 $MN_HW_ASSIGN_ANA_FASTIN (I/O address of the I/O module) (hardware assignment for the fast analog NCK inputs)
- MD10366 $MN_HW_ASSIGN_DIG_FASTIN (I/O address of the I/O module) (hardware assignment for the external digital NCK inputs)

(See also Section "Supplementary conditions > I/O modules (Page 492)").
Test control direction

Proceed as follows to test the control direction of the clearance control function:

- Activate the clearance control via a part program using CLC(1) (see Section "Activating and deactivating clearance control (CLC) (Page 473)"
- Generate an input voltage, e.g. via the following synchronized action:

```plaintext
N100 $AC_TIMER[1]=2.5
N110 ID = 1 EVERY $AC_TIMER[1] >= 2.5 DO $AC_TIMER[1]=0
N120 ID = 2 WHENEVER $AC_TIMER[1] < 2.0 DO $A_OUITA[6] = 100000.0 * ($AC_TIMER[1] - 1.0)
N130 ID = 3 WHENEVER $AC_TIMER[1] >= 2.0 DO $A_OUITA[6] = 0.0
```

![Output voltage for synchronized action](image)

The voltage specification for the analog output $A_OUITA[6]$ used in the synchronized action is subtracted from the clearance sensor input voltage by the clearance control function and therefore has the opposite polarity to the input signal.

Set the following machine data to induce the clearance control function to use analog output 6 ($A_OUITA[6]$) as an additional input overlaid on the sensor input:

MD10366 $MN_CLC_OFFSET_ASSIGN_ANAOUT = 6$ (hardware assignment for the external digital NCK inputs)

**Note**

Before the clearance control function is activated for the first time, check that the entire working range enabled for clearance control is collision-free:

- MD62505 SMC_CLCSENSOR_LOWERLIMIT (lower clearance control motion limit)
- MD62506 SMC_CLCSENSOR_UPPERLIMIT (upper clearance control motion limit)
An incorrect control direction can be corrected using one of the following methods:

- Reversing the polarity of the analog input
- Changing the sign of all values in the following machine data:
  
  - MD62511 $MC_CLC_SENSOR_VELO_TABLE_1 (coordinate velocity of interpolation points sensor characteristic 1)
  
  - MD62513 $MC_CLC_SENSOR_VELO_TABLE_2 (coordinate velocity of interpolation points sensor characteristic 2)

**Sensor signal**

**Signal quality**

The quality of the analog input signal can be checked using the function-specific display data (see Section "Function-specific display data (Page 487)").

**Input voltage range**

The input voltage range for the measuring signal of the sensor can be adapted using the machine data for the weighting factor for the analog NCK inputs:

MD10320 $MN_FASTIO_ANA_INPUT_WEIGHT[<analog input>] = <weighting factor>

The following machine data must be set in order that the weighting factor from the clearance control function is included in the calculation:

MD62508 $MC_CLC_SPECIAL_FEATURE_MASK, Bit 13 = 1

**Function-specific alarm texts**

Function-specific alarm texts must first be integrated into the appropriate HMI data management before they can be displayed (see Section "Creating alarm texts (Page 443)").

**Completion**

A data backup is recommended once the start-up procedure has been completed.

**References:**

Commissioning Manual IBN CNC: NCK, PLC, drive

**Note**

A data backup is recommended once the start-up procedure has been completed.
10.6  Programming

10.6.1  Activating and deactivating clearance control (CLC)

Syntax

CLC(Mode)

Mode

- Format: Integer
- Range of values: -1, 0, 1, 2, 3

CLC(...) is a procedure call and must therefore be programmed in a dedicated part program block.

Functionality

The following modes are available for activating/ deactivating clearance control:

- CLC(1)
  Activation of the clearance control with compensation vector in the direction of the tool orientation
  The evaluation of the sensor collision signal is deactivated.

- CLC(2)
  Activation of the clearance control with compensation vector in the direction of the tool orientation
  The evaluation of the sensor collision signal is activated.

- CLC(3)
  Activation of the clearance control with programmed compensation vector
  The evaluation of the sensor collision signal is deactivated.

- CLC(0)
  Deactivation of clearance control without canceling the position offset.
  If the clearance-controlled axes are still moving at the instant of deactivation due to the sensor signal, they are stopped. The workpiece coordinate system (WCS) is then synchronized with the corresponding standstill positions. An automatic preprocessing stop is executed.

- CLC(-1)
  Deactivation of clearance control with cancellation of the position offset
  If the clearance-controlled axes are still moving at the instant of deactivation due to the sensor signal, they are stopped. A position offset to the last programmed position is canceled automatically with the deactivation command.
RESET behavior

CLC(0) is executed implicitly on a reset (NC RESET or end of program).

Parameterizable RESET behavior

The reset response of a 1D clearance control function can be determined via the channel-specific NCK OEM machine data:

- MD62524 $MC_CLC_ACTIVE_AFTER_RESET (reset response with active CLC)

The following response can be parameterized:

- MD62524 $MC_CLC_ACTIVE_AFTER_RESET = 0
  In the event of a reset, the clearance control function responds as it does to deactivation with CLC(0) (see "Functionality" section).

- MD62524 $MC_CLC_ACTIVE_AFTER_RESET = 1
  The current state of the clearance control function is retained.

⚠️ CAUTION

The following channel-specific NCK OEM machine data is only effective in conjunction with a 1D clearance control function:

- MD62524 $MC_CLC_ACTIVE_RESET (reset response with active CLC)

For 3D clearance control, CLC(0) is always effective in the event of a RESET.

Boundary conditions

Please note the following supplementary conditions:

Continuous-path mode

Activating/ deactivating clearance control (CLC(mode)) during active continuous-path mode (G64/G64x) will induce a drop in velocity for path motions. To avoid voltage drops of this type, clearance control must be activated before a path section with constant path velocity. During the corresponding path section, if necessary, clearance control can be blocked and then re-enabled via the programmed gain factor for clearance control (CLC_GAIN).

Block change with exact stop

If exact stop is active at the end of the block (G60/G09 with G601/G602) the block change may be delayed due to axis movements induced by the clearance control sensor signal.
Sensor collision monitoring

A digital input for an additional collision signal can be configured by the sensor using the following machine data:

MD6204 $MC_CLC_SENSOR_TOUCHED_INPUT (assignment of the input signal for the "sensor collision" signal)

This collision monitor can be activated and deactivated block-synchronously through alternate programming of CLC(1)/CLC(2).

As a reaction to the sensor collision signal, the clearance control moves, irrespective of the feedrate override setting, at maximum velocity in the plus direction until it reaches the currently valid upper limit. The path motion is stopped simultaneously.

NC-START can be used to resume processing.

3D clearance control and 5-axis transformation

If 3D clearance control is activated before the 5-axis transformation required for clearance control in the direction of the tool orientation has been activated, the clearance control function will be dependent on the active working plane (G17/G18/G19):

- G17: Direction of clearance control = Z
- G18: Direction of clearance control = Y
- G19: Direction of clearance control = X

Activation of 5-axis transformation

When 5-axis transformation is activated, the tool orientation specified by means of the rotary axis positions must tally with the control direction specified by the active working plane on activation of clearance control.

If the tool orientation of the 5-axis transformation and the control direction of the clearance control function do not tally, the following CLC alarm will appear:

- Alarm “75016 Channel number Block number CLC: Orientation changed with TRAFOOF”

Deactivation of 5-axis transformation

If 5-axis transformation is deactivated when clearance control is active, the last control direction before 5-axis transformation was deactivated is retained.

Tool radius compensation

3D clearance control can only be deactivated if no tool radius compensation is active in the channel at the time of deactivation (G40). If tool radius compensation is active (G41/G42), the following alarm appears:

- Alarm “75015 Channel number Block number CLC(0) with active TRC.”
### Compensation vector

**Actual position of the direction axes**

If the clearance control is activated with a programmable compensation vector at a position of 0 on all 3 direction axes, a compensation vector cannot be calculated from this information. The following alarm is then displayed:

- Alarm "75019 Channel number, error ID: 1, angle 0.0"

**Referencing of the direction axes**

The direction axes must be referenced before clearance control is activated with programmable compensation vector CLC(3).

**Interface signals of the direction axes**

The following interface signals must be set for all 3 direction axes by the PLC user program, before clearance control is activated with programmable compensation vector CLC(3).

- DBX31, … DBX1.5 = 1 (position measuring system 1)
- DBX31, … DBX2.1 = 1 (servo enable)
- DBX31, … DBX21.7 = 1 (pulse enable)

x = axle number

**Switchover of clearance control**

Direct switchover of clearance control from CLC(1) or CLC(2) to CLC(3) or vice-versa is not possible. Such switchovers are ignored without a checkback message. If a switchover is necessary, the clearance control must first be deactivated with CLC(0) or CLC(-1) and then activated in the desired mode.

**Interpolation of the compensation vector**

If the compensation vector is required to follow a non-linear workpiece surface, such as an arc, with respect to its orientation, this can be achieved by programming the direction axes.

**Example**

Orientation of the compensation vector perpendicular to a semi-circular workpiece surface. The programming of the traversing movement is not considered.
The compensation vector must be oriented by programming the direction axes at [1, 0, 0] before part program block N100. In part program block N100, the end position of the compensation vector is oriented by programming the direction axes at [0, 0, -1]. The intermediate values are generated by path interpolation of all axes programmed in the part program block:

- Geometry axes for the movement of the machining head
- Direction axes of the compensation vector

It is necessary to break the movement down into part program blocks N100 and N200, because an antiparallel orientation of the compensation vector of [1, 0, 0] at the start of the movement and [-1, 0, 0] at the end of the movement (semi-circle) would otherwise result. In this case, the interpolator would interpolate only the X coordinate of the compensation vector, and the orientation of the compensation vector would remain unchanged.

**Antiparallel orientation of the compensation vector**

When an antiparallel orientation of the compensation vector is programmed in a part program block, the following alarm is displayed:

- Alarm "75018 Channel number Block number CLC in programmable direction, error ID: 1"

---

**Note**

**Interpolation of the compensation vector**

The interpolation of the compensation vector is not a genuine vector interpolation, as described above, but results from the interpolation of the actual positions of the direction axes.
Consequently, if the compensation vector changes due to the workpiece contour, the interpolation of the direction axes is included in the path interpolation of the geometry axes. In order to minimize the impact of the direction axes on the path interpolation, it is recommended to configure the dynamic response of the direction axes at least equal to or greater (by a factor of approx. 10) than the dynamic response of the geometry axes.

In the case of a re-orientation (rotation) of the compensation vector, it is also necessary to note the ratio between the programmed traversing path and the configured dynamic response of the direction axes. The ratio should be chosen such that the programmed traversing path is not traversed in one or a small number of interpolation cycles, due to the dynamic response of the axis. This causes heavy loads on the machine and, in certain circumstances, may trigger axial alarms and abort part program execution.

**Example**

Rotation of the compensation vector and thus the machining head through 90°:

- Initial orientation: Parallel to coordinate axis X
- Target orientation: Parallel to coordinate axis Y

  Bad programming of re-orientation:
  - \([1, 0, 0] \rightarrow [0, 1, 0]\)

  Good programming of re-orientation:
  - \([100, 0, 0] \rightarrow [0, 100, 0]\)

**Rotation of the workpiece coordinate system**

As described above, the compensation vector always refers to the basic coordinate system (machine coordinate system). If the workpiece coordinate system (rotation, mirroring) is transformed to machine a workpiece in such a way that the coordinate axes of both coordinate systems are no longer parallel with the same orientation, a corresponding transformation must be carried out for the compensation vector.

| CAUTION |
|---------|
| If the workpiece coordinate system is transformed such that the coordinate axes of the basic and workpiece coordinate systems are no longer parallel with the same orientation, it is the sole responsibility of the user to ensure that an appropriate transformation of the compensation vector is carried out. |
10.6.2 Closed-loop control gain (CLC_GAIN)

Syntax

\[ \text{CLC_GAIN} = \text{Factor} \]

Factor

- Format: Real
- Range of values: \( y \geq 0.0 \)

CLC_GAIN is an NC address and can therefore be written together with other instructions in a part program block.

When a negative factor is programmed, the absolute value is used without an alarm output.

Functionality

The current closed-loop control gain for clearance control is produced by the active characteristic specified via machine data:

- MD62510 $MC_CLC_SENSOR_VOLTAGE_TABLE1 (coordinate voltage of interpolation points sensor characteristic 1)
- MD62511 $MC_CLC_SENSOR_VELO_TABLE1 (coordinate velocity of interpolation points sensor characteristic 1)
  
  or

- MD62512 $MC_CLC_SENSOR_VOLTAGE_TABLE2 (coordinate voltage of interpolation points sensor characteristic 2)
- MD62513 $MC_CLC_SENSOR_VELO_TABLE2 (coordinate velocity of interpolation points sensor characteristic 2)

CLC_GAIN can be used to multiply the closed-loop control gain of the characteristic by a programmable factor.

⚠️ CAUTION

Increasing the gain (CLC_GAIN > 1.0) may lead to oscillation in the controlled axes!

Instant of activation

The modified closed-loop control gain is effective in the part program block in which CLC_GAIN has been programmed or, if this block does not contain any executable instructions, in the next part program block with executable instructions.
Response to characteristic changeover

The programmed factor remains active even when the gain characteristic is changed over with CLC_SEL, i.e. it is immediately applied to the newly selected characteristic.

Response to CLC_GAIN=0.0

If the closed-loop control gain for clearance control is deactivated with CLC_GAIN=0.0, the CLC position offset present at the time of deactivation is retained and is not changed. This can be used for example when laser-cutting sheet steel to "skip over" sections of sheet that are not to be machined without foundering.

If the tool orientation is changed when 3D clearance control is active and the closed-loop control gain has been deactivated (CLC_GAIN=0.0), the CLC offset vector is rotated simultaneously. This generally induces an offset in the CLC operating point on the workpiece surface (see following diagram).

![Diagram of CLC offset vector response when CLC_GAIN=0.0](image.png)

Figure 10-11 Response of the CLC offset vector when CLC_GAIN=0.0

Reset

Within a part program, a modified gain factor must be reset by means of explicitly programming CLC_GAIN=1.0.

RESET response

CLC_GAIN=1.0 becomes effective after a power on reset, NC RESET or end of program.
10.6.3 Limiting the control range (CLC_LIM)

Syntax

 CLC_LIM(lower limit, upper limit)

*Lower limit, upper limit*

Format and value range as machine data:
- MD62505 $MC_CLC_SENSOR_LOWER_LIMIT[n]$ (lower clearance control motion limit)
- MD62506 $MC_CLC_SENSOR_UPPER_LIMIT[n]$ (upper clearance control motion limit)

CLC_LIM(...) is a procedure call and must therefore be programmed in a dedicated part program block.

Functionality

The maximum control range for clearance control can be modified on a block-specific basis using CLC_LIM. The maximum programmable lower/upper limit is limited by the limit value preset in the relevant machine data:
- MD62505 $MC_CLCSENSOR_LOWERLIMIT[1]$ (lower clearance control motion limit)
- MD62506 $MC_CLCSENSOR_UPPERLIMIT[1]$ (upper clearance control motion limit)

![Image of value range limits for lower and upper limit](image)

Figure 10-12 Value range limits for lower and upper limit

The control range limit is effective in relation to the current programmed setpoint position of the axis. If the limits are changed so that the actual position is located outside the limit, the clearance control automatically effects travel back to the limit range.
Reset

Within a part program, a modified control range limit can be reset by explicitly programming CLC_LIM without a "CLC_LIM( )" argument. This reapplies the limits from the following machine data:

- MD62505 $MC_CLC_SENSOR_LOWER_LIMIT[0]$ (lower clearance control motion limit)
- MD62506 $MC_CLCSENSOR_UPPER_LIMIT[0]$ (upper clearance control motion limit)

RESET response

The default setting from the above-mentioned machine data becomes effective after power on reset, NC RESET and end of program.

Error messages

The following programming errors are displayed with an alarm:

- Programming more than 2 arguments
  - CLC alarm "75005 Channel number Block number CLC_LIM: general programming error"
- Programming arguments outside the permissible limits
  - CLC alarm "750010 Channel number Block number CLC_LIM Value greater than MD limit"

10.6.4 Direction-dependent traversing motion disable

Syntax

$A_OUT[number] = enabling signal$

Number
Number of the parameterized digital output (see "Parameter Assignment" section)
- Format: Integer
- Range of values: 1, 2, \ldots \text{max. number of digital outputs}

Enabling signal
Invertable enable signal (see "Parameter Assignment" section)
- Format: Integer
- Range of values: 0, 1

System variable $A_OUT[n]$ can be set block-synchronously in the part program or asynchronously via synchronized actions.
Functionality

Parameterizable digital outputs (system variable $A\_OUT$) can be used for direction-dependent disabling of the traversing motion (manipulated variable) induced via clearance control. As long as e.g. the negative traversing direction is disabled, the clearance-controlled axes will only travel in a positive direction due to the sensor signal.

This can be used for example when laser-cutting sheet steel to "skip over" sections of sheet that are not to be machined without foundering.

Parameterization

The following machine data is used to parameterize the digital outputs:

- MD62523 $MC\_CLC\_LOCK\_DIR\_ASSIGN\_DIGOUT[n]$ (assignment of the digital outputs for disabling the CLC movement)
  - $n = 0 \rightarrow$ Digital output for disabling the negative traversing direction
  - $n = 1 \rightarrow$ Digital output for disabling the positive traversing direction

Example

The following digital outputs are to be used:

- $A\_OUT[3]$ to disable the negative traversing direction
- $A\_OUT[4]$ to disable the positive traversing direction

Parameter settings in the machine data:

- MD62523 $MC\_CLC\_LOCK\_DIR\_ASSIGN\_DIGOUT[0] = 3$ (assignment of the digital outputs for disabling the CLC movement)
- MD62523 $MC\_CLC\_LOCK\_DIR\_ASSIGN\_DIGOUT[1] = 4$

Effect:

- $A\_OUT[3] = 0 \rightarrow$ Negative traversing direction enabled
- $A\_OUT[3] = 1 \rightarrow$ Negative traversing direction disabled
- $A\_OUT[4] = 0 \rightarrow$ Positive traversing direction enabled
- $A\_OUT[4] = 1 \rightarrow$ Positive traversing direction disabled

Inversion of the evaluation

Enter the negative number of the digital output to evaluate the digital output signal with inversion:

Parameter settings in the machine data:

- MD62523 $MC\_CLC\_LOCK\_DIR\_ASSIGN\_DIGOUT[0] = -3$ (assignment of the digital outputs for disabling the CLC movement)
- MD62523 $MC\_CLC\_LOCK\_DIR\_ASSIGN\_DIGOUT[1] = -4$
Effect:
- $A\_OUT[3] = 0 \rightarrow$ Negative traversing direction disabled
- $A\_OUT[3] = 1 \rightarrow$ Negative traversing direction enabled
- $A\_OUT[4] = 0 \rightarrow$ Positive traversing direction disabled
- $A\_OUT[4] = 1 \rightarrow$ Positive traversing direction enabled

10.6.5 Voltage offset, can be set on a block-specific basis (CLC_VOFF)

Syntax

$$\text{CLC\_VOFF} = \text{Voltage offset}$$

Voltage offset
- Format: Real
- Unit: Volts
- Range of values: No restrictions

CLC_VOFF is an NC address and can therefore be written together with other instructions in a part program block.

Functionality

CLC_VOFF can be used to preset a constant voltage offset for clearance control, which is subtracted from the input voltage of the clearance sensor. The programmed voltage offset therefore changes the setpoint distance between the workpiece and the clearance sensor or offsets the operating point for clearance control.

The quantitative effect of the voltage offset is dependent on the additional parameters for clearance control and can therefore not be standardized in a generally valid format.

Instant of activation

The voltage offset is effective in the part program block in which CLC_VOFF has been programmed or, if this block does not contain any executable instructions, in the next part program block with executable instructions.

Reset

Within a part program, a voltage offset must be reset by means of explicitly programming CLC_VOFF=0.0.

RESET response

CLC_VOFF =0.0 becomes effective after a power on reset, NC RESET or end of program.
10.6.6 Voltage offset definable by synchronized action

Syntax

\[$A_{OUTA}[number] = Voltage\ offset\$

Number

Number of the parameterized analog output (see "Parameter Assignment" section)

- Format: Integer
- Range of values: 1, 2, . . . max. number of analog outputs

Voltage offset

Just like the voltage offset for CLC_VOFF (see Section "Voltage offset, can be set on a block-specific basis (CLC_VOFF) (Page 484)").

Functionality

A parameterizable output (system variable $A_{OUTA}$) can be used to apply a voltage offset for clearance control, which, like CLC_OFF, is subtracted from the input voltage of the clearance sensor.

The voltage offset can be modified in the interpolation cycle by programming the analog output within a synchronized action.

Parameterization

The following machine data is used to parameterize the analog output:

MD62522 $MC\_CLC\_OFFSET\_ASSIGN\_ANAOUT$ (modification of the setpoint distance by means of sensor signal override)

Example

An external voltage $U_{ext}$ is present at analog input $A_{INA}[3]$, which is to be overlaid on the sensor voltage as a continuously variable voltage offset e.g. for test or start-up purposes. $A_{OUTA}[2]$ is used as an analog output for the clearance control voltage offset.

Parameter setting for the analog output for clearance control voltage offset:

MD62522 $MC\_CLC\_OFFSET\_ASSIGN\_ANAOUT = 2$ (modification of the setpoint distance by means of sensor signal override)

The analog input $A_{INA}[3]$ is assigned to the clearance control analog output $A_{OUTA}[2]$ within a synchronized action:

ID=1 DO $A_{OUTA}[2] = A_{INA}[3]$
10.6.7 **Selection of the active sensor characteristic (CLC_SEL)**

**Syntax**

CLC_SEL(*characteristic number*)

*Characteristic number*

- Format: Integer
- Range of values: 1, 2

CLC_SEL(...) is a procedure call and must therefore be programmed in a dedicated part program block.

*Characteristic number* = 2 selects characteristic 2. Any other value selects characteristic 1 without alarm.

**Functionality**

CLC_SEL can be used to switch between the sensor characteristics defined in the machine data.

- **Characteristic 1:**
  - MD62510 $MC_CLC_SENSOR_VOLTAGE_TABLE_1 (coordinate voltage of interpolation points sensor characteristic 1)
  - MD62511 $MC_CLC_SENSOR_VELO_TABLE_1 (coordinate velocity of interpolation points sensor characteristic 1)

- **Characteristic 2:**
  - MD62512 $MC_CLC_SENSOR_VOLTAGE_TABLE_2 (coordinate voltage of interpolation points sensor characteristic 2)
  - MD62513 $MC_CLC_SENSOR_VELO_TABLE_2 (coordinate velocity of interpolation points sensor characteristic 2)

**RESET response**

Characteristic 1 becomes effective after a power on reset, NC RESET or end of program.
10.7 Function-specific display data

The "clearance control" technological function provides specific display data for supporting start-up and for service purposes.

Possible applications

Application options for display data include for example:

- Determination of form variances and transient control errors via the variables for the maximum and minimum position offset/sensor voltage.
- Determination of the voltage noise detected by the A/D converter via the variables for the maximum and minimum sensor input voltage. This requires a constant clearance between the clearance sensor and the workpiece surface and the deactivation of clearance control via CLC_GAIN = 0.0.

The minimum and maximum values are detected in the position controller cycle.

Types of variable

The display data is available both as channel-specific GUD (Global User Data) variables and as OPI variables.

10.7.1 Channel-specific GUD variables

The "clearance control" technological function provides the following channel-specific GUD variables for HMI applications:

- SINUMERIK HMI Advanced
- SINUMERIK Operate

| GUD variables       | Description                              | Unit   | Access       |
|---------------------|------------------------------------------|--------|--------------|
| CLC_DISTANCE[0]     | Current position offset                  | mm     | read only    |
| CLC_DISTANCE[1]     | Absolute minimum of position offset      | mm     | read/write   |
| CLC_DISTANCE[2]     | Absolute maximum of position offset      | mm     | read/write   |
| CLC_VOLTAGE[0]      | Current sensor input voltage             | V      | read only    |
| CLC_VOLTAGE[1]      | Absolute minimum of sensor input voltage | V      | read/write   |
| CLC_VOLTAGE[2]      | Absolute maximum of sensor input voltage | V      | read/write   |

Once the technological function has been started up successfully, the GUD variables listed are not displayed automatically on the HMI interface.
SINUMERIK HMI Advanced

Proceed as follows to create and display the GUD variables in HMI Advanced.

1. Setting the password
   Enter the password for protection level 1: (machine manufacturer).

2. Activate the "definitions" display.
   Operating area switchover > Services > Data Selection

3. If no SGUD.DEF file is yet available:
   Operating area switchover > Services > Data admin > New...
   • Name: SGUD
   • Type: Global data/system
     Confirm with OK.
     This opens the file in the editor.

1. Edit the GUD variable definitions
   DEF CHAN REAL CLC_DISTANCE[3] ; Array of real, 3 elements
   DEF CHAN REAL CLC_VOLTAGE[3] ; Array of real, 3 elements
   M30

2. Save the file and close the editor.

3. Activate the SGUD.DEF file.

The GUD variables for clearance control are now displayed under:
   Operating area switchover > Parameters > User data > Channel user data

SINUMERIK Operate

Proceed as follows to create and display the GUD variables for SINUMERIK Operate.

1. Setting the password
   Enter the password for protection level 1: (machine manufacturer).

2. If no SGUD.DEF file is yet available:
   Operating area switchover > Commissioning > System data > Open NC data directory:
   Set cursor to definitions > New...
   • Name: SGUD
   • Type: DEF
     Confirm with OK.
     This opens the file in the editor.
3. Edit the GUD variable definitions

   DEF CHAN REAL CLC_DISTANCE[3] ; Array of real, 3 elements
   DEF CHAN REAL CLC_VOLTAGE[3] ; Array of real, 3 elements
   M30

4. Save the file and close the editor.

5. Activate the SGUD.DEF file.

The GUD variables for clearance control are now displayed under:

**Operating area switchover > Parameters > User variables > GUD channel**

### SINUMERIK NCK

The new GUD variables, which are already being displayed, will only be detected by the clearance control function and supplied with up-to-date values following an NCK POWER ON RESET.

**Note**

Once the GUD variables have been created, an NCK POWER ON RESET must be carried out in order for the clearance control function to update the GUD variables.

#### 10.7.2 OPI variable

The "clearance control" technological function provides the following channel-specific OPI variables as display data for the HMI application:

- SINUMERIK HMI Advanced

| OPI variable | Description                  | Unit     | Access     |
|--------------|------------------------------|----------|------------|
| CLC[0]       | Current position offset      | mm       | read only  |
| CLC[1]       | Absolute minimum of position | mm       | read/write |
|              | offset                       |          |            |
| CLC[2]       | Absolute maximum of position | mm       | read/write |
|              | offset                       |          |            |
| CLC[3]       | Current sensor input voltage | V        | read only  |
| CLC[4]       | Absolute minimum of sensor   | V        | read/write |
|              | input voltage                |          |            |
| CLC[5]       | Absolute maximum of sensor   | V        | read/write |
|              | input voltage                |          |            |
| CLC[6]       | 1. component of the         | -        | read only  |
|              | standardized tool orientation vector | |      |
### OPI variable

Once the technological function has been started up successfully, the OPI variable is not available automatically.

**1. Create the CLC-specific definition file:** `CLC.NSK`

**Note:**
We recommend that you create the file in the `\OEM` directory rather than in the `\MMC2` directory so that it is not overwritten when a new software version is installed.

**2. Define the CLC-specific OPI variables.**

Add the following line to the `CLC.NSK` file:

```
LINK("CLC" ,200,"2 1 1 1 1F# /NC 5 0 1",100)
```

**3. Create/expand the user-specific definition file:** `USER.NSK`

(See item 1: Note)

**4. In file USER.NSK, supplement the call of the CLC-specific definition file:** `CLC.NSK`. To do this, insert the following line:

```
CALL(CLC.NSK)
```

### Using LinkItem

In order to use the OPI variables in a DDE control, the "LinkItem" property of the DDE control must be set in accordance with the following example:

```
label1.LinkItem = "CLC[u1,1,9](" "d%15.4lf "")"
```

The format string can be modified if necessary.

The following code lines provide an example of how the variables supplied by means of NCDDE access can be distributed on a field of labels:

```
For i = 0 To 8
label2.Caption[ i ] = Trim$(Mid$(label1.Caption, 1 + 15 * i, 15) )
Next
```
10.8 Function-specific alarm texts

For details of the procedure for creating function-specific alarm texts, see Section "Creating alarm texts (Page 443)".
10.9 Supplementary conditions

10.9.1 I/O modules

The analog output voltage of the distance sensor for the A/D conversion must be connected to the NC using an I/O module with an analog input.

Connection options

The SIMATIC ET 200S I/O for SINUMERIK 840D sl is connected via PROFIBUS DP. The clearance sensor is connected via an analog S7 I/O module.

Suitable I/O modules

As the A/D conversion time directly affects the deadtime of the clearance control servo loop, only one I/O module may be used with low conversion time.

Suitable SIMATIC S7 I/O modules for the clearance control are:

- Analog I/O module 2 AI, U, high-speed for ET 200S
  Order number (MLFB): 6ES7134-4FB52-0AB0
- Analog I/O module 2 AO, U, high-speed for ET 200S
  Order number (MLFB): 6ES7135-4FB52-0AB0
I/O module connection

The SIMATIC I/O devices of the Production Series ET200, e.g. ET200M, are brought into the S7 project as usual, and configured.

**Note**

To check whether a module selected from the hardware catalog complies with the module in the automation system, the following procedure is recommended:

1. Note the MLFB numbers of all modules used in the automation system.
2. Select the corresponding module in the hardware catalog and compare the MLFB number of the module used in the automation system with the MLFB number that is displayed in the hardware catalog. Both MLFB numbers must be the same.

10.9.1.1 External smoothing filters

If an external filter is to be interconnected to smooth the output voltage of the clearance sensor before the A/D conversion of the output voltage by the I/O module, please ensure that the resulting time constant is small in relation to the NC position controller cycle.

**Note**

It is better for the control if electromagnetic shielding is used to ensure a large signal-noise ratio than if smoothing filters are used in the signal path.

10.9.2 Function-specific supplementary conditions

**Complete NC Stop**

If in conjunction with an NC Stop, not only the programmed path motion but also the traversing movement of the clearance-controlled axes should be stopped, the following NC/PLC interface signals must be set:

- DB21, ... DBX7.3 = 1 (NC Stop)
- DB21, ... DBX7.4 = 1 (NC stop axes and spindles)

**Followup**

If a clearance-controlled axis is to be switched as an alarm response or via the corresponding interface signal from the NC/PLC in “follow-up” mode, setpoint output will cease for clearance control on this axis.
Travel without software limit switches

If the clearance-controlled axes are to travel without referencing (travel without software limit switches), values outside the used traversing range must still be parameterized for the axis-specific software limit switches:

- MD36100 $MA_POS_LIMIT_MINUS (1st software limit switch minus)
- MD36110 $MA_POS_LIMIT_PLUS (1st software limit switch plus)
- MD36120 $MA_POS_LIMIT_MINUS2 (2nd software limit switch minus)
- MD36130 $MA_POS_LIMIT_PLUS2 (2nd software limit switch plus)

Clearance control takes the machine data into account even if an axis is not being referenced.

Disabling digital/ analog inputs

Neither the analog input for the input voltage of the clearance sensor nor the digital input used by the clearance control in the context of the "Lift fast with position controller cycle" special function can be controlled (disabled) via the PLC:

DB10, DBB0 (Disable digital NCK inputs)
DB10, DBB146 (Disable analog NCK inputs)

See also the description of machine data:

- MD62508 $MC_CLC_SPECIAL_FEATURE_MASK, bit 4 and 5 (Special functions and operating modes of the clearance control)

Gantry axes: Only leading axes

Only one of the clearance-controlled axes may be configured as the master axis of a gantry grouping:

MD37100 $MA_GANTRY_AXIS_TYPE (gantry axis definition)

The use of following axes in a gantry grouping is not permitted.

Displaying the axis position

The actual current axis position of a clearance-controlled axis as the sum of an interpolatory axis position and the current position offset of clearance control is not displayed in the main machine screen:

- SINUMERIK HMI Advanced:
  The actual current axis position is displayed in the Service screen: Operating area switchover > Diagnosis > Service displays > Axis/spindle displayed as the "actual position value".
- SINUMERIK Operate:
  The actual current axis position is displayed in the Service screen: Operating area switchover > Diagnosis > Axis dialog > Service axis > displayed as the "measuring system 1 and 2 actual position value".
No virtual axes

Clearance-controlled axes must not be parameterized as virtual axes:
MD30132 $MA_IS_VIRTUAL_AX[<axis>] (axis is virtual axis)

Computing time requirements

The additional computing time required for the "clearance control" technological function must be taken into account on control systems in which the cycle times set for the interpolator and position controller cycle have been substantially optimized in comparison with the default setting:

The additional required computing time results after the activation of the clearance control in the CLC(... part program. If the interpolation or position controller cycle is exceeded, the following alarm appears:
- Interrupt: "4240 Computing time overflow at IPO or position controller level, IP point in part program"

Execution of the part program is aborted.

1D clearance control function

Alarm "1016: System error, ID550010" can occur in the following situation:
- The clearance-controlled axis (e.g. Z axis) is parameterized as geometry axis
- Within any command sequence in which STOPRE is initiated implicitly or explicitly, the clearance control with CLC(0) is switched off

We therefore recommend that the clearance-controlled axis of a 1D clearance control function (e.g. Z axis) is parameterized so that it is no geometry axis of the channel.

Machine data
- MD20050 $MC_AXCONF_GEOAX_ASSIGN_TAB[<Z axis>] = 0
- MD20060 $MC_AXCONF_GEOAX_NAME_TAB[<Z axis>] = "NO_Z_AXIS"

Frame rotations

If the Z axis is not a geometry axis, the CRPL(1,0) command rather than the CROT(Z,0) command must used be for frame rotations around the Z axis.
10.10 Data lists

10.10.1 Machine data

10.10.1.1 NC-specific machine data

| Number | Identifier: $MN_ | Description |
|--------|------------------|-------------|
| 10300  | FASTIO_ANA_NUM_INPUTS | Number of active analog NCK inputs |
| 10350  | FASTIO_DIG_NUM_INPUTS | Number of active digital NCK input bytes |
| 10362  | HW_ASSIGN_ANA_FASTIN | Hardware assignment of external analog NCK inputs: 0...7 |
| 10712  | NC_USER_CODE_CONF_NAME_TAB | List of renamed NC identifiers |

10.10.1.2 Channel-specific machine data

| Number | Identifier: $MC_ | Description |
|--------|------------------|-------------|
| 28090  | MM_NUM_CC_BLOCK_ELEMENTS | Number of compile cycle block elements (DRAM) |
| 28100  | MM_NUM_CC_BLOCK_USER_MEM | Memory for compile cycle block elements (DRAM) in kB |
| 28254  | MM_NUM_AC_PARAM | Number of parameters for synchronized actions |

Clearance control

| Number | Identifier: CLC_ | Description |
|--------|------------------|-------------|
| 62500  | CLC_AXNO | Axis assignment for clearance control |
| 62502  | CLC_ANALOG_IN | Analog input for clearance control function |
| 62504  | CLC_SENSOR_TOUCHED_INPUT | Input bit assignment for the "sensor collision" signal |
| 62505  | CLC_SENSOR_LOWER_LIMIT | Lower motion limit of clearance control |
| 62506  | CLC_SENSOR_UPPER_LIMIT | Upper motion limit of clearance control |
| 62508  | CLC_SPECIAL_FEATURE_MASK | Special functions and operating modes of the clearance control |
| 62510  | CLC_SENSOR_VOLTABE_TABLE_1 | Coordinate voltage of interpolation points sensor characteristic 1 |
| 62511  | CLC_SENSOR_VELO_TABLE_1 | Coordinate velocity of interpolation points sensor characteristic 1 |
| 62512  | CLC_SENSOR_VOLTAGE_TABLE_2 | Coordinate voltage of interpolation points sensor characteristic 2 |
| 62513  | CLC_SENSOR_VELO_TABLE_2 | Coordinate velocity of interpolation points sensor characteristic 2 |
| 62516  | CLC_SENSOR_VELO_LIMIT | Clearance control movement velocity |
| 62516  | CLC_SENSOR_ACCEL_LIMIT | Clearance control movement acceleration |
| 62520  | CLC_SENSOR_STOP_POS_TOL | Positional tolerance for status message "Clearance control zero speed" |
10.10 Data lists

| Number | Identifier: $MC_\_ \_ | Description |
|--------|--------------------------|-------------|
| 62521  | CLC_SENSOR_STOP_DWELL_TIME | Dwell time for status message "Clearance control zero speed" |
| 62522  | CLC_OFFSET_ASSIGN_ANAOUT  | Modification of the setpoint distance by means of sensor signal override |
| 62523  | CLC_LOCK_DIR_ASSIGN_DIGOUT | Assignment of the digital outputs for disabling the CLC movement |
| 62524  | CLC_ACTIVE_AFTER_RESET    | Clearance control remains active after RESET |
| 62525  | CLC_SENSOR_FILTER_TIME    | PT1 filtering time constant of sensor signal |
| 62528  | CLC_PROG_ORI_AX_MASK     | Programmed orientation vector: Axis mask |
| 62529  | CLC_PROG_ORI_MAX_ANGLE    | Programmed orientation vector: Maximum difference angle |
| 62530  | CLC_PROG_ORI             | Programmed orientation vector: Index of the $AC_PARAM variables for the output of the current difference angle |

10.10.1.3 Axis/spindlespecific machine data

| Number | Identifier: $MA_\_ \_ | Description |
|--------|--------------------------|-------------|
| 32070  | CORR_VELO                | Axis velocity for handwheel, external zero offsets, SA clearance control |
| 32410  | AX_JERK_TIME             | Time constant for axis jerk filter |
| 32610  | VELO_FFW_WEIGHT          | Feedforward control factor for velocity feedforward control |
| 36000  | STOP_LIMIT_COARSE        | Exact stop coarse |
| 36010  | STOP_LIMIT_FINE          | Exact stop fine |
| 36040  | STANDSTILL_DELAY_TIME    | Delay time zero speed monitoring |
| 36060  | STANDSTILL VELO_TOL      | Axis/Spindle velocity stopped |
| 36750  | AA_OFF_MODE              | Value calculation mode for axial position override |
10.10 Data lists

10.10.2 Drive parameters (SINAMICS S120)

| Number       | Short name      | Long name                                |
|--------------|-----------------|------------------------------------------|
| p1414[0...n] | n_soll_filt current | Speed setpoint filter activation 1, 2     |
| p1415[0...n] | n_soll_filt 1 type | Speed setpoint filter 1 type             |
| p1416[0...n] | n_soll_filt 1 T   | Speed setpoint filter 1 time constant     |
| p1417[0...n] | n_soll_filt 1 fn_n | Speed setpoint filter 1 denominator natural frequency |
| p1418[0...n] | n_soll_filt 1 D_n | Speed setpoint filter 1 denominator damping |
| p1419[0...n] | n_soll_filt 1 fn_z | Speed setpoint filter 1 numerator natural frequency |
| p1420[0...n] | n_soll_filt 1 D_z | Speed setpoint filter 1 numerator damping |
| p1421[0...n] | n_soll_filt 2 type | Speed setpoint filter 2 type             |
| p1422[0...n] | n_soll_filt 2 T   | Speed setpoint filter 2 time constant     |
| p1423[0...n] | n_soll_filt 2 fn_n | Speed setpoint filter 2 denominator natural frequency |
| p1424[0...n] | n_soll_filt 2 D_n | Speed setpoint filter 2 denominator damping |
| p1425[0...n] | n_soll_filt 2 fn_z | Speed setpoint filter 2 numerator natural frequency |
| p1426[0...n] | n_soll_filt 2 D_z | Speed setpoint filter 2 numerator damping |

10.10.3 Signals

10.10.3.1 Signals to channel

| DB number | Byte.bit | Description     |
|-----------|----------|-----------------|
| 21, ...   | 1.4      | Stop CLC motion |
| 21, ...   | 1.5      | Feedrate override acts on CLC |

10.10.3.2 Signals from channel

| DB number | Byte.bit | Description              |
|-----------|----------|--------------------------|
| 21, ...   | 37.3     | CLC is active            |
| 21, ...   | 37.4-5   | CLC motion has stopped   |
| 21, ...   | 37.4     | CLC motion at lower motion limit |
| 21, ...   | 37.5     | CLC motion at upper motion limit |
11.1 Brief description

A master-slave coupling is a speed setpoint coupling between a master and any number of slave axes - performed at the position controller level - with and without torque equalization control. The coupling can be permanently closed, dynamically closed and opened and reconfigured.

Possible applications of a master-slave coupling include:

- Increase the power for mechanically coupled drives
- Compensating gear and gear tooth flank play by entering a pre-tensioning torque

![Diagram of Dynamic Master-Slave Coupling of Two Axes](image-url)

Figure 11-1 Dynamic master-slave coupling of two axes
11.2 Coupling diagram

If the coupling is closed, the slave axis is traversed only with the load-side setpoint speed of the master axis. It is therefore only speed-controlled, not position-controlled. There is no differential position control between the master and slave axis.

The torque required between the master and slave axis is distributed using the torque equalization controller. When using different motors, this distribution can be adapted to the specific requirements using a weighting factor.

A tension can be established between the master and slave axes by entering a supplementary torque (tensioning torque).

---

Figure 11-2 Control structure
11.3 Configuring a coupling

Static assignment

For a speed setpoint coupling and torque equalization control, the static assignment of master and slave axis is defined separately in the following machine data:

- Speed setpoint coupling
  \[ \text{MD37250 $MA_MS_ASSIGN_MASTER_SPEED_CMD[<slave axis>] = <machine axis number of the master axis for the speed setpoint coupling>} \]

- Torque equalization control
  \[ \text{MD37252 $MA_MS_ASSIGN_MASTER_TORQUE_CTR[<slave axis>] = <machine axis number of the master or slave axis for torque equalization control> (see Section "Tension torque (Page 507)")} \]

Dynamic assignment

The assignment of master and slave axes can be dynamically changed using the following program commands.

**Defining assignments**

Assignment of one or several slave axes to a master axis:

\[ \text{MASLDEF (<slave axis_1>, <slave axis_2>, ..., <master axis>)} \]

In principle, any number of slave axes can be assigned. The assignment is kept even after an operating mode change, reset and end of part program.

**Deleting assignments**

Delete assignment of one or several slave axes to their associated master axis:

\[ \text{MASLDEL (<slave axis_1>, <slave axis_2>, ...)} \]

An active coupling is first implicitly opened before deleting the assignment.

**Axis assignment for speed setpoint coupling and torque equalization control**

For the speed setpoint coupling, the slave axis refers to the master axis specified when defining the assignment (\text{MASLDEF}).

The axis to which the slave axis refers for torque equalization control is defined in the following machine data:

\[ \text{MD37253 $MA_MS_FUNCTION_MASK[<slave axis>], bit 1 = <value>} \]
11.3 Configuring a coupling

### Special functions

| <value> | Description |
|---------|-------------|
| 0       | In the case of dynamic assignment (MASLDEF), the slave axis for the torque equalization control – as well as for speed setpoint coupling – refers to the master axis of the master-slave group. |
| 1       | In the case of dynamic assignment (MASLDEF), the slave axis for the torque equalization control refers to the axis, master or slave axis of the master-slave group specified in machine data: MD37252 $MA_MS_ASSIGN_MASTER_TORQUE_CTR<slave axis> = <axis> |

(See Section "Tension torque (Page 507)")

### Supplementary conditions

The following supplementary conditions must be observed for the dynamic assignment:

- When the coupling is closed, a change to the assignment using MASLDEF has no effect. The change only becomes effective the next time that the coupling is opened.
- The same master axis is always used for the speed setpoint coupling and torque equalization control.
- A plausibility check of the assignment is only made when the coupling is closed.

### User-specific standard assignment after reset

A user-specific standard assignment, which is always effective after a reset, can be defined in PROG_EVENT.SPF using the program commands MASLDEF and MASLDEL.

The following machine data should be set in order that PROG_EVENT.SPF is executed for a reset: MD20108 $MC_PROG_EVENT_MASK.Bit 2 = 1

### Example: Dynamic change of the assignment

The assignment of slave axis AX3 is changed from master axis AX1 to master AX2. To do this, the coupling must be temporarily opened (see Section "Closing/opening a coupling (Page 511)").

| Programming   | Description                                      |
|---------------|--------------------------------------------------|
| ① MASLDEF (AX3, AX1) | Assignment of slave axis AX3 to master axis AX1 |
| ② MASLON (AX3)   | Close the coupling                               |
| ③ MASLDEL (AX3)  | Open the coupling and cancel the assignment between AX3 and AX1 |
| ④ MASLDEF (AX3, AX2) | Assignment of slave axis AX3 to master axis AX2 |
**General supplementary conditions**

Please note the following general supplementary conditions:

- A slave axis can only be assigned to one master axis
- One master axis can be assigned several slave axes
- A slave axis must not be a master axis or be in another master-slave relationship.

**NOTICE**

**Drive optimization**

At a SINAMICS S120 drive unit, a maximum of three drives can be optimized or measured at the same time (speed controller optimization / function generator). Therefore, for a coupling with more than three coupled drives at the same time, we recommend that these are distributed over several drive units.
11.4 Torque compensatory controller

The torque equalization controller (PI control) calculates a load-side initial speed setpoint. The additional speed setpoint can be differently entered via the following machine data:

\[
\text{MD37254 } $\text{MA_MS_TORQUE_CTRL_MODE}[$\text{<slave axis>}$] = <\text{value}> 
\]

| <value> | Input of the additional speed setpoint at: |
|---------|-----------------------------------------|
| 0       | Master and slave axis (default value)    |
| 1       | Slave axis                              |
| 2       | Master axis                             |
| 3       | is not input at any of the axes          |

**NOTICE**

Several slave axes

For a master-slave grouping with one master and several slave axes, the input of the additional speed setpoint to the master and slave axes - set as standard in MD37254 - can result in instability. For several axes, the input of an additional speed setpoint should only be done in the slave axes:

\[
\text{MD37254 } $\text{MA_MS_TORQUE_CTRL_MODE} = 1 
\]

SINAMICS S120: Current setpoint filter

Up to four current setpoint filters (p1656 - p1674, function block diagram [5710]) can be activated in the SINAMICS S120. In a master-slave group, it is recommended that for the master and all of the slave axes, only filters with the same data are activated at any instant in time.

**Scaling**

Scaling the machine data:

- MD37256 $\text{MA_MS_TORQUE_CTRL_P_GAIN}$ (gain factor (P component))
- MD37260 $\text{MA_MS_MAX_CTRL_VELO}$ (speed setpoint limiting)

Must be entered using the following machine data:

\[
\text{MD37253 } $\text{MA_MS_FUNCTION_MASK}[$\text{<slave axis>}$], \text{bit } 0 = <\text{value}> 
\]

| <value> | Description |
|---------|-------------|
| 0       | MD37256 and MD37260 are internally multiplied with the following factor: 1 / Ipo cycle[s] |
| 1       | MD37256 and MD37260 are accepted unchanged |
Note
It is recommended to set MD37253 $MA_MS_FUNCTION_MASK[<slave axis>], bit 0 = 1.

Gain factor (P component)

The gain factor of the torque equalization controller is set in the following machine data as a percentage of the ratio of the maximum load-side axis velocity of the slave axis (MD32000 $MA_MAX_AX VELO) to its rated torque (SINAMICS S120: p2003):

MD37256 $MA_MS_TORQUE_CTRL_P_GAIN[<slave axis>]

Note: Scaling via MD37253 $MA_MS_FUNCTION_MASK[<slave axis>], bit 0

Integral time (I component)

The integral time of the torque equalization controller is set in the following machine data:

MD37258 $MA_MS_TORQUE_CTRL_I_TIME[<slave axis>]

The I component is deactivated as standard.

Speed setpoint limiting

The speed setpoint calculated by the torque equalization controller can be limited using the following machine data. The input is a percentage referred to the maximum, load-side axis velocity of the slave axis (MD32000 $MA_MAX_AX VELO).

MD37260 $MA_MS_MAX_CTRL_VELO[<slave axis>]

Note: Scaling via MD37253 $MA_MS_FUNCTION_MASK[<slave axis>], bit 0

Deactivating the torque equalization controller

If the following settings are made, the torque compensatory controller will be inactive:

- MD37254 $MA_MS_TORQUE_CTRL_MODE[<slave axis>] = 3
- MD37256 $MA_MS_TORQUE_CTRL_P_GAIN[<slave axis>] = 0
Torque weighting

The percentage of the torque generated by the slave axis of the total torque can be set using the torque weighting.

MD37268 $MA_MS_TORQUE_WEIGHT_SLAVE[<slave axis>]

Using the torque weighting, it is possible to configure a different torque distribution between the master and slave axes for motors with different rated torques. For motors of the master and slave axes with the same rated torques, it makes sense to have a 50% torque distribution (default setting).

The torque provided by the master axis is given by:

Torque provided by the master axis = 100% - MD37268

NOTICE

Mechanical coupling

When using the torque equalization controller, it is absolutely necessary to have a mechanical coupling between the master and slave axis. Otherwise, the two drives involved could accelerate in an uncontrollable fashion.

Activating/deactivating via the NC/PLC interface

The torque equalization controller can be activated on an axis-for-axis basis via the NC/PLC interface:

DB31, ... DBX24.4 (torque equalization controller on)

Activating via the NC/PLC interface must be explicitly enabled:

MD37254 $MA_MS_TORQUE_CTRL_MODE[<slave axis>] = 1

The actual activation state can be read back via:

DB31, ... DBX96.4 (master/slave equalization controller active)
11.5 Tension torque

The tension torque is a supplementary torque which is switched to the active torque equalization controller. This means that a mechanical tension can be established between axes within a master-slave grouping. Establishing a tension is not only possible between the master and a slave axis, but also between two slave axes by declaring one of the slave axes the reference axis for the torque equalization controller.

The tension torque is entered as a percentage of the rated torque of the slave axis and is active immediately:

\[
\text{MD37264 } \text{ $MA_MS\_TENSION\_TORQUE[<slave axis>] = <tension torque>}
\]

As can be seen from the structure of the torque equalization controller (Chapter "Torque compensatory controller (Page 504)"), the tension torque is entered via a PT1 filter. The filter time constant is set using the following machine data:

\[
\text{MD37266 } \text{ $MA_MS\_TENSION\_TORQ\_FILTER\_TIME[<slave axis>] = <time constant>}
\]

A value for the filter time constant greater than 0 activates the filter.

---

**Note**

**Minimum tension torque**

The tension torque chosen must be high enough to ensure that the resulting torque does not drop below the minimum required tension even during acceleration.

**Reducing the motor temperature rise**

The tension torque can be reduced when the motor is at a standstill to reduce the motor temperature rise.

---

**NOTICE**

If a tension torque is entered without a mechanical coupling between the master and slave axis then this causes the axes to move.
**Example 1: Static coupling and establishing a tension in pairs**

A master-slave application is parameterized so that one master axis is assigned three slave axes and a tension torque is entered for one axis pair. As a consequence, the master axis alone does not have to establish the tension torque with respect to all of the slave axes.

Assumption regarding machine axes:
- 1. up to the 4th machine axis: AX1, AX2, AX3, AX4

Static coupling for all slave axes
- MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX2] = 1
- MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX3] = 1
- MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX4] = 1

| Axis | Reference axis of the speed setpoint coupling MD37250 = value | Reference axis of the torque equalization controller MD37252 = value | Input of the torque equalization contr. MD37254 = value |
|------|--------------------------------------------------------------|---------------------------------------------------------------|------------------------------------------------------|
| AX1  | 0 No reference axis                                         | 0 No reference axis                                           | 0 Master and slave                                    |
| AX2  | 1 Master axis, AX1                                          | 1 Master axis, AX1                                           | 0 Master and slave                                    |
| AX3  | 1 Master axis, AX1                                          | 0 No reference axis                                           | 3 Not input                                           |
| AX4  | 1 Master axis, AX1                                          | 3 2. Slave axis, AX3                                         | 0 Master and slave                                    |

Figure 11-4  Example 1: Torque equalization control in pairs
Example 2: Dynamic coupling with 1x4 and 2x2 axes grouped in pairs

Assumption regarding machine axes:
- 1. up to the 4th machine axis: AX1, AX2, AX3, AX4

Dynamic coupling for all slave axes:
- MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX2] = 0
- MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX3] = 0
- MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX4] = 0

For a dynamic coupling, the master axis specified when defining the coupling (MASLDEF) is, as standard, the reference axis for speed setpoint coupling and torque equalization control.

The definition of reference axis AX3 for the torque equalization control of the 3rd slave axis AX4 is required for the application "1x4 axes" where the following machine data is set:
MD37253 $MA_MS_FUNCTION_MASK[AX4], Bit 1 = 1 (see part program)

| Axis | Reference axis of the speed setpoint coupling ($MD37250 = value) | Reference axis of the torque equalization controller ($MD37252 = value) | Input of the torque equalization contr. ($MD37254 = value) |
|------|---------------------------------------------------------------|---------------------------------------------------------------|---------------------------------------------------------------|
| AX1  | 0 No reference axis                                           | 0 No reference axis                                           | 0 Master and slave                                           |
| AX2  | 0 No reference axis                                           | 0 No reference axis                                           | 0 Master and slave                                           |
| AX3  | 0 No reference axis                                           | 0 No reference axis                                           | 3 Machine axis AX3                                           |
| AX4  | 0 No reference axis                                           | 3 Machine axis AX3                                           | 0 Master and slave                                           |

Program code

```
PROC MASL_SWITCH IPRTLOCK DISPLOF
IF MASL_REQUEST==4 ; Activate "1x4 axes"
MASLOF(AX2, AX4) ; Open coupling
MASLDEL(AX2, AX4) ; Delete coupling
$MA_MS_FUNCTION_MASK[AX4]= MD37253, Bit1 = 1: ; Reference axis of the torque equalization controller
$MA_MS_FUNCTION_MASK[AX4] B_OR 'B10' ; Reference axis of the torque equalization controller
from AX4, is AX3 (MD37252)
NEWCONF ; Activate machine data change
STOPRE
MASLDEF(AX2, AX3, AX4, AX1) ; Define assignment for "1x4 axes"
MASLON(AX2, AX3, AX4) ; Close coupling
MASL_ACTIVE=4 MASL_REQUEST=0 ; Checkback signal: "1x4 axes" switched-in
ENDIF
IF MASL_REQUEST==2 ; Activate "2x2 axes"
MASLOF(AX2, AX3, AX4) ; Open coupling
MASLDEL(AX2, AX3, AX4) ; Delete coupling
$MA_MS_FUNCTION_MASK[AX4]= ; MD37253, Bit1 = 0:
```
### 11.5 Tension torque

| Program code | Comment |
|--------------|---------|
| $MA_MS_FUNCTION_MASK[AX4] \_B\_AND \_"HFFFD" ; | ; Reference axis of the torque equalization controller; from AX4, is the master axis defined with MASLDEF |
| NEWCONF | |
| STOPRE | |
| MASLDEF(AX2, AX1) ; | ; Define assignment for the 1st "2x2 axes" |
| MASLDEF(AX4, AX3) ; | ; Define assignment for the 2nd "2x2 axes" |
| MASLON(AX2, AX4) ; | ; Close coupling |
| MASL_ACTIVE=2 MASL_REQUEST=0 ; | ; Checkback signal: "2x2 axes" closed |
| ENDIF | |
| RET | |

---

**Figure 11-5** Example 2: alternating coupling with 1x4 and 2x2 axes
11.6 Closing/opening a coupling

Default

After the control has booted, the following machine data defines whether the coupling is permanently closed (static) or can be dynamically closed/opened and reconfigured:

MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[<slave axis>] = <close mode>

Statically closing a coupling

MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[<slave axis>] = 1

After the control boots, the coupling is statically closed.

By writing the machine data in the part program or synchronized action with subsequent warm restart NEWCONF, then a static coupling can also be opened and closed at a later time.

Example: Opening and closing the static coupling for slave axis AX2

| Program code | Comment |
|--------------|---------|
| N100 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX2] = 0 ; | Coupling type: static -> dynamic |
| N110 NEWCONF ; | Activating machine data |
| N120 MASLOF(AX2) ; | Open coupling |
| .... | |
| N200 PRESETON(AX2,...) ; | e.g. Actual value setting of the slave axis |
| .... | ; |
| N300 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX2] = 1 ; | Coupling type: dynamic -> static |
| ; and close coupling |

Note

A statically closed coupling can neither be closed/opened nor reconfigured using the master-slave-specific NC/PLC interface signals and/or program commands.

Dynamically closing/opening a coupling

MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[<slave axis>] = 0

The coupling can be dynamically closed and opened and reconfigured.

The coupling can be closed/opened by:

- Machine data MD37262
  Writing the machine data to close (= 1) or open (= 0) in the part program or synchronized action. The change becomes active immediately.
NC/PLC interface
DB31, ... DBX24.7 = 1 or 0 (master-slave, closed or open)

Program commands
- Close: MASLON (<slave axis_1>, <slave axis_2>, ...)
- Open: MASLOF (<slave axis_1>, <slave axis_2>, ...)
- Opening with braking of the slave spindles:
  MASLOFS (<slave spindle_1>, <slave spindle_2>, ...)
- Opening and deleting the assignment:
  MASLDEL (<slave axis_1>, <slave axis_2>, ...)

Note
Changing the static assignment
The setting:
MD37262 $MA_MS COUPLING ALWAYS_ACTIVE[<slave axis>] = 0
Permits the coupling to be dynamically changed. The means that it is possible to define a new assignment using MASLDEF. The assignment parameterized in the machine data (see static assignment, Section "Configuring a coupling (Page 501)") takes effect only after the control has been rebooted (NCK reset).

Coupling state

System variable
The actual coupling state of a slave axis can be read in the part program and synchronized action using the following system variable:
$AA_MASL_STAT[<slave axis>]

| Value | Description                                      |
|-------|--------------------------------------------------|
| 0     | 1) The coupling of the slave axis is not active.  |
|       | 2) The specific axis is not a slave axis         |
| > 0   | The coupling is active. <value> == Machine axis number of the master axis |

NC/PLC interface signal
The current coupling state of a slave axis can be read using the following axis-specific NC/PLC interface signal:
DB31, ... DBX96.7 (master/slave coupling active)

Setpoint state
Requests to close/open a master-slave coupling (NC/PLC interface signal and/or program commands) that consecutively follow one another mutually overwrite one another. The effective setpoint state results from the last chronological request (see Section "Response on activation/deactivation (Page 513)").
11.7 Response on activation/deactivation

On/off at standstill (axis)

A request to close/open the coupling only becomes effective when the master and slave axis are at a standstill (zero speed):

DB31, ... DBX61.4 == 1 (axis/spindle stationary)

In this case, the coupled axes must be in the closed-loop controlled mode.

![Activation procedure diagram]

Figure 11-6 Activation procedure

When closing the coupling using the program command MASLON, the system waits until the coupling is closed before the block is changed. The "Master-slave switchover active" message will be displayed on the user interface during this time.

Activation/deactivation during motion (spindle)

**Note**

**Activation/deactivation during motion**

During the motion, the coupling can be switched on or off only for spindles in speed control mode.
Switch on

For switch on during the motion, the coupling operation divides itself at different speeds in two phases.

- Phase 1
  
  The PLC user program must request the switch on of the coupling with:

  DB31, ... DBX24.7 = 1 (master/slave ON)

  The slave spindle accelerates or brakes along a ramp to the setpoint speed of the master spindle.

  When the setpoint speed is reached, the coupling is closed and the NC/PLC interface signal is set:

  DB31, ... DBX96.7 == 1 (coupling active)

  If the master spindle is accelerated while the coupling is being closed, then Phase 1 is extended corresponding to the dynamic difference between the master and the slave spindle.

- Phase 2

  The following synchronous signals are generated from the actual speed difference between the master and slave spindle(s):

  - DB31, ... DBX96.3 (speed tolerance, coarse)
  - DB31, ... DBX96.2 (speed tolerance, fine)

  The associated limits are set using the following machine data:

  - MD37270 $MA_MS_VELO_TOL_COARSE ("Tolerance coarse")
  - MD37272 $MA_MS_VELO_TOL_FINE ("Tolerance fine").

---

**Note**

The "Speed tolerance coarse" signal can be used to implement a monitoring function on the PLC side that checks a coupled master-slave group for loss of speed synchronism.

The "Speed tolerance fine" signal can be used to directly derive the time taken to close the coupling mechanically and activate the torque equalization controller.
Switch off without braking

If the coupling is switched off with the `MASLOF` program command, the coupling will be opened immediately for spindles in speed control mode. The slave spindles maintain their speeds at the time that the coupling is opened until a new speed is programmed.

Switch off with braking

If the coupling is switched off with the `MASLOFS` program command, the coupling will be opened immediately for spindles in speed control mode and the slave spindles braked.

Note

The implicit preprocessor stop is omitted for `MASLON` and `MASLOF`. The missing preprocessor stop means that the `$P` system variables of the slave spindles do not supply updated values until reprogrammed.
Coupling characteristics

The behavior regarding the program commands $\text{MASLON, MASLOF, MASLOFS, MASLDEL}$ and the NC/PLC interface signal: DB31, ... DBX24.7 (master/slave close) is, for spindles in the open-loop speed controlled mode, set using the following machine data:

$$\text{MD37263 $\text{SMA_MS_SPIND COUPLING_MODE[<slave spindle>] = <value>}$}$$

| <value> | Description |
|--------|-------------|
| 0      | Coupling and disconnection take place only at standstill. The current coupling state is retained until all axes involved have actually come to a standstill. The $\text{MASLOFS}$ and $\text{MASLOF}$ operations are identical; the slave spindle is not decelerated automatically. |
| 1      | Coupling and disconnection takes place immediately and therefore during motion. During coupling, the slave spindles are accelerated automatically to the current speed of the master spindle. On disconnection, the slave spindles rotating at this time retain their speeds until next speed programming. However, a slave spindle disconnected with $\text{MASLOFS}$ decelerates automatically. |
11.8 Supplementary conditions

11.8.1 Speed/torque coupling (SW 6 and higher)

General

- Master and slave axes must be on the same NCU.
- A coupling is closed or opened independent of the channel state the next time that the axis is at a standstill.
- Slave axes can only be traversed via the master axis when the coupling is closed.
- A coupled slave axis cannot be switched further via the axis container.
- A requested axis interchange of coupled slave axes is not executed.
- When the coupling is switched-in (closed) via the slave axis, the master axis is braked automatically (if the channel axis is in the same channel). This produces an asymmetric behavior when the coupling is switched-in and switched-out. In contrast to switching-in (closing), there is no automatic braking when switching-out (opening).
- The setpoint position of a coupled slave axis corresponds to its actual position.
- For axes and spindles in the positioning mode, the coupling is only switched-in (closed) and switched-out (opened) at standstill (zero speed).
- The master-slave coupling must be deactivated prior to a gear change or a star-delta switchover.

Axial monitoring functions

- Except monitoring the speed setpoint and actual speed, axial monitoring functions like contour and standstill in the slave axis are inactive because of the missing position controller. The position control circuit parameters like gain factor, precontrol, balancing can thus be set differently for master and slave axes without the monitoring functions responding.
- In order to ensure the same braking behavior for all axes of a master-slave group, when the coupling is active, the same alarm reaction is applied to all of the axes of the master-slave group.
- When canceling error states, the slave axes do not reposition to the point of interruption.

Virtual axis

Master and/or slave axes must not be "virtual" axes:

MD30132 $MA_IS_VIRTUAL_AX (axis is virtual axis)
11.8 Supplementary conditions

Modulo rotary axes

- For slave axes, when the coupling is closed, the actual value displayed in the service display is not modulo 360°. Independent of the setting in machine data: MD30310 $MA_ROT_IS_MODULO[<slave axis>]

Spindles

- If a master-slave coupling is activated with spindles, the slave spindle is operated in the open-loop speed controlled mode. The actual value of the slave spindle is not displayed as modulo 360° in the service display. However, in the automatic basic display the actual value is displayed as modulo 360°.
- If the spindles are accelerated at the current limit, this may mean that no adjusting reserves are left over in the coupled state for the torque compensatory controller to use to distribute the torque between the master and the slave as required.
- The maximum master spindle chuck speed must be configured in the following machine data to be lower than or equal to that of the slave spindles: MD35100 $MA_SPIND_VELO_LIMIT[<master spindle>]
- The axial velocity monitoring function should be adapted to the chuck speed: MD36200 $MA_AX_VELO_LIMIT[<master spindle>]

11.8.2 Axial NC/PLC interface signals

- In the brake control logic, the NC/PLC interface signal may longer be evaluated for slave axes: DB3x.DBX61.5 (position controller active). When the coupling is closed, the signal is no longer set. Instead, the following NC/PLC interface signal should be used: DB31, ... DBX96.7 (master-slave coupling active).
- When requested, and for a closed coupling, the following NC/PLC interface signals are directly taken from the master axis for the slave axis:
  DB31, ... DBX2.1 (controller enable)
  DB31, ... DBX21.7 (pulse enable)
- When the controller enable is cancelled for the master axis:
  DB31, ... DBX2.1 (controller enable) = 0 within the configured time of:
  MD36610 $MA_AX_EMERGENCY_STOP_TIME
  then this also results in interpolatory braking of the slave axis. The corresponding speed and current controller enable signals are only cancelled after the configured time has expired
  MD36620 $MA_SERVO_DISABLE_DELAY_TIME
- In order to facilitate a standard braking behavior for all coupled axes, it is recommended that the following parameters are set the same.
  - MD36620 $MA_SERVO_DISABLE_DELAY_TIME
  - p9560 (pulse suppression, shutdown speed)
  - p1228 (pulse suppression, delay time)
11.8 Supplementary conditions

- If, for the master or slave axis, one of the following drive status signals is not set:
  - DB31, ... DBX61.7 (current controller active) == 0 OR
  - DB31, ... DBX61.6 (speed controller active) == 0
then, when the slave axis is at a standstill, the status signal is reset:
  - DB31, ... DBX96.7 (master/slave active) = 0
As soon as the master and slave axis are again in closed-loop control:
  - DB31, ... DBX61.7 (current controller active) == 1 AND
  - DB31, ... DBX61.6 (speed controller active) == 1
then for the slave axis, the status signal is again set:
  - DB31, ... DBX96.7 (master/slave active) = 1

- The torque equalization controller can be activated via the NC/PLC interface using:
  - DB31, ... DBX24.4 = 1 (torque equalization controller on)
The status of the torque equalization controller is displayed in:
  - DB31, ... DBX96.4 (master/slave equalization controller active)

Note
The slave axis is only closed-loop speed controlled when the coupling is closed. The NC/PLC interface signal: DB31, ... DBX61.5 (position controller active) is no longer set.

11.8.3 Interaction with other functions

Function generator
To calibrate the speed control circuit for a closed master-slave coupling, slave axis MD37268 $MA_MS_TORQUE_WEIGHT_SLAVE should be set to a smaller value. Traversing of a coupled-motion slave axis is not prevented by the torque compensatory controller in this case.

Reference point approach
If the coupling is closed, only the master axis can be referenced. Referencing of slave axes is suppressed. The referencing requirement does not have to be explicitly canceled for the slave axis in order to do this. The referencing status of coupled slave axes remains unchanged. The slave axis position is generally not the same as the master axis position. This difference in position is not significant. If the coupling is not closed, each axis can be referenced separately as usual.

Compensations
Position offsets of the slave axis, such as spindle pitch errors, backlash, temperature and sag offsets are computed but not active because there is no position controller.
Correct calculation of the backlash compensation requires that the backlash of the slave axis is always overtraveled by the motion of the master axis in coupled mode. Disconnecting the coupling during an axis reversal error will generate an incorrect actual value for the slave axis.
Dynamic stiffness control

The servo gain factor of the master axis is copied to the slave axis for an existing coupling and is thus also active in the slave drive. This is an attempt to achieve the same control response in the drive of the master and slave axis as far as possible. MD32640 $MA_STIFFNESS_CONTROL_ENABLE must be configured identically in all coupled axes.

Speed/torque feedforward control

The feedforward control in the slave axis does not have to be activated explicitly. The current settings of the master axis apply. The speed feedforward value of the master axis is already incorporated in the speed setpoint of the slave axis. For an active torque feedforward control, the load-side torque feedforward control value of the master axis is also active in the slave drive.

The mechanical situation changes in coupled mode. Axial settings must be adjusted accordingly. All coupled drives should have the same speed control dynamics.

Gantry

If one master-slave relationship is defined on each side of the gantry grouping to increase the gain, only the leading axis or following axis may be operated as master axis.

Moving to fixed end stop

The travel to fixed stop function can be programmed only in the master axis when a coupling is active and has a different effect on the master and slave axes.

- The programmed value is expressed as a percentage of the rated drive torque of the master axis. The master axis detects when the fixed stop has been reached.
- The programmed value is also active on the slave axis, but refers to the drive torque of the slave axis.

If the rated torque values of the master and slave axes are different, they can be adapted to each other through the following slave axis machine data:
MD37014 $MA_FIXED_STOP_TORQUE_FACTOR
Specifying a factor < 1 reduces the programmed clamping torque in the slave axis.

Please note the following supplementary conditions:
- Torque distribution between the master and slave axes is not possible during clamping as the torque compensatory controller is deactivated during clamping operations.
- Status changes to the master-slave coupling have no effect during travel to fixed stop. Specification of a new status is only accepted when the fixed stop function has been completed.

Safety Integrated (new 840D sl)

As the slave axis is traversed via the master axis speed setpoint, the axial setpoint limit MD36933 $MA_SAFE_DES_VELO_LIMIT is inactive in the coupled slave axes. All safety monitoring functions remain active in the slave axes however.
Gear stage change with active master-slave coupling

An automatic gear stage change in a coupled slave spindle is not possible and can only be implemented indirectly using the master spindle. The point in time at which the gear stage is changed is then derived from the master spindle. The oscillating motion of the coupled slave spindle is generated implicitly via the oscillating motion of the master spindle.

In contrast to the master spindle, the associated parameter block must be explicitly selected in the coupled slave spindle. To enable the parameter block to be specified, the following machine data must be set to the value 2:

MD35590 $MA_PARAMSET_CHANGE_ENABLE (parameter set change possible)

In the event of a gear stage change for the master/slave spindle, the associated parameter set index can be activated by the PLC via the VDI interface.

Note
For more information about gear stage change and parameter sets for changes in spindle mode, see:

References:
Function Manual, Basic Functions; Spindles (S1)

Axis container

If a coupled slave axis is configured in an axis container, alarm "4025 Switch axis container %3 not permitted: Master-slave active channel %1 Axis %2" is output. The axis container may not be advanced because the coupling is active.

In the event that masters change, dynamic configuration can be used to make the relevant spindle the master spindle following a rotation of the axis container. Both master and slave spindles can be container spindles.

For a coupling to be closed after container rotation using a different spindle in each case, the old coupling must be disconnected before the rotation, the configuration deleted and the new coupling closed after the rotation.

Example for cyclic coupling sequence (position=3/container=CT1)

| Program code                     | Comment                                      |
|----------------------------------|----------------------------------------------|
| MASLDEF(AUX, SPI(3))             | ; S3 master for AUX                          |
| MASLON(AUX)                      | ; Close coupling for AUX                      |
| M3=3 S3=4000                    | ; Machining ...                              |
| MASLDEL(AUX)                     | ; Clear configuration and release coupling    |
| AXCTSWE(CT1)                     | ; Container rotation                         |
**11.8 Supplementary conditions**

**Hardware and software limit switches**

Crossing of hardware and software limit switches is detected in coupled axes; in the coupled state, the software limit switch is generally crossed on slave axes. The alarm is output on the slave axis, while braking is initiated via the master axis.

The path traveled after detection of the slave software limit switch equals the distance required by the master axis to brake the coupling.

The master axis controls the movement away from the limit switch, since the coupling cannot be disconnected until the cause of the alarm has been eliminated.
Block search

Static coupling
The "block search with calculation" function (SERUPRO) can be used without any restrictions in conjunction with a static master-slave coupling.

Dynamic coupling
For a dynamic coupling, regarding the program commands MASLON and MASLOF, the following restrictions must be observed:

- The coupled axes must be on the same NCU.
- The coupled axes must be in the same channel when the block search is executed.
- After the block search has been completed, the coupling state, the associated axis positions and speeds must be subsequently influenced by the user via the system ASUB "PROGEVENT.SPF". System variables are available for this purpose.
  - $P_PROG_EVENT: This variable provides information about the event which activated the subprogram. A value of 5 stands for block search.
  - $P_SEARCH_MASLC[following axis identifier]: The variable stands for alteration of the coupling status during a block search.
  - $P_SEARCH_MASLD[following axis identifier]: This variable indicates the positional offset calculated in the block search between the slave and master axes at the instant the coupling was closed.
  - $AA_MASL_STAT[following axis identifier]: This variable indicates the current coupling state.
- The system ASUB "PROGEVENT.SPF" must be saved under the following path: /_N_CMA_DIR/_N_PROG_EVENT_SPF
- The following machine data should be parameterized so that PROGEVENT.SPF is started.
  
  NC-specific machine data:
  - MD11450 $MN_SEARCH_RUN_MODE = 'H02'
  - MD11602 $MN_ASUP_START_MASK = 'H01'
  - MD11604 $MN_ASUP_START_PRIO_LEVEL = 100

Channel-specific machine data:
  - MD20105 $MC_PROG_EVENT_IGN_REFP_LOCK = 'H3F'
### 11.8 Supplementary conditions

#### TE3: Speed/torque coupling, master-slave - 840D sl only

| Table 11-1 | PROGEVENT.SPF: Example 1 |
|-------------|--------------------------|
| Program code | Comment |
| N10 IF $P_PROG_EVENT==5 | ; Block search active |
| N20 IF (($P_SEARCH_MASLC[Y]<0) AND ($AA_MASL_STAT[Y]<0)) | ; In the block search, the coupling state has changed AND the actual state is "coupled" |
| N30 MASLOF(Y) | ; Open coupling |
| N40 SUPEA Y=$AA_IM[X]-$P_SEARCH_MASLD[Y] | ; Move through the position offset using the slave axis |
| N50 MASLON(Y) | ; Close coupling |
| N60 ENDF |
| N70 ENDF |
| N80 REPOSA |

#### Table 11-2 | PROGEVENT.SPF: Example 2 |
|-------------|--------------------------|
| Program code | Comment |
| N10 IF $P_PROG_EVENT==5 | ; Block search active |
| N20 IF (($P_SEARCH_MASLC[SPI(2)]<0) AND ($AA_MASL_STAT[SPI(2)]=0)) | ; In the block search, the coupling state of the second spindle has changed AND the actual state is "separated" |
| N30 M2=$P_SEARCH_SDIR[2] | ; Update direction of rotation |
| N40 S2=$P_SEARCH_S[2] | ; Update speed |
| N50 ENDF |
| N60 ENDF |
| N70 REPOSA |

For more application examples, see Section “Examples (Page 525).

#### Note

For an active coupling, it is recommended to only use block search type 5, "Block search via program test" (SERUPRO) for a block search.

More information about event-driven program calls and "block search using the program test" (SERUPRO) is included in:

#### References:

- Function Manual, Basic Functions;
- Auxiliary function outputs to PLC (H2)
- Mode group, channel, program operation, reset response (K1)
- Spindles (S1)
11.9  Examples

11.9.1  Master-slave coupling between AX1=Master and AX2=Slave.

Configuration

Master-slave coupling between AX1=Master and AX2=Slave.
1. Machine axis number of master axis for speed setpoint coupling
   MD37250 $MA_MS_ASSIGN_MASTER_SPEED_CMD[AX2] = 1
2. Master axis with torque distribution identical to master axis with speed setpoint coupling
   MD37252 $MA_MS_ASSIGN_MASTER_TORQUE_CTR[AX2] = 0
3. Permanent coupling
   MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX2] = 1
4. Torque is injected in both the master and slave axes
   MD37254 $MA_MS_TORQUE_CTRL_MODE[AX2] = 0
5. Torque distribution between the master and slave axes is 50% to 50%
   MD37268 $MA_MS_TORQUE_WEIGHT_SLAVE[AX2] = 50
6. Parameters of the torque compensatory controller
   MD37256 $MA_MS_TORQUE_CTRL_P_GAIN[AX2] = 0.5
   MD37258 $MA_MS_TORQUE_CTRL_I_TIME[AX2] = 5.0

11.9.2  Close coupling via the PLC

This application allows you to close or separate a master-slave coupling between the machine axes AX1=Master axis and AX2=Slave axis during operation.

Preconditions

- One configured master axis
  MD37250 $MA_MS_ASSIGN_MASTER_SPEED_CMD ≠ 0
- Activation of a master-slave coupling via
  MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE=0
- The coupling is open.
### Typical sequence of operations

| Action                              | Effect/comment                                      |
|-------------------------------------|-----------------------------------------------------|
| • Approach coupling position        | Each axis moves to the coupling position.           |
| • Close coupling mechanically       | Both axes are mechanically coupled to one another.  |
| • Request to close the coupling     | PLC interface signal "Master/slave on" DB32, ... DBX24.7 is set. |
| • Read back coupling state          | When the axis is at a standstill, the coupled slave axis sets PLC interface signal "Master/slave active" DB32, ... DBX96.7 and clears "Position controller active" DB32, ... DBX61.5. Wait for checkback signal. |
| • Moving the master-slave group     | The master axis is moved.                           |

#### 11.9.3 Close/separate coupling via part program

This application allows you to close or separate a master-slave coupling between the machine axes AX1=Master axis and AX2=Slave via the part program.

**Preconditions**

- A configured master axis MD37250 0 0.
- Not a static master-slave coupling: MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE= 0
- The coupling is open.

**Parts program**

| Program code | Comment                                      |
|--------------|----------------------------------------------|
| N10 G0 AX1=0 AX2=0 | ; Approach the coupling position on an axis-for-axis basis |
| N20 MASLON (AX2)  | ; (mechanically connect the axes)            |
| N30 AX1=100     | ; Close the coupling.                        |
| N40 MASLOF (AX2) | ; Move the master-slave group using the master axis. |
| N50 AX1=200 AX2=200 | ; Open the coupling.                       |
| N60 M30         | ; (Mechanically separate the axes)           |
|                | ; Move the axes separately.                 |
|                | N60 M30                                      |
11.9.4 Release the mechanical brake

This application allows implementation of a brake control for machine axes AX1=Master axis and AX2=Slave axis in a master-slave coupling.

Preconditions

- Master-slave coupling is configured.
- Axes are stationary.
- No servo enable signals.

Typical sequence of operations

| Action                        | Effect/comment                                                                                                                                 |
|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| Request to close the coupling | The following PLC interface signal is set: DB31, ..., DBX24.7 (Master/Slave ON)                                                                |
| Set controller enable         | PLC interface signal "Servo enable" DB31, ... DBX2.1 is set for both axes.                                                                     |
| Interpreting the feedback     | Link PLC interface signals of the master axis using AND:
                               | DB31, ..., DBX61.7 (current controller active)
                               | DB31, ..., DBX61.6 (speed controller active)
                               | DB31, ..., DBX61.5 (position controller active)
                               | Link PLC interface signals of the saster axis using AND:
                               | DB31, ..., DBX61.7 (current controller active)
                               | DB31, ..., DBX61.6 (speed controller active)
                               | DB31, ..., DBX96.7 (master/slave active)                                                                                                     |
| Release brakes                | If the result of the AND operations on the master and slave axes is ≠ 0, the brake may be released.                                             |
11.10 Data lists

11.10.1 Machine data

11.10.1.1 Axis/spindlespecific machine data

| Number | Identifier: $MA_ | Description                                      |
|--------|------------------|--------------------------------------------------|
| 37250  | MS_ASSIGN_MASTER_SPEED_CMD | Leading axis for speed setpoint coupling          |
| 37252  | MS_ASSIGN_MASTER_TORQUE_CTR | Leading axis for torque distribution              |
| 37254  | MS TORQUE_CTRL_MODE | Connection of torque control output               |
| 37255  | MS TORQUE_CTRL_ACTIVATION | Activate torque compensatory control             |
| 37256  | MS TORQUE_CTRL_P_GAIN | Gain factor of torque compensatory controller     |
| 37258  | MS TORQUE_CTRL_I_TIME | Reset time for torque compensatory controller    |
| 37260  | MS_MAX_CTRL_VELO | Limitation of torque compensatory control         |
| 37262  | MS COUPLING_ALWAYS_ACTIVE | Permanent master-slave coupling                   |
| 37263  | MS SPIND COUPLING_MODE | Coupling characteristics of a spindle            |
| 37264  | MS TENSION TORQUE | Master-slave tension torque                      |
| 37268  | MS TORQUE WEIGHT_SLAVE | Torque weighting of the slave axis               |
| 37270  | MS VELO_TOL_COARSE | Master-slave velocity tolerance "coarse"         |
| 37272  | MS VELO_TOL_FINE | Master-slave velocity tolerance "fine"           |
| 37274  | MS MOTION_DIR_REVERSE | Invert master-slave direction of travel         |

11.10.2 System variables

After a block search, the coupling status and associated axis positions can be adjusted subsequently by means of a system ASUB (asynchronous subroutine) "PROGEVENT.SPF". System variables $P_SEARCH_MASL_C, $P_SEARCH_MASL_D and $AA_MASL_STAT are available for this purpose; they can be used to alter the positional offset between the coupled axes and the coupling status:

| Identifier | Meaning                                                                                                                   |
|-----------|---------------------------------------------------------------------------------------------------------------------------|
| $P SEARCH_MASLC[slave axis identifier] | This variable registers a change in the coupling state during the SERUPRO block search.                               |
| $P SEARCH_MASLD[slave axis identifier] | This variable indicates the positional offset between the slave and master axes at the instant the coupling was closed.   |
| $AA_MASL_STAT[slave axis identifier] | This variable indicates the current coupling state. Value ≠ 0: "Master-slave coupling active"                           |
|                                                      | In this case, it contains the current machine number of the master axis and, if the NCU link is active (several operating panel fronts and NCUs), also the NCU No. at the hundreds position. |
|                                                      | Example: 201 for Axis 1 on NCU2.                                                                                                         |
### 11.10.3 Signals

#### 11.10.3.1 Signals to axis/spindle

| DB number | Byte.bit | Description                                      |
|-----------|----------|--------------------------------------------------|
| 31, ...   | 24.4     | Activate torque compensatory controller          |
| 31, ...   | 24.7     | Activate master-slave coupling                   |

#### 11.10.3.2 Signals from axis/spindle

| DB number | Byte.bit | Description                          |
|-----------|----------|--------------------------------------|
| 31, ...   | 96.2     | Differential speed “Fine”            |
| 31, ...   | 96.3     | Differential speed “Coarse”          |
| 31, ...   | 96.4     | State of torque compensatory controller |
| 31, ...   | 96.7     | State of master-slave coupling       |
TE3: Speed/torque coupling, master-slave - 840D sl only

11.10 Data lists
12.1 Brief description

Functionality

The handling transformation package has been designed for use on **manipulators** and **robots**. The package is a type of modular system, which enables the customer to configure the transformation for his machine by setting machine data (provided that the relevant kinematics are included in the handling transformation package).

Section structure

The "Handling transformation package" section deals with the following topics:

- The "Kinematic transformation" section describes the kinematic transformation environment.
- The "Definition of terms" section explains some basic terms.
- The "Configuration of the kinematic transformation" section explains the machine data required for the configuration.
- The "Kinematic descriptions" section uses configuring examples to illustrate the most commonly used 2-axis to 5-axis kinematics that can be configured with the "Handling transformation package".
- The "Tool orientation" to "Tool programming" sections handle the programming, describing orientation programming, the entry of tool parameters and transformation calls.

Abbreviations

| Abbreviation | Description                                      |
|--------------|--------------------------------------------------|
| FL           | Flange coordinate system                         |
| HP           | Wrist point coordinate system                    |
| IRO          | Internal robot coordinate system (= basic coordinate system) |
| p₃, q₃, r₃   | Coordinates of the last basic axis               |
| RO           | Robot or base center point coordinate system     |
| WS           | Workpiece coordinate system                      |
| T            | Tool coordinate system                            |
| X₃, Y₃, Z₃   | Coordinates of the first wrist axis              |
12.2 Kinematic transformation

Task of a transformation

The purpose of a transformation is to transform movements in the tool tip, which are programmed in a Cartesian coordinate system, into machine axis positions.

Fields of application

The handling transformation package described here has been designed to cover the largest possible number of kinematic transformations implemented solely via parameter settings in machine data. Currently, kinematics with two to maximum six axes to be included in the transformation can be configured with this package, corresponding to up to five spatial degrees of freedom. In this case, a maximum of 3 degrees are available for translation and 3 degrees for orientation. This package thus allows a tool (milling cutter, laser beam) to be oriented by a 5-axis machine in any desired relation to the workpiece in every point of the machining space. The workpiece is always programmed in the rectangular workpiece coordinate system; any programmed or set frames rotate and shift this system in relation to the basic system. The kinematic transformation then converts this information into motion instructions for the real machine axes. The kinematic transformation requires information about the design (kinematics) of the machine, which are stored in machine data.

Kinematic categories

The handling transformation package is divided into two categories of kinematics, which can be selected via machine data:

MD62600 $MC_TRAFO6_KINCLASS (kinematic category)
- STANDARD: This category includes the most commonly used kinematics.
- SPECIAL: Special kinematics
12.3 Definition of terms

12.3.1 Units and directions

Lengths and angles
In the transformation machine data, all lengths are specified in millimeters or inches and, unless otherwise stated, all angles in degrees at intervals of \([-180^\circ, 180^\circ]\).

Direction of rotation
In the case of angles, arrows in the drawings always indicate the mathematically positive direction of rotation.

12.3.2 Definition of positions and orientations using frames
In order to make a clear distinction from the term "frame" as it is used in the NC language, the following description explains the meaning of the term "frame" in relation to the handling transformation package.

Frame
A frame can be used to translate one coordinate system into another. In this respect, a distinction must be made between translation and rotation. Translation only effects an offset between the coordinate system and the reference system, while rotation actually rotates the coordinate system in relation to the reference. Coordinates X, Y and Z are used to describe the translation. They are defined so as to result in a legal coordinate system.

Translation
The translation is always specified in relation to the coordinate directions of the initial system. These directions are assigned to machine data as follows:

- X direction: \(\ldots\_\text{POS}[0]\)
- Y direction: \(\ldots\_\text{POS}[1]\)
- Z direction: \(\ldots\_\text{POS}[2]\)
12.3 Definition of terms

**Rotation**

The rotation is described by the RPY angles A, B and C (RPY stands for Roll Pitch Yaw). The positive direction of rotation is defined by the right hand rule, i.e. if the thumb on the right hand is pointing in the direction of the axis of rotation, then the fingers are pointing in the positive angular direction. In this respect, it must be noted that A and C are defined at intervals \([-180; +180]\) and B at intervals \([-90; +90]\).

The definitions of the RPY angles are as follows:

- **Angle A**: 1. rotation about the Z axis of the initial system
- **B angle**: 2. Rotation through the rotated Y axis
- **C angle**: 3. rotation about the twice rotated X axis

The RPY angles are assigned to machine data as follows:

- **Angle A**: \(..._\text{RPY}[0]\)
- **B angle**: \(..._\text{RPY}[1]\)
- **C angle**: \(..._\text{RPY}[2]\)

An example of the rotation of the RPY angles is specified in Fig. "Example of rotation through RPY angles".

In this example, the initial coordinate system X1, Y1, Z1 is first rotated through angle A about axis Z1, then through angle B about axis Y2 and finally through angle C about axis X3.

![Example of rotation through RPY angles](image-url)
12.3.3 Definition of a joint

Meaning

The term joint describes either a translational or a rotary axis. The basic axis identifiers result from the arrangement and sequence of the individual joints. These are described by identifying letters (S, C, R, N), which are explained below.

| Letter | Description                  | Diagram |
|--------|------------------------------|---------|
| S      | Sliding joint                | ![S Diagram](image) |
| C      | Sliding joint II Rotary joint| ![C Diagram](image) |
| R      | Rotary joint II Rotary joint| ![R Diagram](image) |
| N      | Rotary joint Rotary joint   | ![N Diagram](image) |
| FL     | Flange for mounting tool     | ![FL Diagram](image) |
| WZ     | Tool                         | ![WZ Diagram](image) |

Positive axis direction

Positive axis direction into drawing into

Figure 12-2 Joint identifying letters
12.4 Configuration of a kinematic transformation

Meaning
In order to ensure that the kinematic transformation can convert the programmed values into axis motions, it must have access to some information about the mechanical construction of the machine. This information is stored in machine data:

- Axis assignments
- Geometry information

12.4.1 General machine data

MD24100 $MC_TRAFO_TYPE_1 (definition of transformation 1 in the channel)
The value 4100 must be entered in this data for the handling transformation package.

MD24110 $MC_TRAFO_AXES_IN_1 (axis assignment for transformation)
The axis assignment at the transformation input defines which transformation axis is mapped internally onto a channel axis. It is specified in machine data:

```
MD24110 $MC_TRAFO_AXES_IN_1
```

There is a predetermined axis sequence for the handling transformation package. The basic axes are entered in the first three components (index 0…2). The upper three components (index 3…5) are assigned to the manual axes.

- MD24110 $MC_TRAFO_AXES_IN_1[0] = 1 ; 1. Basic axis
- MD24110 $MC_TRAFO_AXES_IN_1[1] = 2 ; 2. Basic axis
- MD24110 $MC_TRAFO_AXES_IN_1[2] = 3 ; 3. Basic axis
- MD24110 $MC_TRAFO_AXES_IN_1[3] = 4 ; 1. Manual axis
- MD24110 $MC_TRAFO_AXES_IN_1[4] = 5 ; 2. Manual axis
- MD24110 $MC_TRAFO_AXES_IN_1[5] = 6 ; 3. Manual axis
MD24120 $MC.TRAFO_GEOAX_ASSIGN_TAB_1 (assignment between geometry axis and channel axis for transformation 1)

The translational degrees of freedom that are available for the transformation are entered via machine data:

MD24120 $MC.TRAFO_GEOAX_ASSIGN_TAB_1

The 3 geometry axes normally correspond to Cartesian axis directions X, Y and Z.

In this case, the first three channel axis numbers must be transferred from MD24110 $MC.TRAFO_AXES_IN_1 to MD24120 $MC.TRAFO_GEO_AX_ASSIGN_TAB_1.

- MD24120 $MC.TRAFO_GEO_AX_ASSIGN_TAB_1[0] = 1
- MD24120 $MC.TRAFO_GEO_AX_ASSIGN_TAB_1[1] = 2
- MD24120 $MC.TRAFO_GEO_AX_ASSIGN_TAB_1[2] = 3

12.4.2 Parameterization using geometry data

Modular principle

The machine geometry is parameterized according to a type of modular principle. With this method, the machine is successively configured in geometry parameters from its base center point to the tool tip, thereby producing a closed kinematic loop. In this case, frames (see Section "Definition of positions and orientations using frames (Page 533") describe the geometry. While the control is powering up, the configuration machine data is checked and alarms generated when necessary.

All axes in the mode group follow-up (are tracked), the alarms can only be reset by a POWER ON.

As shown in Fig. "Closed kinematic loop illustrated by the example of a robot", the kinematic transformation effects a conversion of the tool operating point (tool coordinate system: XWZ, YWZ, ZWZ), that is specified in relation to the basic coordinate system (BCS = robot coordinate system: XRO, YRO, ZRO), in machine axis values (MCS positions: A1, A2, A3, ...). The operating point (XWZ, YWZ, ZWZ) is specified in the part program in relation to the workpiece to be machined (workpiece coordinate system WCS: XWS, YWS, ZWS). The programmable frames make it possible to create an offset between the workpiece coordinate system WCS and the basic coordinate system BCS.
12.4 Configuration of a kinematic transformation

Figure 12-3  Closed kinematic loop illustrated by the example of a robot

**Note**
For more detailed information about coordinate systems, please see:

**References:**
Programming Manual Fundamentals

The following machine data is available for configuring kinematic transformations:

**MD62612, MD62613**

The frame T_IRO_RO links the base center point of the machine (BCS = RO) with the first internal coordinate system (IRO) determined by the transformation.

MD62613 $MC_TTRAFO6_TIRORO_RPY (frame between base center point and internal coordinate system (rotation component), n = 0...2)

MD62612 $MC_TTRAFO6_TIRORO_POS (frame between base center point and internal coordinate system (position component), n = 0...2)

**MD62603**

The type of basic axis arrangement is specified in machine data:

MD62603 $MC_TTRAFO6_MAIN_AXES (basic axis identifier)

The basic axes are generally the first 3 axes to be included in the transformation.
12.4 Configuration of a kinematic transformation

The basic axis lengths A and B are specified with machine data:

MD62607 $MC\_TRAFO6\_MAIN\_LENGTH\_AB (basic axis lengths A and B, \( n = 0...1 \))

As Fig. "Overview of basic axis configurations" illustrates, these are specially defined for each type of basic axis.

Whether the 4th axis is mounted parallel, anti-parallel or perpendicular to the last rotary basic axis is specified in machine data:

MD62606 $MC\_TRAFO6\_A4PAR (axis 4 is parallel/anti-parallel to last basic axis)

Frame T_X3_P3 links the last coordinate system of the basic axes with the first hand coordinate system.

MD62608 $MC\_TRAFO6\_TX3P3\_POS (attachment of hand (position component), \( n = 0...2 \))

MD62609 $MC\_TRAFO6\_TX3P3\_RPY (attachment of hand (rotation component), \( n = 0...2 \))

These parameters describe the hand geometry.

MD62604 $MC\_TRAFO6\_WRIST\_AXES (wrist axis identifier)

MD62616 $MC\_TRAFO6\_DHPAR4\_5.. (Parameter ALPHA for configuring the hand, \( n = 0...1 \))

The hand type is specified in machine data:

MD62604 $MC\_TRAFO6\_WRIST\_AXES (wrist axis identifier)

The term wrist axes generally refers to axes four to six.

Frame T_FL_WP links the last hand coordinate system with the flange coordinate system.

MD62610 $MC\_TRAFO6\_TFLWP\_POS (frame between wrist point and flange coordinate system (position component), \( n = 0...2 \))

MD62611 $MC\_TRAFO6\_TFLWP\_RPY (frame between wrist point and flange coordinate system (rotation component), \( n = 0...2 \))

This data is explained in more detail in the following sections.
Basic axes included in every transformation

MD62603

The first 3 axes included in the transformation are generally referred to as the "basic axes". They must always be mutually parallel or perpendicular. Each of the following basic axis arrangements has its own special identifier (see Section "Definition of a joint (Page 535)"). The basic axis identifier is specified via machine data:

MD62603 $MC_TRAFO6_MAIN_AXES (basic axis identifier)

Figure 12-4  Overview of basic axis configuration
The handling transformation package contains the following basic axis kinematics:

- **SS**: MD62603 = 1, gantry (3 linear axes, rectangular)
- **CC**: MD62603 = 2, SCARA (1 linear axis, 2 rotary axes (in parallel))
- **SC**: MD62603 = 4, SCARA (2 linear axes, 1 rotary axis (swivel axis))
- **CS**: MD62603 = 6, SCARA (2 linear axes, 1 rotary axis (spin axis))
- **NR**: MD62603 = 3, articulated arm (3 rotary axes (2 axes in parallel))
- **NN**: MD62603 = 7, articulated arm (3 rotary axes)
- **RR**: MD62603 = 5, articulated arm (1 linear axis, 2 rotary axes (vertical))

**Wrist axes included in every transformation**

**MD62804**

The fourth axis and all further axes are generally referred to as "wrist axes". The handling transformation package can only identify hands with rotary axes. The wrist axis identifier for three-axis hands is entered in machine data:

MD62804 $MC_TRAFO6_WRIST_AXES (wrist axis identifier)

In the case of hands with fewer than three axes, the identifier for a beveled hand with elbow or a central hand is entered in machine data:

MD62804 $MC_TRAFO6_WRIST_AXES (wrist axis identifier)

The current software supports only wrist axis types beveled hand with elbow or central hand.

![Figure 12-5 Overview of wrist axis configuration](image-url)
Parameterization of wrist axes

MD62614 - MD62616

Hands are parameterized via parameter:

- MD62614 $MC\_TRAFO6\_DHPAR4\_5A$ (parameter A for configuring the hand, n = 0...1)
- MD62615 $MC\_TRAFO6\_DHPAR4\_5D$ (parameter D for configuring the hand, n = 0...1)
- MD62616 $MC\_TRAFO6\_DHPAR4\_5ALPHA$ (parameter ALPHA for configuring the hand, n = 0...1)

This data is special types of frame which describe the relative positions of the coordinate systems in the hand. For this purpose, the machine data corresponds to certain frame components (see Section "Definition of positions and orientations using frames (Page 533)"):

- MD62614 $MC\_TRAFO6\_DHPAR4\_5A$ to the ..._POS[0] (x component)
- MD62615 $MC\_TRAFO6\_DHPAR4\_5D$ dem ..._POS[2] (z component)
- MD62616 $MC\_TRAFO6\_DHPAR4\_5ALPHA$ dem ..._RPY[2] (C angle)

The other components of the frame are then zero.

Central hand (CH)

On a central hand, all wrist axes intersect at one point. All parameters must be set as shown in Table "Configuring data for a central hand".

![Central hand diagram](image)

**Figure 12-6 Central hand**

**Table 12-1 Configuring data for a central hand**

| Machine data                        | Value          |
|-------------------------------------|----------------|
| MD62604 $MC\_TRAFO6\_WRIST\_AXES   | 2              |
| MD62614 $MC\_TRAFO6\_DHPAR4\_5A   | [0.0, 0.0]     |
| MD62615 $MC\_TRAFO6\_DHPAR4\_5D   | [0.0, 0.0]     |
| MD62616 $MC\_TRAFO6\_DHPAR4\_5ALPHA | [-90.0, 90.0]  |
Beveled hand with elbow (BHE)

The beveled hand with elbow differs from the central hand in two respects, i.e. the axes do not intersect nor are they mutually perpendicular. Parameters \(a_4\), \(d_5\), and \(a_4\) are available for this type of hand, as shown in Table "Configuring data for a central hand".

![Figure 12-7 Beveled hand with elbow](image)

Table 12-2 Configuring data for a beveled hand with elbow (5-axis)

| Machine data                        | Value                      |
|-------------------------------------|----------------------------|
| MD62604 $MC\_TRAFO6\_WRIST\_AXES   | 6                          |
| MD62614 $MC\_TRAFO6\_DHPAR4\_5A   | \([a_4, 0.0]\)             |
| MD62615 $MC\_TRAFO6\_DHPAR4\_5D   | \([0.0, d_5]\)             |
| MD62616 $MC\_TRAFO6\_DHPAR4\_5ALPHA | \([a_4, 0.0]\)          |

![Figure 12-8 Link frames](image)
12.4 Configuration of a kinematic transformation

**T_IRO_RO**

Frame T_IRO_RO provides the link between the base center point coordinate system (RO) defined by the user and the internal robot coordinate system (IRO). The internal robot coordinate system is predefined in the handling transformation package for each basic axis type and included in the kinematic diagrams for the basic axis arrangements. The base center point system is in the Cartesian zero point of the machine, corresponding to the basic coordinate system. If no FRAMES are programmed, the basic coordinate system equals the workpiece coordinate system.

**Note**

For more detailed information about FRAMES, please see:

**References:**

Programming Manual Fundamentals

Frame T_IRO_RO is not subject to any restrictions for 5-axis kinematics.

The following restrictions apply in relation to 4-axis kinematics:

- The first rotary axis must always be parallel/anti-parallel to one of the coordinate axes of the base center point coordinate system (RO).
- No further restrictions apply to type SS basic axes.
- In the case of type CC, CS or SC basic axes, no further restrictions apply provided that the 4th axis is parallel to the last rotary basic axis.
- With respect to all other basic axes, and basic axes of type CC, CS or SC if the 4th axis is perpendicular to the last rotary basic axis, the Z axis of RO must be parallel to the Z axis of IRO.

**T_X3_P3**

Frame T_X3_P3 describes the method used to attach the hand to the basic axes. Frame T_X3_P3 is used to link the coordinate system of the last basic axis (p3_q3_r3 coordinate system) with the coordinate system of the first wrist axis (x3_y3_z3 coordinate system). The p3_q3_r3 coordinate system is included in the kinematic diagrams for the basic axis arrangements.

The z3 axis is always on the 4th axis.

Depending on the number of axes to be included in the transformation, frame T_X3_P3 is subject to certain restrictions relating to the hand and basic axes:

- For 5-axis kinematics, frame T_X3_P3 can be freely selected in the following cases:
  - If the basic axes are of the SS type.
  - If the basic axes are of the CC, CS or SC type, the transformation must either include a central hand (ZEH) or the 4th axis must be positioned in parallel to the last rotary basic axis.
If the basic axes are of the NR or RR type, the transformation must either include a central hand (ZEH) or the 4th axis must be positioned in parallel to the last basic rotary axis and an X flange must intersect the 5th axis.

If the basic axes are of the NN type, the transformation must include a central hand.

- With 4-axis kinematics, the z3 axis must always be parallel/anti-parallel or perpendicular to the last basic axis.

**T_FL_WP**

Frame T_FL_WP links the flange with the last internal coordinate system (hand-point coordinate system) preset by the handling transformation package.

For kinematics with less than 6 axes, this frame is subject to certain restrictions. These restrictions are explained by the relevant kinematics.

**Other configuring data**

**Number of transformed axes**

The number of axes included in the transformation is defined in machine data:

MD62605 $MC_TRAFO6_NUM_AXES (number of transformed axes)

The number of transformed axes can currently lie between two and six axes.

**Changing the axis sequence**

MD62620

**Note**

With certain types of kinematics, it is possible to transpose axes without changing the behavior of the kinematic transformation. Machine data:

MD62620 $MC_TRAFO6_AXIS_SEQ (rearrangement of axes)

The axes on the machine are numbered consecutively from 1 to 6 and must be entered in the internal sequence in machine data:

MD62620 $MC_TRAFO6_AXIS_SEQ[0] ...[4]

All other axis-specific machine data refer to the sequence of axes on the machine.

| Basic axis kinematics | Options for changing axis sequence |
|-----------------------|-----------------------------------|
| SS, CC                | Any                               |
| SC                   | 1 and 2                           |
| CS                   | 2 and 3                           |
Example 1

This example involves two kinematics such as those illustrated in Fig. "Rearrangement of axes (example 1). Kinematic 1 is directly included in the handling transformation package. It corresponds to a CC kinematic with a wrist axis parallel to the last basic rotary axis.

Kinematic 2 is the same as kinematic 1 inasmuch as it is irrelevant for the resulting robot movement whether the translational axis is axis 1 or axis 4. In this instance, the data for kinematic 2 must be entered as follows in machine data:

MD62620 $MC_TRAFO6_AXIS_SEQ (rearrangement of axes)

The input is as follows:

MD62620 $MC_TRAFO6_AXIS_SEQ[ 0 ] = 4
MD62620 $MC_TRAFO6_AXIS_SEQ[ 1 ] = 1
MD62620 $MC_TRAFO6_AXIS_SEQ[ 2 ] = 2
MD62620 $MC_TRAFO6_AXIS_SEQ[ 3 ] = 3
Example 2

This example involves a SCARA kinematic transformation as illustrated in Fig. "Rearrangement of axes (example 2)" , in which the axes can be freely transposed. Kinematic 1 is directly included in the handling transformation package. It corresponds to a CC kinematic. As regards the transposition of axes, it is irrelevant how many wrist axes are involved in the transformation.

![Diagram of SCARA kinematic transformation](image)

Figure 12-10 Rearrangement of axes 2

**Changing the directions of axes**

**MD62618**

A rotational or offset direction is preset for each axis in the handling transformation package. This direction is not necessarily the same as the corresponding direction on the machine. The following machine data must be set to -1 for the relevant axis if the direction is to be inverted, or otherwise to +1:

MD62618 $MC.TRAFO6_AXES_DIR[ ] (matching of physical and mathematical directions of rotation [axis no.]: 0...5)

**Adapting the zero points of the axes**

**MD62617**

The mathematical zero points of axes are preset in the handling transformation package. However, the mathematical zero point does not always correspond to the mechanical zero point (calibration point) of axes. In order for the zero point of each axis to fit one another, the difference between the mathematical zero point and the alignment point for each axis must be entered in the following machine data:

MD62617 $MC.TRAFO6_MAMES[ ] (offset between mathematical and mechanical zero points [axis no.]: 0...5)

The deviation to be entered corresponds to the difference between the mechanical zero point and the mathematically positive direction of rotation of the axis.
Example

The example (Fig. "Matching mathematical and mechanical zero points") shows an articulated arm kinematic. The mathematical zero point for axis 2 is 90°. This value must be set for axis 2 in machine data:

MD62617 $MC\_TRAFO6\_MAMES[1]$ (offset between mathematical and mechanical zero points)

Axis 3 is counted relative to axis 2 and therefore has a value of -90° as a mathematical zero point.

Figure 12-11 Matching mathematical and mechanical zero points

Axis types

MD62601

Which axis type is handled is specified in machine data:

MD62601 $MC\_TRAFO6\_AXES\_TYPE$ (axis type for transformation [axis no.]: 0...5)

The transformation package distinguishes between the following axis types:

- Linear axis: MD62601 = 1
- Rotary axis: MD62601 = 3

Velocities and acceleration rates

Separate velocities for the Cartesian movement components are introduced for axes that are traversed with $G00$ and active transformation.

The velocity is preset via path feed F for axis traversal with $G01$ or $G02$. 
MD62629

The velocities for individual translational motion directions for axis traversal with \texttt{G00} can be preset in machine data:

\texttt{MD62629 \$MC\_TRAFO6\_VELCP[i]} (Cartesian velocity \texttt{[no.: 0...2]})

- Index \(i = 0\) : X component of basic system
- Index \(i = 1\) : Y component of basic system
- Index \(i = 2\) : Z component of basic system

MD62630

The acceleration rates for individual translational motion directions for axis traversal with \texttt{G00} can be preset in machine data:

\texttt{MD62630 \$MC\_TRAFO6\_ACCCP[i]} (Cartesian acceleration rates \texttt{[no.: 0...2]})

- Index \(i = 0\) : X component of basic system
- Index \(i = 1\) : Y component of basic system
- Index \(i = 2\) : Z component of basic system

MD62631

The velocities for individual directions of orientation for axis traversal with \texttt{G00} can be preset in machine data:

\texttt{MD62631 \$MC\_TRAFO6\_VELORI[i]} (orientation angle velocities \texttt{[no.: 0...2]})

- Index \(i = 0\) : A angle
- Index \(i = 1\) : B angle
- Index \(i = 2\) : C angle

MD62632

The acceleration rates for individual directions of orientation for axis traversal with \texttt{G00} can be preset in machine data:

\texttt{MD62632 \$MC\_TRAFO6\_ACCORI[i]} (orientation angle acceleration rates \texttt{[no.: 0...2]})

- Index \(i = 0\) : A angle
- Index \(i = 1\) : B angle
- Index \(i = 2\) : C angle

MD62634

A reduction factor for the velocity controller in the JOG mode can be specified with the machine data:

\texttt{MD62633 \$MC\_ROBX\_DYN\_LIM\_REDUCE}

MD62635

A time constant for the PT1 filter of the speed controller can be specified with the machine data:

\texttt{MD62634 \$MC\_ROBX\_VEL\_FILTER\_TIME}
12.5 Descriptions of kinematics

The following descriptions of kinematics for transformations involving 2 to 5 axes explain the general configuring procedure first before describing how the machine data need to be configured, using a configuring example for each kinematic type. These examples do not include all possible lengths and offsets. The direction data refer to the positive directions of traversal and rotation for the transformation. The axis positions correspond to their zero position for the relevant transformation.

12.5.1 3-axis kinematics

3-axis kinematics normally possess 3 translational degrees of freedom, but do not have a degree of freedom for orientation. In other words, they only include basic axes.

Configuration

The procedure for configuring a 3-axis kinematic is as follows:
1. Enter "Standard" kinematic category in machine data:
   MD62600 $MC_TRAFO6_KINCLASS (kinematic category)
2. Set the number of axes for transformation in machine data:
   MD62605 $MC_TRAFO6_NUM_AXES = 3 (number of transformed axes)
3. Compare the basic axes with the basic axes contained in the handling transformation package. → Enter the basic axis identifier in machine data:
   MD62603 $MC_TRAFO6_MAIN_AXES (basic axis identifier)
4. If the axis sequence is not the same as the normal axis sequence, it must be corrected in machine data:
   MD62620 $MC_TRAFO6_AXIS_SEQ (rearrangement of axes)
5. As identifier for the wrist axes, the following machine data must be set to 1 (no hand):
   MD62604 $MC_TRAFO6_WRIST_AXES = 1 (wrist axis identifier)
6. Enter the axis types for the transformation in machine data:
   MD62601 $MC_TRAFO6_AXES_TYPE (axis type for transformation)
7. Compare the directions of rotation of axes with the directions defined in the handling transformation package and correct in machine data:
   MD62618 $MC_TRAFO6_AXES_DIR (matching of physical and mathematical directions of rotation)
8. Enter the mechanical zero offset in machine data:
   MD62617 $MC_TRAFO6_MAMES (offset between mathematical and mechanical zero points)
9. Enter the basic axis lengths in machine data:
   MD62607 $MC_TRAFO6_MAIN_LENGTH_AB (basic axis lengths A and B)

10. Define frame T_IRO_RO and enter the offset in machine data:
    MD62612 $MC_TRAFO6_TIRORO_POS (frame between base center point and internal system (position component))
    Enter the rotation in machine data:
    MD62613 $MC_TRAFO6_TIRORO_RPY (frame between base center point and internal system (rotation component))

11. Determine the flange coordinate system. For this purpose, the p3_q3_r3 coordinate system must be regarded as the initial system. The offset is entered in machine data:
    MD62610 $MC_TRAFO6_TFLWP_POS (frame between wrist point and flange (position component))
    The rotation is entered in machine data:
    MD62611 $MC_TRAFO6_TFLWP_RPY (frame between wrist point and flange (rotation component))

**SCARA kinematics**

SCARA kinematics are characterized by the fact that they possess both translational and rotary axes. The basic axes are divided into 3 categories depending on how they are mutually positioned.

- CC types
- CS types
- SC types (cf. Fig. "Overview of basic axis configurations")
3-axis CC kinematics

![Diagram of 3-axis CC kinematics]

Table 12-4 Configuration data for 3-axis CC kinematics

| Machine data                        | Value                          |
|-------------------------------------|-------------------------------|
| MD62600 $MC\_TRAFO6\_KINCLASS      | 1                             |
| MD62605 $MC\_TRAFO6\_NUM\_AXES     | 3                             |
| MD62603 $MC\_TRAFO6\_MAIN\_AXES    | 2                             |
| MD62604 $MC\_TRAFO6\_WRIST\_AXES   | 1                             |
| MD62601 $MC\_TRAFO6\_AXES\_TYPE    | \([3, 1, 3, \ldots]\)         |
| MD62620 $MC\_TRAFO6\_AXIS\_SEQ     | \([2, 1, 3, 4, 5, 6]\)        |
| MD62618 $MC\_TRAFO6\_AXES\_DIR     | \([1, 1, 1, 1, 1]\)           |
| MD62617 $MC\_TRAFO6\_MAMES         | \([0.0, 0.0, 0.0, 0.0, 0.0, 0.0]\) |
| MD62607 $MC\_TRAFO6\_MAIN\_LENGTH\_AB | \([0.0, 300.0]\)       |
| MD62612 $MC\_TRAFO6\_TIRORO\_POS  | \([0.0, 0.0, 500.0]\)         |
| MD62613 $MC\_TRAFO6\_TIRORO\_RPY  | \([0.0, 0.0, 90.0]\)          |
| MD62608 $MC\_TRAFO6\_TX3P3\_POS   | \([0.0, 0.0, 0.0]\)           |
| MD62609 $MC\_TRAFO6\_TX3P3\_RPY   | \([0.0, 0.0, 0.0]\)           |
| MD62610 $MC\_TRAFO6\_TFLWP\_POS   | \([200.0, 0.0, 0.0]\)         |
| MD62611 $MC\_TRAFO6\_TFLWP\_RPY   | \([0.0, 0.0, -90.0]\)         |
3-axis SC kinematics

Table 12-5 Configuration data for 3-axis SC kinematics

| Machine data                          | Value                  |
|---------------------------------------|------------------------|
| MD62600 $MC_TRAFO6_KINCLASS           | 1                      |
| MD62605 $MC_TRAFO6_NUM_AXES           | 3                      |
| MD62603 $MC_TRAFO6_MAIN_AXES          | 4                      |
| MD62604 $MC_TRAFO6_WRIST_AXES         | 1                      |
| MD62601 $MC_TRAFO6_AXES_TYPE          | [1, 1, 3, …]           |
| MD62620 $MC_TRAFO6_AXIS_SEQ           | [1, 2, 3, 4, 5, 6]     |
| MD62618 $MC_TRAFO6_AXES_DIR           | [1, 1, 1, 1, 1, 1]     |
| MD62617 $MC_TRAFO6_MAMES              | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD62607 $MC_TRAFO6_MAIN_LENGTH_AB     | [500.0, 0.0]           |
| MD62612 $MC_TRAFO6_TIRORO_POS         | [0.0, 0.0, 500.0]      |
| MD62613 $MC_TRAFO6_TIRORO_RPY         | [0.0, 0.0, 0.0]        |
| MD62608 $MC_TRAFO6_TX3P3_POS          | [0.0, 0.0, 0.0]        |
| MD62609 $MC_TRAFO6_TX3P3_RPY          | [0.0, 0.0, 0.0]        |
| MD62610 $MC_TRAFO6_TFLWP_POS          | [300.0, 0.0, 0.0]      |
| MD62611 $MC_TRAFO6_TFLWP_RPY          | [0.0, 0.0, 0.0]        |
3-axis CS kinematic

![3-axis CS kinematic diagram]

Table 12-6  Configuration data for 3-axis CS kinematics

| Machine data                        | Value       |
|-------------------------------------|-------------|
| MD62600 $MC\_TRAFO6\_KINCLASS      | 1           |
| MD62605 $MC\_TRAFO6\_NUM\_AXES      | 3           |
| MD62603 $MC\_TRAFO6\_MAIN\_AXES     | 6           |
| MD62604 $MC\_TRAFO6\_WRIST\_AXES   | 1           |
| MD62601 $MC\_TRAFO6\_AXES\_TYPE    | [3, 1, 1, ...] |
| MD62620 $MC\_TRAFO6\_AXIS\_SEQ     | [1, 2, 3, 4, 5, 6] |
| MD62618 $MC\_TRAFO6\_AXES\_DIR     | [1, 1, 1, 1, 1, 1] |
| MD62617 $MC\_TRAFO6\_MAMES         | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD62607 $MC\_TRAFO6\_MAIN\_LENGTH\_AB | [500.0, 0.0] |
| MD62612 $MC\_TRAFO6\_TIOROR\_POS   | [0.0, 0.0, 500.0] |
| MD62613 $MC\_TRAFO6\_TIOROR\_RPY   | [0.0, 0.0, 0.0] |
| MD62608 $MC\_TRAFO6\_TX3P3\_POS    | [0.0, 0.0, 0.0] |
| MD62609 $MC\_TRAFO6\_TX3P3\_RPY    | [0.0, 0.0, 0.0] |
| MD62610 $MC\_TRAFO6\_TFLWP\_POS    | [300.0, 0.0, 0.0] |
| MD62611 $MC\_TRAFO6\_TFLWP\_RPY    | [0.0, 0.0, 0.0] |
Articulated-arm kinematics

3-axis NR kinematics

![3-axis NR kinematics diagram](image)

Figure 12-15 3-axis NR kinematics

Table 12-7 Configuration data 3-axis NR kinematic

| Machine data                        | Value                                      |
|-------------------------------------|--------------------------------------------|
| MD62600 $MC\_TRAFO6\_KINCLASS       | 1                                          |
| MD62605 $MC\_TRAFO6\_NUM\_AXES       | 3                                          |
| MD62603 $MC\_TRAFO6\_MAIN\_AXES      | 3                                          |
| MD62604 $MC\_TRAFO6\_WRIST\_AXES     | 1                                          |
| MD62601 $MC\_TRAFO6\_AXES\_TYPE     | [3, 3, 3, ...]                              |
| MD62620 $MC\_TRAFO6\_AXIS\_SEQ      | [1, 2, 3, 4, 5, 6]                          |
| MD62618 $MC\_TRAFO6\_AXES\_DIR       | [1, 1, 1, 1, 1, 1]                          |
| MD62617 $MC\_TRAFO6\_MAMES          | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]              |
| MD62607 $MC\_TRAFO6\_MAIN\_LENGTH\_AB| [300.0, 500.0]                              |
| MD62612 $MC\_TRAFO6\_TIOROR\_POS     | [0.0, 0.0, 500.0]                           |
| MD62613 $MC\_TRAFO6\_TIOROR\_RPY     | [0.0, 0.0, 0.0]                             |
| MD62608 $MC\_TRAFO6\_TX3P3\_POS     | [0.0, 0.0, 0.0]                             |
| MD62609 $MC\_TRAFO6\_TX3P3\_RPY     | [0.0, 0.0, 0.0]                             |
| MD62610 $MC\_TRAFO6\_TFLWP\_POS     | [300.0, 0.0, 0.0]                           |
| MD62611 $MC\_TRAFO6\_TFLWP\_RPY     | [0.0, 0.0, 0.0]                             |
3-axis RR kinematics

![3-axis RR kinematics diagram](image)

Figure 12-16 3-axis RR kinematics

Table 12-8 Configuration data for 3-axis RR kinematics

| Machine data                              | Value                                      |
|-------------------------------------------|--------------------------------------------|
| MD62600 $MC\_TRAFO6\_KINCLASS            | 1                                          |
| MD62605 $MC\_TRAFO6\_NUM\_AXES           | 3                                          |
| MD62603 $MC\_TRAFO6\_MAIN\_AXES          | 5                                          |
| MD62604 $MC\_TRAFO6\_WRIST\_AXES         | 1                                          |
| MD62601 $MC\_TRAFO6\_AXES\_TYPE         | [3, 1, 3, ...]                             |
| MD62620 $MC\_TRAFO6\_AXES\_SEQ          | [1, 2, 3, 4, 5, 6]                         |
| MD62618 $MC\_TRAFO6\_AXES\_DIR          | [1, 1, 1, 1, 1, 1]                         |
| MD62617 $MC\_TRAFO6\_MAMES              | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]        |
| MD62607 $MC\_TRAFO6\_MAIN\_LENGTH\_AB   | [300.0, 0.0]                               |
| MD62612 $MC\_TRAFO6\_TIRORO\_POS        | [0.0, 0.0, 300.0]                          |
| MD62613 $MC\_TRAFO6\_TIRORO\_RPY        | [0.0, 0.0, 0.0]                            |
| MD62608 $MC\_TRAFO6\_TX3P3\_POS         | [0.0, 0.0, 0.0]                            |
| MD62609 $MC\_TRAFO6\_TX3P3\_RPY         | [0.0, 0.0, 0.0]                            |
| MD62610 $MC\_TRAFO6\_TFLWP\_POS         | [200.0, 0.0, 0.0]                          |
| MD62611 $MC\_TRAFO6\_TFLWP\_RPY         | [0.0, 0.0, 0.0]                            |
3-axis NN kinematics

![Diagram of 3-axis NN kinematics]

**Figure 12-17 3-axis NN kinematics**

**Table 12-9 Configuration data for 3-axis NN kinematics**

| Machine data                        | Value                           |
|-------------------------------------|---------------------------------|
| MD62600 $MC\_TRAFO6\_KINCLASS      | 1                               |
| MD62605 $MC\_TRAFO6\_NUM\_AXES     | 3                               |
| MD62603 $MC\_TRAFO6\_MAIN\_AXES    | 7                               |
| MD62604 $MC\_TRAFO6\_WRIST\_AXES   | 1                               |
| MD62601 $MC\_TRAFO6\_AXES\_TYPE    | [3, 3, 3, ...]                   |
| MD62620 $MC\_TRAFO6\_AXIS\_SEQ     | [1, 2, 3, 4, 5, 6]              |
| MD62618 $MC\_TRAFO6\_AXES\_DIR     | [1, 1, 1, 1, 1, 1]              |
| MD62617 $MC\_TRAFO6\_MAMES         | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]   |
| MD62607 $MC\_TRAFO6\_MAIN\_LENGTH\_AB | [300.0, 500.0]               |
| MD62612 $MC\_TRAFO6\_TIORO\_POS    | [0.0, 0.0, 300.0]               |
| MD62613 $MC\_TRAFO6\_TIORO\_RPY    | [0.0, 0.0, 90.0]                |
| MD62608 $MC\_TRAFO6\_TX3P3\_POS    | [0.0, 0.0, 0.0]                 |
| MD62609 $MC\_TRAFO6\_TX3P3\_RPY    | [0.0, 0.0, 0.0]                 |
| MD62610 $MC\_TRAFO6\_TFLWP\_POS   | [400.0, 0.0, 0.0]               |
| MD62611 $MC\_TRAFO6\_TFLWP\_RPY   | [0.0, 0.0, -90.0]               |
12.5.2 4-axis kinematics

4-axis kinematics usually imply 3 translational degrees of freedom and one degree of freedom for orientation.

Restrictions

The following restrictions apply to 4-axis kinematics:

The frame T_FL_WP is subject to the following condition:

- MD62611 $MC\_TRAFO6\_TFL\_WP\_RPY = [0.0, 90.0, 0.0]$ (frame between wrist point and flange (rotation component))
- X flange and X tool must be parallel to the 4th axis.
- Two successive basic axes must be parallel or orthogonal.
- The 4th axis must only be mounted in a parallel or orthogonal way to the last basic axis.

Configuration

The procedure for configuring a 4-axis kinematic is as follows:

1. Enter "Standard" kinematic category in the machine data:
   MD62600 $MC\_TRAFO6\_KINCLASS$ (kinematic category)
2. Set the number of axes for transformation in the machine data:
   MD62605 $MC\_TRAFO6\_NUM\_AXES=4$ (number of transformed axes)
3. Compare the basic axes with the basic axes contained in the handling transformation package.
   - Enter the basic axis identifier in machine data:
     MD62603 $MC\_TRAFO6\_MAIN\_AXES$ (basic axis identifier)
4. If the axis sequence is not the same as the normal axis sequence, it must be corrected in machine data:
   MD62620 $MC\_TRAFO6\_AXIS\_SEQ$ (rearrangement of axes)
5. As identifier for the wrist axes, the following machine data must be set (no hand):
   MD62604 $MC\_TRAFO6\_WRIST\_AXES = 1$ (wrist axis identifier)
6. Whether axis 4 runs parallel/anti-parallel to the last rotary basic axis is entered into the machine data:
   MD62606 $MC\_TRAFO6\_A4PAR$ (axis 4 is parallel/anti-parallel to last basic axis)
7. Enter the axis types for the transformation in machine data:
   MD62601 $MC\_TRAFO6\_AXES\_TYPE$ (axis type for transformation)
8. Compare the directions of rotation of axes with the directions defined in the handling transformation package and correct in the machine data:
   MD62618 $MC_TRAFO6_AXES_DIR (matching of physical and mathematical directions of rotation)

9. Enter the mechanical zero offset in the machine data:
   MD62617 $MC_TRAFO6_MAMES (offset between mathematical and mechanical zero points)

10. Enter the basic axis lengths in the machine data:
    MD62607 $MC_TRAFO6_MAIN_LENGTH_AB (basic axis lengths A and B)

11. Define frame T_IRO_RO and enter the offset in the machine data:
    MD62612 $MC_TRAFO6_TIRORO_POS (frame between base center point and internal system (position component))
    Enter the rotation in the machine data:
    MD62613 $MC_TRAFO6_TIRORO_RPY (frame between base center point and internal system (rotation component))

12. Specification of frame T_X3_P3 to attach hand. For this purpose, the p3_q3_r3 coordinate system must be regarded as the initial system. The offset is entered in the machine data:
    MD62608 $MC_TRAFO6_TX3P3_POS (attachment of hand (position component))
    The rotation is entered in the machine data:
    MD62609 $MC_TRAFO6_TX3P3_RPY (attachment of hand (rotation component))
    is entered.

13. Determine the flange coordinate system. For this purpose, the hand-point coordinate system must be regarded as the initial system. The offset is entered in the machine data:
    MD62610 $MC_TRAFO6_TFLWP_POS (frame between wrist point and flange (position component))
    The rotation is entered in the machine data:
    MD62611 $MC_TRAFO6_TFLWP_RPY (frame between wrist point and flange (rotation component))
SCARA kinematics

4-axis CC kinematics

Figure 12-18 4-axis CC kinematics

Table 12-10 Configuration data for 4-axis CC kinematics

| Machine data                          | Value          |
|---------------------------------------|----------------|
| MD62600 $MC\_TRAFO6\_KINCLASS        | 1              |
| MD62605 $MC\_TRAFO6\_NUM\_AXES        | 4              |
| MD62603 $MC\_TRAFO6\_MAIN\_AXES       | 2              |
| MD62604 $MC\_TRAFO6\_WRIST\_AXES      | 1              |
| MD62606 $MC\_TRAFO6\_A4PAR            | 1              |
| MD62601 $MC\_TRAFO6\_AXES\_TYPE      | [3, 1, 3, 3, ...] |
| MD62620 $MC\_TRAFO6\_AXIS\_SEQ       | [2, 1, 3, 4, 5, 6] |
| MD62618 $MC\_TRAFO6\_AXES\_DIR       | [1, 1, 1, 1, 1] |
| MD62617 $MC\_TRAFO6\_MAMES           | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD62607 $MC\_TRAFO6\_MAIN\_LENGTH\_AB| [0.0, 300.0]   |
| MD62612 $MC\_TRAFO6\_TIORO\_POS      | [0.0, 0.0, 500.0] |
| MD62613 $MC\_TRAFO6\_TIORO\_RPY      | [0.0, 0.0, 90.0] |
| MD62608 $MC\_TRAFO6\_TX3P3\_POS      | [300.0, 0.0, -200.0] |
| MD62609 $MC\_TRAFO6\_TX3P3\_RPY      | [-90.0, 90.0, 0.0] |
| MD62610 $MC\_TRAFO6\_TFLWP\_POS      | [0.0, 0.0, 200.0] |
| MD62611 $MC\_TRAFO6\_TFLWP\_RPY      | [0.0, -90.0, 0.0] |
4-axis SC kinematics

Figure 12-19 4-axis SC kinematics

Table 12-11 Configuration data for 4-axis SC kinematics

| Machine data                   | Value                  |
|--------------------------------|------------------------|
| MD62600 $MC\_TRAFO6\_KINCLASS | 1                      |
| MD62605 $MC\_TRAFO6\_NUM\_AXES | 4                      |
| MD62603 $MC\_TRAFO6\_MAIN\_AXES | 4                      |
| MD62604 $MC\_TRAFO6\_WRIST\_AXES | 1                      |
| MD62606 $MC\_TRAFO6\_A4PAR   | 1                      |
| MD62601 $MC\_TRAFO6\_AXES\_TYPE | [1, 1, 3, 3, ...]      |
| MD62620 $MC\_TRAFO6\_AXIS\_SEQ | [1, 2, 3, 4, 5, 6]     |
| MD62618 $MC\_TRAFO6\_AXES\_DIR | [1, 1, 1, 1, 1, 1]    |
| MD62617 $MC\_TRAFO6\_MAMES    | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD62607 $MC\_TRAFO6\_MAIN\_LENGTH\_AB | [300.0, 0.0] |
| MD62612 $MC\_TRAFO6\_TIRORO\_POS | [0.0, 0.0, 300.0]     |
| MD62613 $MC\_TRAFO6\_TIRORO\_RPY | [0.0, 0.0, 0.0]     |
| MD62608 $MC\_TRAFO6\_TX3P3\_POS | [200.0, 0.0, 0.0]   |
| MD62609 $MC\_TRAFO6\_TX3P3\_RPY | [0.0, 0.0, -90.0]     |
| MD62610 $MC\_TRAFO6\_TFLWP\_POS | [200.0, 0.0, 0.0]   |
| MD62611 $MC\_TRAFO6\_TFLWP\_RPY | [0.0, -90.0, 180.0] |
4-axis CS kinematic

Figure 12-20 4-axis CS kinematic

Table 12-12 Configuration data for 4-axis CS kinematics

| Machine data                      | Value                                      |
|-----------------------------------|--------------------------------------------|
| MD62600 $MC_TRAFO6_KINCLASS       | 1                                          |
| MD62605 $MC_TRAFO6_NUM_AXES       | 4                                          |
| MD62603 $MC_TRAFO6_MAIN_AXES      | 6                                          |
| MD62604 $MC_TRAFO6_WRIST_AXES     | 1                                          |
| MD62606 $MC_TRAFO6_A4PAR          | 1                                          |
| MD62601 $MC_TRAFO6_AXES_TYPE      | [3, 1, 1, 3, ...]                          |
| MD62620 $MC_TRAFO6_AXIS_SEQ       | [1, 2, 3, 4, 5, 6]                         |
| MD62618 $MC_TRAFO6_AXES_DIR       | [1, 1, 1, 1, 1, 1]                         |
| MD62617 $MC_TRAFO6_MAMES          | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]             |
| MD62607 $MC_TRAFO6_MAIN_LENGTH_AB | [400.0, 0.0]                               |
| MD62612 $MC_TRAFO6_TIRORO_POS     | [0.0, 0.0, 400.0]                          |
| MD62613 $MC_TRAFO6_TIRORO_RPY     | [0.0, 0.0, 0.0]                            |
| MD62608 $MC_TRAFO6_TX3P3_POS      | [500.0, 0.0, -200.0]                       |
| MD62609 $MC_TRAFO6_TX3P3_RPY      | [90.0, 0.0, 180.0]                         |
| MD62610 $MC_TRAFO6_TFLWP_POS      | [200.0, 0.0, 0.0]                          |
| MD62611 $MC_TRAFO6_TFLWP_RPY      | [0.0, -90.0, 0.0]                          |
Articulated-arm kinematics

4-axis NR kinematics

Table 12-13 Configuration data 4-axis NR kinematic

| Machine data                  | Value                      |
|-------------------------------|----------------------------|
| MD62600 $MC.TRAFO6_KINCLASS  | 1                          |
| MD62605 $MC.TRAFO6_NUM_AXES   | 4                          |
| MD62603 $MC.TRAFO6_MAIN_AXES  | 3                          |
| MD62604 $MC.TRAFO6_WRIST_AXES | 1                          |
| MD62606 $MC.TRAFO6_A4PAR      | 1                          |
| MD62601 $MC.TRAFO6_AXES_TYPE  | [3, 3, 3, 3, ...]          |
| MD62620 $MC.TRAFO6_AXIS_SEQ   | [1, 2, 3, 4, 5, 6]         |
| MD62618 $MC.TRAFO6_AXES_DIR   | [1, 1, 1, 1, 1, 1]         |
| MD62617 $MC.TRAFO6_MAMES      | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD62607 $MC.TRAFO6_MAIN_LENGTH_AB | [300.0, 300.0]            |
| MD62612 $MC.TRAFO6_TIRORO_POS | [0.0, 0.0, 500.0]          |
| MD62613 $MC.TRAFO6_TIRORO_RPY | [0.0, 0.0, 0.0]            |
| MD62608 $MC.TRAFO6_TX3P3_POS  | [300.0, 0.0, 0.0]          |
| MD62609 $MC.TRAFO6_TX3P3_RPY  | [0.0, 0.0, -90.0]          |
| MD62610 $MC.TRAFO6_TFLWP_POS  | [200.0, 0.0, 0.0]          |
| MD62611 $MC.TRAFO6_TFLWP_RPY  | [0.0, -90.0, 180.0]        |
12.5.3 5-axis kinematics

5-axis kinematics usually imply 3 degrees of freedom for translation and 2 for orientation.

Restrictions

The following restrictions apply to 5-axis kinematics:

1. There are restrictions for the flange coordinate system because the X flange axis must intersect the 5th axis, nevertheless, it must not be parallel to it.

2. The frame T_FL_WP is subject to the following condition as far as 5-axis articulated-arm kinematics are concerned:
   - MD62610 $MC_TRAFO6_TFLWP_POS = [0.0, 0.0, Z]$ (frame between wrist point and flange (position component))
   - MD62611 $MC_TRAFO6_TFLWP_RPY = [A, 0.0, 0.0]$ (frame between wrist point and flange (rotation component))

3. There are restrictions for the tool as far as 5-axis articulated-arm kinematics are concerned:
   - 4. Axis parallel to the 3rd axis: 2-dimensional tool is possible [X, 0.0, Z]
   - 4. Axis perpendicular to the 3rd axis: Only 1-dimensional tool is possible [X, 0.0, 0.0]

4. There are restrictions for the tool as far as 5-axis Scara kinematics are concerned:
   - 4. Axis perpendicular to the 3rd axis: 1-dimensional tool is possible [X, 0.0, 0.0]

5. Two successive basic axes must be parallel or orthogonal.

6. The 4th axis must only be mounted in a parallel or orthogonal way to the last basic axis.

Configuration

The procedure for configuring a 5-axis kinematic is as follows:

1. Enter "Standard" kinematic category in machine data:
   MD62600 $MC_TRAFO6_KINCLASS (kinematic category)

2. Set the number of axes for transformation in the machine data:
   MD62605 $MC_TRAFO6_NUM_AXES = 5 (number of transformed axes)

3. Compare the basic axes with the basic axes contained in the handling transformation package.
   - Enter the basic axis identifier in machine data:
     MD62603 $MC_TRAFO6_MAIN_AXES (basic axis identifier)

4. If the axis sequence is not the same as the normal axis sequence, it must be corrected in machine data:
   MD62620 $MC_TRAFO6_AXIS_SEQ (rearrangement of axes)

5. ID specification for the wrist axes. If axis 4 and 5 intersect, a central hand (ZEH) is present. In all other cases, the ID for beveled hand with elbow (BHE) must be entered in the machine data:
   MD62604 $MC_TRAFO6_WRIST_AXES (wrist axis identifier)
6. Whether axis 4 runs parallel/anti-parallel to the last rotary basic axis must be entered into the machine data:
   MD62606 $MC\_TRAFO6\_A4PAR (axis 4 is parallel/anti-parallel to last basic axis)

7. Enter the axis types for the transformation in machine data:
   MD62601 $MC\_TRAFO6\_AXES\_TYPE (axis type for transformation)

8. Compare the directions of rotation of axes with the directions defined in the handling transformation package and correct in machine data:
   MD62618 $MC\_TRAFO6\_AXES\_DIR (matching of physical and mathematical directions of rotation)

9. Enter the mechanical zero offset in the machine data:
   MD62617 $MC\_TRAFO6\_MAMES (offset between mathematical and mechanical zero points)

10. Enter the basic axis lengths in machine data:
    MD62607 $MC\_TRAFO6\_MAIN\_LENGTH\_AB (basic axis lengths A and B)

11. Define frame T_IRO_RO and enter the offset in machine data:
    MD62612 $MC\_TRAFO6\_TIRORO\_POS (frame between base center point and internal system (position component))
    Define frame T_IRO_RO and enter the rotation in machine data:
    MD62613 $MC\_TRAFO6\_TIRORO\_RPY (frame between base center point and internal system (rotation component))

12. Specification of frame T_X3_P3 to attach hand. The offset is entered in machine data:
    MD62608 $MC\_TRAFO6\_TX3P3\_POS (attachment of hand (position component))
    The rotation is entered in the machine data:
    MD62609 $MC\_TRAFO6\_TX3P3\_RPY (attachment of hand (rotation component))

13. Specification of wrist axes parameters. For this purpose, only the parameters for axis 4 must be entered in the machine data:
    MD62614 $MC\_TRAFO6\_DHPAR4\_5A[0] (parameter A for configuring the hand)
    MD62616 $MC\_TRAFO6\_DHPAR4\_5ALPHA[0] (parameter A for configuring the hand)
    All other parameters must be set to 0.0.

14. Determine the flange coordinate system. For this purpose, the hand-point coordinate system must be regard as the initial system. The offset is entered in the machine data:
    MD62610 $MC\_TRAFO6\_TFLWP\_POS (frame between wrist point and flange (position component))
    The rotation is entered in the machine data:
    MD62611 $MC\_TRAFO6\_TFLWP\_RPY (frame between wrist point and flange (rotation component))
SCARA kinematics

5-axis CC kinematics

![Diagram of 5-axis CC kinematics](image)

Figure 12-22 5-axis CC kinematics

Table 12-14 Configuration data for 5-axis CC kinematics

| Machine data                  | Value                  |
|-------------------------------|------------------------|
| MD62600 $MC\_TRAFO6\_KINCLASS | 1                      |
| MD62605 $MC\_TRAFO6\_NUM\_AXES | 5                      |
| MD62603 $MC\_TRAFO6\_MAIN\_AXES | 2                      |
| MD62604 $MC\_TRAFO6\_WRIST\_AXES | 5                      |
| MD62606 $MC\_TRAFO6\_A4PAR | 1                      |
| MD62601 $MC\_TRAFO6\_AXES\_TYPE | [3, 1, 3, 3, ...]       |
| MD62620 $MC\_TRAFO6\_AXIS\_SEQ | [2, 1, 3, 4, 5, 6]      |
| MD62618 $MC\_TRAFO6\_AXES\_DIR | [1, 1, 1, 1, 1]         |
| MD62617 $MC\_TRAFO6\_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD62607 $MC\_TRAFO6\_MAIN\_LENGTH\_AB | [0.0, 500.0]     |
| MD62612 $MC\_TRAFO6\_TIRORO\_POS | [0.0, 0.0, 500.0]    |
| MD62613 $MC\_TRAFO6\_TIRORO\_RPY | [0.0, 0.0, 90.0]      |
| MD62608 $MC\_TRAFO6\_TX3P3\_POS | [300.0, 0.0, -200.0]  |
| MD62609 $MC\_TRAFO6\_TX3P3\_RPY | [0.0, 0.0, -90.0]     |
| MD62610 $MC\_TRAFO6\_TFLWP\_POS | [200.0, 0.0, 0.0]   |
| MD62611 $MC\_TRAFO6\_TFLWP\_RPY | [0.0, 0.0, 0.0]       |
| MD62614 $MC\_TRAFO6\_DHPAR4\_5A | [200.0, 0.0]        |
| MD62615 $MC\_TRAFO6\_DHPAR4\_5D | [0.0, 0.0]          |
| MD62616 $MC\_TRAFO6\_DHPAR4\_5ALPHA | [-90.0, 0.0]    |
5-axis NR kinematics

![Diagram of 5-axis NR kinematics]

**Table 12-15 Configuration data 5-axis NR kinematic**

| Machine data                        | Value                  |
|-------------------------------------|------------------------|
| MD62600 $MC\_TRAF06\_KINCLASS       | 1                      |
| MD62605 $MC\_TRAF06\_NUM\_AXES      | 5                      |
| MD62603 $MC\_TRAF06\_MAIN\_AXES     | 3                      |
| MD62604 $MC\_TRAF06\_WRIST\_AXES    | 2                      |
| MD62606 $MC\_TRAF06\_A4PAR          | 0                      |
| MD62601 $MC\_TRAF06\_AXES\_TYPE     | [3, 3, 3, 3, 3, ...]   |
| MD62620 $MC\_TRAF06\_AXIS\_SEQ      | [1, 2, 3, 4, 5, 6]     |
| MD62618 $MC\_TRAF06\_AXES\_DIR      | [1, 1, 1, 1, 1, 1]     |
| MD62617 $MC\_TRAF06\_MAMES          | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD62607 $MC\_TRAF06\_MAIN\_LENGTH\_AB | [30.0, 300.0]       |
| MD62612 $MC\_TRAF06\_TIOROR\_POS    | [0.0, 0.0, 500.0]      |
| MD62613 $MC\_TRAF06\_TIOROR\_RPY    | [0.0, 0.0, 0.0]        |
| MD62608 $MC\_TRAF06\_TX3P3\_POS     | [500.0, 0.0, 0.0]      |
| MD62609 $MC\_TRAF06\_TX3P3\_RPY     | [0.0, 90.0, 0.0]       |
| MD62610 $MC\_TRAF06\_TFLWP\_POS     | [0.0, -300.0, 0.0]     |
| MD62611 $MC\_TRAF06\_TFLWP\_RPY     | [-90.0, 0.0, 0.0]      |
| MD62614 $MC\_TRAF06\_DHPAR4\_5A     | [0.0, 0.0]             |
| MD62615 $MC\_TRAF06\_DHPAR4\_5D     | [0.0, 0.0]             |
| MD62616 $MC\_TRAF06\_DHPAR4\_5ALPHA | [-90.0, 0.0]           |
12.5.4 6-axis kinematics

6-axis kinematics usually imply 3 degrees of freedom for translation and 3 more for orientation. This allows for the tool direction to be manipulated freely in space. Also, the tool can be rotated along its own axis to the machining surface or inclined with a tilting angle.

This kinematic type is supported by the software, but only for very specific solutions as the auxiliary method for this handling transformation package. A general, comprehensive and universally applicable software version is still not available for 6-axis kinematics.

12.5.5 Special kinematics

MD62602 $MC_TRAFO6_SPECIAL_KIN (special kinematic type)

Special kinematics are kinematics that are not directly included in the building block system of the Handling transformation package. They are frequently missing a degree of freedom or are characterized by mechanical links between the axes or with the tool. Set the following machine data for these kinematics:

MD62600 $MC_TRAFO6_KINCLASS = 2 (kinematic category)

Which special kinematic is handled is specified in machine data:

MD62602 $MC_TRAFO6_SPECIAL_KIN (special kinematic type)
Special 2-axis SC kinematic

This special kinematic is characterized by the fact that the tool is always maintained in the same orientation via a mechanical linkage. It implies two Cartesian degrees of protection. The identifier for this kinematic is machine data:

MD62602 $MC_TARFO6_SPECIAL_KIN = 3$ (special kinematic type)

![Special 2-axis SC kinematic diagram](image)

Figure 12-24 Special 2-axis SC kinematic

Table 12-16 Configuring data for a special 2-axis SC kinematic

| Machine data                  | Value                      |
|-------------------------------|----------------------------|
| MD62600 $MC_TRAFO6_KINCLASS  | 2                          |
| MD62602 $MC_TRAFO6_SPECIAL_KIN | 3                          |
| MD62605 $MC_TRAFO6_NUM_AXES   | 2                          |
| MD62603 $MC_TRAFO6_MAIN_AXES  | 2                          |
| MD62604 $MC_TRAFO6_WRIST_AXES | 1                          |
| MD62601 $MC_TRAFO6_AXES_TYPE  | [1, 3, 3, ...]             |
| MD62620 $MC_TRAFO6_AXIS_SEQ   | [1, 2, 3, 4, 5, 6]         |
| MD62618 $MC_TRAFO6_AXES_DIR   | [1, 1, 1, 1, 1, 1]         |
| MD62617 $MC_TRAFO6_MAMES      | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD62607 $MC_TRAFO6_MAIN_LENGTH_AB | [400.0, 500.0]     |
| MD62612 $MC_TRAFO6_TIRORO_POS | [0.0, 0.0, 300.0]      |
| MD62613 $MC_TRAFO6_TIRORO_RPY | [0.0, 0.0, 0.0]         |
| MD62608 $MC_TRAFO6_TX3P3_POS  | [0.0, 0.0, 0.0]           |
| MD62609 $MC_TRAFO6_TX3P3_RPY  | [0.0, 0.0, 0.0]           |
| MD62610 $MC_TRAFO6_TFLWP_POS  | [0.0, 0.0, 0.0]           |
| MD62611 $MC_TRAFO6_TFLWP_RPY  | [0.0, 0.0, 0.0]           |
Special 3-axis SC kinematic

The special kinematic has 2 Cartesian degrees of freedom and one degree of freedom for orientation. The identifier for this kinematic is:

\[ \text{MD62602 } \$MC\_TRAFO6\_SPECIAL\_KIN = 4 \text{ (special kinematic type)} \]

![Figure 12-25 Special 3-axis SC kinematic](image)

Table 12-17 Configuring data for a special 3-axis SC kinematic

| Machine data                          | Value                                      |
|---------------------------------------|--------------------------------------------|
| MD62600 $MC\_TRAFO6\_KINCLASS        | 2                                          |
| MD62602 $MC\_TRAFO6\_SPECIAL\_KIN    | 4                                          |
| MD62605 $MC\_TRAFO6\_NUM\_AXES       | 3                                          |
| MD62603 $MC\_TRAFO6\_MAIN\_AXES      | 2                                          |
| MD62604 $MC\_TRAFO6\_WRIST\_AXES     | 1                                          |
| MD62601 $MC\_TRAFO6\_AXES\_TYPE      | \[1, 3, 3, ...\]                           |
| MD62620 $MC\_TRAFO6\_AXIS\_SEQ       | \[1, 2, 3, 4, 5, 6\]                       |
| MD62618 $MC\_TRAFO6\_AXES\_DIR       | \[1, 1, 1, 1, 1\]                          |
| MD62617 $MC\_TRAFO6\_MAMES           | \[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0\]     |
| MD62607 $MC\_TRAFO6\_MAIN\_LENGTH\_AB| \[0.0, 0.0\]                               |
| MD62612 $MC\_TRAFO6\_TIRORO\_POS     | \[0.0, 0.0, 400.0\]                        |
| MD62613 $MC\_TRAFO6\_TIRORO\_RPY     | \[0.0, 0.0, 0.0\]                          |
| MD62608 $MC\_TRAFO6\_TX3P3\_POS      | \[400.0, 0.0, 0.0\]                        |
| MD62609 $MC\_TRAFO6\_TX3P3\_RPY      | \[0.0, 0.0, -90.0\]                        |
| MD62610 $MC\_TRAFO6\_TFLWP\_POS      | \[200.0, 0.0, 0.0\]                        |
| MD62611 $MC\_TRAFO6\_TFLWP\_RPY      | \[0.0, -90.0, 180.0\]                     |
**Special 4-axis SC kinematic**

This special kinematic is characterized by the fact that axis 1 and axis 2 are mechanically coupled. This coupling ensures that axis 2 is maintained at a constant angle when axis 1 is swiveled. This kinematic also guarantees that axes 3 and 4 always remain perpendicular, irrespective of the positions of axes 1 and 2. The identifier for this kinematic is:

MD62602 $MC\_TRAFO6\_SPECIAL\_KIN = 7$ (special kinematic type)

![Figure 12-26 Special 4-axis SC kinematic](image)

| Machine data              | Value                      |
|---------------------------|----------------------------|
| MD62600 $MC\_TRAFO6\_KINCLASS | 2                          |
| MD62602 $MC\_TRAFO6\_SPECIAL\_KIN | 7                          |
| MD62605 $MC\_TRAFO6\_NUM\_AXES | 4                          |
| MD62603 $MC\_TRAFO6\_MAIN\_AXES | 2                          |
| MD62604 $MC\_TRAFO6\_WRIST\_AXES | 1                          |
| MD62601 $MC\_TRAFO6\_AXES\_TYPE     | [3, 3, 1, 3, ...]         |
| MD62620 $MC\_TRAFO6\_AXIS\_SEQ     | [1, 2, 3, 4, 5, 6]        |
| MD62618 $MC\_TRAFO6\_AXES\_DIR     | [1, 1, 1, 1, 1, 1]        |
| MD62617 $MC\_TRAFO6\_MAMES         | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD62607 $MC\_TRAFO6\_MAIN\_LENGTH\_AB | [100.0, 400.0]   |
| MD62612 $MC\_TRAFO6\_TIRORO\_POS   | [100.0, 0.0, 1000.0]      |
| MD62613 $MC\_TRAFO6\_TIRORO\_RPY   | [0.0, 0.0, 0.0]           |
| MD62608 $MC\_TRAFO6\_TX3P3\_POS   | [300.0, 0.0, 0.0]         |
| MD62609 $MC\_TRAFO6\_TX3P3\_RPY   | [0.0, 0.0, 0.0]           |
| MD62610 $MC\_TRAFO6\_TFLWP\_POS   | [0.0, 0.0, -600.0]        |
| MD62611 $MC\_TRAFO6\_TFLWP\_RPY   | [0.0, 90.0, 0.0]          |
Special 2-axis NR kinematics

This special kinematic is characterized by the fact that axis 1 and axis 2 are mechanically coupled. Another special feature is the tool. With this kinematic, it maintains its orientation in space irrespective of the positions of the other axes.

The identifier for this kinematic is:

\[
\text{MD62602 \$MC\_TRAFO6\_SPECIAL\_KIN = 5 (special kinematic type)}
\]

Without a mechanical coupling between axis 1 and 2, it has the following identifier:

\[
\text{MD62602 \$MC\_TRAFO6\_SPECIAL\_KIN = 8 (special kinematic type)}
\]

---

**Figure 12-27** Special 2-axis NR kinematics

**Table 12-19** Configuration data for special 2-axis NR kinematics

| Machine data | Value |
|--------------|-------|
| MD62600 $MC\_TRAFO6\_KINCLASS | 2 |
| MD62602 $MC\_TRAFO6\_SPECIAL\_KIN | 5 (8) |
| MD62605 $MC\_TRAFO6\_NUM\_AXES | 2 |
| MD62603 $MC\_TRAFO6\_MAIN\_AXES | 3 |
| MD62604 $MC\_TRAFO6\_WRIST\_AXES | 1 |
| MD62601 $MC\_TRAFO6\_AXES\_TYPE | [3, 3, ...] |
| MD62620 $MC\_TRAFO6\_AXIS\_SEQ | [1, 2, 3, 4, 5, 6] |
| MD62618 $MC\_TRAFO6\_AXES\_DIR | [1, 1, 1, 1, 1] |
| MD62617 $MC\_TRAFO6\_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD62607 $MC\_TRAFO6\_MAIN\_LENGTH\_AB | [100.0, 400.0] |
| MD62612 $MC\_TRAFO6\_TIRORO\_POS | [100.0, 500.0, 0.0] |
| MD62613 $MC\_TRAFO6\_TIRORO\_RPY | [0.0, 0.0, -90.0] |
| MD62608 $MC\_TRAFO6\_TX3P3\_POS | [400.0, 0.0, 0.0] |
| MD62609 $MC\_TRAFO6\_TX3P3\_RPY | [0.0, 0.0, 0.0] |
| MD62610 $MC\_TRAFO6\_TFLWP\_POS | [0.0, 0.0, 0.0] |
| MD62611 $MC\_TRAFO6\_TFLWP\_RPY | [0.0, 0.0, 0.0] |
12.6 Tool orientation

12.6.1 Tool orientation

Programming

Three possible methods can be used to program the orientation of the tool:
- Directly as "orientation axes" A, B and C in degrees
- via Euler or RPY angles in degrees via A2, B2, C2
- Using direction vectors A3, B3, C3.

The identifiers for Euler angles and direction vectors can be set in machine data:

Euler angle via machine data:
MD10620 $MN_EULER_ANGLE_NAME_TAB (name of Euler angles)

Direction vectors via machine data:
MD10640 $MN_DIR VECTOR_NAME_TAB (name of normal vectors)

The tool orientation can be located in any block. Above all, it can be programmed alone in a block, resulting in a change of orientation in relation to the tool tip which is fixed in its relationship to the workpiece.
12.6 Tool orientation

Euler or RPY

Can be switched between Euler and RPY input via the machine data:

MD21100 $MC_ORIENTATION_IS_EULER (angle definition for orientation programming)

Note

It is not possible to program using Euler angles, RPY angles or direction vectors for kinematics involving fewer than 5 axes. In such cases, only one degree of freedom is available for orientation. This orientation angle can only be programmed with "Orientation axis angle" "A".

Orientation reference

A tool orientation at the start of a block can be transferred to the block end in the workpiece coordinate system only using the ORIWKS command.

ORIWKS command

The tool orientation is programmed in the workpiece coordinate system (WCS) and is thus not dependent on the machine kinematics. In the case of a change in orientation with the tool tip at a fixed point in space, the tool moves along a large arc on the plane stretching from the start vector to the end vector.

ORIMKS command

The tool orientation is programmed in the machine coordinate system and is thus dependent on the machine kinematics. In the case of a change in orientation of a tool tip at a fixed point in space, linear interpolation takes place between the rotary axis positions.

Note

Transferring an orientation using ORIMKS is not allowed in the handling transformation package. With an active transformation, the machine axis angles are not programmed and traversed, but the "orientation angles" (RPY angles according to robotics definition, see Section "Definition of positions and orientations using frames (Page 533)").

The orientation is selected via NC language commands ORIWKS and ORIMKS.

ORIMKS is the basic setting The initial setting can be modified via machine data:

MD20150 $MC_GCODE_RESET_VALUES (RESET position of G groups)

GCODE_RESET_VALUES [24] = 1 → ORIWKS is initial setting

GCODE_RESET_VALUES [24] = 2 → ORIMKS is initial setting

GCODE_RESET_VALUES [24] = 3 → ORIPATH

When ORIPATH is active, the orientation is calculated from the lead and side angles relative to the path tangent and surface normal vector.
Improper tool orientation

If the tool orientation is programmed in conjunction with the following functions

- G04 Dwell time
- G33 Thread cutting with constant lead
- G74 Approach reference point
- G75 Approach fixed point
- REPOSIL Repositioning
- REPOSQ Repositioning
- REPOSH Repositioning

Alarm 12130 "Illegal tool orientation" is output when Euler angles and direction vectors are selected. The NC program stops (this alarm can also occur in connection with G331, G332 and G63). Alarm 17630 or 17620 is output for G74 and G75 if a transformation is active and the axes to be traversed are involved in the transformation. This applies irrespective of orientation programming.

If the start and end vectors are inverse parallel when ORIWKS is active, then no unique plane is defined for the orientation programming, resulting in the output of alarm 14120.

Alarm 14400 is output if the transformation is switched on or off when a tool offset is active.

In the reverse situation, i.e. a tool offset is selected or deselected when a transformation is active, no alarm message is output.

Multiple input of tool orientation

According to DIN 66025, only one tool orientation may be programmed in a block, e.g. with direction vectors:

N50 A3=1 B3=0 C3=0

If the tool orientation is input several times, e.g. with direction vectors and Euler angles:

N60 A3=1 B3=1 C3=1 A2=0 B2=1 C2=3

error message 12240 "Channel X block Y tool orientation xx defined more than once" is displayed and the NC part program stops.
12.6.2 Orientation programming for 4-axis kinematics

Tool orientation for 4-axis kinematic

4-axis kinematics possess only one degree of freedom for orientation. When the orientation is programmed using RPY angles, Euler angles or direction vectors, it is not generally possible to guarantee that the specified orientation can be approached. If used at all, this type of orientation programming is only suitable for certain types of kinematic, i.e. those which feature an invariance in orientation angles relative to the basic axes. This is the case for the SCARA kinematic, for example.

For this reason, orientation programming is only permitted via "orientation angle" A for 4-axis kinematics. This angle corresponds to the RPY angle C according to the robotics definition, i.e. one rotation about the Z-RO axis, as illustrated in Fig. "Orientation angle for 4-axis kinematic".

![Figure 12-29 Orientation angle for 4-axis kinematic](image)

12.6.3 Orientation programming for 5-axis kinematics

Tool orientation for 5-axis kinematics

For 5-axis kinematics, when programming via orientation vector, it is assumed that the orientation vector corresponds to the x component of the tool.

When programming via orientation angle (RPY angle according to robotics definition), the x component of the tool is considered as the initial point for rotations.

For this purpose, the vector in the x tool direction, as shown in Fig. "Orientation angle for 5-axis kinematic", is first rotated around the Z axis by the angle A and then around the rotated Y axis by the angle B. The rotation by the angle C is not possible for 5-axis kinematics because of the restricted degrees of freedom for the orientation.
The following machine data can be used to set the flange coordinate system on the user side so that Z can be set as tool direction for 5-axis kinematics:

MD62636 $MC_TRAFO6_TFL_EXT_RPY (adaptation of the flange coordinate system)

The following machine data can be used to specify whether the tool direction should be set according to the robotics convention (0 = default setting) or the NC convention:

MD62637 $MC_TRAFO6_TOOL_DIR (definition of the tool direction)

To provide a tool calculation function as usually available for machine tools, set the following machine data:

MD62637 $MC_TRAFO6_TOOL_DIR = 1 (standard definition of the tool direction)

This machine data also has an effect on the rotation sequence of the virtual orientation axes. For this purpose, the vector in the Z tool direction is first turned through angle A at the X axis and then through angle B at the rotated Y axis. The rotation by the angle C is not possible for 5-axis kinematics because of the restricted degrees of freedom for the orientation.

Note

Additional information can be found in:

Function Manual, Special Functions, Section "Orientation axes"
and
Programming Manual, Job Planning, Section "Orientation axes".

Figure 12-30  Orientation angle for 5-axis kinematic

The following machine data can be used to set the flange coordinate system on the user side so that Z can be set as tool direction for 5-axis kinematics:

MD62636 $MC_TRAFO6_TFL_EXT_RPY (adaptation of the flange coordinate system)

The following machine data can be used to specify whether the tool direction should be set according to the robotics convention (0 = default setting) or the NC convention:

MD62637 $MC_TRAFO6_TOOL_DIR (definition of the tool direction)

To provide a tool calculation function as usually available for machine tools, set the following machine data:

MD62637 $MC_TRAFO6_TOOL_DIR = 1 (standard definition of the tool direction)

This machine data also has an effect on the rotation sequence of the virtual orientation axes. For this purpose, the vector in the Z tool direction is first turned through angle A at the X axis and then through angle B at the rotated Y axis. The rotation by the angle C is not possible for 5-axis kinematics because of the restricted degrees of freedom for the orientation.

Note

Additional information can be found in:

Function Manual, Special Functions, Section "Orientation axes"
and
Programming Manual, Job Planning, Section "Orientation axes".
12.7 Singular positions and how they are handled

The calculation of the machine axes to a preset position, i.e. position with orientation, is not always clear. Depending on the machine kinematic, there may be positions with an infinite number of solutions. These positions are called "singular".

Singular positions

- A singular position is, for example, characterized by the fact that the fifth axis is positioned at 0°. In this case, the singular position does not depend on a specified orientation. The fourth axis is not specified in this position, i.e., the fourth axis does not have any influence on the position or the orientation.
- A singular position also applies for articulated arm and SCARA kinematics if the third axis is at 0° or 180°. These positions are called leveling/diffraction singularity.
- Another singular position exists for articulated arm kinematics if the hand point is above the rotary axis of axis 1. This position is called overhead singularity.

Extreme velocity increase

If the path runs in close vicinity to a pole (singularity), one or several axes may traverse at a very high velocity. Alarm 10910 "Irregular velocity run in a path axis" is then triggered.

Behavior at pole

The unwanted behavior of fast compensating movements can be improved by reducing the velocity in the proximity of a pole. Traveling through the pole with active transformation is usually not possible.
12.8 Call and application of the transformation

Switch on
The transformation is activated by means of the TRAORI(1) command.

Once the TRAORI(1) command has been executed and the transformation thus activated, the interface signal switches to "1":

DB21, … DBX33.6 (transformation active)

If the machine data have not been defined for an activated transformation grouping, the NC program stops and the control displays the alarm 14100 "Orientation transformation does not exist".

For more information, go to:

References:
Programming Manual, Production Planning,
Section: "5-axis machining"

Deactivation
The currently active transformation is deactivated by means of TRAPOOF or TRAPOOF().

Note
When the "Handling transformation package" transformation is deactivated, no preprocessing stop and no synchronization of the preprocessing with the main run is performed.

RESET/end of program
The control behavior in terms of transformation following run-up, end of program or RESET depends on machine data:

MD20110 $MC_RESET_MODE_MASK (definition of control basic setting na)

Bit 7: Reset behavior of "active kinematic transformation"
Bit 7 = 0 For this reason, the initial setting is defined for active transformation after the end of part program or RESET, according to the following machine data:

MD20140 $MC_TRAFO_RESET_VALUE (transformation data block run-up (reset/part program end))

Meaning:
0: After RESET no transformation is active.
1 to 8: The set transformation is active according to machine data:

MD24100 $MC_TRAFO_TYPE_1 (definition of transformation 1 in the channel)
To machine data:
MD24460 $MC_TRAFO_TYPE_8 (channel transformation 8)

Bit 7 = 1 The current setting for the active transformation remains unchanged after a RESET or end of part program.
12.9 Actual value display

MCS machine coordinate system

The machine axes are displayed in mm/inch and/or degrees in MCS display mode.

WCS workpiece coordinate system

If the transformation is active, the tool tip (TCP) is specified in mm/inch and the orientation by the RPY angles A, B and C in WCS display mode. The tool orientation results from turning a vector in the Z direction firstly with A around the Z axis, then with B around the new Y axis and lastly with C around the new X axis.

If the transformation is deactivated, the axes will be displayed with the channel axis coordinates. Otherwise the geometry axis coordinates will be displayed.
12.10 Tool programming

Meaning

The tool lengths are specified in relation to the flange coordinate system. Only 3-dimensional tool compensations are possible. Depending on the kinematic type, there are additional tool restrictions for 5-axis and 4-axis kinematics. For a kinematic as illustrated in Fig. "5-axis NR kinematic", only a 1-dimensional tool with lengths in the x direction is permitted.

The tool direction is dependent on the machine's initial setting, as specified with G codes G17, G18 and G19. The tool lengths refer to the zero position specified by G17. This zero position should not be modified in the program.

Example

An example of a 2-dimensional tool mounted on a 5-axis Scara is described below (see Fig. "5-axis CC kinematic"). Type 100 (cutting tool) is specified as the tool identifier. The tool lengths result from the specifications shown in Fig. "Tool length programming". X-TOOL must be entered as tool length x and Y-TOOL must be entered as tool length y in the tool parameters.

\[
\begin{align*}
&\text{TC\_DP1}[1,1] = 100; \quad \text{Cutting tool type} \\
&\text{TC\_DP3}[1,1] = 0.0; \quad \text{(z) Length offset vector} \\
&\text{TC\_DP4}[1,1] = \text{Y-TOOL}; \quad \text{(y) Length offset vector} \\
&\text{TC\_DP5}[1,1] = \text{X-TOOL}; \quad \text{(x) Length offset vector}
\end{align*}
\]

Figure 12-31 Tool length programming
12.11 Cartesian PTP travel with handling transformation package

It is possible to use the "Cartesian PTP travel" function with the handling transformation package (see Section "Cartesian PTP travel (Page 29)"). For this, the following machine data must be set:

MD24100 $MC_TRAFO_TYPE_1 = 4100 (definition of transformation 1 in the channel)

References

For additional information on the function "Cartesian PTP travel" see:

Programming Manual, Job Planning; Transformations,
Section: Cartesian PTP travel
12.12 Supplementary conditions

12.12.1 Function-specific alarm texts

For details of the procedure for creating function-specific alarm texts, see Section "Creating alarm texts (Page 443)".

12.12.2 Functional restrictions

NCU 572.2

The handling transformation package can be utilized on NCU 572.2 hardware only on condition that is has been specifically enabled for the customer.

Clearance control

The handling transformation package cannot be operated together with the technology function: "clearance control", as generally the three basic axes are not arranged perpendicular to one another.

Moving to fixed end stop

The handling transformation package cannot be operated in conjunction with the "travel to fixed stop" function.

Several transformations

The handling transformation package can be activated just once per channel in all channels.

Tool programming

Tools can only be parameterized by specifying tool lengths. It is not possible to program an orientation for the tool.
12.12 Supplementary conditions

Programming of orientation

The programming possibilities of the orientation depend on the number of axes available on the machine:

Number < 5:
- Orientation axis angle

Number = 5:
- Orientation axis angle
- Orientation vector

Singularities

A pole cannot be crossed when a transformation is active. Singular positions can cause axis overloads. The feedrate is not automatically adjusted. The user must reduce the feedrate appropriately at the relevant points.
12.13 Startup

12.13.1 General information about start-up

The compile cycles are supplied as loadable modules. For the general commissioning of the compile cycles, see Section "TE01: Installation and activation of loadable compile cycles (Page 433)".

For the specific commissioning measures of this compile cycle, see Section "Starting up a kinematic transformation (Page 585)".

12.13.2 Starting up a kinematic transformation

The next step necessary to start up the kinematic transformation is to activate the handling transformation package (option).

Set the option data for handling transformation package.

Alarms

Record the alarm texts in the corresponding language-specific text files.

Set option data for transformation.

Configure the transformation

1. Enter transformation type 4100 in the machine data:
   MD24100 $MC_TRAFO_TYPE_1 (definition of transformation 1 in the channel)
2. Enter the assignment of the channel axes involved in the transformation in the machine data:
   MD24110 $MC_TRAFO_AXES_IN_1[0 to 5] (axis assignment for transformation)
   Axis numbers beginning with 1.
3. Enter the geometry axes corresponding to the Cartesian degrees of freedom of the machine in the machine data:
   MD24120 $MC_TRAFO_GEOAX_ASSIGN_TAB_1[0 to 2] (assignment between geometry axis and channel axis for transformation 1)
4. Enter the kinematic identifier in the machine data:
   MD62600 $MC_TRAFO6_KINCLASS (kinematic category)
5. Enter the identifier for special kinematics, if there are any, in machine data:
   MD62602 $MC_TRAFO6_SPECIAL_KIN (special kinematic type)
6. Enter the number of axes in the machine data:
   MD62605 $MC_TRAFO6_NUM_AXES (number of transformed axes)
7. If the travel direction of the involved axes is opposed to the transformation definition, then change the factory setting in the machine data:

   MD62618 $MC_TRAFO6_AXES_DIR[ ] (matching of physical and mathematical directions of rotation)

8. Enter the data which define the basic axes:
   – Basic axis identifier in machine data:
     MD62603 $MC_TRAFO6_MAIN_AXES (basic axis identifier)
   – Basic axis lengths in machine data:
     MD62607 $MC_TRAFO6_MAIN_LENGTH_AB (basic axis lengths A and B)

9. Enter any changes to the axis sequence in the machine data:
    MD62620 $MC_TRAFO6_AXIS_SEQ (rearrangement of axes)

10. Enter the data which define the hand:
    – Wrist axis identifier in the machine data:
      MD62604 $MC_TRAFO6_WRIST_AXES (wrist axis identifier)
    – Parameters for hand in the machine data:
      MD62614 $MC_TRAFO6_DHPAR4_5A (parameter A for configuring the hand)
      MD62615 $MC_TRAFO6_DHPAR4_5D (parameter D for configuring the hand)
      MD62616 $MC_TRAFO6_DHPAR4_5ALPHA (parameter ALPHA for configuring the hand)
    – MD62606 $MC_TRAFO6_A4PAR (axis 4 is parallel/anti-parallel to last basic axis)

11. Enter the geometry parameters:
    – Frame T_IRO_RO
    – Frame T_X3_P3
    – Frame T_FL_WP

12. Enter the position in relation to the calibration point in the machine data:
    MD62617 $MC_TRAFO6_MAMES (offset between mathematical and mechanical zero points)

13. Enter the Cartesian velocities and acceleration rates.
12.14 Data lists

12.14.1 Machine data

12.14.1.1 General machine data

| Number | Identifier: $MN_ | Description |
|--------|-----------------|-------------|
| 10620  | EULER_ANGLE_NAME_TAB[n] | Name of Euler angle |
| 19410  | TRAFO_TYPE_MASK, bit 4 | Option data for OEM transformation |
| 60943  | CC_ACTIVE_IN_CHAN_RCTR | Activation of the handling transformation for the corresponding channels |
| 19610  | TECHNO_EXTENSION_MASK | Option data for handling transformation |

12.14.1.2 Channelspecific machine data

| Number | Identifier: $MC_ | Description |
|--------|-----------------|-------------|
| 21100  | ORIENTATION_IS_EULER | Angle definition for orientation programming |
| 21110  | X_AXIS_IN_OLD_X_Z_PLANE | Coordinate system with automatic FRAME definition |
| 24100  | TRAFO_TYPE_1 | Definition of transformation |
| 24110  | TRAFO_AXES_IN_1 | Axis assignment for transformation 1 |
| 24120  | TRAFO_GEOAX_ASSIGN_TAB_1 | Assignment of geometry axes to channel axes |

12.14.1.3 Channel-specific machine data for compile cycles

| Number | Identifier: $MC_ | Description |
|--------|-----------------|-------------|
| 62600  | TRAFO6_KINCLASS | Kinematic category |
| 62601  | TRAFO6_AXES_TYPE | Axis type for transformation |
| 62602  | TRAFO6_SPECIAL_KIN | Special kinematic type |
| 62603  | TRAFO6_MAIN_AXES | Basic axis identifier |
| 62604  | TRAFO6_WRIST_AXES | Wrist axis identifier |
| 62605  | TRAFO6_NUM_AXES | Number of transformed axes |
| 62606  | TRAFO6_A4PAR | Axis 4 is parallel/anti-parallel to last basic axis |
| 62607  | TRAFO6_MAIN_LENGTH_AB | Basic axis lengths A and B |
| 62608  | TRAFO6_TX3P3_POS | Attachment of hand (position component) |
| 62609  | TRAFO6_TX3P3_RPY | Attachment of hand (rotation component) |
| 62610  | TRAFO6_TFLWP_POS | Frame between wrist point and flange (position component) |
| 62611  | TRAFO6_TFLWP_RPY | Frame between wrist point and flange (rotation component) |
12.14 Data lists

| Number | Identifier: SMC_ | Description |
|--------|-----------------|-------------|
| 62612  | TRAFO6_TIRORO_POS | Frame between base center point and internal system (position component) |
| 62613  | TRAFO6_TIRORO_RPY | Frame between base center point and internal system (rotation component) |
| 62614  | TRAFO6_DHPAR4_5A  | Parameter A for configuring the hand |
| 62615  | TRAFO6_DHPAR4_5D  | Parameter D for configuring the hand |
| 62616  | TRAFO6_DHPAR4_5ALPHA | Parameter ALPHA for configuring the hand |
| 62617  | TRAFO6_MAMES     | Offset between mathematical and mechanical zero points |
| 62618  | TRAFO6_AXES_DIR  | Matching of physical and mathematical directions of rotation |
| 62619  | TRAFO6_DIS_WRP   | Mean distance between wrist point and singularity |
| 62620  | TRAFO6_AXIS_SEQ  | Rearrangement of axes |
| 62621  | TRAFO6_SPIN_ON   | Configuration includes triangular or trapezoidal spindles |
| 62622  | TRAFO6_SPIND_AXIS | Axis that is controlled by triangular spindle |
| 62623  | TRAFO6_SPINDEL_RAD_G | Radius G for triangular spindle |
| 62624  | TRAFO6_SPINDEL_RAD_H | Radius H for triangular spindle |
| 62625  | TRAFO6_SPINDEL_SIGN | Sign for triangular spindle |
| 62626  | TRAFO6_SPINDEL_BETA | Angular offset for triangular spindle |
| 62627  | TRAFO6_TRP_SPIND_AXIS | Axes driven via trapezoidal connection |
| 62628  | TRAFO6_TRP_SPIND_LEN | Trapezoid lengths |
| 62629  | TRAFO6_VELCP     | Cartesian velocities |
| 62630  | TRAFO6_ACCCP     | Cartesian acceleration rates |
| 62631  | TRAFO6_VELORI    | Orientation angle velocities |
| 62632  | TRAFO6_ACCORI    | Orientation angle acceleration rates |
| 62634  | TRAFO6_DYN_LIM_REDUCE | Reduction factor for velocity controller during the JOG travel |
| 62635  | TRAFO6_VEL_FILTER_TIME | Time constant for PT1 filter during the JOG travel |
| 62636  | TRAFO6_TFL_EXT_RPY | Adaptation of the flange coordinate system |
| 62637  | TRAFO6_TOOL_DIR   | Definition of the tool direction |

12.14.2 Signals

12.14.2.1 Signals from channel

| DB number | Byte.bit | Description |
|-----------|----------|-------------|
| 21, ...   | 29.4     | Activate PTP traversal |
| 21, ...   | 33.6     | Transformation active |
| 21, ...   | 232      | Number of active G function of G function group 25 (ORIWORKS, ORIMKS, ORIPATH) |
| 21, ...   | 317.6    | PTP traversal active |
13.1 Brief description

If there are two or more separately traversable machining heads on a machine tool and a transformation is required for machining, the orientation axes of the machining heads cannot be coupled via the standard coupling functions COPON, TRAILON. The coupling is performed in the workpiece coordinate system (WCS). However, under the aforementioned conditions the coupling of the orientation axes must be performed in the machine coordinate system (MCS).

The compile cycle "MCS coupling" can be used to parameterize coupling in the machine coordinate system (MCS) with coupling factors "1" and "-1" for a master axis, referred to as CC_Master in the following, and several slave axes, referred to as CC_Slave in the following.

The couplings can be switched on and off via commands in the part program.

![Figure 13-1 Application example: Double slide gantry-type milling machine](image)
13.1 Brief description

Master and slave axes

A CC_Master can be master for several CC_Slaves. A CC_Slave cannot be a CC_Master at the same time.

The following functions are not possible for a CC_Slave:

- To be a PLC axis
- To be a command axis
- To be traversed separately from the CC_Master in JOG mode

Tolerance window

When the coupling is active, the actual values of the CC_Master and CC_Slave are monitored for compliance with a parameterizable tolerance window.

Programming a CC-Slave

If a CC_Slave is programmed in a channel, either an alarm is displayed or the axis is requested for the channel via an implicitly triggered axis interchange (GET), depending on the setting in machine data:

MD30552 $MA_AUTO_GET_TYPE = <GET type> (automatic GET for axis)

Switching the coupling on/off

A coupling is switched on and off via OEM-specific language commands and can be active in all operating modes.

Collision protection

In order to protect the machining heads against collision in decoupled operation or in mirrored coupling, a collision protection can be parameterized. The activation is performed via machine data or the NC/PLC interface. The assignment of the protected pairs is not dependent on the CC_Master and CC_Slave pairs.
13.2 Description of MCS coupling functions

13.2.1 Defining coupling pairs

A CC_Slave axis is matched to its CC_Master axis via the following axial machine data:

MD63540 $MA_CC_MASTER_AXIS (specifies the CC_Master axis assigned to a CC_Slave axis)

The coupling's axes can only be changed when the coupling is not active.

A CC_Slave axis is displayed in axial VDI-Out byte:

DB31, … DBX97.0 (axis is slave axis)

Requirements

• The CC_Master and CC_Slave axes must be either both rotary axes or both linear axes.
• Spindles cannot be coupled by this function.
• Neither the CC_Master nor CC_Slave axis may be an exchange axis ($MA_MASTER_CHAN[AXn]=0$)
13.2 Description of MCS coupling functions

13.2.2 Switching the coupling ON/OFF

Switching the coupling on

- Switching the 1:1 coupling on. Monitoring of the tolerance window is active.
  \[ \text{CC\_COPON}(\text{[<axis1>][<axis2>][<axis3>][<axis4>][<axis5>]}) \]
- Switching the 1:-1 coupling (mirror) on. Monitoring of the tolerance window is not active.
  \[ \text{CC\_COPONM}(\text{[<axis1>][<axis2>][<axis3>][<axis4>][<axis5>]}) \]

Axis identifier

An axis can be specified via:

- Machine axis identifier
- Channel axis identifier
- Geometry axis identifier

Supplementary conditions

- A programmed axis must be involved in a coupling.
- CC_Master axes, CC_Slave axes or both can be programmed simultaneously. All defined couplings are switched on with CC_COPON or CC_COPONM.
- An alarm is output if an axis not involved in a coupling is programmed.

NC/PLC interface signals

All NC/PLC interface signals of a coupling refer to the CC_Slave axes.

- DB31, … DBX97.1 (coupling active)
- DB31, … DBX97.2 (mirroring active)
- DB31, … DBX24.2 (suppress CC_Slave axis coupling)

  If the coupling is suppressed, an alarm is not output for the CC_Slave axis.

Switching the coupling off

\[ \text{CC\_COPOFF}(\text{[<axis1>][<axis2>][<axis3>][<axis4>][<axis5>]}) \]

As for switching on the coupling with the difference that no alarm is output if an axis is programmed that is not involved in a coupling.

An existing coupling can also be switched off by the axial NC/PLC interface signal of the CC_Slave axis.

Supplementary condition

A coupling can only be switched on or off when the axes are at standstill.
13.2.3 **Tolerance window**

A monitoring window is specified via axial machine data:

MD63541 $MA_CC_POSITION_TOL (monitoring window)

The absolute difference between the actual values of CC_Slave axis and CC_Master axis must never be greater than this value. Alarm 70010 is output if the tolerance window is violated.

The monitoring function is not active:

- if the machine data is set to 0,
- if the coupling is switched off,
- if axis/spindle inhibit is set for one of the axes,
- if an axis is in follow-up mode,
- for the 1:1 coupling.

If the offset stored at the instant of coupling activation changes when 1:1 coupling is active, the change is indicated by NC => PLC VDI-SS:

DB31, … DBX97.3 (offset after start-up)

---

**Note**

The offset might change:

- if the SW limit monitor was active for one axis during the main run,
- if one axis has been switched to follow-up mode,
- if collision protection was active for one axis.

---

13.2.4 **Memory configuration: Block memory**

The technological function requires additional data in the NC-internal block memory. The values must be increased for the following memory configuring channel-specific machine data:

- MD28090 $MC_MM_NUM_CC_BLOCK_ELEMENTS += 1 (number of block elements for compile cycles)
- MD28100 $MN_MM_NUM_CC_BLOCK_USER_MEM += 1 (size of block memory for compile cycles (DRAM) in KB)
13.3 Description of collision protection

13.3.1 Defining protection pairs

A ProtecSlave axis (PSlave) is matched to its ProtecMaster (PMaster) axis via the following axial machine data:

MD63542 $MA_CC_PROTECT_MASTER (specifies the PMaster axis assigned to a PSlave axis)

The protection pairs can thus be defined independently of the coupling pairs. A PSlave axis may act as the PMaster axis for another axis. The axes must be either both rotary axes or both linear axes.

13.3.2 Switching collision protection ON / OFF

The minimum clearance between PSlave and PMaster is provided in the axial machine data of the PSlave axis:

MD63544 $MA_CC_COLLISION_WIN (collision protection window)

No collision protection is implemented if the value entered here is less than 0. The 0 position offset between PSlave and PMaster is provided in the axial machine data (PSlave axis):

MD63545 $MA_CC_OFFSET_MASTER (zero point offset between PSlave and PMaster)

Before the collision protection is switched on, the monitoring function for each individual axis must be enabled in the following machine data:

MD63543 $MA_CC_PROTECT_OPTIONS

In the same machine data for the PSlave axis, a setting is entered to specify whether the collision protection must be active continuously or whether it is activated via VDI interface signal (PLC => NC):

DB31, … DBX24.3

If collision protection is active, the setpoint positions of the PSlave and PMaster in the next IPO cycle are extrapolated and monitored in the IPO clock cycle using the current setpoint position and current velocity.

If the axes violate the minimum clearance, they are braked at the configured maximum acceleration rate:

MD32300 $MA_MAX_AX_ACCEL (Axis acceleration)

Or at a 20% faster acceleration rate, defined in machine data:

MD63543 $MA_CC_PROTECT_OPTIONS
An alarm is output as soon as the axes reach zero speed.

| WARNING |
|------------------|
| If the axes are forced to brake, the positions displayed in the workpiece coordinate system are incorrect! |
| These are not re-synchronized again until a system RESET. |

If the axes are already violating the minimum clearance when collision protection is activated, they can only be traversed in one direction (retraction direction). The retraction direction is configured in machine data:

MD63543 $MA_CC_PROTECT_OPTIONS

The collision protection status is optionally displayed in axial VDI-Out byte of the PSlave:

DB31, … DBX66.0 (activate monitoring)

- DB31, … DBX66.0=1 → collision protection active
- DB31, … DBX66.0=0 → collision protection inactive

This output is activated via Bit7 in machine data of the PSlave axis:

MD63543 $MA_CC_PROTECT_OPTIONS
13.3 Description of collision protection

13.3.3 Configuring example

![Diagram of configuration example]

Figure 13-2 Configuring example

---

**Note**

Since the collision protection function extrapolates the target positions from the "current velocity + maximum acceleration (or +20%)", the monitoring alarm may be activated unexpectedly at reduced acceleration rates:

**Example:**

PMaster = X, PSlave = X2, $MA_CC_COLLISION_WIN = 10mm

Starting point in part program: X=0.0 X2=20.0

N50 G0 X100 X2=90

; the monitoring alarm is activated because X and X2 are interpolating together: for this reason, the acceleration rate of X2 < maximum acceleration.

**Remedy:**

- N50 G0 POS[X]=100 POS[X2]=90
- or switch the monitoring function off.
13.4 User-specific configurations

Parking the machining head

In this context, "parking" means that the relevant machining head is not involved in workpiece machining. All axes are operating under position control and positioned at exact stop.

Even if a machining head is being used in production, coupling should be active! This is essential primarily if only the second head (Y2,...) is being used. "Axis/spindle inhibit" must then be set axially (PLC -> NCK) for the "parked" head.

Note

When an axis/spindle inhibit is active, a part program can be executed if this axis is not operating under position control.

Spindle functionalities

Since an MCS coupling cannot be activated for spindles, other types of solutions should be configured for these.

- Positioning the spindle ($SPOS = ...$)
  
  Instead, a cycle is called from $SPOS$. $SPOS$ is called for all active spindles in this cycle.

- Speed default
  
  Speed and direction of rotation inputs can be detected via synchronized actions or PLC and passed on to all other active spindles.

- Synchronous spindle function
13.5 Special operating states

Reset

The couplings can remain active after a \texttt{RESET}.

Reorg

No non-standard functionalities.

Block search

During a block search, the last block containing an OEM-specific language command is always stored and then output with the last action block. This feature is illustrated in the following examples. The output positions of the axes are always 0.

\textbf{Example 1:}

\begin{verbatim}
N01 M3 S1000
N02 G01 F1000 X10 Y10
N03 CC_COPON( X, Y)
\end{verbatim}

TARGET:

If this program is started normally, axes X and Z traverse to \texttt{X10 Z10} in the decoupled state. After block search to TARGET: axes X and Y traverse to this position in the coupled state!

\textbf{Example 2:}

\begin{verbatim}
N01 M3 S1000
N02 CC_COPON( X)
N03 G01 F1000 X100 Y50
N04 CC_COPOFF( X)
N05 CC_COPON( Y)
N06 Y100
N10 CC_COPOFF()
\end{verbatim}

TARGET:

After block search to TARGET: the axes traverse to \texttt{X100 Y100} in the decoupled state.

\textbf{Example 3:}

\begin{verbatim}
N01 CC_COPON( X, Y, Z)
N02 ...
...
N10 CC_COPOFF( Z)
\end{verbatim}

TARGET:

After block search to TARGET: no coupling is active!

\textbf{Single block}

There are no nonstandard functionalities.
13.6 Boundary conditions

Validity

The function is configured only for the first channel.

Braking behavior

Braking behavior at the SW limit with path axes

The programmable acceleration factor $\text{ACC}$ for braking at the SW limit corresponds to the path axes.

The axes in an MCS coupling are principal axes that are referred to as geometry axes due to their geometric arrangement.

Braking geometry axes using synchronized actions

The faster deceleration capacity as required for path axes can be implemented for geometry axes as follows using a synchronized action:

$$\text{ACC}[x] = 190$$

Axial acceleration limitation with G0

The following machine data is not considered by the MCS coupling:

MD32434 $\text{SMA\_G00\_ACCEL\_FACTOR}$ (scaling of the acceleration limitation with G0)

A value deviating from the standard value affects the braking ramp up to the software limit switch.
13.7 Data lists

13.7.1 Machine data

13.7.1.1 Channelspecific machine data

| Number | Identifier: $MC_ | Description                                           |
|--------|------------------|-------------------------------------------------------|
| 28090  | NUM_CC_BLOCK_ELEMENTS | Number of block elements for compile cycles.          |
| 28100  | NUM_CC_BLOCK_USER_MEM  | Total size of usable block memory for compile cycles |

13.7.1.2 Axis/spindlespecific machine data

| Number | Identifier: $MA_ | Description                                           |
|--------|------------------|-------------------------------------------------------|
| 63540  | CC_MASTER_AXIS   | Specifies the CC_Master axis assigned to a CC_Slave axis. |
| 63541  | CC_POSITION_TOL | Monitoring window                                       |
| 63542  | CC_PROTEC_MASTER | Specifies the PMaster axis assigned to a PSlave axis.   |
| 63543  | CC_PROTEC_OPTIONS |                                                |
| 63544  | CC_COLLISION_WIN | Collision protection window                           |
| 63545  | CC_OFFSET_MASTER | Zero point offset between PSlave and PMaster          |
14.1 Brief description

Function

The "Continue machining - Retrace support (RESU)" technological function supports the retracing of uncompleted 2-dimensional machining processes such as laser cutting, water jet cutting, etc.

In the event of a fault during the machining process, e.g. loss of the laser, RESU can be used even by machine operators who do not have specific knowledge of the active part program to interrupt machining and travel back along the contour from the interruption point to a program continuation point necessary for machining purposes.

After reaching the "continue machining" point, the machine operator triggers the re-machining. As part of the retrace process, an implicit block search takes place along the contour with calculation followed by repositioning on the contour and automatic retracing of the part program machining process.

The retrace option is selected and deselected in advance using part program commands within the machining program. The program continuation point can be selected at any position within the contour ranges specified in this way.

Figure 14-1 Programmed contour with program continuation and interruption points
Precise retracing of contours is possible on all programmed contours comprising straight and circular elements. During retracing, other programmed contour elements such as splines or automatically inserted non-linear contour elements (circle, parable, etc. e.g. through tool radius compensation) are mapped as straight lines through the start and end points of the corresponding contour element, thereby preventing precise retracing of contours.

Function code

The code for the "Continue Machining - Retrace support" technological function for function-specific identifiers of program commands, machine data, etc. is:

RESU (= REtrace SUpport)

Restrictions

The use of the "Continue Machining Retrace support" technological function is subject to the following restrictions:

- The technological function is available only in the 1st channel of NC.
- Program continuation or reverse travel is only possible for part program blocks that contain traversing blocks in the configured RESU working plane (e.g. 1st and 2nd geometry axis of the channel, see Section "Definition of the RESU working plane (Page 609)").

References

The "Continue Machining - Retrace support" technological function is a compile cycle. For the handling of compile cycles, see Section "TE01: Installation and activation of loadable compile cycles (Page 433)".
14.2 Function description

14.2.1 Function

Block search with calculation on contour
To be able to resume interrupted machining at a specific point in a part program, a block search can be carried out using the "Block search with calculation on contour" standard function. However, detailed knowledge of the part program is required to be able to enter the block number of the part program block required for the block search (i.e. the number of the block the search needs to locate).

Continue machining - Retrace support
The "Continue machining - Retrace support" technological function supports the continuation of the machining operation by means of an implicit block search with calculation on the contour without the machine operator needing to know the part program block required.

Continue machining might be required for example in a laser cutting application if the laser is lost during the machining operation and machining needs to resume at the point at which it was interrupted.

RESU supports the machining continuation via the following automatically running sub-functions:
- Function-specific reverse travel along the contour to the required program continuation point
- Automatic identification of the part program block associated with the program continuation point
- Block search with calculation on the contour for the part program block identified
- Repositioning on the contour at the program continuation point
- Continuation of part program machining

To approach the required program continuation point exactly, it is possible to switch several times between reverse and forward travel along the contour during the continue machining process.

Retraceable contour areas
RESU is activated via programming of the function-specific part program command CC_PREPRE (1). Only the contour range lying between the RESU start (CC_PREPRE (1)) and the interruption point (NC stop) is retraceable in the sense of "continue machining".

Once RESU has been launched, all part program blocks in which traversing movements are programmed are logged by RESU for possible subsequent reverse travel. Contour ranges for which continuing machining is irrelevant can be excluded from the log using RESU stop CC_PREPRE (0).
Contour ranges which are not logged are bridged by straight lines between the starting and end points during reverse/forward travel.

Figure 14-2  Retraceable contour range

### 14.2.2 Definition of terms

**Interruption point**

The interruption point is the point of the contour at which the traversing movement comes to a standstill following an NC stop and reverse travel is activated.

**Program continuation point**

The program continuation point is the point of the contour at which reverse travel terminates and program continuation is activated.

**Retraceable contour range**

Retraceable contour ranges consist of traversing blocks in the configured RESU working plane (e.g. 1st and 2nd geometry axis of the channel), that are programmed in the part program between the RESU start command `CC_PREPRE(1)` and the RESU stop command `CC_PREPRE(0)` *(see "Figure 14-2 Retraceable contour range (Page 604)").*
14.2.3 Functional sequence (principle)

The principle sequence of the RESU function between the interruption point, program continuation point and continuation of part program processing is described below.

Requirements

A part program with traversing blocks in the configured RESU working plane (e.g. 1st and 2nd geometry axis of the channel) as well as the part program command for the RESU start has been started in the 1st channel.

Functional sequence

1. Interrupt part program processing:
   The part program processing / traversing movement may be interrupted any number of traversing blocks after RESU start by NC stop.

2. Select reverse travel:
   The selection of the reverse travel is by PLC interface signal:
   \[ \text{DB21, … DBX0.1 = 1} \] (forward / reverse)

3. Reverse travel:
   The contour is traversed in the RESU working plane in the reverse direction with NC start. Instead of the current machining program, RESU selects the automatically generated RESU main program.
   For more about RESU programs, see Section "RESU-specific part programs (Page 617)".

4. End reverse travel:
   Once the required program continuation point on the contour has been reached, reverse travel is ended using NC stop.

5. Select forward travel (optional):
   For forward travel, reverse travel must be deselected via PLC interface signal:
   \[ \text{DB21, … DBX0.1 = 0} \]

6. Forward travel (optional):
   The contour is traversed in the RESU working plane in the forward direction with NC start.

7. End forward travel (optional):
   Once the required program continuation point on the contour has been reached, forward travel is ended using NC stop.
8. Retrace support:
   Continue machining is initiated via PLC interface signal:
   DB21, … DBX0.2 = 1 (start retrace support)
   For retrace support, RESU automatically selects the original machining program and
   launches a block search with calculation as far as the program continuation point.

9. Continuation of part program machining:
   Part program processing continues at the program continuation point in accordance with
   the "Block search with calculation" standard function when two NC start commands occur
   one after the other.
   The first NC start command processes the action blocks. Retrace support ASUB
   "CC_RESU_BS_ASUP.SPF" is initiated when the last action block is reached:
   DB21, … DBX32.6 = 1 (last action block active)
   To ASUB (see Section "RESU-specific part programs (Page 617)").
   The second NC start command processes the approach block before part program
   processing is resumed.

Note
Points 3 to 8 can be repeated as often as required.
Following retrace support, a new reverse travel is possible up to a maximum of the last
program continuation point.
Signal chart for interface signals

The principle sequence of the RESU function is illustrated in the following figure as a signal chart of the interface signals involved:

1. Reverse travel is started.
2. Forward travel is started (optional).
3. Continue machining is started (block search).
4. Search destination (target block) was found.
5. NC start → Action blocks are output.
6. Last action block is activated.

The RESU ASUB "CC_RESU_BS_ASUP.SPF" is triggered when the last action block is activated.
7. NC start → Return travel to approach block for program continuation point.
8. Part program processing (target block) resumed

Figure 14-3   Signal chart
14.2.4 Maximum retraceable contour area

In multiple machining continuation within a contour area, the reverse travel on the contour is always possible only up to the last machining continuation point (W). In first-time reverse travel after RESU start, reverse travel up to the start of the contour range is possible.

This response must be illustrated via the following graph. For the sake of simplicity, the interruption point (U) is always the same:

![Graph](image)

Figure 14-4 Maximum Retraceable contour range

1st reverse travel

In first-time reverse travel, reverse travel is possible maximum up to the start of the first contour element \(N20\) after RESU start \(N15\) \(W_{1\text{max}}\).

If reverse travel takes place up to the machining continuation point \(W1\), \(W1\) defines the maximum RESU range for a possible further reverse travel after machining continuation and forward travel.

2nd reverse travel

A renewed reverse travel can now be undertaken maximum only up to the last machining continuation point \(W_{2\text{max}} = W1\).

If reverse travel goes as far back as program continuation point \(W2\), the maximum RESU range is restricted further.
14.3 Startup

14.3.1 Activation

Before commissioning the technological function, ensure that the corresponding compile cycle has been loaded and activated (see also Section "TE01: Installation and activation of loadable compile cycles (Page 433)").

Activation

The "Continue machining - Retrace support" technology function is activated via the following machine data:

MD60900+i $MN_CC_ACTIVE_IN_CHAN_RESU[0], bit 0 = 1

Note

The "Continue machining - Retrace support" technological function is available only in the 1st channel of NC.

14.3.2 Definition of the RESU working plane

Machining continuation/reverse travel is possible only for part program blocks that contain traversing blocks in the configured RESU working plane.

The RESU working plane is defined with the following machine data:

MD62580 $MC_RESU_WORKING_PLANE

| Value | Meaning |
|-------|---------|
| 1     | The RESU working plane is formed by the 1st and 2nd Geometry axes of the 1st channel (for G17). |
| 2     | The RESU working plane is formed by the 1st and 3rd Geometry axes of the 1st channel (for G18). |
| 3     | The RESU working plane is formed by the 2nd and 3rd Geometry axes of the 1st channel (for G19). |

14.3.3 Memory configuration: Block memory

The technological function requires additional data in the NCK-internal block memory. The values must be increased for the following memory configuring channel-specific machine data:

- MD28090 $MC_MM_NUM_CC_BLOCK_ELEMENTS += 4 (number of block elements for compile cycles)
- MD28100 $MN_MM_NUM_CC_BLOCK_USER_MEM += 20 (size of block memory for compile cycles (DRAM) in KB)
14.3 Startup

14.3.4 Memory configuration: Heap memory

Memory requirements

RESU requires compile cycles heap memory for the following function-specific buffers:

- **Block buffer**
  
  The larger the block buffer (see "Figure 14-6 RESU-specific part programs (Page 617)"), the larger the number of part program blocks that can be traversed backwards.
  
  32 bytes are required per part program block.
  
  The block buffer can be parameterized directly.

- **Block search buffer**
  
  The block search buffer contains the information required for processing subprogram searches in the context of RESU.
  
  180 bytes are required for each subprogram. The block search buffer requires at least 2880 bytes (16 subprogram calls with 180 bytes each).
  
  The block search buffer cannot be parameterized directly.

A function-specific GUD variable displays the size of the block search buffer (for creation of GUD variables, see Section "Channel-specific GUD variables (Page 627)").

![Diagram of heap memory division for compile cycles](image-url)
Memory configuration

Size of the compile cycle heap memory
The size of the heap memory in KB that can be used by the user for compile cycles is defined via the memory configuring channel-specific machine data:

MD28105 $MC_MM_NUM_CC_HEAP_MEM

For RESU, the already existing machine data value (x) is adjusted as follows:

MD28105 $MC_MM_NUM_CC_HEAP_MEM = x + 50

Size of the block buffer
The size of the block buffer is adjusted via the machine data:

MD62571 $MC_RESU_RING_BUFFER_SIZE

Default setting:

MD62571 $MC_RESU_RING_BUFFER_SIZE = 1000

RESU portion of the total heap memory
The RESU portion of the total heap memory that can be used for compile cycles is set via the machine data:

MD62572 $MC_RESU_SHARE_OF_CC_HEAP_MEM

Default setting:

MD62572 $MC_RESU_SHARE_OF_CC_HEAP_MEM = 100

Error messages
The block search buffer requires at least 2880 bytes (corresponding to 16 subprogram calls with 180 bytes each). Otherwise, the following alarm will be generated during NC power-up:

Alarm 75600 "Channel 1 Retrace Support: Incorrect MD configuration, error no. 5"

If the block search buffer is not big enough during operation, the following alarm appears:

Alarm 75606 "Channel 1 retraceable contour shortened"
14.3.5 RESU main program memory area

Memory configuration

The storage location of the RESU main program CC_RESU.MPF can be set with the following machine data (see Section "Main program (CC_RESU.MPF) (Page 618)"):

\[ \text{MD62574 $MC\_RESU\_SPECIAL\_FEATURE\_MASK$ (additional RESU features)} \]

| Bit | Value | Meaning                                      |
|-----|-------|----------------------------------------------|
| 0   | 0     | The RESU main program is stored in the **dynamic NC memory** (default). |
| 1   | 1     | The RESU main program is stored in the **static NC memory** (default). |

Storage in the dynamic NC memory (default)

If the RESU main program is created in the dynamic NC memory, the memory area available to the user for file storage must be increased as follows:

\[ \text{MD18351 $MN\_MM\_DRAM\_FILE\_MEM\_SIZE = x^{(1)} + 100$} \]

\(^{(1)}\) already available machine data value

Storage in the static NC memory

If the RESU main program is created in the static memory area of the NC, it is retained even after a POWER OFF. However, since RESU regenerates the RESU main program every time the retrace support function is used, this parameter setting is not recommended.
14.3.6 Storage of the RESU subroutines

Storage as user or manufacturer cycles

The following RESU-specific subroutines can be stored as user cycles or manufacturer cycles:

- INI program: CC_RESU_INI.SPF
- END program CC_RESU_END.SPF
- Retrace support ASUB CC_RESU_BS_ASUP.SPF
- RESU ASUB CC_RESU_BS_ASUP.SPF

The setting is done using machine data:

MD62574 $MC_RESU_SPECIAL_FEATURE_MASK (additional RESU features)

| Bit | Value | Meaning                                      |
|-----|-------|----------------------------------------------|
| 2   | 0     | The RESU-specific subroutines are stored as user cycles (default). |
| 1   | 1     | The RESU-specific subroutines are stored as manufacturer cycles. |

Series commissioning

Due to the default setting of MD62574 Bit 2, the RESU-specific subroutines along with their contents are stored as user cycles the first time the NC is powered up after the activation of the technological function.

If the setting is then changed to specify that the RESU-specific subroutines are to be stored as manufacturer cycles, the RESU-specific subroutines already created as user cycles are retained even after a new power-up and must be deleted.

As support for series commissioning, the RESU-specific subroutines present as user cycles can be deleted without prompting when the NC is powered up:

MD62574 $MC_RESU_SPECIAL_FEATURE_MASK, Bit 3 = 1

14.3.7 ASUB enable

Note

A requirement for using ASUBs is that the "Cross-mode actions" option must be available.

The following machine data must be set for the start enable for the RESU-specific ASUB CC_RESU_ASUP.SPF while the channel is in the NC STOP state:

MD11602 $MN_ASUP_START_MASK, bit 0 = 1 (ignore stop reason for ASUB)
MD11604 $MN_ASUP_START_PRIO_LEVEL = 1 (priorities from which MD11602 is effective)
14.3.8  PLC user program

Requirements

The following functionality is necessary for the sequential coordination of the RESU function in the PLC user program:

IF        DB21, … DBX32.2        "Retrace support active" == 1
THEN       DB21, … DBX0.1        "Forward/Reverse" = 0
            DB21, … DBX0.2        "Start retrace support" = 0

IF        DB11, … DBX0.7        "Mode group reset" == 1
OR         DB21, … DBX7.7        "Reset" == 1
THEN       DB21, … DBX0.1        "Forward/Reverse" = 0
            DB21, … DBX0.2        "Start retrace support" = 0

The following signals should be reset for safety reasons:

IF        DB21, … DBX0.2        "Start retrace support" == 1
THEN       DB21, … DBX0.1        "Forward/Reverse" = 0

IF        DB21, … DBX0.1        "Forward/Reverse" == 1
THEN       DB21, … DBX0.2        "Start retrace support" = 0

Program example

The following program extract implements the changes described above:

```plaintext
U DB21, … DBX32.2 // IF "Retrace support active" == 1
R DB21, … DBX0.1 // THEN "Forward/Reverse" = 0
R DB21, … DBX0.2 // "Start retrace support" = 0
O DB11, … DBX0.7 // IF "Mode group reset" == 1
O DB21, … DBX7.7 // OR "Reset" == 1
R DB21, … DBX0.1 // THEN "Forward/Reverse" = 0
R DB21, … DBX0.2 // "Start retrace support" = 0
U DB21, … DBX0.2 // IF "Start retrace support" == 1
R DB21, … DBX0.1 // THEN "Forward/Reverse" = 0
U DB21, … DBX0.1 // IF "Forward/Reverse" == 1
R DB21, … DBX0.2 // THEN "Start retrace support" = 0
```

Special functions

Function Manual, 07/2012, 6FC5397-2BP40-3BA0
14.4 Programming

14.4.1 RESU Start/Stop/Reset (CC_PREPRE)

Start / stop / reset / of RESU is done with the program instruction:

```
CC_PREPRE (Prepare Retrace)
```

**Programming**

**Syntax:**

```
CC_PREPRE(<mode>)
```

**Parameters:**

- Mode: Type: INTEGER
- Range of values: -1, 0, 1

`CC_PREPRE(...)` is a procedure call and must therefore be programmed in a dedicated part program block.

**Functionality**

The following modes are available:

| Statement   | Meaning |
|-------------|---------|
| `CC_PREPRE(1)` | Starts logging the traversing blocks. The information required for reverse travel is logged on a block-specific basis in a RESU-internal block buffer. The travel information is related to the two geometry axes of the RESU working plane e.g. the 1st and 2nd geometry axes of the channel: MD20050 $MC_AXCONF_GEOAX_ASSIGN_TAB[x]; with x = 0 and 1 Or, if transformation is active: MD24120 $MC_TRAFO_GEOAX_ASSIGN_TAB[x]; with x = 0 and 1 |
| `CC_PREPRE(0)` | Stops logging the traversing blocks. Thus irrelevant contour areas can be excluded from the log. These excluded contour areas are bridged by a straight line between the starting point and end point during reverse travel. |
| `CC_PREPRE(-1)` | Deactivates logging of the traversing blocks and deletes the function-internal block buffer. Contour areas located before the instant of deactivation of the part program are therefore no longer available to RESU. |
RESET response

In reset events:

• NCK POWER ON RESET (warm start)
• NCK Reset
• End of program (M30)

CC_PREPRE(-1) is executed implicitly.

Error messages

The following programming errors are detected and displayed with alarms:

• Invalid mode programmed:
  RESU alarm 75601 "Channel number Block number Invalid parameter for CC_PREPRE( )"

• More than one parameter programmed:
  Alarm 12340 "Channel number Block number Too many parameters"

• RESU technology function is not available:
  The technological function is not available. The compile cycle is possibly not loaded or it was not activated.
  Alarm 12340 "Channel number Block number Name CC_PREPRE not defined or option not available"
14.5 RESU-specific part programs

14.5.1 Overview

RESU uses the following, automatically generated and partially adjustable part programs:

| Program                  | Name                        |
|--------------------------|-----------------------------|
| Main program             | CC_RESU.MPF                 |
| INI program              | CC_RESU_INI.SPF             |
| END program              | CC_RESU_END.SPF             |
| Continue machining ASUB | CC_RESU_BS_ASUP.SPF         |
| RESU ASUB                | CC_RESU_ASUP.SPF            |

The following figure provides an overview of the internal structure of the technological function and the relationship between the various part programs.

Figure 14-6 RESU-specific part programs
14.5.2 Main program (CC_RESU.MPF)

Function

In addition to the calls for the RESU-specific subprograms, the RESU main program "CC_RESU.MPF" contains the traversing blocks generated from the traversing blocks logged in the block buffer for reverse/forward travel along the contour. The program is always regenerated by the RESU function if, once the part program has been interrupted, the status of the following interface signal changes:

DB21, … DBX0.1 (Reverse / Forward)

Note

CC_RESU.MPF must not be changed. User-specific adjustments are to be made in the corresponding RESU-specific subprograms.

Error messages

By default, RESU generates traversing blocks for the entire retraceable contour range logged in the block buffer. If there is not enough memory for all traversing blocks to be generated in the parameterized memory area of the RESU main program (see Section "RESU main program memory area (Page 612)"), RESU reduces the number of generated traversing blocks.

The lacking memory and/or reduction in the number of the generated traversing blocks is indicated by an alarm:

RESU alarm 75608 "Channel number NC memory limit reached, RAM type type"

If the RESU main program is created in the static user memory, the following system alarm appears at the same time as the RESU alarm:

Alarm 6500 "NC memory full"

Note

If the number of traversing blocks generated is reduced due to insufficient memory, the entire retraceable contour can still be retraced for retrace support. To do this, proceed as follows:

• Reverse travel up to the end of the RESU main program
• Two-time change of the interface signal:
  
  DB21, … DBX0.1 (Reverse/Forward)

Using the current position as a new interruption point enables RESU to generate a new RESU main program.

Subsequently, travel is possible as far as the end of the retraceable contour range or, if the limits have changed, as far as the starting point of the last traversing block that can be generated.

The procedure described can be repeated as many times as required both for reverse and forward travel.
14.5 RESU-specific part programs

14.5.3 INI program (CC_RESU_INI.SPF)

Function

The RESU-specific subprogram "CC_RESU_INI.SPF" contains the defaults required for the reverse travel:

- Metric input system: G71
- Absolute dimensions: G90
- To switch off the adjustable zero-point offsets / frames (see Section "Frames (Page 633)"): G500
- To switch off the active tool offsets (see Section "Tool offsets (Page 633)"): T0
- Switching off the tool radius offset: G40
- Traversing velocity: F200

Program structure

CC_RESU_INI.SPF has the following content by default:

```plaintext
PROC CC_RESU_INI
    G71 G90 G500 T0 G40 F200
    ;system frames that are present are deactivated
    ;actual value and scratching
    if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H01'
        $P_SETFRAME = ctrans()
    endif
    ;external zero point offset
    if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H02'
        $P_EXTFRAME = ctrans()
    endif
    ;tool carrier
    if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H04'
        PAROTOF
    endif
    if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H08'
        TOROTOF
    endif
    ;workpiece reference points
    if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H10'
        $P_WPFRAME = ctrans()
    endif
    ;cycles
    if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H20'
        $P_CYCFRAME = ctrans()
    endif
    ;transformations
```
14.5 RESU-specific part programs

```c
if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H40'
  $P_TRAFRAME = ctrans()
endif
; bit mask for global basic frames
$P_NCBFRMASK = 0
; bit mask for channel-specific basic frames
$P_CHBFRMASK = 0
; programmable frame
$P_PFRAME = ctrans()
```

**Note**

CC_RESU_INI.SPF may be changed.

CC_RESU_INI.SPF must not contain any RESU part program commands. CC_PREPRE(x).

**CAUTION**

By changing the content of the RESU-specific subprogram "CC_RESU_INI.SPF", the user (machine manufacturer) takes over the responsibility for the correct sequence of the technological function.

14.5.4 END program (CC_RESU_END.SPF)

**Function**

The task of the RESU-specific subroutine "CC_RESU_END.SPF" is to stop reverse travel once the end of the retraceable contour is reached. If the RESU function is parameterized appropriately, this scenario will not arise under normal circumstances.

**Program structure**

CC_RESU_END.SPF has the following content by default:

```c
PROC CC_RESU_END
  M0
ML7
```

Special functions
TE7: Continue machining at the contour (retrace support) - 840D si only

14.5 RESU-specific part programs

---

**Note**

CC_RESU_END.SPF may be changed.

CC_RESU_END.SPF must not contain any RESU part program commands. CC_PREPRE(x).

---

⚠️ **CAUTION**

By changing the content of the RESU-specific subroutine "CC_RESU_END.SPF", the user (machine manufacturer) takes over the responsibility for the correct sequence of the technological function.

---

14.5.5 Retrace support ASUB (CC_RESU_BS_ASUP.SPF)

**Function**

The NC is forced to approach the current path point during machining continuation with the help of the RESU-specific ASUB "CC_RESU_BS_ASUP.SPF":

- Reapproach next point on path: RMN
- Approach along line on all axes: REPOSA

**Program structure**

CC_RESU_BS_ASUP.SPF has the following content by default:

```plaintext
PROC CC_RESU_BS_ASUP SAVE
  RMN
  REPOSA
M17
```

**Note**

CC_RESU_BS_ASUP.SPF may be changed.

User-specific modifications must be inserted before the part program block RMN.

---

⚠️ **CAUTION**

By changing the content of the RESU-specific subroutine "CC_RESU_BS_ASUP.SPF", the user (machine manufacturer) takes over the responsibility for the correct sequence of the technological function.
14.5.6 RESU ASUB (CC_RESU_ASUP.SPF)

Function

The RESU-specific ASUB "CC_RESU_ASUP.SPF" is required internally for the function. The ASUB is initiated if the following RESU interface signal is switched over in the NC Stop state:
DB21, … DBX0.1 (Forward/Reverse)

Program structure

CC_RESU_ASUP.SPF has the following content:

```
PROC CC_RESU_ASUP
   ; siemens system asub - do not change
   G4 F0.001
   M0
   REPOSA
   M17
```

Note

CC_RESU_ASUP.SPF must not be changed.
14.6 Retrace support

14.6.1 General

Retrace support refers to the entire operation from initiation of retracing through the interface signal DB21, … DBX0.2 = 1 (start machining continuation) up to to the continuation of the part program processing of the programmed contour.

Requirement

In order for retrace support to function, the retrace mode, launched by means of the request for reverse travel, must be active in the channel:

DB21, … DBX32.1 = 1 (retrace mode active)

(See Section "Functional sequence (principle) (Page 605)"

Subfunctions

The two essential subfunctions of retrace support are the standard NC functions:

- Block search with calculation on contour
- Repositioning on the contour via shortest route (REPOS RMN)

14.6.2 Block search with calculation on contour

Function

The block search with calculation on the contour initiated implicitly by RESU within the framework of the retrace support has the following tasks:

- Set the program pointer on the part program block on which repositioning was done with the help of Reverse / Forward travel
- Calculates the axis positions on the basis of the programmed traversing blocks from the start of the part program to the target block
- Collates the instructions programmed from the start of the part program to the target block, which are executed in the action block. These include:
  - Auxiliary functions
  - Tool change
  - Spindle functions
  - Feedrate programming
All part program instructions which are not executed in the action block but are required for retrace support in the part program must be entered manually in the RESU-specific retrace support ASUB CC_RESU_BS_ASUP.SPF, e.g.:

- Synchronized actions
- M functions

**References**

The complete description of the Block search is available in:
Basic Functions Function Manual; Mode Group, Channel, Program Mode (K1), Program test

**14.6.3 Reposition**

**Function**

Following the end of the last action block (last traversing block before repositioning), NC Start outputs the approach block for repositioning all channel axes programmed in the part program as far as the target block.

**Geometry axes**

In the approach block, the geometry axes of the RESU working plane (e.g. 1st and 2nd geometry axes of the channel) traverse the shortest route along the contour to the program continuation point.

![Diagram](image-url)
Channel axes

All other channel axes programmed in the part program travel to the relevant position calculated in the block search.

14.6.4 Temporal conditions concerning NC start

NC start should be initiated twice by the machine operator within the framework of continue machining (see Section "Functional sequence (principle) (Page 605)").

The following conditions must be met:

- In NC start for outputting the action blocks:
  - The block search must be completed:
    DB21, … DBX33.4 = 0 (block search active)
- In NC start for outputting the approach blocks:
  - The RESU ASUB "CC_RESU_BS_ASUP" must be completed:
    DB21, … DBX318.0 = 1 (ASUB stopped)

14.6.5 Block search from last main block

The block search with calculation on the contour executed within the framework of the machining continuation via the use of the most powerful NCU in very large part programs can itself lead to computation times of several minutes up to the reaching of the target block.

A significant reduction of this waiting period is possible through the use of the "Block search from the last main block".

Functionality

For retrace support with block search from the last main block, the search for the target block takes place in 2 stages:

1. Block search without calculation from start of machining program to last main block before target block. Subprograms are ignored during this search, i.e. it takes place exclusively in the main program.
2. Block search with calculation on contour from main block to target block. This block search does not ignore subprograms.

Requirement

To enable a search from last main block for retrace support, at least one main block must be programmed after the RESU start CC_PREPRE(1).
Main block

All instructions required for processing the subsequent section of the part program must be programmed in one main block.

The main blocks are to be designated with a Main Block No. consisting of the sign ":" and a positive whole number (block number).

References:
Programming Manual, Fundamentals; Fundamentals of NC Programming, Language Elements of the Programming Language

Activation

Activation of the block search from the last main block is performed using the RESU-specific machine data:

MD62575 $MC_RESU_SPECIAL_FEATURE_MASK_2, bit 0 (additional RESU features)

| Bit | Value | Meaning                                           |
|-----|-------|---------------------------------------------------|
| 0   | 0     | Retrace support is performed using block search with calculation on contour. |
| 1   | 1     | Retrace support is performed using block search from the last main block. |

Supplementary conditions

In order that a new retrace support operation can take place following a retrace support operation with block search from last main block, the RESU start CC_PREPRE(1) must be programmed in the retrace support ASUB "CC_RESU_BS_ASUP.SPF".

Programming example:

```
PROC CC_RESU_BS_ASUP SAVE
   ; (synchronized actions, M functions, etc. required for retrace support )
   CC_PREPRE(1)
   RMN
   REPOSA
   M17
```
14.7 Function-specific display data

14.7.1 Channel-specific GUD variables

As display data for the size of the block search buffer, RESU provides the following channel-specific GUD variables:

| GUD variable               | Meaning                        | Value | Access   |
|----------------------------|--------------------------------|-------|----------|
| CLC_RESU_LENGTH_BS_BUFFER  | Size of block search buffer    | Byte  | read only|

After the startup of the technological function, the GUD variable is not displayed automatically on the operator panel.

Creating and displaying the GUD variable

Perform the following steps to create and display the GUD variables in the operator panel:

1. Set password:
   - The password of protection level 1 (machine manufacturer) is to be input.
2. Activate the "Definitions" display
3. The file is recreated if an "SGUD.DEF" file is still not available:
   - Name: SGUD
   - Type: Global data /system
   - Confirm entry with OK.
   - This opens the file in the editor.
4. Edit the GUD variable definitions:
   ```
   DEF CHAN REAL CLC_RESU_LENGTH_BS_BUFFER
   M30
   ```
5. Save the file and close the editor.
6. Activate the "SGUD.DEF" file.

The GUD variable is not displayed on the operator panel.

Note

The new GUD variable, which is already being displayed, will be detected by the RESU function and supplied with an up-to-date value only following an NCK POWER ON Reset. Hence, a NCK POWER-ON must be initiated after the creation.

References

The exact procedure to be following while creating and displaying GUD variables depends on the software version of the existing operator panel and is described in the:

- Operating Manual
14.8 Function-specific alarm texts

For details of the procedure for creating function-specific alarm texts, see Section "Creating alarm texts (Page 443)".
14.9 Boundary conditions

14.9.1 Function-specific supplementary conditions

14.9.1.1 Continue machining within subprograms

Subprogram call outside or inside a program loop

Clear retrace support within subprograms depends on whether the subprogram call is made outside or inside a program loop:

- **Outside**
  
  Clear retrace support is possible if a subprogram is called outside a program loop.

- **Inside**
  
  Clear retrace support may not be possible if a subprogram is called inside a program loop (see Section "Continue machining within program loops (Page 630)").

Number of passes $P$

Subprogram repetitions using number of passes $P$ are taken into account for retrace support. This means that retrace support is performed in the part program with the correct reference to the part program block and number of passes $P$ to the program continuation point of the contour.
14.9 Boundary conditions

14.9.1.2 Continue machining within program loops

In NC high-level language, program loops can be programmed using:

- LOOP  ENDLOOP
- FOR  ENDFOR
- WHILE  ENDMILE
- REPEAT  UNTIL
- CASE/IF-ELSE-ENDIF in conjunction with GOTO

If retrace support is performed within program loops, the retrace support is always effective in the first loop run.

⚠️ WARNING

If the machining continuation point on the programmed contour is the result of a loop run not equal to the first loop run, there can be considerable contour deviations in the further course of the machining under certain circumstances, which pose a threat to man and machine.

14.9.1.3 Machining continuation on full circles

In full circles, the block starting and end points coincide at one contour point. As no clear differentiation is possible in this case, one always starts from the block start point during machining continuation on this kind of a contour point. The first part program block following retrace support is then the circular block.

A contour point just before the block end point of the circular block must be selected to avoid traversing the circular block after the machining continuation.

14.9.1.4 Automatically generated contour elements

The automatic generation of non-linear / non-circular contour elements by the NC takes place, for example, when programming the following NC functions in the part program:

- RND
- G641/G642
- Tool radius compensation

For reverse / forward travel as part of RESU these contour elements can be replaced by straight lines between the start and end of the block.
14.9.2 Supplementary conditions for standard functions

14.9.2.1 Axis replacement

As long as RESU is active, the two geometry axes of the RESU working plane (e.g. the 1st and 2nd geometry axes of the channel) must not be transferred to another channel via axis replacement (RELEASE(x)/GET(x)).

The RESU activity:

- Starts:
  - with the part program command CC_PREPRE(1)
- Ends with:
  - the program end
  - or
  - with the part program command CC_PREPRE(-1)

14.9.2.2 Traversing movements of the channel axes

Other channel axes, except the two geometry axes of the RESU working plane, are not considered by RESU.

If traversing movements in other channel axes are required for machining continuation or reverse travel, these can either be undertaken by the machined operator manually, or programmed as travel block in the RESU-specific subroutine "CC_RESU_INI.SPF".

⚠️ WARNING

The machine operator must ensure that collision of the associated traversing movements does not take place during the entire machining continuation operation within the framework of the technological function of RESU.

14.9.2.3 Block numbers

The following RESU-specific subroutines and their subroutines must not contain block numbers:

- CC_RESU_INI.SPF
- CC_RESU_END.SPF

The following alarm appears in the event of an error:

Alarm 75604 "Reverse travel not possible, error no. number"
14.9 Boundary conditions

14.9.2.4 Block search

Block search with calculation

RESU is subject to the following constraints in the context of the "block search with calculation (on contour/at end of block)" standard function:

- The last block of the RESU part program command \texttt{CC\_PREPRE(x)} run during the block search is effective in the target block.
- The retraceable contour range starts with the \texttt{REPOS} approach block.

Block search without calculation

RESU part program commands \texttt{CC\_PREPRE(x)} have no effect in the "block searches without calculation" function.

14.9.2.5 Transformations

RESU can also be used for active kinematic transformation (e.g. 5-axis transformation) subject to restrictions, as the traversing movements of the two geometry axes of the RESU working plane are recorded in the basic coordinate system (BCS) and therefore before the transformation (see also Section "TE4: Handling transformation package - 840D sl only (Page 531)").

Transformation changeover

While RESU is active, no transformation changes are permitted to take place and transformation must not be activated/deactivated.

The RESU activity:

- Starts:
  - with the part program command \texttt{CC\_PREPRE(1)}

- Ends with:
  - the program end
  - or
  - with the part program command \texttt{CC\_PREPRE(-1)}

References

A full description of the transformations is available in:
Function Manual, Extended Functions; Kinematic Transformation (M1)
14.9.2.6 Compensations

RESU can be used in interaction with compensations, because the traversing movements of the two geometry axes of the RESU working plane is recorded in the basic coordinate system (BCS) and therefore before the compensation.

A full description of the compensations can be found in:

References:
Function Manual, Extended Functions; Compensations (K3)

14.9.2.7 Frames

RESU can be used in conjunction with frames.

However, as the traversing movements of the two geometry axes of the RESU working plane are recorded in the basic coordinate system (BCS) and therefore after the frames have been taken into account, the frame offsets must be deactivated during retrace support (reverse / forward travel).

The frame offsets are deactivated during retrace support via the standard default settings of the RESU-specific subprogram "CC_RESU_INI.SPF" (see Section "INI program (CC_RESU_INI.SPF) (Page 619)").

A full description of the frames can be found in:
Reference:
Function Manual, Basic Functions; Axes, Coordinate Systems, Frames (K2)

14.9.2.8 Tool offsets

RESU can be used in conjunction with tool offsets.

However, as the traversing movements of the two geometry axes of the RESU working plane are recorded in the basic coordinate system (BCS) and therefore after the tool offsets have been taken into account, the tool offsets must be deactivated during retrace support (reverse / forward travel).

The tool offsets are deactivated during retrace support via the standard default settings of the RESU-specific subprogram "CC_RESU_INI.SPF" (see Section "INI program (CC_RESU_INI.SPF) (Page 619)").

Specific instances of tool radius compensation, e.g. compensation on outside corners $G450 \text{DISC}=x$ may generate contour deviations between the contour traversed during retrace support and the contour programmed in the machining program.

Contour deviations are always generated if tool radius compensation produces contour elements that are non-linear or circular. For example, $G450\text{DISC}=x$, where $x > 0$ produces parabolic or hyperbolic contour elements.

A full description of the tool radius compensation can be found in:
Reference:
Function Manual, Basic Functions; Tool Compensation (W1)
14.10 Data lists

14.10.1 Machine data

14.10.1.1 General machine data

| Number | Identifier: $MN_ | Meaning |
|--------|-----------------|---------|
| 11602  | ASUP_START_MASK | Ignore stop reasons if an ASUB is running. |
| 11604  | ASUP_START_PRIO_LEVEL | Defines the ASUB priority from which MD11602 is effective. |
| 18351  | MM_DRAM_FILE_MEM_SIZE | Size of the part program memory in DRAM (in kByte) |

14.10.1.2 Channelspecific machine data

| Number | Identifier: $MC_ | Description |
|--------|-----------------|-------------|
| 20050  | AXCONF_GEOAX_ASSIGN_TAB | Assignment 'geometry/channel axis' |
| 24120  | TRAFO_GEOAX_ASSIGN_TAB_1 | Assignment of geometry - channel axes for transformation 1 |
| 28090  | MM_NUM_CC_BLOCK_ELEMENTS | Number of block elements for compile cycles (CC) |
| 28100  | MM_NUM_CC_BLOCK_USER_MEM | Size of block memory for CC in KB |
| 28105  | MM_NUM_CC_HEAP_MEM | Heap memory in KB for CC applications (DRAM) |
| 62571  | RESU_RING_BUFFER_SIZE | Size of ring buffer (RESU-internal block buffer) |
| 62572  | RESU_SHARE_OF_CC_HEAP_MEM | RESU portion of the total CC heap memory |
| 62573  | RESU_INFO_SA_VAR_INDEX | Indices of the synchronized action variables |
| 62574  | RESU_SPECIAL_FEATURE_MASK | Additional RESU features |
| 62575  | RESU_SPECIAL_FEATURE_MASK_2 | Additional RESU properties |
| 62580  | RESU_WORKING_PLANE | Selection of the RESU working plane |
15.1 Brief description

Function

The "Cycle-independent path synchronized switching signal output" technological function serves the purpose of switching time-critical, position-based machining processes on and off quickly, e.g. high speed laser cutting (HSLC; High Speed Laser Cutting).

The switching signal output can be block-related or path length-related:

- **Block-related switching signal output**
  - The switching signal output and therefore, the activation / deactivation of the machining is undertaken independently of the status changes:
    - Rapid travel mode G00 active / inactive
    - Programmed feed threshold fell short / exceeded

- **Path length-related switching signal output**
  - The switching signal output and therefore, the activation / deactivation of the machining takes place in a continuous change and is monitored along the traversed path.
  - This enables a regular machining without the individual switching positions having to be programmed explicitly.

The activation or selection of the mentioned options leads to the monitoring of the output of the digital signal which is programmed with a part program command.

I/Os

Only the on-board I/Os of the NC module can be used as the digital I/O via which the switching signal is output: The switching signal can only be output via one of the 4 on-board digital outputs on the NCU module.

Restrictions

The use of the "Cycle-independent path synchronized switching signal output" technological function is subject to the following restriction:

- The technological function is available only in one channel of NC.

References

The "cycle-independent path-synchronous switching signal output" technological function is a compile cycle. For the handling of compile cycles, see Section "TE01: Installation and activation of loadable compile cycles (Page 433)".
15.2 Functional description

15.2.1 General

Note
The functionality is described with examples, with the help of the "High speed laser cutting technology (HSLC, High Speed Laser Cutting).

15.2.2 Calculating the switching positions

15.2.2.1 Block-related switching signal output

Switching criteria
During high-speed laser cutting, e.g. as used to manufacture perforated sheets, it is absolutely essential to switch the laser beam on/off exactly at the programmed setpoint positions during the machining process.

In order to minimize programming overheads, the switching positions of the technology function are calculated using the velocity of the geometry axes programmed in the part program block.

The following criteria define the setpoint position programmed in the part program block (end of block position) as a switching position:
1. G0 edge change
2. Overshooting/undershooting a freely programmable velocity threshold

G0 edge change as switching criterion
If G0 (rapid traverse) is active in a part program block (programmed or modal), the switching signal is switched off. On the other hand: If G0 (rapid traverse) is not active in a part program block, the switching signal is switched on. The G0 edge change marks the programmed end of block position of the previous block as the switching position.

Example:
The following block positions function as switching positions:

- Position X30 for **GO**-edge change from \textbf{N10} to \textbf{N20}
- Position X100 for **GO**-edge change from \textbf{N30} to \textbf{N40}

**Freely programmable velocity threshold value as switching criterion**

A freely programmable velocity threshold value is used to define the setpoint velocity programmed in the part program block at and above which the switching signal is activated/deactivated.

- If the setpoint velocity programmed in the part program block is **above** the programmed threshold, the switching signal is switched **off**.
- If the setpoint velocity is **at** or **below** the threshold value, the switching signal is switched **on**.

The edge change marks the programmed end of block position of the previous block as the switching position.

**Example:**

The following block positions function as switching positions:

- Position X30 for edge change from \textbf{N10} to \textbf{N20}
- Position X70 for edge change from \textbf{N20} to \textbf{N30}

**Note**

**GO** always deactivates the switching signal, regardless of the threshold value.
15.2.2.2 Path length-related switching signal output

Programmable paths as the switching criterion

In path-related switching signal output, the switching positions are defined with the help of the two freely programmable paths $s_1$ and $s_2$.

Functional sequence

The path length-related switching signal output starts with a switch on signal at the beginning of the first traversing block after activation with `CC_FAST_CONT` (refer to ① in the screen).

The processing is active till a deactivation signal is set after the traversing of a programmable path $s_1$ (refer to ② in screen). This way the processing is interrupted till a switch on signal is set again after the traversing of a programmable path $s_2$ (refer to ③ in screen).

The changeover between activation and deactivation signal, and therefore between processing and interruption phase is path length dependent at the end of either of the stretch sections $s_1$ and $s_2$. This enables a continuous and regular machining without the individual switching positions having to be programmed explicitly.

The path length-related switching signal output ends with the start of the first traversing block (or of another executable block) after deactivation with `CC_FASTOFF` (refer to ④ in screen).

![Diagram of path length-related switching signal output](image-url)
15.2.3 Calculating the switching instants

In order for the switching to be as precise as possible at the switching positions calculated, the control calculates the positional difference between the actual position of the geometry axes involved and the switching difference in every position controller cycle.

If the positional difference is less than 1.5 position controller cycles, the control converts it into a temporal difference taking into account the current path velocity and acceleration rate of the geometry axes.

With the temporal difference specified, a hardware timer is started, which triggers the switching signal at exactly the instant calculated in advance regardless of the position controller cycle.

15.2.4 Switching frequency and switching position distance

Maximum switching frequency

The maximum switching frequency is: 1 signal edge change per IPO cycle

Note

Special case: IPO cycle time = position controller cycle time

In this case, the maximum switching frequency is:

1 signal edge change per 2 IPO cycles

Minimum switching position clearance

The minimum possible distance between the switching positions placed one above the other depends on:

- The duration of an IPO cycle
- The feed rate

The theoretically possible minimum distance can be determined from these dimensions as follows:

\[
\text{Minimum switching position clearance} = \text{Programmed feed rate} \times \text{IPO cycle time}
\]

Example:

For IPO cycle times of 2 ms and position controller cycle times of 1 ms as well as a feed rate of 20000 mm/min, the theoretically possible minimum distance between switching positions one above the other is limited to:

\[
20000 \text{ mm/min} \times 2 \text{ ms} = 0.667 \text{ mm}
\]
Value below minimum switching position distance

For path length-related switching signal output, the value may fall below the minimum switching position distance, e.g. due to:

- Increase in feed rate
- Decrease in programmable switching position distance $s_1$ and $s_2$

The following reactions take place if the value falls below the minimum:

- Alarm 75501 "Channel %1 HSLC: CC_FASTON_CONT speed too high" is displayed.
- The switching signal at the current switching position is omitted. To maintain the machining rhythm, the function also suppresses the switching signal at the following switching position.

The position of all the following switching positions is not affected by this. In the further course of the path, one can witness alternating output and suppression of two switching signals following each other.

15.2.5 Approaching switching position

If in block-related switching signal output a switching position is not reached exactly, e.g. in continuous-path mode and travel in more than one geometry axis, then switching takes place at the instant at which the positional difference between the actual position of the geometry axes involved and the programmed switching position increases again.

![Figure 15-2 Switching position offset in path control mode](image-url)
15.2.6 Programmed switching position offset

Programmed switching position offset

For block-related switching signal output, a positional offset of the switching position can be programmed:

- Offset distance **negative** = lead
  
  With a negative offset distance, the switching position is offset **before** the set point position programmed in the part program block.

  If an excessively large negative offset distance is programmed, i.e. the setpoint has already been exceeded by the time the edge is detected, the signal is switched immediately.

- Offset distance **positive** = follow-up
  
  With a positive offset distance, the switching position is offset **behind** the set point position programmed in the part program block.

---

**Figure 15-3 Programmed switching position offset**

Path reference

The offset distance is a positional specification that refers to the programmed path. This way one can start from a simplified linear motion. Path curves are not taken into account.

Response to single block and G60

Due to the internal motion logic, negative offset distances (lead) have no effect when used with the following standard functions:

- Single Block
- Exact stop at block end \((G60)\)

15.2.7 Response to part program interruption

Following an interruption in the part program (NC-STOP) and subsequent change to JOG mode, the technological function is deactivated or switching signals cease to be output.

The technology function is reactivated or switching signals are output again only after switching to the AUTOMATIC mode and continuing the part program (NC-START).
15.3 Startup

15.3.1 Activation

Before commissioning the technological function, ensure that the corresponding compile cycle has been loaded and activated (see also Section "TE01: Installation and activation of loadable compile cycles (Page 433)").

Activation

The "Cycle-independent path-synchronized switching signal output" technology function is activated via the following machine data:

MD60948 $MN_CC_ACTIVE_IN_CHAN_HSLC[0], Bit 0 = 1

Note

The "Cycle-independent path-synchronized switching signal output" is available only in one channel of NC.

15.3.2 Memory configuration

The technological function requires additional data in the NCK-internal block memory. The values must be increased for the following memory configuring channel-specific machine data:

- MD28090 $MC_MM_NUM_CC_BLOCK_ELEMENTS += 1 (number of block elements for compile cycles)
- MD28100 $MN_MM_NUM_CC_BLOCK_USER_MEM += 10 (size of block memory for compile cycles (DRAM) in KB)

15.3.3 Parameterizing the digital on-board outputs

Parameter assignment

A digital output from the local I/O is required for the switching signal.

For this, at least 1 digital output byte must be defined through the following machine data:

MD10360 $MN_FASTIO_DIG_NUM_OUTPUTS ≥ 1 (number of active digital output bytes)
References

A full description of the parameterization of a digital output can be found in:

- Function Manual, Extended Functions, Digital and Analog NCK I/Os (A4)

15.3.4 Parameterizing the switching signal

Output number of the switching signal

Once the compile cycle has started up, the following function-specific machine data appears in the channel-specific machine data:

MD62560 $MC_FASTON_NUM_DIG_OUTPUT (number of the digital output of the switching signal)

The number n of the on-board digital output through which the switching signal is to be output, must be entered in it:

n = 1, 2, 3 or 4

Disable

Entering the number of the digital output n = 0 deactivates the function. No message or alarm is output.

Effect on other output signals

The hardware-timer-controlled output of the switching signal at the parameterized output delays the signal output for the other digital on-board outputs, e.g. due to synchronized actions, by 2 IPO cycles.
15.3.5 Parameterization of the geometry axes

Standard setting

Machines for high-speed laser cutting normally have two geometry axes that are configured in the following two machine data:

MD20050_$MC_AXCONF_GEOAX_ASSIGN_TAB[0]
MD20050_$MC_AXCONF_GEOAX_ASSIGN_TAB[1]

The calculation of the switching instants is derived from these two geometry axes.

Note

The configured axis selection for calculating the switching instants can be changed by redefining the first and second geometry axes in the part program with the help of program instructions \texttt{GEOAX(1, <axis name>)} and \texttt{GEOAX(2, <axis name>)}.

Important:
For the function change to be considered correctly, it should occur before the interpretation of the \texttt{CC_FASTON} command.

Changing the default setting

For a deviating machine configuration (e.g. definition of a third geometry axis), the default setting can be adjusted via the following machine data:

MD60948 $MN_CC_ACTIVE_IN_CHAN_HSCL[1]

| Value | Meaning |
|-------|---------|
| $MN_CC_ACTIVE_IN_CHAN_HSCL[1]="H3" | Default setting. The calculation of the switching instants is derived from the first and second geometry axis. |
| $MN_CC_ACTIVE_IN_CHAN_HSCL[1]="H7" | The calculation of the switching instants is derived from the first, second and third geometry axis. |

Any change is effective only after the next NCK booting.
15.4 Programming

15.4.1 Activating the block-related switching signal output (CC_FASTON)

Syntax

\[ CC_{\text{FASTON}} \left( \text{DIFFON}, \text{DIFFOFF} [,\text{FEEDTOSWITCH}] \right) \]

**CC_FASTON** is a procedure call and must therefore be programmed in a dedicated part program block.

Parameter

The parameters for the **CC_FASTON** procedure have the following meaning:

| Parameter    | Meaning                                      |
|--------------|----------------------------------------------|
| <DIFFON>     | Length* of the offset distance for setting the switching signal. |
| <DIFFOFF>    | Length* of the offset distance for deactivating the switching signal. |
| <FEEDTOSWITCH> | This parameter is optional.  
                  | If the parameter is not specified in the procedure call, the G0 edge change is used as the switching criterion.  
                  | If the parameter is specified in the procedure call, it contains as a switching criterion the velocity threshold value, which, when undershot or exceeded, activates or deactivates the switching signal accordingly. |

* The basic unit (inch or mm) depends on the current dimensions programming (G70 / G71 / G700 / G710).

Programming example

| Programming | Comment |
|-------------|---------|
| DEF REAL DIFFON= -0.08 ; Length* of the offset distance for activating the switching signal = -0.08 |
| DEF REAL DIFFOFF= 0.08 ; Length* of the offset distance for deactivating the switching signal = 0.08 |
| DEF REAL FEEDTOSWITCH= 20000 ; Speed threshold value = 20000 |
| CC_FASTON(DIFFON, DIFFOFF, FEEDTOSWITCH) ; Activating block-related switching signal output (with speed threshold value as switching criterion) |
Changing parameters

The parameters for the `CC_FASTON()` procedure can be modified at any time during the execution of the part program. To do this, enter the procedure call again with the new parameter values. The switching criterion (edge change / velocity threshold value) may also be changed.

Reset response

A reset (NC RESET or end of program) deactivates the function.

15.4.2 Activating the path length-related switching signal output (CC_FASTON_CONT)

Syntax

```plaintext
CC_FASTON_CONT (PATH_DISTANCE_ON, PATH_DISTANCE_OFF)
```

`CC_FASTON_CONT()` is a procedure call and must therefore be programmed in a dedicated part program block.

Parameter

The parameters for the `CC_FASTON_CONT()` procedure have the following meaning:

| Parameter                | Meaning                                      |
|--------------------------|----------------------------------------------|
| `< PATH_DISTANCE_ON >`   | Length* of the stretch section with machining (s1). |
| `< PATH_DISTANCE_OFF>`   | Length* of the stretch section without machining (s2). |
| * The basic unit (inch or mm) depends on the current dimensions programming (`G70`, `G71`, `G700`, `G710`). |

Programming example

| Programming                     | Comment                                                                 |
|---------------------------------|-------------------------------------------------------------------------|
| DEF REAL PATH_DISTANCE_ON = 0.5 | ; Length* of the stretch section with machining = 0.5                   |
| DEF REAL PATH_DISTANCE_OFF = 1.0| ; Length* of the stretch section without machining = 1.0                |
| CC_FASTON_CONT (PATH_DISTANCE_ON, PATH_DISTANCE_OFF) | ; Activating path length-related switching signal output. |
15.4 Programming

Changing parameters

The parameters for the CC_FASTON_CONT( ) procedure can be modified at any time during the execution of the part program. For this, the procedure call must be specified again with the new parameter values.

Reset response

A reset (NC RESET or end of program) deactivates the function.

15.4.3 Deactivation (CC_FASTOFF)

Syntax

CC_FASTOFF

CC_FASTOFF is a procedure call and must therefore be programmed in a dedicated part program block.

Functionality

The CC_FASTOFF procedure call deactivates the "Cycle independent, path synchronized switching signal output" function.
15.5 Function-specific alarm texts

For details of the procedure for creating function-specific alarm texts, see Section "Creating alarm texts (Page 443)".
15.6 Supplementary conditions

15.6.1 Block search

Switching signal output for block search

If a block search is on a part program block which lies after a `CC_FASTON()` procedure call for activating the technology function, then the switching signal is activated with the next traversing motion. One of the specific consequences of this is to initiate travel along the contour from the start position of the geometry axes back to the program continuation point with an activated switching signal.

Example

Normal process:

In the normal process of the part program machining, the switching signal is activated for the first time at the beginning of part program block `N60`.

![Figure 15-4 Switching signal for part program machining operation](image1)

Process sequence after block search:

If a block search is executed for the block end point of part program block `N60` the switching signal is activated on reaching the start position of the geometry axes.

![Figure 15-5 Switching signal after block search](image2)
Suppressing the switching signal output

The user (machine manufacturer) must take appropriate measures, e.g. disable the switching signal, in order to suppress the activation of the switching signal in the REPOS block in the constellation described above.

Note

It is the sole responsibility of the user (machine manufacturer) to suppress the output of the switching signal during repositioning, e.g. after a block search.

References

You will find a description of the block search in:

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

15.6.2 Transformations

The function will only run correctly with deactivated transformation. There is no monitoring function.

For a description of the transformations (see Section "TE4: Handling transformation package - 840D sl only (Page 531)").

References:
Function Manual Extended Functions; Kinematic Transformation (M1)

15.6.3 Compensations

The following compensations are considered while calculating the switching positions:

- Temperature compensation
- Sag compensation

A description of the compensations can be found in:

Reference:
Function Manual, Extended Functions; Compensations (K3)
15.6.4 Tool radius compensation (TRC)

As part of tool radius compensation, control-internal part program blocks (compensation blocks) are inserted into the part program. With reference to the switching signal output, a compensation block is always added to the next programmed part program block.

A description of the tool radius compensation can be found in:
Reference: Function Manual, Basic Functions; Tool Compensations (W1), Section: Tool radius compensation

15.6.5 Continuous-path mode

Continuous-path mode

Although the CC_FASTON(), CC_FASTON_CONT() and CC_FASTOFF procedure calls must be programmed in dedicated part program blocks, this will not lead to a drop in velocity while continuous-path mode is active (G64, G641, ...).

Continuous-path mode (ADIS)

If in continuous-path mode with programmable rounding response (G641 ADIS) a part program block is inserted in the part program, then the originally programmed switching position is not reached and the switching signal output takes place nevertheless in the center of the rounding block.

References

You will find a description of the continuous-path mode in:
Function Manual, Basic Functions, Continuouspath Mode, Exact Stop and Look Ahead (B1)

15.6.6 Software cams

Because the hardware timer is also used for the "software cam" function, it is not possible to use the "clock-independent switching signal output" function with software cams at the same time.

The following alarm appears in the event of an error:
Alarm 75500 "Channel Channel No., wrong configuration of function: clock-independent switching signal output"

A description of the software cams can be found in:
Reference: Function Manual, Extended Functions; Software cams, Position switching signals (N3)
15.7 Data lists

15.7.1 Machine data

15.7.1.1 General machine data

| Number | Identifier: $MN_ | Description                     |
|--------|------------------|---------------------------------|
| 10360  | FASTO_NUM_DIG_OUTPUTS   | Number of digital output bytes  |

15.7.1.2 Channelspecific machine data

| Number | Identifier: $MC_ | Description                                                     |
|--------|------------------|-----------------------------------------------------------------|
| 20050  | AXCONF_GEOAX.Assign_TAB | Assignment of geometry axis to channel axis                  |
| 28090  | MM_NUM_CC_BLOCK_ELEMENTS | Number of block elements for CC                                 |
| 28100  | MM_NUM_CC_BLOCK_USER_MEM | Size of block memory for CC                                    |
| 62560  | FASTON_NUM_DIG_OUTPUT | Number of the on-board digital output for the switching signal |
TE9: Axis pair collision protection

16.1 Brief description

16.1.1 Brief description

Function

The "axis pair collision protection" technology function enables machine axes which are arranged on the same guide element of a machine to be monitored in pairs to ensure that no collisions occur and that the maximum distance between the two axes is not exceeded.

Function code

The code for the "axis pair collision protection" technology function for function-specific identifiers of machine data, system variables, etc., is:

PROTECT (axial collision PROTECTion)

References

The "axis pair collision protection" technology function is a compile cycle. For the handling of compile cycles, see Section "TE01: Installation and activation of loadable compile cycles (Page 433)".
16.2 Functional description

The "axis pair collision protection" function is a protection function for machine axes which are arranged in a machine tool in such a way (on the same guide rail, for example) that incorrect operation or programming could cause them to collide with one another.

The machine axes are always monitored in pairs, i.e. parameters always need to be assigned for two machine axes in each case, which are then monitored in relation to one another. The machine axes being monitored may be located in different machine coordinate systems.

Collision protection

The function uses the current actual positions and actual velocities, as well as the offset for the machine coordinate systems and the axis-specific brake acceleration values, to calculate the distance between the standstill positions of the machine axes, on a cyclical basis. If the resulting distance is shorter than the protection window that has been set, the machine axes are decelerated to a standstill. This ensures that the minimum distance defined using the protection window is not undershot.

Distance monitoring

If the offset vector is selected accordingly, the function can also be used to monitor the maximum distance between the two machine axes (maximum distance vector).

Monitoring status

The current status of a pair of machine axes can be read from an item of global user data (GUD), which you must define in the NC program if you wish to make use of this option.
16.3 Startup

16.3.1 Enabling the technology function (option)

The technology function is enabled via the following optional item of data:

\[ \text{MD19610 \$ON\_TECHNO\_EXTENSION\_MASK[6], BIT4 = 1} \]

16.3.2 Activating the technology function

Before commissioning the technological function, ensure that the corresponding compile cycle has been loaded and activated (see also Section "TE01: Installation and activation of loadable compile cycles (Page 433)").

Activating the technology function

The "axis pair collision protection" technology function must be activated for all NC channels to which the machine axes to be monitored will be assigned once the NC has powered up or to which they will be assigned during the course of machining, for example, due to an axis interchange. Activation is carried out via the following item of machine data:

\[ \text{MD60972 \$MN\_CC\_ACTIVE\_IN\_CHAN\_PROT[0], BITn = 1 where} \]
\[ \text{BIT0} = 1: \text{activation in the 1st channel of the NC} \]
\[ \text{BIT1} = 1: \text{activation in the 2nd channel of the NC} \]
, etc.
16.3 Startup

16.3.3 Definition of an axis pair

Definition

A pair of machine axes to be monitored is defined in the following item of machine data:

MD61516 $MN_CC_PROTECT_PAIRS[n] = yy xx
xx (1st and 2nd decimal place): axis number of the 1st machine axis
yy (3rd and 4th decimal place): axis number of the 2nd machine axis

Note

Axis number

The axis number $m$ of a machine axis relates to machine data MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[i]$ where $i = (m - 1)$.

| NOTICE |
|--------|
| Same axis types |
| The machine axes in an axis pair must be of the same axis type: |
| • Linear axis |
| • Rotary axis |
| Modulo rotary axes |
| The machine axes in an axis pair must not be of the "modulo rotary axis" axis type. |

16.3.4 Retraction direction

The direction of travel for retracting the corresponding machine axis is entered in the following item of machine data:

MD61517 $MN_CC_PROTECT_SAFE_DIR[n] = yy xx
xx = retraction direction for the 1st axis in the pair
yy = retraction direction for the 2nd axis in the pair

Retraction in the positive direction of travel of the machine axis: $xx$ or $yy > 0$
Retraction in the negative direction of travel of the machine axis: $xx$ or $yy = 0$

| NOTICE |
|--------|
| Machine data change |
| A change to machine data item $MN_{...SAFE_DIR}[n]$ must only be activated when the protection function is inactive ($MN_{...PAIRS[n]} = 0$). |

Special functions

Function Manual, 07/2012, 6FC5397-2BP40-3BA0
16.3.5 Offset for the machine coordinate systems
The offset vector for the machine coordinate systems of both machine axes in the axis pair is specified in the following item of machine data:

\[ \text{MD61518 $MN\_CC\_PROTECT\_OFFSET[n] = \langle offset \rangle} \]

If both machine axes are located in separate machine coordinate systems, the offset vector is specified as a vector from the origin of the machine coordinate system for the 2nd axis in the pair to the origin of the machine coordinate system for the 1st axis in the pair, with reference to the machine coordinate system for the 1st axis of the pair.

If both machine axes in the axis pair are located in the same machine coordinate system, an offset vector of 0 must be specified.

**NOTICE**

| Machine data change |
|---------------------|
| A change to machine data item $MN\_...\_OFFSET[n]$ must only be activated when the protection function is inactive ($MN\_...\_PAIRS[n] = 0$). |

16.3.6 Protection window
The protection window is used to define the minimum distance, which the two machine axes specified in $MN\_...\_PAIRS[n]$ must not undershoot. Parameters are assigned for the protection window in the following item of machine data:

\[ \text{MD61519 $MN\_CC\_PROTECT\_WINDOW[n] = \langle minimum\ distance \rangle} \]

**Note**

| Machine data change |
|---------------------|
| A change to machine data item $MN\_...\_WINDOW[n]$ may also be activated when the protection function is active ($MN\_...\_PAIRS[n] \neq 0$). |
16.3.7 Orientation

The orientation of the machine axes to one another is specified in the following item of machine data:

\[ \text{MD61532} \; \text{MN CC PROTECT_DIR IS REVERSE}[n] = \langle \text{orientation} \rangle \]

\[ \text{orientation} = 0: \text{orientation in the same direction}, \]
\[ \text{orientation} = 1: \text{orientation in the opposite direction} \]

**NOTICE**

**Machine data change**

A change to machine data item $\text{MN}_\ldots_\text{DIR IS REVERSE}[n]$ must only be activated when the protection function is inactive ($\text{MN}_\ldots_\text{PAIRS}[n] = 0$).

16.3.8 Protection window extension

The protection window for which parameters are assigned in $\text{MN}_\ldots_\text{WINDOW}[n]$ can be expanded using the protection window extension function. The protection window extension is set in the following item of machine data:

\[ \text{MD61533} \; \text{MN CC PROTECT_WINDOW EXTENSION}[n] = \langle \text{extension} \rangle \]

The resulting effective protection window is as follows:

- Effective protection window$[n] = \text{protection window} \; \text{MN}_\ldots_\text{WINDOW}[n] + \text{protection window extension} \; \text{MN}_\ldots_\text{WINDOW EXTENSION}[n]$

**Note**

**Machine data change**

A change to machine data item $\text{MN}_\ldots_\text{WINDOW EXTENSION}[n]$ may also be made and activated (NewConfig) when the protection function is active, for example, from the part program.

16.3.9 Activating the protection function

The protection function is active if the following conditions are met:

- A valid pair of machine axes has been entered in machine data item $\text{MN}_\ldots_\text{PAIRS}[n]$
- Machine data item $\text{MN}_\ldots_\text{PAIRS}[n]$ is active
- Both machine axes have been referenced

If, at the time when the protection function is activated, the distance between the two machine axes is shorter than the minimum distance for which parameters have been assigned, the machine operator must retract the machine axes. The protection function prevents the axes from being traversed toward one another in this case.
16.3.10 Axis-specific acceleration

Parameters are assigned for the accelerations which the protection function uses in order to decelerate the machine axes in the following axis-specific item of machine data:

MD63514 $MA_CC_PROTECT_ACCEL = <acceleration of the protection function>

16.3.11 Monitoring status (GUD)

The current statuses of the individual protection functions are shown in an item of global user data. This item of user data is optional and does not have to be defined.

Definition

DEF NCK INT _PROTECT_STATUS[ n ]
where n = maximum number of possible axis pairs (currently, n = 5)

| Value | Meaning                                                      |
|-------|--------------------------------------------------------------|
| 0     | Not active                                                  |
| 1     | Activated, but not yet active (machine axes have not yet been referenced) |
| 2     | Active                                                      |
| 3     | Decelerate axes                                             |
16.4 Limitations and constraints

16.4.1 Precedence of function-specific acceleration

The protection function only uses the function-specific acceleration of the machine axes MD63514 $MA_CC_PROTECT_ACCEL to calculate the time at which deceleration should be performed. This means that the protection function does not take the current acceleration of the machine axis in the channel into account.

**NOTICE**

Path reference

If machine axes being monitored by the protection function are traversed by a channel with a path reference to other axes, this path reference is lost as soon as the protection function causes the axis grouping to decelerate. The machine axes being monitored by the protection function are decelerated using their function-specific acceleration values from machine data item MD63514 $MA_CC_PROTECT_ACCEL. The remaining axes in the axis grouping are decelerated using the current path acceleration value of the channel.

16.4.2 Axis container

If the assignment of the machine axes to be monitored changes dynamically during the course of a machining process - when using axis containers, for example - the protection function must be deactivated, its parameters reassigned, and then reactivated prior to the change (in this case, the axis container rotation) being made.

**Example**

The protection function is to monitor logical machine axes 1 and 13. These relate to slots 1 and 2 of axis container CT1. The associated real machine axes are AX1 and AX13.

In an axis container rotation, the axis container makes a transition by one step, which causes the real machine axes to be changed.

Machine axis AX13 is retracted in the positive direction of travel. Machine axis AX1 is retracted in the negative direction of travel.
Assigning parameters for the NC
Logical machine axes: axis numbers 1 and 13

- MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB [ 0 ] = "CT1_SL1" (log. mach. axis 1)
- MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB [ 12 ] = "CT1_SL2" (log. mach. axis 13)

Axis container CT1, slot 1 and slot 2

- MD12750 $MN_AXCT_NAME_TAB[ 0 ] = "CT1"
- MD12701 $MN_AXCT_AXCONF_ASSIGN_TAB1[ 0 ] = "AX1" (slot 1)
- MD12701 $MN_AXCT_AXCONF_ASSIGN_TAB1[ 1 ] = "AX13" (slot 2)

Real machine axes: machine axis identifiers AX1 and AX13

- MD10000 $MN_AXCONF_MACHAX_NAME_TAB[ x ] = "AX1"
- MD10000 $MN_AXCONF_MACHAX_NAME_TAB[ y ] = "AX13"

Assigning parameters for the protection function prior to the axis container rotation

- MD61516 $MN_CC_PROTECT_PAIRS[0] = 01 13
- MD61517 $MN_CC_PROTECT_SAVE_DIR[0] = 01 00

Performing the axis container rotation
1. Deactivate the protection function
   MD61516 $MN_CC_PROTECT_PAIRS[0] = 00 00
2. Trigger a reset in the 1st NC channel in order to adopt the change made to the machine data
   MD60972 $MN_CC_ACTIVE_IN_CHAN_PROT[0], BITx, ...
3. Execute the axis container rotation
   AXCTSWED(CT1)

Reassigning parameters for the protection function after the axis container rotation

- MD61516 $MN_CC_PROTECT_PAIRS[0] = 13 01
- MD61517 $MN_CC_PROTECT_SAFE_DIR[0] = 01 00
  or
- MD61516 $MN_CC_PROTECT_PAIRS[0] = 01 13
- MD61517 $MN_CC_PROTECT_SAFE_DIR[0] = 00 01

Trigger a reset in the 1st NC channel in order to adopt the change made to the machine data
16.5 Examples

16.5.1 Collision protection

The figure shows the arrangement of 3 machine axes and the offset and orientation of the machine coordinate systems (machine).

![Diagram showing the arrangement of 3 machine axes and the offset and orientation of the machine coordinate systems.]

Figure 16-2 Collision protection for 2 axis pairs

Parameter assignment: Protection function 1

Axis pair: 1st machine axis A3, 2nd machine axis A1

- MD61516 $MN_CC_PROTECT_PAIRS[0] = 01 03

Retraction direction: A1 in negative direction, A3 in positive direction

- MD61517 $MN_CC_PROTECT_SAFE_DIR[0] = 00 01

Offset vector from machine coordinate system machine_A1 to machine_A3 with reference to machine_A3

- MD61518 $MN_CC_PROTECT_OFFSET[0] = 70.0

Example protection window, 10.0 mm

- MD61519 $MN_CC_PROTECT_WINDOW[0] = 10.0

Orientation of the machine coordinate systems to one another: same direction

- MD61532 $MN_CC_PROTECT_DIR_IS_REVERSE[0] = 0

Protection window extension: none

- MD61533 $MN_CC_PROTECT_WINDOW_EXTENSION[0] = 0.0
Parameter assignment: Protection function 2

Axis pair: 1st machine axis A1, 2nd machine axis A12
- MD61516 $MN_CC_PROTECT_PAIRS[1] = 12 01

Retraction direction: A12 in positive direction, A1 in positive direction
- MD61517 $MN_CC_PROTECT_SAFE_DIR[1] = 01 01

Offset vector from machine coordinate system machine_A12 to machine_A1 with reference to machine_A1
- MD61518 $MN_CC_PROTECT_OFFSET[1] = 32.0

Example protection window, 5.0 mm
- MD61519 $MN_CC_PROTECT_WINDOW[1] = 5.0

Orientation of the machine coordinate systems to one another: opposite direction
- MD61532 $MN_CC_PROTECT_DIR_IS_REVERSE[1] = 1

Protection window extension: by 5.0 mm to give a total of 10.0 mm
- MD61533 $MN_CC_PROTECT_WINDOW_EXTENSION[1] = 5.0
16.5.2 Collision protection and distance limiter

The figure shows the arrangement of two machine axes, the offset and orientation of the machine coordinate systems (machine), and the minimum and maximum distance vectors.

Parameter assignment: Protection function 1 - Collision protection

Axis pair: 1st machine axis A1, 2nd machine axis A3

- MD61516 $MN_CC_PROTECT_PAIRS[0] = 03 01

Retraction direction: A1 in negative direction, A3 in positive direction

- MD61517 $MN_CC_PROTECT_SAFE_DIR[0] = 01 00

Offset vector from machine coordinate system machine_A3 to machine_A1 with reference to machine_A1

- MD61518 $MN_CC_PROTECT_OFFSET[0] = -100.0

Example protection window, 40.0 mm

- MD61519 $MN_CC_PROTECT_WINDOW[0] = 40.0

Orientation of the machine coordinate systems to one another: same direction

- MD61532 $MN_CC_PROTECT_DIR_IS_REVERSE[0] = 0

Protection window extension: none

- MD61533 $MN_CC_PROTECT_WINDOW_EXTENSION[0] = 0.0
Parameter assignment: Protection function 2 - Distance limiter

Axis pair: 1st machine axis A1, 2nd machine axis A3

- MD61516 $MN_CC_PROTECT_PAIRS[1] = 03 01

Retraction direction: A1 in positive direction, A3 in negative direction

- MD61517 $MN_CC_PROTECT_SAFE_DIR[1] = 00 01

Offset vector = "offset vector from machine coordinate system machine_A3 to machine_A1 with reference to machine_A1" - "maximum distance vector with reference to machine_A1"

**Note**

**Maximum distance vector**

The maximum distance vector from the 1st machine axis to the 2nd machine axis is the vector from the origin of the machine coordinate system for the 1st machine axis to the maximum permissible position of the 2nd machine axis, with reference to the machine coordinate system for the 1st machine axis.

- MD61518 $MN_CC_PROTECT_OFFSET[1] = -100.0 - 500.0 = 400.0

Example protection window, 20.0 mm

- MD61519 $MN_CC_PROTECT_WINDOW[1] = 20.0

Orientation of the machine coordinate systems to one another: same direction

- MD61532 $MN_CC_PROTECT_DIR_IS_REVERSE[1] = 0

Protection window extension: none

- MD61533 $MN_CC_PROTECT_WINDOW_EXTENSION[1] = 0.0

If machine axis A1 has a value of 0, the settings above will limit the traversing range of machine axis A3 to between -60.0 and 380.0, with reference to machine_A3.
16.6 Data lists

16.6.1 Option data

| Number | Description | Identifier: $ON_ |
|--------|-------------|------------------|
| 19610  | Enable the technology function via BIT4 = 1 | TECHNO_EXTENSION_MASK[6] |

16.6.2 Machine data

16.6.2.1 NC-specific machine data

| Number | Description | Identifier: $MN_ |
|--------|-------------|------------------|
| 60972  | Channel-specific activation of the technology function BIT0 = 1 => activation in the 1st NC channel BIT1 = 1 => activation in the 2nd NC channel, etc. | CC_ACTIVE_IN_CHAN_PROT[n] |
| 61516  | Definition of the axis pair = yy xx xx = axis number of 1st machine axis (1st/2nd decade) yy = axis number of 2nd machine axis (3rd/4th decade) | CC_PROTECT_PAIRS[n] |
| 61517  | Retraction direction = yy xx xx = retraction direction of 1st machine axis (1st/2nd decade) yy = retraction direction of 2nd machine axis (3rd/4th decade) | CC_PROTECT_SAFE_DIR[n] |
| 61518  | Offset of the two machine coordinate systems | CC_PROTECT_OFFSET[n] |
| 61519  | Protection window or minimum distance | CC_PROTECT_WINDOW[n] |
| 61532  | Orientation of the machine coordinate systems of the two machine axes to one another 0: orientation in the same direction 1: orientation in the opposite direction | CC_PROTECT_DIR_IS_REVERSE[n] |
| 61533  | Protection window extension | CC_PROTECT_WINDOW_EXTENSION[n] |

16.6.2.2 Axis/Spindle-specific machine data

| Number | Description | Identifier: $MA_ |
|--------|-------------|------------------|
| 61514  | Protection function-specific brake acceleration | CC_PROTECT_ACCEL |

16.6.3 User data

16.6.3.1 Global user data (GUD)

| Identifier | Description |
|------------|-------------|
| _PROTECT_STATUS[n] | Status of the protection function (optional) |
17.1 Brief description

Preprocessing

The programs stored in the directories for standard and user cycles can be preprocessed to reduce runtimes.

Preprocessing is activated via machine data.

Standard and user cycles are preprocessed when the power is switched on, i.e. as an internal control function, the part program is translated (compiled) into a binary intermediate code optimized for processing purposes.

All program errors that can be corrected by means of a compensation block are detected during preprocessing. In addition, when the program includes branches and check structures, a check is made to ensure that the branch destinations are present and that structures are nested correctly.

The full scope of control functionality is available:

- Override influence
- Reactions to data and signals that are input by the PLC or the operator
- Current block display
- The programs can be processed in single block mode (SBL1 and SBL2). Block searches can be executed. The compilation cannot be stored; it is concealed from the user and regenerated every time the power is switched on.

Preprocessing can be used:

- To optimize the runtimes of part programs with high-level language components (branches, check structures, motion-synchronous actions)
- CPU time intensive part programs (e.g. stock removal cycles)
- Faster processing of time-critical sections (e.g. program continuation after preprocessing stop during rapid deletion of distance-to-go, or return stroke, or in the tool change cycle).
**General information**

Preprocessing standard and user cycles is possible. The processing time of part programs can then be reduced without restricting the control functionality.

The standard and user cycles are preprocessed when machine data is set accordingly:

- \texttt{MD10700 \$MN\_PREPROCESSING\_LEVEL} (program preprocessing level)

Preprocessing is carried out program-specifically. It is possible to mix preprocessed part programs and part programs interpreted in ASCII format. Preprocessing serves to reduce incidental times.

Memory is required for preprocessing cycles. You can optimize your memory in two ways:

- The program to be executed can be shortened with the command \texttt{DISPLOF} (display off).
- \texttt{MD10700 \$MN\_PREPROCESSING\_LEVEL} has been expanded by bit 2 and 3. This allows selective cycle preprocessing of the individual directories (e.g. user cycles).

\texttt{MD10700 \$MN\_PREPROCESSING\_LEVEL} has been expanded by bit 4. This allows you to select preprocessing for user cycles from the \_N\_CMA\_DIR directory.

\texttt{MD10700 \$MN\_PREPROCESSING\_LEVEL} has been expanded by bit 5. This allows selective preprocessing of the specific individual user cycles that have the command \texttt{PREPRO} after the \texttt{PROC} instruction.

Pretranslated cycles are stored in the dynamic NC memory by default. \texttt{MD10700 \$MN\_PREPROCESSING\_LEVEL} has been expanded by bit 6. This allows specifying that the compiled programs that are now stored in the dynamic NC memory and no longer have enough space can be stored in static NC memory.

**Functionality**

The programs stored in the directories for standard and user cycles are preprocessed when the power is switched on, i.e. the part program is translated (compiled) into an intermediate binary code optimized for processing purposes. The compilation is processed when called.
Runtime optimization

The preprocessing function is primarily suited for optimizing the runtimes of part programs with high-level language components (branches, check structures, motion-synchronous actions).

While branches and check structures are invalidated by a search through all blocks (block start) when part programs are interpreted in ASCII format (active as default), a branch is made directly to the destination block in a preprocessed part program.

The runtime differences between branches and check structures are thus eliminated.

Example of a preprocessing runtime:

Runtime reduction by 30% with active compressor

```plaintext
DEF INT COUNTER
Target: G1 G91 COMPON
G1 X0.001 Y0.001 Z0.001 F100000
COUNTER=COUNTER +1
COUNTER=COUNTER -1
COUNTER=COUNTER +1
IF COUNTER<= 100000 GOTOB TARGET
```

CPU time intensive programs and programs with symbolic names are processed faster.

Runtime-critical sections (e.g. continuation of processing after deletion of distance-to-go or preprocessing stop in cycles) can be processed faster.

If the interrupt routine is available as a preprocessed cycle, processing can be continued more rapidly after the program interrupt.
17.2 Program handling

Activation/Deactivation

Cycles are preprocessed on POWER ON if the following machine data is set:
MD10700 $MN_PREPROCESSING_LEVEL, bit 1 (program preprocessing level)

| Bit | Value | Meaning |
|-----|-------|---------|
| 0   | 0     | No preprocessing |
| 1   | 1     | During control power-up, the call description of the cycles is generated: All user cycles (_N_CUS_DIR directory) and Siemens cycles (_N_CST_DIR directory) with transfer parameters can be called up without external statement. Changes to the cycle-call interface do not take effect until the next POWER ON. The following machine data must be set: MD18170 $MN_MM_NUM_MAX_FUNC_NAMES (number of auxiliary actions) MD18180 $MN_MM_NUM_MAX_FUNC_PARAM (number of auxiliary parameters for cycles) |
| 1   | 2     | During control power-up, the standard cycles in the _N_CST_DIR directory are preprocessed in a compilation optimized for processing. |
| 1   | 3     | During control power-up, the user cycles in the _N_CUS_DIR directory are preprocessed in a compilation optimized for processing. |
| 1   | 4     | Preprocessing of user cycles from the directory _N_CMA_DIR |
| 1   | 5     | Preprocessing the user cycles with the PREPRO command in the PROC instruction line. Unmarked files of directories marked with bits 1-4 are not preprocessed. If bit 0, then control of the preprocessing is carried out in accordance with the specifications of bits 0-4. |
| 0   | 6     | The compilation is stored in the dynamic NC memory as long as free memory is still available. If sufficient memory is not available, preprocessing is aborted. The size of the dynamic NC memory is obtained with this machine data: MD18351 $MN_MM_DRAM_FILE_MEM_SIZE. |

The areas occupied in the dynamic NC memory by the compilation are visible to the user. Bit combinations are permissible.
Compiling

In the directories for standard cycles: _N_CST_DIR, _N_CMA_DIR and user cycles: _N_CUS_DIR, _N_CUS_DIR and user cycles: _N_CUS_DIR, and, if necessary, the subprograms (_SPF file extension) located in _N_CUS_DIR, and, if necessary, the subprograms marked with PREPRO are compiled. The name of the compilation corresponds to the original cycle with extension _CYC.

Note

Program changes to precompiled programs do not take effect until the next POWER ON.

Access rights

The preprocessed program can only be executed, but not read or written. The compilation cannot be modified or archived. The original cycles _SPF files are not deleted.

The compilation is not changed when the ASCII cycle is altered, i.e. changes do not take effect until after the next POWER ON.

Memory requirements

The memory requirement for compiled cycles is approximately factor 2 in addition to the ASCII part program.

The memory requirements for variables defined in the part programs are defined by the following machine data:

- MD28020 $MC_MM_NUM_LUD_NAMES_TOTAL (number of local user variables)
- MD28010 $MC_MM_NUM_REORG_LUD_MODULES (number of modules for local user variables with REORG)
- MD28040 $MC_MM_LUD_VALUES_MEM (memory size for local user variables)
- MD18242 $MC_MM_MAX_SIZE_OF_LUD_VALUE (memory block size for LUD/GUD values)

References:

Function Manual, Extended Functions; Memory Configuration (S7)

While preprocessing is in progress, the amount of memory required is the same as if the preprocessed program were called on the first subprogram level.

When programs are preprocessed after POWER ON, a name is counted for each branch destination/label as if it were a variable. These names must be taken into account in the following machine data:

- MD28020 $MC_MM_NUM_LUD_NAMES_TOTAL (number of local user variables)
17.2 Program handling

Example:

| Program code                                      | Comment                          |
|--------------------------------------------------|---------------------------------|
| PROC NAMES ; 1 name                              |                                 |
| DEF INT VARIABLE, ARRAY[2] ; 2 names             |                                 |
| START: ; 1 name, only for preprocessing           |                                 |
| FOR VARIABLE = 1 TO 9 ; 1 name, only for preprocessing |                          |
| G1 F10 X=VARIABLE*10-56/86EX4+4*SIN(VARIABLE/3)   |                                 |
| ENDFOR ; 1 name, only for preprocessing           |                                 |
| M17                                              |                                 |

In order to execute this program normally, the following machine data must specify at least 3 names:

MD28020 $MC_MM_NUM_LUD_NAMES_TOTAL

Six names are required to compile this program after POWER ON.

Preprocessed programs/cycles are stored in the dynamic NC memory. The space required for each program must be flashed over unmodified as outlined above. Adjustment of the memory mapping in the static NC memory is required only if bit 6 = 1 is set in the following machine data:

MD10700 $MN_PREPROCESSING_LEVEL (program processing level)

In this case, the program compilations for which there is insufficient space in the dynamic NC memory are stored in the static NC memory.

Examples for appropriate machine data settings can be found under "Examples" in the Subsection "Preprocessing in the dynamic NC memory".
17.3 Program call

Overview

Figure 17-1  Generation and call of preprocessed cycles without parameters

Figure 17-2  Generation and call of preprocessed cycles with parameters
Start

- Compiled cycle: A compiled cycle is called in exactly the same way as a normal subroutine.
  
  Example: CYCLE

- Preprocessing is activated: The compiled cycle is called instead of the ASCII cycle.
  - If the subroutine is called explicitly with extension _SPF, then the ASCII cycle is called even if a compilation is available.
    
    Example: CYCLE_SPF; ASCII cycle call
  - If the subroutine is called explicitly with extension _CYC, then the preprocessed cycle is called if available. An error message is output if no compilation is available.
    
    Example: CYCLE_CYC; Preprocessed cycle call
  - If bit 5 is set and a file that is not marked with PREPRO is called explicitly with the extension _CYC, an error message is issued with Alarm 14011.

- If a subroutine is called without explicit extension, an attempt is first made to load the program. If this is not possible (not marked with PREPRO), an attempt is made to load the SPF program.

- The change to an external language mode with G291 is rejected and an alarm issued. When the pre-compiled cycle is called, an explicit change is made to the Siemens language mode.

- When the subroutine is called, it is checked whether the compiled file is older than the cycle. If so, the compile file is deleted and an alarm issued. The user must preprocess the cycles again.

Note

Only cycles without parameters may be called with the extension _SPF or _CYC.

Do not use PUDs in cycles that are preprocessed. The PUDs are created in the calling main program. At the time of compilation after POWER ON, this data is not known to the cycles.

The current program display shows whether the current ASCII cycle or the compilation has been called (extension _SPF or _CYC).
Call condition

All cycles in the cycle directories must be compiled before preprocessing is activated. Non-compiled cycles in _N_CUS_DIR and _N_CST_DIR which were loaded after POWER ON, for example, can only be called through explicit specification of extension _SPF.

If preprocessing is active and bit 5 is set, all programs that do not start with the `PREPRO PROC` instruction are not precompiled.

Syntax check

All program errors that can be corrected by means of a compensation block are detected during preprocessing. In addition, when the program includes branches and check structures, a check is made to ensure that the branch destinations are present and that structures are nested correctly.

Branch destinations and labels must be unique in the program.

After the errors detected during preprocessing have been corrected, preprocessing must be started again by means of an NCK POWER ON.
17.4 Constraints

Availability of the "preprocessing" function

The function is an option ("Program pre-processing"), which must be assigned to the hardware via the license management.

Vocabulary

The full vocabulary of the NC language is available in the part program.
There are no restrictions on the calculation of measured process variables and in the reaction to signals from the process and other channels (override, deletion of distance-to-go, motion-synchronous actions, channel coordination, interrupt processing, etc.).

Axis identifier

Part programs are compiled independently of channels. For this reason, the geometry and channel identifiers set in the following machine data must be identical in all channels if they are used directly in the precompiled cycles:

- MD20060 $MC_AXCONF_GEOAX_NAME_TAB (name of the geometry axis in the channel)
- MD20080 $MC_AXCONF_CHANAX_NAME_TAB (name of the channel axis in the channel)

Generally speaking, axis identifiers are not used directly in machining cycles since cycles are written as follows:
- independently of channels and
- independently of the axis identifiers defined on the machine.

The axes to be traversed are addressed indirectly via machine data or transferred as parameters:
- Indirect axis programming:
  - IF $AA_IM[AXNAME($MC_AXCONF_CHANAX_NAME_TAB[4])] > 5
    ; This branch will pass through if the actual value of the 5th channel axis
    ; with reference to the machine coordinate system is greater than 5.
  - G1 AX[AXNAME($MC-AXCONF-GEOAX-NAME-TAB[0])] = 10
    F1000 G90.
    ; Traverse the 1st geometry axis to the value 10.
  ENDF
• Transfer of axis to be traversed from the main program:
  - Cycle definition
    PROC DRILL(AXIS DRILL_AXIS)
    WHILE $AA_IW[DRILL_AXIS] > -10
    G1 G91 F250 AX[DRILL_AXIS] = -1
    ENDWHILE
  - Call from the main program
    DRILL(Z)
17.5 Examples

17.5.1 Preprocessing individual files

| Program code                                                                 | Comment                                      |
|------------------------------------------------------------------------------|----------------------------------------------|
| PROC UP1 PREPRO                                                              | ; Preprocessing if bit 5 = 1                |
| ; in PREPROCESSING_LEVEL                                                      |                                              |
| N1000 DEF INT COUNTER                                                       |                                              |
| N1010 TARGET: G1 G91 COMPON                                                 |                                              |
| N1020 G1 X0.001 Y0.001 Z0.001 F100000                                       |                                              |
| N1030 COUNTER=COUNTER+1                                                     |                                              |
| N1040 COUNTER=COUNTER-1                                                     |                                              |
| N1050 COUNTER=COUNTER+1                                                     |                                              |
| N1060 IF COUNTER<=10 GOTOB TARGET                                            |                                              |
| N1070 M30                                                                  |                                              |
| PROC UP2                                                                    |                                              |
| N2000 DEF INT VARIABLE, ARRAY[2]                                            |                                              |
| N2010 IF $AN_NCK_Version < 3.4                                              |                                              |
| N2020 SETAL(61000)                                                          |                                              |
| N2030 ENDIF                                                                 |                                              |
| N2040 START:                                                                |                                              |
| N2050 FOR VARIABLE = 1 TO 5                                                 |                                              |
| N2060 G1 F1000 X=VARIABLE*10-56/86`EX4+4*SIN(VARIABLE/3)                    |                                              |
| N2070 ENDFOR                                                                |                                              |
| N2080 M17                                                                   |                                              |
| PROC MAIN                                                                   |                                              |
| N10 G0 X0 Y0 Z0                                                             |                                              |
| N20 UP1                                                                     |                                              |
| N30 G0 X10 Y10 Z10                                                          |                                              |
| N40 UP2                                                                     |                                              |
| N50 G0 X100 Y100                                                            |                                              |
| N60 UP3                                                                     |                                              |
| N70 G0 X10 Y10                                                              |                                              |
| N80 M30                                                                     |                                              |
Sample constellations:
a) Bit 5 = 1

MD10700 $MN_PREPROCESSING_LEVEL=45 ; bit 0, 2, 3, 5
Subprogram UP1 is pretranslated, and the call description is generated.
Subprogram UP2 is not pretranslated, but the call description is generated.
b) Bit 5 = 0

MD10700 $MN_PREPROCESSING_LEVEL=13 ; bit 0, 2, 3,
Both subprograms are pretranslated, and the call description is generated.
c) Example of an invalid subprogram with activated compiling:

| Program code               | Comment                           |
|----------------------------|-----------------------------------|
| PROC SUB1 PREPRO G291     | ← Alarm when compiling, G291 not possible |
| G0 X0 Y0 Z0               |                                   |
| M17                       |                                   |

17.5.2 Preprocessing in the dynamic NC memory

Machine data for preprocessing only in the dynamic NC memory with selective selection:

| Program code | Comment                           |
|--------------|-----------------------------------|
| ; Bit 5 = 1  | Selective program selection       |
| ; Bit 6 = 0  | no switch to                      |
| ; static NC memory if dynamic NC memory is full |
| N30 $MN_MM_DRAM_FILE_MEM_SIZE = 800 | Reserve space |
| N40 $MN_PREPROCESSING_LEVEL = 63  | Bit 0-5 = 1                       |
| M17          |                                   |

Machine data for preprocessing in the dynamic NC memory with the option of using the static NC memory and selective selection:

| Program code | Comment                           |
|--------------|-----------------------------------|
| ; Bit 5 = 1  | Selective program selection       |
| ; Bit 6 = 1  | switch to static                  |
| ; NC memory if dynamic NC memory full |
| N30 $MN_MM_DRAM_FILE_MEM_SIZE = 800 | Reserve space |
| N40 $MN_PREPROCESSING_LEVEL = 127  | Bit 0-6 = 1                       |
| M17          |                                   |
17.6 Data lists

17.6.1 Machine data

17.6.1.1 General machine data

| Number | Identifier: $MN_ | Description                        |
|--------|------------------|------------------------------------|
| 10700  | PREPROCESSING_LEVEL | Program preprocessing level         |
| 18242  | MM_MAX_SIZE_OF_LUD_VALUE | Maximum LUD-variable array size   |

17.6.1.2 Channelspecific machine data

| Number | Identifier: $MC_ | Description                                                                 |
|--------|------------------|-----------------------------------------------------------------------------|
| 28010  | MM_NUM_REORG_LUD_MODULES | Number of blocks for local user variables for REORG (DRAM)                 |
| 28020  | MM_NUM_LUD_NAMES_PER_PROG | Number of local user variables (DRAM)                                        |
| 28040  | MM_LUD_VALUES_MEM    | Memory size for local user variables (DRAM)                                 |
18.1 Brief description

18.1.1 General

Why 3D TRC?

3D tool radius compensation is used to machine contours with tools that can be controlled in their orientation independently of the tool path and shape.

---

Note

This description is based on the specifications for 2D tool radius compensation.

References:
Function Manual Basic Functions; Tool Offset (W1)

---

How 2½ D and 3D TRC differ

- **With 2½D TRC**, it is assumed that the tool is always space-bound. Tools with constant orientation (cylindrical tools) are used for circumferential milling operations.

  While the orientation of the machining surface is not constant when other tools are used, it is determined by the contour and cannot thus be controlled independently of it.

- **With 3D TRC**, surfaces with variable orientation are generated.

  The prerequisite for circumferential milling is that the tool orientation can be changed, i.e. in addition to the 3 degrees of freedom needed to position the tool (normally 3 linear axes), a further two degrees of freedom (2 rotary axes) are required to set the tool orientation (5-axis machining).

  End faces can be milled with 3 or 5 degrees of freedom.
Circumferential milling, face milling

The following diagram shows the differences between 2½D and 3D TRC with respect to circumferential milling operations.

![Diagram showing 2½D and 3D tool radius compensation](image)

**Figure 18-1 2½D and 3D tool radius compensation**

The parameters for display in the "Face milling" screen are described in detail in Chapter "Face milling (Page 694)".

![Diagram showing face milling](image)

**Figure 18-2 Face milling**
Orientation

With 3D TRC, a distinction must be drawn between:

- Tools with space-bound orientation
- Tools with variable orientation

18.1.2 Machining modes

There are two modes for milling spatial contours:

- Circumferential milling
- Face milling

Circumferential milling mode is provided for machining so-called ruled surfaces (e.g. taper, cylinder, etc.) while face milling is used to machine curved (sculptured) surfaces.

Circumferential milling

Tools will be applied as follows for circumferential milling:

- with space-bound orientation (2.5D TRC) and
- with variable orientation (3D TRC)

3D TRC can therefore be applied in circumferential milling only if the tool orientation is variable.

Intermediate blocks that are required from non-tangential transitions for mathematical reasons can be avoided using the intersection procedure. In these cases, the two curves in question are extended; the intersection of both extended curves is approached.

Face milling

Tools of both types, i.e. with constant or variable orientation, can be used for face milling operations.

Tools with variable orientation offer the following advantages:

- Better approximation of end contour
- Greater cutting capability
- Wider selection of tool shapes
- Wider range of surfaces can be machined (relief cuts).
18.2 Circumferential milling

Circumferential milling

The variant of circumferential milling used here is implemented through the definition of a path (directrix) and the associated orientation. In this machining mode, the tool shape is irrelevant on the path and at the outside corners. The only decisive factor is the radius at the tool contact point.

![Circumferential milling diagram](image)

Figure 18-3 Circumferential milling

Insertion depth (ISD)

Program command ISD (insertion depth) is used to program the tool insertion depth for circumferential milling operations. This makes it possible to change the position of the machining point on the outer surface of the tool.

ISD defines the distance between cutter tip FS and cutter construction point FH. Point FH is obtained by projecting the programmed machining point onto the tool axis. ISD is evaluated only when 3D TRC is active.

![Insertion depth diagram](image)

Figure 18-4 Insertion depth
18.2.1 Corners for circumferential milling

Outside corners/inside corners

Outside corners and inside corners must be treated separately. The terms inside corner and outside corner are dependent on the tool orientation. When the orientation changes at a corner, for example, the corner type may change while machining is in progress. Whenever this occurs, the machining operation is aborted with an error message.

Figure 18-5 Corner type

Figure 18-6 Change of corner type during machining
18.2.2 Behavior at outer corners

In the same manner as 21/2D tool radius compensation procedures, a circle is inserted at outer corners for G450 and the intersection of the offset curves is approached for G451.

With nearly tangential transitions, the procedure for active G450 is as with G451 (limit angle is set via machine data). Conversely, if G451 is active, a circle is also inserted (procedure as for G450) if there is no intersection or if the corner angle exceeds a specific value (MD).

If there is a change in orientation between the two traversing blocks, a circle is always inserted.

G450

Outside corners are treated as if they were circles with a 0 radius. The tool radius compensation acts on these circles in the same way as on any other programmed path.

The circle plane extends from the final tangent of the first block to the start tangent of the second block.

The orientation can be changed during block transition.

A change in orientation between two programmed blocks is executed either before the circle block or in parallel to it. Circles are always inserted. The command DISC is not evaluated.

Programming

- ORIC: Change in orientation and path movement in parallel
  (ORIentation Change Continuously)
- ORID: Change in orientation and path movement consecutively
  (ORIentation Change Discontinuously)

The ORIC and ORID program commands are used to determine whether changes in orientation programmed between two blocks are executed before the inserted circle block is processed or at the same time.

When the orientation needs to be changed at outside corners, the change can be implemented in parallel to interpolation or separately from the path motion. When ORID is programmed, the inserted blocks are executed first without a path motion (blocks with changes in orientation, auxiliary function outputs, etc.). The circle block is inserted immediately in front of the second of the two traversing blocks which form the corner.

ORIC

If ORIC is active and there are two or more blocks with changes in orientation (e.g. A2= B2= C2=) programmed between the traversing blocks, then the inserted circle block is distributed among these intermediate blocks according to the absolute changes in angle.
Change in orientation

The method by which the orientation is changed at an outer corner is determined by the program command that is active in the first traversing block of an outer corner.

![Diagram showing change in orientation and path movement in parallel.]

**Example:**

| Program code | Comment |
|--------------|---------|
| N10 A0 B0 X0 Y0 Z0 F5000 ; Radius=5 |
| N20 T1 D1 ; Transformation selection |
| N30 TRAORI (1) ; 3D TRC selection |
| N40 CUT3DC |
| N50 ORIC ; TRC selection |
| N60 G42 X10 Y10 |
| N70 X60 ; Change in orientation to the for N70 |
| N80 A3=1 B3=0 C3=1 |
| N90 Y60 ; and N90 generated outside corner |
| N100 X10 |
| N110 G40 X0 Y0 |
| N120 M30 |

The circular motion and change in orientation are executed in parallel in block N80 (ORIC active).
Special case

Intermediate blocks without traversing and orientation motions are executed at the programmed positions, e.g. auxiliary functions.

Example:

| Program code   | Comment                                      |
|----------------|----------------------------------------------|
| N70 X60        |                                              |
| N75 M20        | ; Auxiliary function call                    |
| N80 A3=1 B3=0 C3=1 | ; Change in orientation to the for N70 |
| N90 Y60        | ; and N90 generated outside corner           |

Blocks N75 and N80 are executed after N70. The circle block is then executed with the current orientation.

ORID

If ORID is active, then all blocks between the two traversing blocks are executed at the end of the first traversing block. The circle block with constant orientation is executed immediately before the second traversing block.

Figure 18-8 ORID: Change in orientation and path movement consecutively
Example:

| Program code     | Comment                                      |
|------------------|----------------------------------------------|
| N10 A0 B0 X0 Y0 Z0 F5000 |                                             |
| N20 T1 D1        | ; Radius=5                                   |
| N30 TRAORI(1)    | ; Transformation selection                  |
| N40 CUT3DC       | ; 3D TRC selection                           |
| N50 ORID         |                                              |
| N60 G42 X10 Y10  | ; TRC selection                              |
| N70 X60          |                                              |
| N80 A3=1 B3=0 C3=1 | ; Change in orientation to the for N70       |
| N90 Y60          | ; and N90 generated outside corner           |
| N100 X10         |                                              |
| N110 G40 X0 Y0   |                                              |
| N120 M30         |                                              |

**Note**

The command **DISC** is not evaluated.

**G451**

The intersection is determined by extending the offset curves of the two participating blocks and defining the intersection of the two blocks at the corner in the plane perpendicular to the tool orientation. If no such intersection is available, a circle is inserted.

If an intersection is found in the plane perpendicular to the tool, this does not mean that the curves also intersect in space. Rather the curves in the direction of the tool longitudinal axis are considered, which are generally a certain distance apart. The positional offset is eliminated over the entire block length in direction of the tool.

The way this offset is processed in tool direction at outside corners is the same as for inside corners.

**No intersection procedure**

The intersection procedure is not used when at least one block containing a change to the tool orientation was inserted between the traversing blocks in question. In this case a circle is always inserted at the corner.

**Blocks without traversing information**

Blocks without relevant traversing information (neither tool orientation nor position of geometry axes are changed) are permissible. The intersection procedure is applied to the adjacent blocks as if these intermediate blocks did not exist. In the same manner, tool direction motions in the tool direction may also be programmed in intermediate blocks.
18.2.3 Behavior at inside corners

Collision monitoring

With the 3D compensation function, only adjacent traversing blocks are taken into account in the calculation of intersections.

Path segments must be long enough to ensure that the contact points of the tool do not cross the block limits into other blocks when the orientation changes at an inside corner.

![Diagram showing behavior at inside corners](image)

**Figure 18-9** The contact points of the tool must not cross the limits of block N70 or N90 as a result of the change in orientation in block N80

**Example:**

| Program code | Comment |
|--------------|---------|
| N10 A0 B0 X0 Y0 Z0 F5000 |  |
| N20 T1 D1 | Radius=5 |
| N30 TRAGRI(1) | Transformation selection |
| N40 CUT3DC | 3D TRC selection |
| N50 ORID |  |
| N60 G42 X10 Y10 | TRC selection |
| N70 X60 |  |
| N80 A3=1 B3=0 C3=1 | Change in orientation to the for |
| N90 X10 | N70 and N90 generated inside corner |
| N100 G40 X0 Y0 |  |
| N120 M30 |  |
Without change in orientation

If the orientation is not changed at the block limit, then the contour need only be considered in the plane vertical to the tool axis. In this case, the tool cross-section is a circle which touches the two contours. The geometric relations in this plane are identical to those for 2½D compensation.

With change in orientation

If the orientation changes on a block transition, the tool moves in the inside corner so that it is constantly in contact with the two blocks forming the corner.

When the orientation changes in a block that is one of the two blocks forming the inside corner, then it is no longer possible to adhere to the programmed relationship between path position and associated orientation. This is because the orientation must reach its end value even though the path end position is not reached. This response is identical to the response of synchronized axes with 2½ D tool radius compensation.

Figure 18-10  Path end position and change in orientation at inside corners
Change in insertion depth

Generally speaking, the contour elements that form an inside corner are not positioned on the plane perpendicular to the tool. This means that the contact points between the two blocks and the tool are at different distances from the tool tip.

This means: the insertion depth (ISD) changes abruptly from the 1st to the 2nd block at an inside corner.

To ensure that this difference in depth is not an abrupt step change, it is distributed continuously among the blocks involved during interpolation. The depth-compensating motion is executed in the current tool direction.

This solution prevents the contour from being violated by cylindrical tools if the length of the tool prevents the cutter contact point on the lateral surface of the cutter leaving the range in which machining is possible.
Example of inside corners

Figure 18-12 Change in orientation at an inside corner

Example:

| Program code       | Comment                                           |
|--------------------|---------------------------------------------------|
| N10 A0 B0 X0 Y0 Z0 F5000 |                                                   |
| N20 T1 D1          | ; Radius=5                                        |
| N30 TRAORI(1)      | ; Transformation selection                       |
| N40 CUT3DC         | ; 3D TRC selection                                |
| N50 ORID           |                                                   |
| N60 G42 X10 Y10 G451 | ; TRC selection                                  |
| N70 Y60            |                                                   |
| N80 A3=1 B3=0 C3=1 | ; Change in orientation to the for               |
| N90 X60 Y90        | ; N70 and N90 generated inside corner             |
| N100 G40 X... Y... |                                                   |
| ...                |                                                   |
| N190 CDOF          |                                                   |
| N200 M30           |                                                   |
18.3 Face milling

The face milling function allows surfaces with any degree or form of curvature to be machined. In this case, the longitudinal axis of the tool and the surface normal vector are more or less parallel. In contrast, the longitudinal axis and the surface normal vector of the surface to be machined in a circumferential milling operation are at right angles to one another.

Information about the surfaces to be machined is absolutely essential for face milling operations, i.e., a description of the linear path in space is not sufficient. The tool shape must also be known in order to implement the tool offset (the term “Tool radius compensation” is not appropriate in this case).

The relations in face milling are shown in the Figure below.

Figure 18-13 Face milling with a torus

18.3.1 Cutter shapes

The following table lists the possible tool shapes that may be used for face milling with their dimensions.

| Cutter type                              | Tool No. | d  | r  | a  |
|------------------------------------------|----------|----|----|----|
| Ball end mill (cylindrical die sinker)   | 110      | >0 | X  | X  |
| Ball end mill (tapered die sinker)       | 111      | >0 | >d | X  |
| End milling cutter without corner rounding| 120, 130 | >0 | X  | X  |
| End mill with corner rounding (torus)     | 121, 131 | >r | >0 | X  |
| Bevel cutter without corner rounding     | 155      | >0 | X  | >0 |
| Bevel cutter with corner rounding        | 156      | >r | >0 | >0 |

If a tool number other than any of those specified in the table above is used in the NC program, then the tool type is assumed to be a ball end mill (tool type 110). Tool parameters marked with an X in the tool table are not evaluated. A value other than zero is meaningless for the tool offset for face milling.

An alarm is output if tool data are programmed that violate the limits specified in the table above.
The shaft characteristics are not taken into account on any of the tool types. For this reason, the two tool types 120 (end mill) and 155 (bevel cutter), for example, have an identical machining action since only the section at the tool tip is taken into account. The only difference between these tools is that the tool shape is represented differently (dimensions).

![Tool types for face milling](image)

Figure 18-14 Tool types for face milling

The tool data are stored under the following tool parameter numbers:

| Tool data | Geometry     | Wear       |
|-----------|--------------|------------|
| d         | $\text{TC\_DP6}$ | $\text{TC\_DP15}$ |
| r         | $\text{TC\_DP7}$ | $\text{TC\_DP16}$ |
| a         | $\text{TC\_DP11}$ | $\text{TC\_DP20}$ |

**Note**

The geometry and wear values of a tool data are added.

The reference point for tool length compensation (also referred to as tool tip or tool center point (TCP)) on all tool types is the point at which the longitudinal axis of the tool penetrates the surface.

A new tool with different dimensions may be programmed only when the tool compensation is activated for the first time (i.e. on transition from 
\[G40\] to 
\[G41\] or 
\[G42\]) or, if the compensation is already active, only when 
\[G41\] or 
\[G42\] are reprogrammed.

In contrast to circumferential milling, therefore, there are no variable tool dimensions in one block.

This restriction applies only to the tool shape (tool type, dimensions d, r and a).
A change in tool involving only a change in other tool data (e.g. tool length) is permitted provided that no other restrictions apply. An alarm is output if a tool is changed illegally.

18.3.2 Orientation

The options for programming the orientation have been extended for 3D face milling.

The tool offset for face milling cannot be calculated simply by specifying the path (e.g. a line in space). The surface to be machined must also be known. The control is supplied with the information it requires about this surface by the surface normal vector.

The surface normal vector at the block beginning is programmed with A4, B4, C4, and the vector at the block end with A5, B5, C5. Components of the surface normal vector that are not programmed are set to zero. The length of a vector programmed in this way is irrelevant. A vector of zero length (all three components are zero) is ignored, i.e. the direction programmed beforehand remains valid, no alarm is generated.

If only the start vector is programmed (A4, B4, C4) in a block, then the programmed surface normal vector remains constant over the entire block. If only the end vector is programmed (A5, B5, C5), then large-circle interpolation is used to interpolate between the end value of the preceding block and the programmed end value. If both the start and end vectors are programmed, then interpolation takes place between both directions using the large-circle interpolation method. The fact that the start vector may be reprogrammed in a block means that the direction of the surface normal vector can change irregularly on a block transition. Irregular transitions of the surface normal vector always occur in cases where there is no tangential transition between the surfaces (planes) involved, i.e. if they form an edge.

Once a surface normal vector has been programmed, it remains valid until another vector is programmed. In the basic setting, the surface normal vector is set to the same values as the vector in the z direction. This basic setting direction is independent of the active plane (G17-G19). If ORIWKS is active, surface normal vectors refer to the active frame, i.e. when the frame is rotated, the vectors rotate simultaneously. This applies both to programmed orientations as well as to those derived from the active plane. If ORIWKS is active, the surface normal vectors are adjusted when a new frame becomes active. An orientation modified as the result of frame rotations is not returned to its original state on switchover from ORIWKS to ORIMKS.
It must be noted that the programmed surface normal vectors may not necessarily be the same as those used internally. This always applies when the programmed surface normal vector is not perpendicular to the path tangent. A new surface normal vector is then generated which is positioned in the plane extending from the path tangent to the programmed surface normal vector, but which is at right angles to the path tangent vector. This orthogonalization is necessary because the path tangent vector and surface normal vector for a real surface must always be perpendicular to one another. However, since the two values can be programmed independently, they may contain mutually contradictory information. Orthogonalization ensures that the information contained in the path tangent vector has priority over the data in the surface normal vector. An alarm is output if the angle between the path tangent vector and the programmed surface normal vector is smaller than the limit value programmed in machine data:

MD21084 $MC_CUTCOM_PLANE_PATH_LIMIT (minimum angle between surface normal vector and path tangent vector)

If a block is shortened (inside corner), then the interpolation range of the surface normal vector is reduced accordingly, i.e. the end value of the surface normal vector is not reached as it would be with other interpolation quantities such as, for example, the position of an additional synchronized axis.

In addition to the usual methods of programming orientation, it is also possible to refer the tool orientation to the surface normal vector and path tangent vector using the addresses LEAD (lead or camber angle) and TILT (side angle). The lead angle is the angle between the tool orientation and the surface normal vector. The side angle is the angle between the path tangent and the projection of the tool vector into the surface to be machined. Specification of the angle relative to the surface normal is merely an additional option for programming tool orientation at the block end. It does not imply that the lead and side angles reach their programmed values before the path end point is reached.

The final tool orientation is calculated from the path tangent, surface normal vector, lead angle and side angle at the block end. This orientation is always implemented by the end of the block, particularly in cases where the block is shortened (at an inside corner). If the omitted path section is not a straight line in a plane, the lead and side angles generally deviate from their programmed values at the path end point. This is because the orientation has changed relative to the surface normal vector or path tangent vector when the absolute orientation of the tool is the same as at the original path end point.
18.3.3 Compensation on path

Tool longitudinal axis parallel to surface normal

A special case must be examined with respect to face milling operations, i.e. that the machining point on the tool surface moves around. This may be the case on a torus cutter whenever surface normal vector \( \mathbf{n}_F \) and tool vector \( \mathbf{w} \) become collinear (i.e. the tool is at exact right angles to the surface) since it is not a single point on the tool that corresponds to this direction, but the entire circular surface on the tool end face. The contact point is not, therefore, defined with this type of orientation. A path point in which tool longitudinal axis and surface normal are parallel is therefore referred to below as a singular point or a singularity.

The above case is also meaningful in practical terms, e.g. in cases where a convex surface, which may have a vertical surface normal (e.g. hemisphere), must be machined with a perpendicular tool (e.g. face milling with constant orientation). The machining point on the contour remains fixed, but the machine must be moved to bring the machining point from one side of the tool to the other.

The problem described is only a borderline case (lead angle \( \beta = 0 \) and side angle \( \gamma = 0 \)). If the lead angle \( \beta = 0 \) and the side angle \( \gamma \) has a low value, then the tool must be moved very rapidly (in borderline case in steps) to keep the machining point resulting from the milling conditions close to the arc-line forming the end face, see the following Figure.

![Singular point](image)

Figure 18-15 Change in the machining point on the tool surface close to a point in which surface normal vector and tool orientation are parallel

The problem is basically solved as follows: If the angle \( d \) between the surface normal vector \( \mathbf{n}_F \) and tool orientation \( \mathbf{w} \) is smaller than a limit value (machine data) \( \delta_{min} \), then the side angle \( \gamma \) on tools with a flat end face (e.g. torus cutter or cylindrical mill) must be 0. This restriction does not apply to tool types with a spherical end face (e.g. ball end mill, die sinker) since angular changes close to the singular point do not lead to abrupt changes in the machining point on the surface of such tools. If \( \delta \) now becomes 0, i.e. the sign of lead angle \( \beta \) changes, the machining point moves from its current position to the opposite side of the tool. This movement is executed in an inserted linear block.

The machining operation is aborted with an alarm if an attempt is made to machine within the illegal angular range for the side angle \( \gamma \) (i.e. \( \delta < \delta_{min} \) and \( \gamma < 0 \)).
The insertion of linear blocks makes it necessary to split the original blocks at the singular points. The partial blocks created in this way are treated as if they were original, which means, for example, that a concave path containing a singularity is treated like an inside corner, i.e. there is no contour violation. Each new partial block must contain at least one tool contact point since this is always calculated on the basis of adjacent traversing blocks.

Singularities do not just occur at isolated points, but along whole curves. This is the case, for example, if the curve to be interpolated is a plane curve (i.e. a curve with a constant osculating plane) and the tool is constantly aligned in parallel to the binormal vector, i.e. perpendicular to the osculating plane. A simple example is a circular arc in the x-y plane that is machined by a tool aligned in parallel to the z axis. On paths of this type, the tool offset is reduced to a tool length compensation, i.e. the tool is moved so that its tip FS is positioned on the programmed path.

On transition between singular and non-singular curves, linear blocks must be inserted in the same way as for isolated singular points so that the machining point on the tool can move from the tool tip FS to the periphery (on outside corners and convex surfaces) or the paths must be shortened to avoid contour violations (on inside corners and concave surfaces).
18.3.4 Corners for face milling

Two surfaces which do not merge tangentially form an edge. The paths defined on the surfaces make a corner. This corner is a point on the edge.

The corner type (inside or outside corner) is determined by the surface normal of the surfaces involved and by the paths defined on them.

The surface normals of the two surfaces forming the edge may point in opposite directions of the overall surface (the front edge of one surface is continued on the rear edge of the second surface), see also the following Figure. Such transitions are not permissible and are rejected with an alarm.

The scalar product of the surface normal vector and (possibly variable) tool orientation on one corner/path must be positive at each point, i.e. it is not permissible to machine from the rear face of the surface. Failure to observe this rule results in an alarm. The permissible ranges of validity of tool orientation for inside and outside corners are illustrated in the following Figure. These ranges are further restricted by the condition that the angle between the surfaces to be machined and the "steepest" surface line of the tool surface must not be lower than a particular machine data setting. The "steepest" surface line is a line at angle \( \alpha \) to the tool longitudinal axis (this line is in the same direction as the tool longitudinal axis on cylindrical tools). This restriction must be imposed to ensure that the contact point on the tool does not leave the permissible range.

![Range of validity of tool orientation](image)

Figure 18-16 Corners for face milling

It is possible to insert blocks that contain no motion commands (e.g. auxiliary function outputs) and/or that include motions of axes not involved in the path between two blocks which contain a path definition. Changes in orientation can also be programmed in such intermediate blocks. The only exception applies to the activation and deactivation of the 3D tool radius compensation function, i.e. intermediate blocks with changes in orientation may not be inserted between the activation block and the first corrected block or between the last corrected block and the deactivation block. Other intermediate blocks are, however, permitted.
18.3.5 Behavior at outer corners

Outside corners are treated as if they were circles with a 0 radius. The tool radius compensation acts on these circles in the same way as on any other programmed path.

The circle plane extends from the final tangent of the first block to the start tangent of the second block.

The orientation can be changed during block transition.

A circle block is always inserted at an outside corner.

A change in orientation between two programmed blocks is executed either before the circle block or in parallel to it.

Programming

- **ORIC**: Change in orientation and path movement in parallel
  (ORIentation Change Continuously)
- **ORID**: Change in orientation and path movement consecutively
  (ORIentation Change Discontinuously)

The **ORIC** and **ORID** program commands are used to determine whether changes in orientation programmed between two blocks are executed before the inserted circle block is processed or at the same time.

When the orientation needs to be changed at outside corners, the change can be implemented in parallel to interpolation or separately from the path motion. When **ORID** is programmed, the inserted blocks are executed first without a path motion (blocks with changes in orientation, auxiliary function outputs, etc.). The circle block is inserted immediately in front of the second of the two traversing blocks which form the corner.

**ORIC**

If **ORIC** is active and there are two or more blocks with changes in orientation (e.g. A2 = B2 = C2 =) programmed between the traversing blocks, then the inserted circle block is distributed among these intermediate blocks according to the absolute changes in angle.

**Change in orientation**

The method by which the orientation is changed at an outer corner is determined by the program command that is active in the first traversing block of an outer corner.

If the tool orientation at an outside corner is not constant, then the change in orientation is implemented in exactly the same way as described in Subsection "Behavior at outer corners" for circumferential milling operations.
18.3.6 Behavior at inside corners

The position of the tool in which it is in contact with the two surfaces forming the corner must be determined at an inside corner. The contact points must be on the paths defined on both surfaces. It is not usually possible to solve this problem exactly since, when the tool is moved along the path on the first surface, it normally touches a point on the second surface which is not on the path.

For this reason, the tool is not moved along the path on the first surface, but deviates from the path in such a way as to ensure that the distance between the contact points and the relevant contours in the position in which the tool contacts both surfaces is minimal, see also the following Figure.

![Figure 18-17 Inside corner with face milling (view in direction of longitudinal axis of tool)](image)

**Note**

The amount by which the contact points deviate from the programmed contour will generally be small since the explanatory example shown in the Figure, in which the machining point "changes" the cutter side at an inside corner (the value of the angular difference $\Psi$ about the tool longitudinal axis between the two contact points on the tool surface is approximately $180^\circ$) is more likely to be the exception (see also the following Figure, on the right). The angle $\Psi$ will normally stay almost constant so that the distance between the contact points on the tool surface will be relatively small (see also the following Figure, on the left).

![Figure 18-18 Machining at inside corners](image)
The difference between the programmed point on the path and the point actually to be approached (path offset $p$) is eliminated linearly over the entire block length. Differences resulting from inside corners at the block start and block end are overlaid. The current difference in a path point is always perpendicular to the path and in the surface defined by the surface normal vector.

If the tool orientation at an inside corner is not constant, the change in orientation is implemented in the same way as described in Subsection "Behavior at inside corners" for 3D circumferential milling, i.e. the tool is moved in the corner so that it contacts the two adjacent surfaces at the block start, block end and at two points $\frac{1}{3}$ and $\frac{2}{3}$ of the change in orientation. A 3rd-degree polynomial is used to interpolate between these 4 points.

A variable tool orientation in a block that is shortened owing to an inside corner is also treated in the same way as described in Subsection "Behavior at inside corners" for 3D milling, i.e. the entire change in orientation is executed in the shortened block. Consequently, the functional relationship between path tangent, surface normal and tool orientation also changes. This results in new, previously nonexistent singularities or impermissible side angles (at points which are virtually singular) occurring in the shortened block. If this type of situation is detected during processing of an inside corner, the machining operation is aborted with an alarm. No block division takes place at the singular points since the compensatory motions this would involve frequently cause contour violations and the change in machining side on the tool is not generally intended or even foreseen by the user. The alarm is also output during examination of an inside corner if the singularity occurs in the second of the two blocks without the transition to the next block being considered. The system does not therefore detect that a block of this type will form an inside corner in conjunction with the following block and that the singularity would be eliminated again by the second block reduction.

The surface normal vector $n_F$ is not affected by the reduction of a block. This means that in contrast to the tool orientation, the change in orientation that may need to be executed for this vector will not be imaged onto the reduced traversing interval. This is necessary because a surface other than that programmed would be machined. Unlike the tool orientation, no problems arise as the result of an abrupt change in the surface normal vector at a block transition since it does not reflect any axis motions.

Analogously to 3D circumferential milling, (see Section "Behavior at inside corners (Page 690)"), the two traversing blocks that form an inside corner must contain contact points. There is no evaluation of several traversing blocks (i.e. no bottleneck detection), CDON/CDOF are not evaluated. If no contact point can be found, the machining operation is aborted with an alarm (risk of collision).

### 18.3.7 Monitoring of path curvature

The path curvature is not monitored, i.e. the system does not usually detect any attempt to machine a concave surface that is curved to such a degree that the tool currently in use is not capable of performing the machining operation. A possible exception are blocks that are split owing to a singularity. The transition between the two partial blocks created after the split is then treated like an inside corner. Except for such special cases, the user is responsible for ensuring that only tools that can machine along the entire contour without violating it are used.
18.4 Selection/deselection of 3D TRC

The following commands are used to select/deselect 3D tool radius compensation for circumferential milling or face milling:

- **CUT3DC** (circumferential milling)
- **CUT3DFS** (face milling)
- **CUT3DFF** (face milling)
- **CUT3DF** (face milling)

18.4.1 Selection of 3D TRC

**CUT3DC**

3D radius compensation for circumferential milling (only when 5-axis transformation is active).

**CUT3DFS**

3D tool offset for face milling with constant orientation. The tool orientation is determined by G17 - G19 and is not influenced by frames.

**CUT3DFF**

3D tool offset for face milling with constant orientation. The tool orientation is the direction defined by G17 - G19 and, in some case, rotated by a frame.

**CUT3DF**

This programming command selects the 3D tool offset for face milling with change in orientation (only when 5-axis transformation is active).

TRC selection

The program commands used to select 3D TRC are the same as those for 2D TRC. G41/G42 specify the compensation on the left or right in the direction of motion (the response on selection of G41 and G42 for 3D face milling is identical). Tool radius compensation is deactivated with G40. The approach response is always controlled with NORM. Activation must be in a linear block.
Example:

| Program code |
|--------------|
| N10 A0 B0 X0 Y0 Z0 F5000 |
| N20 T1 D1 ; Radius=5 |
| N30 TRAORI(1) ; Transformation selection |
| N40 CUT3DC ; 3D TRC selection (circumferential milling) |
| N50 G42 X10 Y10 ; TRC selection |
| N60 X60 |
| N70 ... |

Intermediate blocks

Intermediate blocks are permitted when 3D TRC is active. The specifications for 2D TRC apply equally to 3D TRC.

18.4.2 Deselection of 3D TRC

Deselection

The 3D tool radius compensation function is deselected in a linear block G0/G1 with geometry axes by means of G40.

Example:

| Program code |
|--------------|
| N10 A0 B0 X0 Y0 Z0 F500 |
| N20 T1 D1 |
| N30 TRAORI(1) ; Transformation selection |
| N40 CUT3DC ; 3D TRC selection |
| N50 G42 X10 Y10 ; TRC selection |
| N60 X60 |
| N70 G40 X100 Y0 Z20 ; 3D TRC deselection |
| N80 ... |

Note

D0 is programmed via active tool radius compensation, so there is no selection. If no geometry axis is programmed for the current plane in the block with the selection, no selection takes place.
18.5 Constraints

The following supplementary conditions apply for the use of the "3D tool radius compensation" function:

- Higher-order geometry information cannot be used in the function (e.g. compressor, polynomials or splines).
- Bottleneck detection is not possible with this function.
- Movements in the orientation direction of the tool force error messages (machining stop).
### 18.6 Examples

Example program for 3D circumferential milling:

| Program code | Comment |
|--------------|---------|
| ; Definition of tool D1 |
| $TC_DP1[1,1] = 120 ; Type (end mill) |
| $TC_DP3[1,1] = ; Length offset vector |
| $TC_DP6[1,1] = 8 ; Radius |
| N10 X0 Y0 Z0 T1 D1 F12000 ; Selection of the tool |
| N20 TRAORI(1) ; Activate transformation |
| N30 G42 ORIC ISD=10 CUT3DC G64 X30 ; Activate 3D circumferential milling, |
| Orientation changes at outside corners |
| Constant, 10 mm insertion depth. |
| N40 ORIWKS A30 B15 ; Change of orientation at a corner |
| by specification of axis positions. |
| N50 Y20 A3=1 C3=1 ; Traversing block with change in orientation, |
| specification |
| ; of the orientation with direction vectors |
| N60 X50 Y30 ; Traversing block with constant orientation |
| N70 Y50 A3=0.5 B3=1 C3=5 ; Traversing block with change in orientation |
| N80 M63 ; Block without traversing information |
| N90 X0 ISD=20 ; Traversing block with change of the insertion depth |
| N100 G40 Y0 ; Deactivating the tool radius compensation |
| N110 M30 |
Example program for 3D face milling:

| Program code | Comment |
|--------------|---------|
| N10          | ; Definition of tool D1 |
| N20 $TC_DP1[1,1] = 121 | ; Tool type (torus cutter) |
| N30 $TC_DP3[1,1]=20 | ; Tool length offset |
| N40 $TC_DP6[1,1]=5 | ; Radius |
| N50 $TC_DP7[1,1]=3 | ; Smoothing radius |
| N80 X0 Y0 Z0 A0 B0 C0 G17 T1 D1 F12000 | ; Selection of the tool |
| N90 TRAORI(I) | |
| N100 B4=-1 C4=1 | ; Plane definition |
| N110 G41 ORID CUT3DF G64 X10 Y0 Z0 | ; Activate tool offset |
| N120 X30 | |
| N130 Y20 A4=1 C4=1 | ; Outside corner, new plane definition |
| N140 B3=1 C3=5 | ; Change in orientation with ORID |
| N150 B3=1 C3=1 | ; Change in orientation with ORID |
| N160 X-10 A5=1 C5=2 ORIC | |
| N170 A3=2 C3=1 | ; Change in orientation with ORIC |
| N180 A3=1 C3=1 | ; Change in orientation with ORIC |
| N190 Y-10 A4=1 C4=3 | ; New plane definition |
| N200 X-20 Y-20 Z10 | ; Inside corner with previous block |
| N210 X-30 Y10 A4=1 C4=1 | ; Inside corner, new plane definition |
| N220 A3=1 B3=0.5 C3=1.7 | ; Change in orientation with ORIC |
| N230 X-20 Y30 A4=1 B4=2 C4=3 ORID | |
| N240 A3 = 0.5 B3=-0.5 C3=1 | ; Change in orientation |
| N250 X0 Y30 C4=1 | ; Path movement, new plane, |
| | ; Orientation |
| | ; With relative programming |
| N260 BSPLINE X20 Z15 | ; Spline onset, relative programming |
| N270 X30 Y25 Z18 | ; of the orientation remains during |
| N280 X40 Y20 Z13 | ; Spline active |
| N290 X45 Y0 PR=2 Z8 | |
| N300 Y-20 | |
| N310 G2 ORIMKS A30 B45 i-20 X25 Y-40 Z0 | ; Helix, orientation with axis program |
| N320 G1 X0 A3=-0.123 B3=0.456 C3 =2.789 B4=-1 C4=5 | ; Path movement, orientation, non-constant plane |
| | B5=1 C5=2 |
| N330 X-20 G40 | ; Deactivation of the tool offset |
| N340 M30 | |
18.7 Data lists

18.7.1 General machine data

| Number | Identifier: $MN_  | Description                                           |
|--------|------------------|-------------------------------------------------------|
| 18094  | MM_NUM_CC_TDA_PARAM | Number of TDA data                                    |
| 18096  | MM_NUM_CC_TOA_PARAM | Number of TOA data, which can be set up per tool and evaluated by the CC |
| 18100  | MM_NUM_CUTTING_EDGES_IN_TOA | Tool offsets per TOA module                           |
| 18110  | MM_NUM_TOA_MODULES | Number of TOA modules                                 |

18.7.2 Channelspecific machine data

| Number | Identifier: $MC_  | Description                                                                                                      |
|--------|------------------|-----------------------------------------------------------------------------------------------------------------|
| 20110  | RESET_MODE_MASK | Definition of control basic setting after powerup and RESET / part program end                                  |
| 20120  | TOOL_RESET_VALUE | Definition of tool for which tool length compensation is selected during powerup or on reset or parts program end as a function of MD 20110 |
| 20130  | CUTTING_EDGE_RESET_VALUE | Definition of tool cutting edge for which tool length compensation is selected during powerup or on reset or parts program end as a function of MD20110 |
| 20140  | TRAFO_RESET_VALUE | Definition of transformation block which is selected during powerup and or RESET or parts program end as a function of MD20110 |
| 20210  | CUTCOM_CORNER_LIMIT | Max. angle for intersection calculation with tool radius compensation                                          |
| 20220  | CUTCOM_MAX_DISC | Maximum value for tool radius compensation                                                                  |
| 20230  | CUTCOM_CURVE_INSERT_LIMIT | Minimum value for intersection calculation with tool radius compensation                                      |
| 20240  | CUTCOM_MAXNUM_CHECK_BLOCKS | Blocks for predictive contour calculation with tool radius compensation                                         |
| 20250  | CUTCOM_MAXNUM DUMMY BLOCKS | Max. no. of dummy blocks with no traversing movements                                                       |
| 20270  | CUTTING_EDGE_DEFAULT | Selected cutting edge after tool change                                                                       |
| 20610  | ADD_MOVE_ACCEL_RESERVE | Acceleration reserve for overlaid movements                                                                 |
| 21080  | CUTCOM_PARALLEL_ORI_LIMIT | Limit angle between path tangent and tool orientation with 3D tool radius compensation                        |
| 21082  | CUTCOM_PLANE_ORI_LIMIT | Minimum angle between surface normal and tool orientation with side angle not equal to 0                    |
| 21084  | CUTCOM_PLANE_PATH_LIMIT | Minimum angle between surface normal vector and path tangent vector, for 3D face milling                      |
| 22550  | TOOL_CHANGE_MODE | New tool offsets with M function                                                                            |
| 22560  | TOOL_CHANGE_M_CODE | M function for tool change                                                                                  |
19.1 Brief description

Function

With the "Path length evaluation" function, the NCK specific machine axis data is made available as the system and OPI variables, with whose help it is possible to assess the strain on the machine axes and thereby make an evaluation on the state of the machine’s maintenance.

Recorded data

The following data is recorded:

- Total traverse path
- Total travel time
- Total travel count
- Total traverse path at high axis speeds
- Total travel time at high axis speeds
- Travel count at high axis speeds
- Total sum of jerks
- Axis travel time with jerk
- Travel count with jerk

Evaluation

When the function is activated, the selected control data is automatically sent and made available via the system and OPI variables for evaluation in the part program or synchronized actions, as with user-specified HMI functions.

Meaning

The data remains saved in the control, so that it may continue to be used after POWER OFF / ON. Consequently, the actual value of a data item represents the sum of all measured values since the function was activated.
## 19.2 Data

The following data is available:

| System variable 1) | OPI variable | Meaning |
|--------------------|--------------|---------|
| $\$AA\_TRAVEL\_DIST | aaTravelDist | Total traverse path: sum of all set position changes in MCS 2) in [mm] or [deg.]. |
| $\$AA\_TRAVEL\_TIME | aaTravelTime | Total travel time: sum of IPO clock cycles from set position changes in MCS 2) in [s] (solution: 1 IPO clock cycle) |
| $\$AA\_TRAVEL\_COUNT | aaTravelCount | Total travel count: a complete machine axis trip is defined by the following succession of states, as based on set position: standstill > traversing > standstill |
| $\$AA\_TRAVEL\_DIST\_HS | aaTravelDistHS | Total traversing distance at high axis velocities 3) |
| $\$AA\_TRAVEL\_TIME\_HS | aaTravelTimeHS | Total traversing time at high axis velocities 3) |
| $\$AA\_TRAVEL\_COUNT\_HS | aaTravelCountHS | Total number of traversing operations at high axis velocities 3) |
| $\$AA\_JERK\_TOT | aaJerkTotal | Total sum of axis jerks: Sum of all jerk setpoints in [m/s³] or [deg./s³]. |
| $\$AA\_JERK\_TIME | aaJerkTime | Total travel time with jerk: sum of IPO clock cycles from jerk setpoint changes in [s] (solution: 1 IPO clock cycle) |
| $\$AA\_JERK\_COUNT | aaJerkCount | Total number of traversing operations with jerk |

1) System variables can be read from part programs and synchronized actions  
2) MCS: Machine coordinate system  
3) Actual machine axis velocity ≥ 80% of the maximum parameterized axis velocity (MD32000 MAX_AX_VELO)
19.3 Parameterization

19.3.1 General activation

The function is generally activated via the NCK-specific machine data:
MD18860 $MN_MM_MAINTENANCE_MON (Activate recording of maintenance data)

19.3.2 Data groups

The data has been collected into data groups.
The data groups are activated via the axis-specific machine data:
MD33060 $MA_MAINTENANCE_DATA (configuration to record maintenance data)

| Bit | Value | Activation of the following data: System variable / OPI variable |
|-----|-------|---------------------------------------------------------------|
| 0   | 1     | • Total traverse path: $AA_TRAVEL_DIST / aaTravelDist        |
|     |       | • Total travel time: $AA_TRAVEL_TIME / aaTravelTime          |
|     |       | • Total travel count: $AA_TRAVEL_COUNT / aaTravelCount       |
| 1   | 1     | • Total traversing distance for high velocities:             |
|     |       | $AA_TRAVEL_DIST_HS / aaTravelDistHS                          |
|     |       | • Total traversing time for high velocities:                 |
|     |       | $AA_TRAVEL_TIME_HS / aaTravelTimeHS                          |
|     |       | • Total number of traversing operations at high velocities:  |
|     |       | $AA_TRAVEL_COUNT_HS / aaTravelCountHS                        |
| 2   | 1     | • Total sum of jerks: $AA_JERK_TOT / aaJerkTotal             |
|     |       | • Travel time with jerk: $AA_JERK_TIME / aaJerkTime          |
|     |       | • Total travel count with jerk:                              |
|     |       | $AA_JERK_COUNT / aaJerkCount                                 |
19.4 Examples

19.4.1 Traversal per part program

Three geometry axes AX1, AX2 and AX3 exist in a machine. For geometry axis AX1, the part program-driven total traverse path, total travel time and travel count should be calculated.

Parameter assignment

Activation of the overall function:

\[\text{MD18860 $MN\_MM\_MAINTENANCE\_MON = TRUE}\]

Group activation: "Total travel distance, total travel time and number of travel operations" for geometry axis AX1:

\[\text{MD33060 $MA\_MAINTENANCE\_DATA[AX1] = 1}\]

Programming

To determine the values referred to the part program, the actual value of the system variables at the beginning of the part program must be saved in a calculation variable. The difference is acquired at the end of the part program.

Part program (extract):

| Program code | Comment |
|--------------|---------|
| ...          | ; Current values: |
| R10=$AA\_TRAVEL\_DIST[AX1] | ; Total traversing distance AX1 |
| R11=$AA\_TRAVEL\_TIME[AX1]  | ; Total traversing time AX1 |
| R12=$AA\_TRAVEL\_COUNT[AX1] | ; Number of traversing operations AX1 |
| ...          | ; Traversing motion of axes |
| R10=R10-$AA\_TRAVEL\_DIST[AX1] | ; Traversing distance AX1 in the part program |
| R11=R11-$AA\_TRAVEL\_TIME[AX1]  | ; Traversing time AX1 in the part program |
| R12=R12-$AA\_TRAVEL\_COUNT[AX1] | ; Number of traversing operations AX1 in the part program |
19.5 Data lists

19.5.1 Machine data

19.5.1.1 NC-specific machine data

| Number | Identifier: $MN_ | Description                          |
|--------|------------------|--------------------------------------|
| 18860  | MM_MAINTENANCE_MON | Activate recording of maintenance data |

19.5.1.2 Axis/spindlespecific machine data

| Number | Identifier: $MA_ | Description                                      |
|--------|------------------|--------------------------------------------------|
| 33060  | MAINTENANCE_DATA | Configuration, recording maintenance data         |
19.5 Data lists
20.1.1 Signals from channel (DB21, ...)

| DB21, ... | activate PTP traversal |
|-----------|------------------------|
| DBX29.4   |                        |
| Edge evaluation: Yes | Signal(s) updated: |
| Signal state 1 or edge change 0 → 1 | activate PTP traversal. |
| Signal state 0 or edge change 1 → 0 | Activate CP traversal. |
| Signal irrelevant for ... | No handling transformations active. |
| Additional references | Function Manual, Special Functions; 3-Axis to 5-Axis Transformation (F2) |

| DB21, ... | Transformation active |
|-----------|-----------------------|
| DBX33.6   |                       |
| Edge evaluation: No | Signal(s) updated: Cyclic |
| Signal state 1 or edge change 0 → 1 | Active transformation. |
| Signal state 0 or edge change 1 → 0 | Transformation not (no longer) active. |
| Signal irrelevant for ... | No transformation used. |
| Additional references | Programming Manual, Fundamentals |

| DB21, ... | Number of active G function of G function group 25 (tool orientation reference) |
|-----------|----------------------------------------------------------------------------------|
| DBB232    |                                                                                 |
| Edge evaluation: | Signal(s) updated: |
| Signal state 1 or edge change 0 → 1 | ORIMKS: The tool orientation is implemented in a workpiece coordinate system and is thus not dependent on the machine kinematics. This is the default setting. |
| Signal state 0 or edge change 1 → 0 | ORIMKS: The tool orientation is defined in the machine coordinate system and is thus dependent on the machine kinematics. This is the default setting. |
### DB21, … DBX317.6

| PTP traversal active |
|----------------------|
| Edge evaluation: Yes |
| Signal(s) updated:   |
| Signal state 1 or edge change 0 → 1 |
| PTP traversal active. |
| Signal state 0 or edge change 1 → 0 |
| CP traversal active. |
| Signal irrelevant for ... |
| No handling transformations active. |
| Additional references |
| Function Manual, Special Functions; 3-Axis to 5-Axis Transformation (F2) |

### DB21, … DBX318.2

| TOFF active |
|-------------|
| Edge evaluation: Yes |
| Signal(s) updated: |
| Signal state 1 or edge change 0 → 1 |
| Activate online tool length offset. |
| Signal state 0 or edge change 1 → 0 |
| Reset online tool length offset. |
| Signal irrelevant for ... |
| The "generic 5-axis transformation" option is not available and no handling transformations are active. |
| Additional references |
| Function Manual, Special Functions; 3-Axis to 5-Axis Transformation (F2) |

### DB21, … DBX318.3

| TOFF motion active |
|--------------------|
| Edge evaluation: Yes |
| Signal(s) updated: |
| Signal state 1 or edge change 0 → 1 |
| Activate offset motion. |
| Signal state 0 or edge change 1 → 0 |
| Deactivate offset motion. |
| Signal irrelevant for ... |
| The "generic 5-axis transformation" option is not available and no handling transformations are active. |
| Additional references |
| Function Manual, Special Functions; 3-Axis to 5-Axis Transformation (F2) |
## 20.2 Gantry Axes (G1)

### 20.2.1 Signals to axis/spindle (DB31, ...)

| DB31, ... DBX29.4 | Start gantry synchronization |
|-------------------|------------------------------|
| Edge evaluation: No | Signal(s) updated: Cyclic |
| Signal state 1 or edge change 0 → 1 | Request from PLC user program to synchronize the leading axis with the assigned synchronized axes: MD37100 $MA_GANTRY_AXIS_TYPE (gantry axis definition) (i.e. all gantry axes approach the reference position of the gantry grouping in the decoupled state). Synchronization of the gantry axes can be started only under the following conditions: |
|                     | • REF machine function must be active |
|                     | • DB11, ... DBX5.2 (REF active machine function) = 1 |
|                     | • DB31, ... DBX101.5 (gantry grouping is synchronized) = 0 |
|                     | • DB31, ... DBX101.4 (gantry synchronization ready to start) = 1 |
|                     | • No axis is being referenced in the appropriate NC channel: DB21, ... DBX33.0 (referencing active) = 0 |
| Signal state 0 or edge change 1 → 0 | The PLC user program can then, for example, reset the interface signal to signal state "0" on completion of gantry synchronization (DB31, ... DBX101.5 = 1). If the IS is set continuously to "1", the gantry synchronization run would be started automatically as soon as the above conditions are fulfilled. |
| Signal irrelevant for ... | Gantry synchronized axis |
| Application example(s) | If the deviation between the position actual values and the reference position is greater than the gantry warning threshold after referencing of the gantry axes, automatic gantry synchronization is not started and IS "Gantry synchronization ready to start" is set to "1". Synchronization of the gantry axes can be started by the user or the PLC user program with IS "Start gantry synchronization". |
| Corresponding to .... | DB31, ... DBX101.5 (gantry grouping is synchronized) DB31, ... DBX101.4 (gantry synchronization ready to start) DB11, ... DBX5.2 (REF active machine function) DB21, ... DBX33.0 (referencing active) |

| DB31, ... DBX29.5 | Disable automatic synchronization |
|-------------------|----------------------------------|
| Edge evaluation: No | Signal(s) updated: |
| Signal state 1 or edge change 0 → 1 | No automatic synchronization. |
| Signal state 0 or edge change 1 → 0 | The automatic synchronization process is active. |
### 20.2 Gantry Axes (G1)

#### 20.2.2 Signals from axis/spindle (DB31, ...)

| DB31, ...  |
|-----------|
| DBX101.2  |

| Signal(s) updated: Cyclic |
|---------------------------|

| **Gantry trip limit exceeded** |
|------------------------------|

| Edge evaluation: No Signal(s) updated: Cyclic |
|-----------------------------------------------|

- **Signal state 1 or edge change 0 → 1**
  - The difference between the position actual values of the leading and synchronized axes has exceeded the maximum permissible limit value. The axes in the gantry grouping are shut down internally in the control. Alarm 10653 "Error limit exceeded" is also output.
  - The monitored limit value is derived from the following machine data:
    - MD37120 $MA_GANTRY_POS_TOL_ERROR (gantry trip limit)
    - ... if gantry grouping is synchronized.
    - MD37130 $MA_GANTRY_POS_TOL_REF (gantry trip limit for referencing)
    - ... if gantry grouping is not yet synchronized.
  - **Note:**
    - IS "Gantry trip limit exceeded" is output to the PLC via the PLC interface of the synchronized axis.

- **Signal state 0 or edge change 1 → 0**
  - The difference between the position actual values of the leading and synchronized axes is still within the permissible tolerance range.

| **Gantry warning limit exceeded** |
|----------------------------------|

| Edge evaluation: No Signal(s) updated: Cyclic |
|-----------------------------------------------|

- **Signal state 1 or edge change 0 → 1**
  - The difference between the position actual values of the leading and synchronized axes has exceeded the limit value defined by the following machine data:
    - MD37110 $MA_GANTRY_POS_TOL_WARNING (gantry warning limit)
  - The message "Warning limit exceeded" is also output.
  - **Note:**
    - IS "Gantry warning limit exceeded" is output to the PLC via the PLC interface of the synchronized axis.
### Gantry warning limit exceeded

| DB31, ... DBX101.3 | Gantry warning limit exceeded |
|---------------------|-------------------------------|
| Signal state 0 or edge change 1 → 0 | The difference between the position actual values of the leading and synchronized axes is less than the limit value defined with machine data: MD37110 $MA_GANTRY_POS_TOL_WARNING (gantry warning limit) |
| Signal irrelevant for ... | Gantry leading axis |
| Application example(s) | When the gantry warning limit is exceeded, the necessary measures (e.g. program interruption at block end) can be initiated by the PLC user program. |
| Special cases, errors, ... | Warning limit monitoring is rendered inactive when the value 0 is input in the following machine data: MD37110 $MA_GANTRY_POS_TOL_WARNING (gantry warning limit) |
| Corresponding to .... | MD37110 $MA_GANTRY_POS_TOL_WARNING (gantry warning limit) |

### Gantry synchronization ready to start

| DB31, ... DBX101.4 | Gantry synchronization ready to start |
|---------------------|--------------------------------------|
| Edge evaluation: No | Signal(s) updated: Cyclic |
| Signal state 1 or edge change 0 → 1 | After gantry axis referencing, the monitoring function has detected that the position actual value deviation between the leading and synchronized axes is greater than the gantry warning limit: MD37110 $MA_GANTRY_POS_TOL_WARNING (gantry warning limit) It is therefore not possible to start the automatic synchronization compensatory motion of the gantry axes internally in the control. The compensatory motion must be started by the user or the PLC user program (IS “Start gantry synchronization”). The signal is processed for the gantry leading axis only. |
| Signal state 0 or edge change 1 → 0 | After the synchronization compensatory motion has been started by the PLC user program (IS “Start gantry synchronization” = “1”). |
| Signal irrelevant for ... | Gantry synchronized axis |
| Corresponding to .... | MD37110 $MA_GANTRY_POS_TOL_WARNING (gantry warning limit) DB31, ... DBX29.4 (gantry synchronization ready to start) DB31, ...DBX60.4 and 60.5 (referenced / synchronized 1/2) |
### 20.2 Gantry Axes (G1)

#### Z3: NC/PLC interface signals

| DB31, ... | DBX101.5 | Gantry grouping is synchronized |
|-----------|----------|---------------------------------|
| Edge evaluation: No | Signal(s) updated: Cyclic |
| **Signal state 1 or edge change 0 → 1** | The gantry axis grouping defined with the following machine data is synchronized: MD37100 $MA_GANTRY_AXIS_TYPE (gantry axis definition) |
| Any existing misalignment between the leading and synchronized axes (e.g. after start-up of the machine) is eliminated by gantry axis synchronization (see Section "Start-up of gantry axes (Page 162)"). The synchronization process is initiated either automatically once the gantry axes have been referenced or via the PLC user program (IS "Start gantry synchronization"). The compensation values for temperature and sag do not become effective internally in the control until the gantry grouping is synchronized. **Note:** IS "Gantry grouping is synchronized" is output to the PLC via the PLC interface of the leading axis. |

| Signal state 0 or edge change 1 → 0 | The gantry axis grouping defined with the following machine data is not synchronized: MD37100 $MA_GANTRY_AXIS_TYPE (gantry axis definition) which means that the positions of the leading and synchronized axes may not be ideally aligned (e.g. gantry misalignment). Workpiece machining with a non-synchronized gantry axis grouping will result in impaired machining accuracy or mechanical damage to the machine. The gantry grouping becomes desynchronized if: |
|-----------------------------------|-----------------------------------------------|
| - The gantry axes were in "Follow-up" mode |
| - The reference position of a gantry axis is no longer valid or the axis is referenced again (IS "Referenced/Synchronized") |
| - The gantry grouping has been invalidated via the following machine data: MD37140 $MA_GANTRY_BREAK_UP (invalidate gantry axis grouping) |
| **Signal irrelevant for ...** | Gantry synchronized axis |
| **Application example(s)** | Machining should be enabled only if the gantry axes are already synchronized. This can be implemented in the PLC user program by combining NC Start with IS "Gantry grouping is synchronized"). |
| **Corresponding to ....** | DB31, ... DBX29.4 (gantry synchronization ready to start) DB31, ... DBX60.4 and 60.5 (referenced/synchronized 1 / 2) MD37140 $MA_GANTRY_BREAK_UP (invalidate gantry axis grouping) |

| DB31, ... | DBX101.6 | Gantry leading axis |
|-----------|----------|---------------------|
| Edge evaluation: No | Signal(s) updated: Cyclic |
| **Signal state 1 or edge change 0 → 1** | The axis is defined as the **leading axis** within a gantry axis grouping (see MD37100). **Note:** The following interface signals are evaluated or output to the PLC via the PLC interface of the gantry leading axis: DB31, ... DBX29.4 (gantry synchronization ready to start) DB31, ... DBX101.5 (gantry grouping is synchronized) |
### Gantry leading axis

| DB31, ... DBX101.6 | Gantry leading axis |
|---------------------|---------------------|
| Signal state 0 or edge change 1 → 0 | The axis is defined as the **synchronized axis** within a gantry axis grouping (see MD37100). It is not possible to traverse a synchronized axis directly by hand (in JOG mode) or to program it in a part program. **Note:** The following interface signals are output to the PLC via the PLC interface of the gantry synchronized axis: DB31, ... DBX101.3 (warning limit exceeded) DB31, ... DBX101.2 (gantry trip limit exceeded) The NCK does not evaluate individual axial PLC interface signals for the synchronized axis. |

| Corresponding to .... | MD37100 $MA_GANTRY_AXIS_TYPE (gantry axis definition)  DB31, ...DBX101.7 (gantry axis) |

### Gantry axis

| DB31, ... DBX101.7 | Gantry axis |
|---------------------|-------------|
| Edge evaluation: No | Signal(s) updated: Cyclic |
| Signal state 1 or edge change 0 → 1 | The axis is defined as a gantry axis within a gantry axis grouping (see MD37100). The PLC user program can read IS "Gantry leading axis" to detect whether the axis has been declared as a leading or synchronized axis. |
| Signal state 0 or edge change 1 → 0 | The axis is not defined as a gantry axis (see MD37100). |

| Corresponding to .... | MD37100 $MA_GANTRY_AXIS_TYPE (gantry axis definition)  DB31, ... DBX101.6 (gantry leading axis) |
20.3 Axis Couplings and ESR (M3)

20.3.1 Signals to axis (DB31, ...)

| DB31, ... DBX26.4 | Active following axis overlay |
|-------------------|------------------------------|
| Edge evaluation: No | Signal(s) updated: Cyclic |

Signal state 1 or edge change 0 → 1
An additional traversing motion can be overlaid on the following axis. This signal is required for the flying synchronization of master and slave axes. Until the "enable following axis overlay" signal stays set at 1, the following axis selected with EGONSYN will travel to synchronization in the EG coupling. Modulo axes included in the EG coupling reduce their position values in the modulo, thereby ensuring that they approach the next possible synchronization.

Signal state 0 or edge change 1 → 0
The following axis cannot be overlaid and traversed. If the "enable following axis overlay" signal has not been set for the following axis, the axis will not travel to synchronization. Instead, the program is stopped at the EGONSYN block and the self-clearing alarm 16771 is issued until the "enable following axis overlay" signal is set to 1.

20.3.2 Signals from axis (DB31, ...)

| DB31, ... DBX98.5 | Velocity warning threshold |
|-------------------|-----------------------------|
| Edge evaluation: No | Signal(s) updated: Cyclic |

Signal state 1 or edge change 0 → 1
When the following axis velocity in the axis grouping of the electronic gear (set in machine data MD32000) reaches or exceeds the velocity % block entered in machine data MD37550, the signal is set to 1.

Signal state 0 or edge change 1 → 0
The following axis velocity in the axis grouping of the electronic gear is less than the threshold value described above.

Signal irrelevant ...
Without electronic gear.

Corresponding to ...
The following machine data:
MD37550 $MA_EG_VEL_WARNING (threshold value velocity alarm threshold)
MD32000 $MA_MAX_AX_VELO (maximum axis velocity)

| DB31, ... DBX98.6 | Acceleration warning threshold |
|-------------------|-------------------------------|
| Edge evaluation: No | Signal(s) updated: Cyclic |

Signal state 1 or edge change 0 → 1
When the following axis acceleration in the axis grouping of the electronic gear (set in machine data MD32300) reaches or exceeds the acceleration % block entered in machine data MD37550, the signal is set to 1.
### Acceleration warning threshold

| Signal state 0 or edge change 1 → 0 | The following axis acceleration in the axis grouping of the electronic gear is less than the threshold value described above. |
|-----------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| Signal irrelevant ...             | Without electronic gear.                                                                                                 |
| Corresponding to ...              | The following machine data: MD37550 $MA_EG_VEL_WARNING (threshold value velocity alarm threshold) MD32300 $MA_MAX_AX_ACCEL (axis acceleration) |

### Axis accelerated

| Edge evaluation: No | Signal(s) updated: Cyclic |
|---------------------|----------------------------|
| Signal state 1 or edge change 0 → 1 | When the following axis acceleration in the axis grouping of the electronic gear (set in machine data MD32300) reaches or exceeds the acceleration % block entered in machine data MD37560, the signal is set to 1. |
| Signal state 0 or edge change 1 → 0 | The following axis acceleration in the axis grouping of the electronic gear is less than the operating value described above. |
| Signal irrelevant ... | Without electronic gear.                                                                                                 |
| Corresponding to ... | The following machine data: MD37560 $MA_EG_ACC_TOL (threshold value for "Accelerate axis") MD32300 $MA_MAX_AX_ACCEL (axis acceleration) |
20.4 Extended stop and retract (R3)

20.4.1 Signals from axis/spindle (DB31, ...)

### ESR: DC link undervoltage

| DB31, ... | Edge evaluation: No | Signal(s) updated: Cyclically |
|-----------|---------------------|------------------------------|
| DBX95.1   |                     |                              |

- **Signal state 1**: The drive signals that the DC link voltage $V_{\text{DC link}}$ is less than the lower DC link voltage threshold set with parameter $p_{1248}$.
- **Signal state 0**: The drive signals that the DC link voltage $V_{\text{DC link}}$ is greater than the lower DC link voltage threshold set with parameter $p_{1248}$.

**Application example(s)**: The PLC user program can initiate measures in order to safely end machining and/or to buffer the DC link voltage, e.g., initiate a drive-autonomous and/or control-managed extended stop and retract (ESR).

**Corresponding to**:
- Drive parameter $p_{1248}$ (lower DC link voltage threshold)
- Drive telegram MELDW.Bit 4

**Additional references**: Commissioning Manual IBN CNC: NCK, PLC, Drive

### ESR: Reaction triggered or generator operation active

| DB31, ... | Edge evaluation: No | Signal(s) updated: Cyclically |
|-----------|---------------------|------------------------------|
| DBX95.2   |                     |                              |

- **Signal state 1**: The drive signals that the configured ESR response has been triggered or generator operation is active.
- **Signal state 0**: The drive signals that neither the configured ESR response has been triggered nor that generator operation is active.

**Corresponding to**: Drive telegram MELDW.Bit 9

**Additional references**: Commissioning Manual IBN CNC: NCK, PLC, Drive

### Generator operation – minimum speed fallen below

| DB31, ... | Edge evaluation: No | Signal(s) updated: Cyclically |
|-----------|---------------------|------------------------------|
| DBX95.3   |                     |                              |

- **Signal state 1**: The axis is parameterized as generator axis. The drive signals that the actual speed is less than the minimum speed set using parameter $p_{2161}$ (speed threshold value 3).
- **Signal state 0**: The axis is parameterized as generator axis. The drive signals that the actual speed is greater than the minimum speed set using parameter $p_{2161}$ (speed threshold value 3).

**Application example(s)**: The PLC user program can initiate measures in order to safely end machining and/or to buffer the DC link voltage, e.g., initiate a drive-autonomous and/or control-managed extended stop and retract (ESR).

**Corresponding to**:
- Drive parameter $p_{2161}$ (speed threshold value 3)
- Drive telegram MELDW.Bit 2

**Additional references**: Commissioning Manual IBN CNC: NCK, PLC, Drive
20.5 Setpoint Exchange (S9)

20.5.1 Signals to axis/spindle (DB31, ...)

| DB31, ... DBX24.5 | Activate setpoint exchange |
|-------------------|-----------------------------|
| Edge evaluation: No | Signal(s) updated: Cyclic |
| Signal state = 1 | Request to axis to take over drive control. |
| Signal state = 0 | Request to axis to relinquish drive control. |

20.5.2 Signals from axis/spindle (DB31, ...)

| DB31, ... DBX96.5 | Status of setpoint exchange |
|-------------------|-----------------------------|
| Edge evaluation: No | Signal(s) updated: Cyclic |
| Signal state 1 | The axis has taken over control of the drive. |
| Signal state 0 | The axis has relinquished control of the drive. |
| Signal relevant for ... | All axes involved in setpoint exchange. |
20.6 Tangential Control (T3)

20.6.1 Special response to signals

The movement of the axis under tangential follow-up control to compensate for a tangent jump at a corner of the path (defined by the movements of the leading axis) can be stopped by the following signals (e.g. for test purposes):

- NC Stop and override = 0
- Removal of the axis-specific feed enable

The signals are described in:

Reference:
Lists (Book 2)
20.7 Clearance Control (TE1)

20.7.1 Signals to channel (DB21, ...)

| DB21, ... DBX1.4 | CLC stop |
|------------------|----------|
| Edge evaluation: No | Signal(s) updated: Cyclic |
| Signal state 1 or edge change 0 → 1 | Clearance control is deactivated in the same way as the part program command \( \text{CLC\_GAIN}=0.0 \). |
| Signal state 0 or edge change 1 → 0 | Clearance control is enabled. |

| DB21, ... DBX1.5 | CLC_Override |
|------------------|--------------|
| Edge evaluation: no | Signal(s) updated: Cyclic |
| Signal state 1 or edge change 0 → 1 | The channel-specific override DB21, … DBB4 (feed rate override) also applies to the clearance control. Override settings \(< 100\%\) reduce corresponding to the velocity limitation for the overriding motion, as defined in the following machine data: \( \text{MD62516 \$MC\_CLC\_SENSOR\_VELO\_LIMIT} \) (velocity of clearance control motion). Override settings \(> 100\%\) apply the limitation from the machine data. |
| Signal state 0 or edge change 1 → 0 | The maximum velocity of the control motion is not dependent on the override setting. |
| Application example | The difference for the operator is particularly dependent on whether the sensor motion is stopped or not with a 0 override. |
| Corresponding to .... | channel-specific override settings: DB21, ... DBB4 (feed rate override) DB21, ... DBX6.7 (feed rate override active) |
### 20.7.2 Signals from channel (DB21, ...)

| DB21, ... DBX37.3 | CLC is active |
|-------------------|---------------|
| Edge evaluation: No | Signal(s) updated: Cyclic |
| Signal state 1 or edge change 0 → 1 | Clearance control is activated. |
| Signal state 0 or edge change 1 → 0 | Clearance control is deactivated. |

| DB21, ... DBX37.4 | CLC motion at lower motion limit |
|-------------------|---------------------------------|
| Edge evaluation: No | Signal(s) updated: Cyclic |
| Signal state 1 or edge change 0 → 1 | The traversing movement of the clearance-controlled axes based on clearance control has been stopped at the upper movement limit set in MD62505 $MC_CLC_SENSOR_LOWER_LIMIT (lower clearance control motion limit) or programmed with CLC_LIM(..).  
**Note:**  
The following signal is set at the same time: DB21, ... DBX37.5 (CLC-stopped upper limit)  
→ see "CLC motion has stopped" signal. |
| Signal state 0 or edge change 1 → 0 | Clearance control has left the lower limitation. |

| DB21, ... DBX37.5 | CLC motion at upper motion limit |
|-------------------|---------------------------------|
| Edge evaluation: No | Signal(s) updated: Cyclic |
| Signal state 1 or edge change 0 → 1 | The traversing movement of the clearance-controlled axes based on clearance control has been stopped at the upper movement limit set in MD62506 $MC_CLC_SENSOR_UPPER_LIMIT (upper clearance control motion limit) or programmed with CLC_LIM(..).  
**Note:**  
The following signal is set at the same time: DB21, ... DBX37.4 (CLC-stopped lower limit)  
→ see "CLC motion has stopped" signal. |
| Signal state 0 or edge change 1 → 0 | Clearance control has left the upper limitation. |
### DB21, ... DBX37.4-5: CLC motion has stopped

| Edge evaluation: No | Signal(s) updated: Cyclic |
|---------------------|---------------------------|
| Signal state 1 or edge change 0 → 1 |

The traversing movement of the clearance-controlled axes based on clearance control is at a standstill. The following conditions will set the signal:

- The standstill conditions set in the following machine data are met:
  - \texttt{MD62520 $MC\_CLC\_SENSOR\_STOP\_POS\_TOL} (positional tolerance for status message "Clearance control zero speed")
  - \texttt{MD62521 $MC\_CLC\_SENSOR\_STOP\_DWELL\_TIME} (wait time for status message "Clearance control zero speed")

- Programming of \texttt{CLC\_GAIN=0.0}

- PLC interface signal:
  - \texttt{DB21, … DBX1.4 (stop CLC motion)}

**Note:**

The "CLC motion at standstill" signal is only set if bits 4 and 5 are set at the same time. If only one of the two bits is set → see below.

| Signal state 0 or edge change 1 → 0 |

Clearance control generates traversing movements directly in the clearance-controlled axes. As long as the axes are traversing due to clearance control, the axial interface signals cannot be set:

- \texttt{DB31, … DBX60.6/7 (position reached with exact stop coarse/fine)}

**Corresponding to ...**

- \texttt{DB3x DBX30.6/7 "Position reached, exact stop coarse/fine"}
20.8 Speed/Torque Coupling, Master-Slave (TE3)

20.8.1 Signals to axis/spindle (DB31, ...)

| DB31, ... | Torque compensatory controller on |
|-----------|-----------------------------------|
| **DBX24.4** | Edge evaluation: Yes | Signal(s) updated: Cyclically |
| **Signal state 1 or edge change 0 → 1** | Torque compensatory controller is to be activated. |
| | The following conditions must be fulfilled for activation: |
| | DB31, ... DBX96.2 (**fine** differential speed reached) |
| **Signal state 0 or edge change 1 → 0** | Torque compensatory controller is to be deactivated. |

| DB31, ... | Master/slave on |
|-----------|-----------------|
| **DBX24.7** | Edge evaluation: Yes | Signal(s) updated: Cyclic |
| **Signal state 1 or edge change 0 → 1** | Master-slave coupling should be activated. |
| **Signal state 0 or edge change 1 → 0** | Master-slave coupling should be deactivated. |
| | The following conditions must be fulfilled for activation and deactivation: |
| | • Leading and following axes under position control (DB31, ... DBX61.5) |
| | • Leading and following axes are at standstill (DB31, ... DBX61.4) |
| | • The channels of the leading and following axes are in the reset state (DB21, ... DBX35.7) |
| | If a condition is not fulfilled, the coupling will not be activated or deactivated. No alarm appears and the status of the coupling remains the same. |
| | If, at a later point, all conditions are fulfilled, the coupling will be activated or deactivated depending on the status of the signal. |
| | The signal for the following axis of a coupling is relevant. |
### 20.8.2 Signals from axis/spindle (DB31, ...)  

**DBX96.2**  
**Master/slave fine**  
Edge evaluation: No  
Signal(s) updated: Cyclic  

| Signal state 1 or edge change 0 → 1 | The differential speed is in the range defined by the following machine data: MD37272 $MA_MS_VELO_TOL_FINE |
|-----------------------------------|--------------------------------------------------------------------------------|
| Signal state 0 or edge change 1 → 0 | The differential speed has not reached the range defined in MD37272. |

**DBX96.3**  
**Master/slave coarse**  
Edge evaluation: No  
Signal(s) updated: Cyclic  

| Signal state 1 or edge change 0 → 1 | The differential speed is in the range defined by the following item of machine data: MD37270 $MA_MS_VELO_TOL_COARSE |
|-----------------------------------|--------------------------------------------------------------------------------|
| Signal state 0 or edge change 1 → 0 | The differential speed has not reached the range defined in MD37270. |

**DBX96.4**  
**Master/slave compensatory cont. active**  
Edge evaluation: No  
Signal(s) updated: Cyclic  

| Signal state 1 or edge change 0 → 1 | Torque compensatory control is active. |
|-----------------------------------|--------------------------------------------------------------------------------|
| Signal state 0 or edge change 1 → 0 | Torque compensatory control is not active.  
The signal at the slave axis of a coupling is relevant. |

**DBX96.7**  
**Master/slave active**  
Edge evaluation: No  
Signal(s) updated: Cyclic  

| Signal state 1 or edge change 0 → 1 | Master-slave coupling is active. |
|-----------------------------------|--------------------------------------------------------------------------------|
| Signal state 0 or edge change 1 → 0 | Master-slave coupling is not active.  
The signal at the following axis of a coupling is relevant. |
## 20.9 Handling Transformation Package (TE4)

### 20.9.1 Signals from channel (DB21, ...)

| DB21, ... | Activate PTP traversal |
|-----------|------------------------|
| DBX29.4   |                        |

Edge evaluation: Yes

| Signal state 1 or edge change 0 → 1 | Activate PTP traversal. |
|-------------------------------------|--------------------------|
| Signal state 0 or edge change 1 → 0 | Activate CP traversal.   |

Signal irrelevant for ...

No handling transformations active.

Additional references Function Manual, Special Functions; 3-Axis to 5-Axis Transformation (F2)

| DB21, ... | Transformation active |
|-----------|-----------------------|
| DBX33.6   |                       |

Edge evaluation: Yes

| Signal state 1 or edge change 0 → 1 | Active transformation. |
|-------------------------------------|-------------------------|
| Signal state 0 or edge change 1 → 0 | Transformation not (no longer) active. |

Signal irrelevant for ...

No transformation used.

Additional references Function Manual, Special Functions; 3-Axis to 5-Axis Transformation (F2)
### Handling Transformation Package (TE4)

#### Special functions

- **DB21, ...**
  - **DBB232**
  - **Number of active G function of G function group 25 (tool orientation reference)**
  - **Edge evaluation:**
    - **Meaning 1** ORIKWS: The tool orientation is implemented in a workpiece coordinate system and is thus not dependent on the machine kinematics.
    - **Meaning 2** ORIKMS: The tool orientation is defined in the machine coordinate system and is thus dependent on the machine kinematics. This is the default setting.
    - **Meaning 3** ORIPATH: The tool orientation is implemented with the programmed lead and side angles relative to the path tangent and surface normal vector.

- **DB21, ...**
  - **DBX317.6**
  - **PTP traversal active**
  - **Edge evaluation:** Yes
    - **Signal state 1 or edge change 0 → 1** PTP traversal active.
    - **Signal state 0 or edge change 1 → 0** CP traversal active.
    - **Signal irrelevant for** No handling transformations active.
    - **Additional references** Function Manual, Special Functions; 3-Axis to 5-Axis Transformation (F2)
20.10 MCS Coupling (TE6)

20.10.1 Signals to axis/spindle (DB31, ...)

### DB31, ... DBX24.2
| Deactivate or disable coupling |
|--------------------------------|
| Edge evaluation: No Signal(s) updated: |
| Signal state 1 An active coupling is not deactivated until the relevant axes are stationary. If CC_COPON is programmed for this axis, no error message is generated. |
| Signal state 0 Coupling may be activated |
| Signal irrelevant for ... |
| Application example(s) Evaluated only on the CC_Slave axis. |

### DB31, ... DBX24.3
| Switch on collision protection |
|--------------------------------|
| Edge evaluation: Yes Signal(s) updated: |
| Signal state 1 Collision protection ON |
| Signal state 0 Collision protection OFF |
| Signal irrelevant for ... This signal is processed only if collision protection is not activated via machine data: MD65543 $MA_CC_PROTECT_OPTIONS |
| Application example(s) Evaluated only on the PSlave axis. |

20.10.2 Signals from axis/spindle (DB31, ...)

### DB31, ... DBX66.0
| Activate monitor |
|------------------|
| Edge evaluation: No Signal(s) updated: |
| Signal state 1 Monitoring is active. This readout must be activated by the PSlave axis in the the following machine data: MD63543 $MD_CC_PROTECT_OPTIONS Note: Conflicts may occur in connection with customer-specific compile cycles. |
| Signal state 0 Monitor is not active. |
### Z3: NC/PLC interface signals

#### 20.10 MCS Coupling (TE6)

| DB31, ... | DBX97.0 | **Axis is a slave axis** |
|-----------|---------|-------------------------|
| Edge evaluation: No | Signal(s) updated: |
| Signal state 1 | Axis is a CC_Slave axis. The associated CC_Master axis can be found in the machine data. |
| Signal state 0 | Axis is not a CC_Slave axis |

| DB31, ... | DBX97.1 | **Activate coupling** |
|-----------|---------|-----------------------|
| Edge evaluation: No | Signal(s) updated: |
| Signal state 1 | Coupling active |
| Signal state 0 | Coupling not active |
| Signal irrelevant for ... | |
| Application example(s) | Displayed only for the CC_Slave axis. |

| DB31, ... | DBX97.2 | **Activate mirroring** |
|-----------|---------|-----------------------|
| Edge evaluation: No | Signal(s) updated: |
| Signal state 1 | Mirroring active (1:-1) |
| Signal state 0 | 1:1 coupling active |
| Signal irrelevant for ... | Relevant only if coupling is active (DBX97.1 = 1) |
| Application example(s) | Displayed only for the CC_Slave axis. |

| DB31, ... | DBX97.3 | **Offset after point of activation** |
|-----------|---------|------------------------------------|
| Edge evaluation: Yes | Signal(s) updated: |
| Signal state 1 | New offset after point of activation. This bit is set to 1 if a particular event (SW/HW limit switch on CC_Slave axis) causes a change in the offset between CC_Master and CC_Slave stored when the coupling was activated. |
| Signal state 0 | No new offset since activation. |
| Signal irrelevant for ... | The bit is not set in the RESET phase. |
| Application example(s) | Displayed only for the CC_Slave axis. |
20.11 Retrace Support (TE7)

20.11.1 Signals to channel

| DB21, ...  | Reverse/Forward |
|------------|-----------------|
| DBX0.1     |                 |

Edge evaluation: Yes
Signal(s) updated:

Signal state 1 or edge change 0 → 1
Activate reverse travel.
The RESU main program CC_RESU.MPF is generated from the traversing blocks recorded in the RESU-internal block buffer in order to initiate travel back along the contour on the next NC START.

Edge change 1 → 0
Activate forward travel.
The RESU main program CC_RESU.MPF is generated from the traversing blocks recorded in the RESU-internal block buffer in order to initiate travel forwards along the contour on the next NC START.

Signal state 0
Not relevant.
Signal irrelevant for ...
RESU technological function not loaded or not activated.

| DB21, ...  | Start retrace support |
|------------|-----------------------|
| DBX0.2     |                       |

Edge evaluation: No
Signal(s) updated:

Signal state 1
Start retrace support:
The original machining program is reselected and a block search is carried out to locate the program continuation point.

Signal state 0
Not relevant.
Signal irrelevant for ...
The NC is not in Retrace mode or RESU is not active.
### 20.11.2 Signals from channel

| DB21, ... | DBX32.1 | Retrace mode active |
|-----------|---------|---------------------|
| Edge evaluation: No | Signal(s) updated: |
| Signal state 1 | The "Retrace mode active" signal is active as long as the control is in Retrace mode. This is the case from initial activation of the "reverse/forward" signal until activation of the "start retrace support" signal. |
| Signal state 0 | The machining program is executed. |
| Signal irrelevant for ... | RESU technological function not active. |

| DB21, ... | DBX32.2 | Retrace support active |
|-----------|---------|-----------------------|
| Edge evaluation: No | Signal(s) updated: |
| Signal state 1 | The "retrace support active" signal is set as soon as signal state 1 is detected for the "start retrace support" signal. The "retrace support active" signal is reset at the end of retrace support once the last action block has been completed. |
| Signal state 0 | Retrace support not active |
| Signal irrelevant for ... | RESU technological function not active. |
Z3: NC/PLC interface signals

20.11 Retrace Support (TE7)
## Appendix

### A.1 List of abbreviations

| A          | Description                                      |
|-----------|--------------------------------------------------|
| O         | Output                                           |
| ADI4      | (Analog drive interface for 4 axes)             |
| AC        | Adaptive Control                                |
| ALM       | Active Line Module                              |
| ARM       | Rotating induction motor                        |
| AS        | PLC                                             |
| ASCII     | American Standard Code for Information Interchange |
| ASIC      | Application-Specific Integrated Circuit: User switching circuit |
| ASUB      | Asynchronous subprogram                         |
| AUXFU     | Auxiliary Function                              |
| STL       | Statement List                                  |
| UP        | User Program                                    |

| B          | Description                                      |
|-----------|--------------------------------------------------|
| BA        | Mode                                             |
| BAG       | Mode group                                       |
| BCD       | Binary Coded Decimals: Decimal numbers encoded in binary code |
| BERO      | Proximity limit switch with feedback oscillator  |
| BI        | Binector Input                                  |
| BICO      | Binector Connector                              |
| BIN       | Binary files                                    |
| BIOS      | Basic Input Output System                       |
| BCS       | Basic Coordinate System                         |
| BO        | Binector Output                                 |
| OPI       | Operator Panel Interface                        |

| C          | Description                                      |
|-----------|--------------------------------------------------|
| CAD       | Computer-Aided Design                            |
| CAM       | Computer-Aided Manufacturing                     |
| CC        | Compile Cycle                                   |
| CI        | Connector Input                                 |
| CF Card   | Compact Flash Card                               |
## Appendix
### A.1 List of abbreviations

| Abbreviation | Description |
|--------------|-------------|
| CNC          | Computerized Numerical Control |
| CO           | Connector Output |
| CoL          | Certificate of License |
| COM          | Communication |
| CPA          | Compiler Projecting Data: Configuring data of the compiler |
| CRT          | Cathode Ray Tube |
| CSB          | Central Service Board: PLC module |
| CU           | Control Unit |
| CP           | Communication Processor |
| CPU          | Central Processing Unit |
| CR           | Carriage Return |
| CTS          | Clear To Send: Ready to send signal for serial data interfaces |
| CUTCOM       | Cutter radius Compensation: Tool radius compensation |

| Abbreviation | Description |
|--------------|-------------|
| DAC          | Digital-to-Analog Converter |
| DB           | Data Block (PLC) |
| DBB          | Data Block Byte (PLC) |
| DBD          | Data Block Double word (PLC) |
| DBW          | Data Block Word (PLC) |
| DBX          | Data block bit (PLC) |
| DDE          | Dynamic Data Exchange |
| DIN          | Deutsche Industrie Norm |
| DIO          | Data Input/Output: Data transfer display |
| DIR          | Directory |
| DLL          | Dynamic Link Library |
| DO           | Drive Object |
| DPM          | Dual Port Memory |
| DPR          | Dual Port RAM |
| DRAM         | Dynamic memory (non-buffered) |
| DRF          | Differential Resolver Function (handwheel) |
| DRIVE-CLiQ   | Drive Component Link with IQ |
| DRY          | Dry Run: Dry run feedrate |
| DSB          | Decoding Single Block |
| DSC          | Dynamic Servo Control / Dynamic Stiffness Control |
| DW           | Data Word |
| DWORD        | Double Word (currently 32 bits) |
## Appendix
### A.1 List of abbreviations

| Abbreviation | Description |
|--------------|-------------|
| E            | I Input     |
|              | I/O Input/Output       |
| ENC          | Encoder: Actual value encoder |
| EFP          | Compact I/O module (PLC I/O module) |
| ESD          | Electrostatic Sensitive Devices |
| EMC          | ElectroMagnetic Compatibility |
| EN           | European standard |
| EnDat        | Encoder interface |
| EPROM        | Erasable Programmable Read Only Memory |
| ePS Network Services | Services for Internet-based remote machine maintenance |
| EQN          | Designation for an absolute encoder with 2048 sine signals per revolution |
| ES           | Engineering System |
| ESR          | Extended Stop and Retract |
| ETC          | ETC key "->"; softkey bar extension in the same menu |

| Abbreviation | Description |
|--------------|-------------|
| F            | FB Function Block (PLC) |
|              | FC Function Call (PLC) |
| FEPROM       | Flash EPROM: Read and write memory |
| FIFO         | First In First Out: Memory that works without address specification and whose data is read in the same order in which they was stored |
| FPO          | Fine interpolator |
| FPU          | Floating Point Unit |
| CRC          | Cutter Radius Compensation |
| FST          | Feed Stop: Feedrate stop |
| FBD          | Function Block Diagram (PLC programming method) |
| FW           | Firmware |

| Abbreviation | Description |
|--------------|-------------|
| G            | GC Global Control (PROFIBUS: Broadcast telegram) |
| GEO          | Geometry, e.g. geometry axis |
| GIA          | Gear Interpolation Data |
| GND          | Signal Ground |
| GP           | Basic program (PLC) |
| GS           | Gear Stage |
| GSD          | Device master file for describing a PROFIBUS slave |
## Appendix

### A.1 List of abbreviations

| Abbreviation | Description |
|--------------|-------------|
| GSDML        | Generic Station Description Markup Language: XML-based description language for creating a GSD file |
| GUD          | Global User Data |
| HEX          | Abbreviation for hexadecimal number |
| AuxF         | Auxiliary Function |
| HLA          | Hydraulic linear drive |
| HMI          | Human Machine Interface: SINUMERIK user interface |
| MSD          | Main Spindle Drive |
| HW           | Hardware |
| IBN          | Commissioning |
| ICA          | Interpolatory compensation |
| IM           | Interface Module |
| IMR          | Interface Module Receive: Interface module for receiving data |
| IMS          | Interface Module Send: Interface module for sending data |
| INC          | Increment |
| INI          | Initializing Data |
| IPO          | Interpolator |
| ISA          | Industry Standard Architecture |
| ISO          | International Standardization Organization |
| JOG          | Jogging: Setup mode |
| KV           | Gain factor of control loop |
| Kp           | Proportional gain |
| Ku           | Transformation ratio |
| LAD          | Ladder Diagram (PLC programming method) |
| LAI          | Logic Machine Axis Image |
| LAN          | Local Area Network |
### A.1 List of abbreviations

| Abbreviation | Description |
|--------------|-------------|
| LCD          | Liquid Crystal Display |
| LED          | Light Emitting Diode  |
| LF           | Line Feed      |
| PMS          | Position Measuring System |
| LR           | Position controller |
| LSB          | Least Significant Bit |
| LUD          | Local User Data  |
| MAC          | Media Access Control |
| MAIN         | Main program (OB1, PLC) |
| MB           | Megabyte       |
| MCI          | Motion Control Interface |
| MCIS         | Motion Control Information System |
| MCP          | Machine Control Panel |
| MD           | Machine Data   |
| MDA          | Manual Data Automatic: Manual input |
| MSGW         | Message Word   |
| MCS          | Machine Coordinate System |
| MLFB         | Machine-readable product code |
| MM           | Motor Module   |
| MPF          | Main Program File (NC) |
| MCP          | Machine Control Panel |
| NC           | Numerical Control |
| NCK          | Numerical Control Kernel: NC kernel with block preparation, traversing range, etc. |
| NCU          | Numerical Control Unit: NCK hardware unit |
| NRK          | Name for the operating system of the NCK |
| IS           | Interface Signal |
| NURBS        | Non-Uniform Rational B-Spline |
| ZO           | Zero Offset    |
| NX           | Numerical Extension: Axis expansion board |
| OB           | Organization block in the PLC |
| OEM          | Original Equipment Manufacturer |
| OP           | Operator Panel |
| OPI          | Operator Panel Interface |

Special functions

Function Manual, 07/2012, 6FC5397-2BP40-3BA0
### Appendix

#### A.1 List of abbreviations

| OPT       | Options                      |
|-----------|------------------------------|
| OLP       | Optical Link Plug: Fiber optic bus connector |
| OSI       | Open Systems Interconnection: Standard for computer communications |

| P          |                              |
|------------|------------------------------|
| PIQ        | Process Image Output         |
| PII        | Process Image Input          |
| PC         | Personal Computer            |
| PCIN       | Name of the SW for data exchange with the controller |
| PCMCIA     | Personal Computer Memory Card International Association: Plug-in memory card standardization |
| PCU        | PC Unit                      |
| PG         | Programming device           |
| PKE        | Parameter identification: Part of a PIV |
| PIV        | Parameter Identifier Value (parameterization part of a PPO) |
| PLC        | Programmable Logic Control: Adaptation control |
| PN         | PROFINET                     |
| PNO        | PROFIBUS user organization   |
| PO         | POWER ON                     |
| POU        | Program Organization Unit    |
| POS        | Position/positioning         |
| POSMO A    | Positioning Motor Actuator   |
| POSMO CA   | Positioning Motor Compact AC: Complete drive unit with integrated power and control module as well as positioning unit and program memory; AC infeed |
| POSMO CD   | Positioning Motor Compact DC: Like CA but with DC infeed |
| POSMO SI   | Positioning Motor Servo Integrated: Positioning motor, DC infeed |
| PPO        | Parameter Process data Object: Cyclic data telegram for PROFIBUS DP transmission and "Variable speed drives" profile |
| PPU        | Panel Processing Unit (central hardware for a panel-based CNC, e.g SINUMERIK 828D) |
| PROFIBUS   | Process Field Bus: Serial data bus |
| PRT        | Program Test                 |
| PSW        | Program control word         |
| PTP        | Point-To-Point               |
| PUD        | Program global User Data     |
| PZD        | Process data: Process data part of a PPO |

| Q          |                              |
|------------|------------------------------|
| QEC        | Quadrant Error Compensation  |
### Appendix

**A.1 List of abbreviations**

| R | RAM | Random Access Memory: Read/write memory |
|---|-----|---------------------------------------|
| | REF | REFerence point approach function |
| | REPOS | REPOSitioon function |
| | RISC | Reduced Instruction Set Computer: Type of processor with small instruction set and ability to process instructions at high speed |
| | ROV | Rapid Override: Input correction |
| | RP | R Parameter, arithmetic parameter, predefined user variable |
| | RPA | R Parameter Active: Memory area on the NCK for R parameter numbers |
| | RPY | Roll Pitch Yaw: Rotation type of a coordinate system |
| | RTLI | Rapid Traverse Linear Interpolation: |
| | RTS | Request To Send: Control signal of serial data interfaces |
| | RTCP | Real Time Control Protocol |

| S | SA | Synchronized Action |
|---|----|---------------------|
| | SBC | Safe Brake Control |
| | SBL | Single Block |
| | SBR | Subprogram (PLC) |
| | SD | Setting Data |
| | SDB | System Data Block |
| | SEA | Setting Data Active: Identifier (file type) for setting data |
| | SERUPRO | SEarch RU by PROgram test |
| | SFB | System Function Block |
| | SFC | System Function Call |
| | SGE | Safety-related input |
| | SGA | Safety-related output |
| | SH | Safe standstill |
| | SIM | Single Inline Module |
| | SK | Softkey |
| | SKP | Skip: Function for skipping a part program block |
| | SLM | Synchronous Linear Motor |
| | SM | Stepper Motor |
| | SMC | Sensor Module Cabinet Mounted |
| | SME | Sensor Module Externally Mounted |
| | SMI | Sensor Module Integrated |
| | SPF | Subprogram File (NC) |
| | PLC | Programmable Logic Controller |
| | SRAM | Static RAM (non-volatile) |
| | TNRC | Tool Nose Radius Compensation |
| | SRM | Synchronous Rotary Motor |
### A.1 List of abbreviations

| Abbreviation | Description |
|--------------|-------------|
| LEC          | Leadscrew Error Compensation |
| SSI          | Synchronous Serial Interface |
| SSL          | Block search |
| STW          | Control word |
| GWPS         | Grinding Wheel Peripheral Speed |
| SW           | Software |
| SYF          | System Files |
| SYNACT       | SYNchronized ACTion: Synchronized Action |
| TB           | Terminal Board (SINAMICS) |
| TCP          | Tool Center Point |
| TCP/IP       | Transport Control Protocol / Internet Protocol |
| TCU          | Thin Client Unit |
| TEA          | Testing Data Active: Identifier for machine data |
| TIA          | Totally Integrated Automation |
| TM           | Terminal Module (SINAMICS) |
| TO           | Tool Offset |
| TOA          | Tool Offset Active: Identifier (file type) for tool offsets |
| TRANSMIT     | Transform Milling Into Turning: Coordination transformation for milling operations on a lathe |
| TTL          | Transistor-Transistor Logic (interface type) |
| TZ           | Technology cycle |
| UFR          | User Frame: Zero Offset |
| SR           | Subprogram |
| USB          | Universal Serial Bus |
| UPS          | Uninterruptible Power Supply |

| Abbreviation | Description |
|--------------|-------------|
| VDI          | Internal communication interface between NCK and PLC |
| VDE          | Verein Deutscher Ingenieure [Association of German Engineers] |
| VDI          | Verband Deutscher Elektrotechniker [Association of German Electrical Engineers] |
| VI           | Voltage Input |
| VO           | Voltage Output |
| FDD          | Feed Drive |
| W                  | Description                                    |
|--------------------|------------------------------------------------|
| SAR                | Smooth Approach and Retraction                 |
| WCS                | Workpiece Coordinate System                    |
| T                  | Tool                                           |
| TLC                | Tool Length Compensation                       |
| WOP                | Workshop-Oriented Programming                  |
| WPD                | Workpiece Directory                            |
| TRC                | Tool Radius Compensation                       |
| T                  | Tool                                           |
| TO                 | Tool Offset                                    |
| TM                 | Tool Management                                |
| TC                 | Tool change                                    |

| X                  | Description                                    |
|--------------------|------------------------------------------------|
| XML                | Extensible Markup Language                     |

| Z                  | Description                                    |
|--------------------|------------------------------------------------|
| ZOA                | Zero Offset Active: Identifier for zero offsets |
| ZSW                | Status word (of drive)                         |
Appendix

A.1 List of abbreviations
A.2 Overview

General documentation

- Sales Brochure
  - SINUMERIK 840D sl
  - SINUMERIK 828D
  - SINUMERIK 826D
  - BASIC T
- Catalog NC 62
- Catalog NC 82
- Catalog PM 21 SIMOTION,
  SINAMICS S120 and Motors for
  Production Machines
- Configuration Manual
  EMC Installation
  Guidelines
- System Manual
  Ctrl Energy

User documentation

- Operating Manual
  - Universal
  - Turning
  - Milling
  - HMI-Advanced
- Operating Manual
  - Turning
  - Milling
- Programming Manual
  - Fundamentals
  - Job Planning
  - Measuring Cycles
  - ISO Turning
  - ISO Milling
- Diagnostics Manual
  - System Manual

Manufacturer/service documentation

- Manual
  - NCU
  - Operator Components
    and Networking
- Commissioning Manual
  - CNC: NCK, PLC,
    Drive
  - Basic Software
    and Operating Software
  - Basic Software and
    HMI-Advanced
- List Manual
  - Part 1
  - Part 2
  - Detailed
    Machine Data
    Description
  - System Variables
- List Manual
  - Machine Data
    and Interface Signals
  - Parameter
    Description
- System Manual
  Guidelines for
  Machine Configuration

Manufacturer/service documentation

- Function Manual
  - Basic Functions
  - Extended Functions
  - Synchronized Actions
  - Tool Management
- Function Manual
  - ISO Dialects
  - Drive Functions
- Function Manual
  - Safety Integrated
  - Safety Integrated

Info/Training

- Getting Started
  Milling and Turning
- Training Documents
  - Easy Milling with ShopMill
  - Easy Turning with ShopTurn
- Manuals
  Tool and
  Mold Making

Electronic documentation

- DOConCD
- My Documentation
  Manager
- Industry Mall
Glossary

Absolute dimensions
A destination for an axis motion is defined by a dimension that refers to the origin of the currently active coordinate system. See → Incremental dimension

Acceleration with jerk limitation
In order to optimize the acceleration response of the machine whilst simultaneously protecting the mechanical components, it is possible to switch over in the machining program between abrupt acceleration and continuous (jerk-free) acceleration.

Address
An address is the identifier for a certain operand or operand range, e.g. input, output, etc.

Alarms
All → messages and alarms are displayed on the operator panel in plain text with date and time and the corresponding symbol for the cancel criterion. Alarms and messages are displayed separately.
1. Alarms and messages in the part program:
   Alarms and messages can be displayed in plain text directly from the part program.
2. Alarms and messages from the PLC:
   Alarms and messages for the machine can be displayed in plain text from the PLC program. No additional function block packages are required for this purpose.

Archiving
Reading out of files and/or directories on an external memory device.

Asynchronous subprogram
Part program that can be started asynchronously to (independently of) the current program status using an interrupt signal (e.g. "Rapid NC input" signal).

Automatic
Operating mode of the controller (block sequence operation according to DIN): Operating mode for NC systems in which a → subprogram is selected and executed continuously.
Auxiliary functions

Auxiliary functions enable → part programs to transfer → parameters to the → PLC, which then trigger reactions defined by the machine manufacturer.

Aaxes

In accordance with their functional scope, the CNC axes are subdivided into:

- Axes: Interpolating path axes
- Auxiliary axes: Non-interpolating feed and positioning axes with an axis-specific feedrate. Auxiliary axes are not involved in actual machining, e.g. tool feeder, tool magazine.

Axis address

See → Axis identifier

Axis identifier

Axes are identified using X, Y, and Z as defined in DIN 66217 for a dextrorotory, right-angled → coordinate system.

Rotary axes rotating around X, Y, and Z are identified using A, B, and C. Additional axes situated parallel to the specified axes can be designated using other letters.

Axis name

See → Axis identifier

Backlash compensation

Compensation for a mechanical machine backlash, e.g. backlash on reversal for ball screws. Backlash compensation can be entered separately for each axis.

Backup battery

The backup battery ensures that the → user program in the → CPU is stored so that it is safe from power failure and so that specified data areas and bit memory, timers and counters are stored retentively.

Basic axis

Axis whose setpoint or actual value position forms the basis of the calculation of a compensation value.

Basic Coordinate System

Cartesian coordinate system which is mapped by transformation onto the machine coordinate system.
The programmer uses axis names of the basic coordinate system in the part program. The basic coordinate system exists parallel to the machine coordinate system if no transformation is active. The difference between the two coordinate systems lies in the axis identifiers.

**Baud rate**

Rate of data transfer (bits/s).

**Blank**

Workpiece as it is before it is machined.

**Block**

"Block" is the term given to any files required for creating and processing programs.

**Block search**

For debugging purposes or following a program abort, the "Block search" function can be used to select any location in the part program at which the program is to be started or resumed.

**Booting**

Loading the system program after power ON.

**C axis**

Axis around which the tool spindle describes a controlled rotational and positioning motion.

**C spline**

The C spline is the most well-known and widely used spline. The transitions at the interpolation points are continuous, both tangentially and in terms of curvature. 3rd order polynomials are used.

**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. Part program runs of different channels can be coordinated through synchronization.

**Circular interpolation**

The tool moves on a circle between specified points on the contour at a given feedrate, and the workpiece is thereby machined.
CNC
See → NC

COM
Component of the NC for the implementation and coordination of communication.

Compensation axis
Axis with a setpoint or actual value modified by the compensation value

Compensation table
Table containing interpolation points. It provides the compensation values of the compensation axis for selected positions on the basic axis.

Compensation value
Difference between the axis position measured by the encoder and the desired, programmed axis position.

Continuous-path mode
The objective of continuous-path mode is to avoid substantial deceleration of the path axes at the part program block boundaries and to change to the next block at as close to the same path velocity as possible.

Contour
Contour of the → workpiece

Contour monitoring
The following error is monitored within a definable tolerance band as a measure of contour accuracy. An unacceptably high following error can cause the drive to become overloaded, for example. In such cases, an alarm is output and the axes are stopped.

Coordinate system
See → Machine coordinate system, → Workpiece coordinate system

CPU
Central processing unit, see → PLC
Curvature

The curvature $k$ of a contour is the inverse of radius $r$ of the nestling circle in a contour point ($k = \frac{1}{r}$).

Cycles

Protected subprograms for execution of repetitive machining operations on the → workpiece.

Data block

1. Data unit of the → PLC that → HIGHSTEP programs can access.
2. Data unit of the → NC: Data modules contain data definitions for global user data. This data can be initialized directly when it is defined.

Data word

Two-byte data unit within a → data block.

Diagnostics

1. Operating area of the controller.
2. The controller has a self-diagnostics program as well as test functions for servicing purposes: status, alarm, and service displays

Dimensions specification, metric and inches

Position and pitch values can be programmed in inches in the machining program. Irrespective of the programmable dimensions ($G70/G71$), the controller is set to a basic system.

DRF

Differential Resolver Function: NC function which generates an incremental zero offset in Automatic mode in conjunction with an electronic handwheel.

Drive

The drive is the unit of the CNC that performs the speed and torque control based on the settings of the NC.

Dynamic feedforward control

Inaccuracies in the → contour due to following errors can be practically eliminated using dynamic, acceleration-dependent feedforward control. This results in excellent machining accuracy even at high → path velocities. Feedforward control can be selected and deselected on an axis-specific basis via the → part program.
Editor

The editor makes it possible to create, edit, extend, join, and import programs/texts/program blocks.

Exact stop

When an exact stop statement is programmed, the position specified in a block is approached exactly and, if necessary, very slowly. To reduce the approach time, → exact stop limits are defined for rapid traverse and feed.

Exact stop limit

When all path axes reach their exact stop limits, the controller responds as if it had reached its precise destination point. A block advance of the → part program occurs.

External zero offset

Zero offset specified by the → PLC.

Fast retraction from the contour

When an interrupt occurs, a motion can be initiated via the CNC machining program, enabling the tool to be quickly retracted from the workpiece contour that is currently being machined. The retraction angle and the distance retracted can also be parameterized. An interrupt routine can also be executed following the fast retraction.

Feed override

The programmed velocity is overridden by the current velocity setting made via the → machine control panel or from the → PLC (0 to 200%). The feedrate can also be corrected by a programmable percentage factor (1 to 200%) in the machining program.

Finished-part contour

Contour of the finished workpiece. See → Raw part.

Fixed machine point

Point that is uniquely defined by the machine tool, e.g. machine reference point.

Fixed-point approach

Machine tools can approach fixed points such as a tool change point, loading point, pallet change point, etc. in a defined way. The coordinates of these points are stored in the controller. The controller moves the relevant axes in → rapid traverse, whenever possible.
Frame
A frame is an arithmetic rule that transforms one Cartesian coordinate system into another Cartesian coordinate system. A frame contains the following components: → zero offset, → rotation, → scaling, → mirroring.

Geometry
Description of a → workpiece in the → workpiece coordinate system.

Geometry axis
Geometry axes are used to describe a 2- or 3-dimensional area in the workpiece coordinate system.

Ground
Ground is taken as the total of all linked inactive parts of a device which will not become live with a dangerous contact voltage even in the event of a malfunction.

Helical interpolation
The helical interpolation function is ideal for machining internal and external threads using form milling cutters and for milling lubrication grooves.

The helix comprises two motions:
- Circular motion in one plane
- A linear motion perpendicular to this plane

High-level CNC language
The high-level language offers: → user-defined variables, → system variables, → macro techniques.

High-speed digital inputs/outputs
The digital inputs can be used for example to start fast CNC program routines (interrupt routines). High-speed, program-driven switching functions can be initiated via the digital CNC outputs

HIGHSTEP
Summary of programming options for → PLCs of the AS300/AS400 system.

HW Config
SIMATIC S7 tool for the configuration and parameterization of hardware components within an S7 project
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.

Inch measuring system

Measuring system which defines distances in inches and fractions of inches.

Inclined surface machining

Drilling and milling operations on workpiece surfaces that do not lie in the coordinate planes of the machine can be performed easily using the function "inclined-surface machining".

Increment

Travel path length specification based on 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).

Incremental dimension

Also incremental dimension: A destination for axis traversal is defined by a distance to be covered and a direction referenced to a point already reached. See → Absolute dimension.

Intermediate blocks

Motions with selected → tool offset (G41/G42) may be interrupted by a limited number of intermediate blocks (blocks without axis motions in the offset plane), whereby the tool offset can still be correctly compensated for. The permissible number of intermediate blocks which the controller reads ahead can be set in system parameters.

Interpolator

Logic unit of the → NCK that defines intermediate values for the motions to be carried out in individual axes based on information on the end positions specified in the part program.

Interpolatory compensation

Interpolatory compensation is a tool that enables manufacturing-related leadscrew error and measuring system error compensations (SSFK, MSFK).
Interrupt routine

Interrupt routines are special subprograms that can be started by events (external signals) in the machining process. A part program block which is currently being worked through is interrupted and the position of the axes at the point of interruption is automatically saved.

Inverse-time feedrate

The time required for the path of a block to be traversed can also be programmed for the axis motion instead of the feed velocity (G93).

JOG

Control operating mode (setup mode): In JOG mode, the machine can be set up. Individual axes and spindles can be traversed in JOG mode by means of the direction keys. Additional functions in JOG mode include: → Reference point approach, → Repos, and → Preset (set actual value).

Key switch

The key switch on the → machine control panel has four positions that are assigned functions by the operating system of the controller. The key switch has three different colored keys that can be removed in the specified positions.

Keywords

Words with specified notation that have a defined meaning in the programming language for → part programs.

KÜ

Transformation ratio

KV

Servo gain factor, a control variable in a control loop.

Leading axis

The leading axis is the → gantry axis that exists from the point of view of the operator and programmer and, thus, can be influenced like a standard NC axis.

Leadscrew error compensation

Compensation for the mechanical inaccuracies of a leadscrew participating in the feed. The controller uses stored deviation values for the compensation.
Limit speed

Maximum/minimum (spindle) speed: The maximum speed of a spindle can be limited by specifying machine data, the → PLC or → setting data.

Linear axis

In contrast to a rotary axis, a linear axis describes a straight line.

Linear interpolation

The tool travels along a straight line to the destination point while machining the workpiece.

Load memory

The load memory is the same as the → working memory for the CPU 314 of the → PLC.

Look Ahead

The Look Ahead function is used to achieve an optimal machining speed by looking ahead over an assignable number of traversing blocks.

Machine axes

Physically existent axes on 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

A channel structure can be used to shorten idle times by means of parallel motion sequences, e.g. moving a loading gantry simultaneously with machining. Here, a CNC channel must be regarded as a separate CNC control system with decoding, block preparation and interpolation.
Macro techniques

Grouping of a set of statements under a single identifier. The identifier represents the set of consolidated statements in the program.

Main block

A block prefixed by ":" introductory block, containing all the parameters required to start execution of a → part program.

Main program

The term "main program" has its origins during the time when part programs were split strictly into main and → subprograms. This strict division no longer exists with today's SINUMERIK NC language. In principle, any part program in the channel can be selected and started. It then runs through in → program level 0 (main program level). Further part programs or → cycles as subprograms can be called up in the main program.

MDA

Control operating mode: Manual Data Automatic. In the MDA mode, individual program blocks or block sequences with no reference to a main program or subprogram can be input and executed immediately afterwards through actuation of the NC start key.

Messages

All messages programmed in the part program and → alarms detected by the system are displayed on the operator panel in plain text with date and time and the corresponding symbol for the cancel criterion. Alarms and messages are displayed separately.

Metric measuring system

Standardized system of units: For length, e.g. mm (millimeters), m (meters).

Mirroring

Mirroring reverses the signs of the coordinate values of a contour, with respect to an axis. It is possible to mirror with respect to more than one axis at a time.

Mode

An operating concept on a SINUMERIK controller. The following modes are defined: → Jog, → MDA, → Automatic.

Mode group

Axes and spindles that are technologically related can be combined into one mode group. Axes/spindles of a mode group can be controlled by one or more → channels. The same → mode type is always assigned to the channels of the mode group.
NC
Numerical Control: Numerical control (NC) includes all components of machine tool control: → NCK, → PLC, HMI, → COM.

Note
A more correct term for SINUMERIK controllers would be: Computerized Numerical Control

NCK
Numerical Control Kernel: Component of NC that executes the → part programs and basically coordinates the motion operations for the machine tool.

Network
A network is the connection of multiple S7-300 and other end devices, e.g. a programming device via a → connecting cable. A data exchange takes place over the network between the connected devices.

NRK
Numeric robotic kernel (operating system of → NCK)

NURBS
The motion control and path interpolation that occurs within the controller is performed based on NURBS (Non Uniform Rational B-Splines). This provides a uniform procedure for all internal interpolations.

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 technology-specific functions in the controller.

Offset memory
Data range in the control, in which the tool offset data is stored.

Oriented spindle stop
 Stops the workpiece spindle in a specified angular position, e.g. in order to perform additional machining at a particular location.
Oriented tool retraction

RETTOOL: If machining is interrupted (e.g. when a tool breaks), a program command can be used to retract the tool in a user-specified orientation by a defined distance.

Overall reset

In the event of an overall reset, the following memories of the → CPU are deleted:
- → Working memory
- Read/write area of → load memory
- → System memory
- → Backup memory

Override

Manual or programmable control feature which enables the user to override programmed feedrates or speeds in order to adapt them to a specific workpiece or material.

Part program

Series of statements 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.

Part program block

Part of a → part program that is demarcated by a line feed. There are two types: → main blocks and → subblocks.

Part program management

Part program management can be organized by → workpieces. The size of the user memory determines the number of programs and the amount of data that can be managed. Each file (programs and data) can be given a name consisting of a maximum of 24 alphanumeric characters.

Path axis

Path axes include all machining axes of the → channel that are controlled by the → interpolator in such a way that they start, accelerate, stop, and reach their end point simultaneously.

Path feedrate

Path feed affects → path axes. It represents the geometric sum of the feedrates of the → geometry axes involved.
Path velocity

The maximum programmable path velocity depends on the input resolution. For example, with a resolution of 0.1 mm the maximum programmable path velocity is 1000 m/min.

PCIN data transfer program

PCIN is an auxiliary program for sending and receiving CNC user data (e.g. part programs, tool offsets, etc.) via a serial interface. The PCIN program can run in MS-DOS on standard industrial PCs.

Peripheral module

I/O modules represent the link between the CPU and the process.

I/O modules are:

- → Digital input/output modules
- → Analog input/output modules
- → Simulator modules

PLC

Programmable Logic Controller: → Programmable logic controller. Component of → NC: Programmable control for processing the control logic of the machine tool.

PLC program memory

SINUMERIK 840D sl: The PLC user program, the user data and the basic PLC program are stored together in the PLC user memory.

PLC programming

The PLC is programmed using the STEP 7 software. The STEP 7 programming software is based on the WINDOWS standard operating system and contains the STEP 5 programming functions with innovative enhancements.

Polar coordinates

A coordinate system which defines the position of a point on a plane in terms of its distance from the origin and the angle formed by the radius vector with a defined axis.

Polynomial interpolation

Polynomial interpolation enables a wide variety of curve characteristics to be generated, such as straight line, parabolic, exponential functions (SINUMERIK 840D sl).
Positioning axis

Axis that performs an auxiliary motion on a machine tool (e.g. tool magazine, pallet transport). Positioning axes are axes that do not interpolate with → path axes.

Pre-coincidence

Block change occurs already when the path distance approaches an amount equal to a specifiable delta of the end position.

Program block

Program blocks contain the main program and subprograms of → part programs.

Program level

A part program started in the channel runs as a → main program on program level 0 (main program level). Any part program called up in the main program runs as a → subprogram on a program level 1 ... n of its own.

Programmable frames

Programmable → frames enable dynamic definition of new coordinate system output points while the part program is being executed. A distinction is made between absolute definition using a new frame and additive definition with reference to an existing starting point.

Programmable logic controller

Programmable logic controllers (PLCs) are electronic controllers, the function of which is stored as a program in the control unit. This means that the layout and wiring of the device do not depend on the function of the controller. The programmable logic control has the same structure as a computer; it consists of a CPU (central module) with memory, input/output modules and an internal bus system. The peripherals and the programming language are matched to the requirements of the control technology.

Programmable working area limitation

Limitation of the motion space of the tool to a space defined by programmed limitations.

Programming key

Characters and character strings that have a defined meaning in the programming language for → part programs.

Protection zone

Three-dimensional zone within the → working area into which the tool tip must not pass.
**Quadrant error compensation**

Contour errors at quadrant transitions, which arise as a result of changing friction conditions on the guideways, can be virtually entirely eliminated with the quadrant error compensation. Parameterization of the quadrant error compensation is performed by means of a circuit test.

**R parameters**

Arithmetic parameter that can be set or queried by the programmer of the → part program for any purpose in the program.

**Rapid traverse**

The highest traverse rate of an axis. For example, rapid traverse is used when the tool approaches the → workpiece contour from a resting position or when the tool is retracted from the workpiece contour. The rapid traverse velocity is set on a machine-specific basis using a machine data element.

**Reference point**

Machine tool position that the measuring system of the → machine axes references.

**Rotary axis**

Rotary axes apply a workpiece or tool rotation to a defined angular position.

**Rotation**

Component of a → frame that defines a rotation of the coordinate system around a particular angle.

**Rounding axis**

Rounding axes rotate a workpiece or tool to an angular position corresponding to an indexing grid. When a grid index is reached, the rounding axis is "in position".

**RS-232-C**

Serial interface for data input/output. Machining programs as well as manufacturer and user data can be loaded and saved via this interface.

**Safety functions**

The controller is equipped with permanently active monitoring functions that detect faults in the → CNC, the → PLC, and the machine in a timely manner so that damage to the workpiece, tool, or machine is largely prevented. 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 has been triggered.
Scaling

Component of a → frame that implements axis-specific scale modifications.

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 (softkeys) are assigned defined functions in the software.

Software limit switch

Software limit switches limit the traversing range of an axis and prevent an abrupt stop of the slide at the hardware limit switch. Two value pairs can be specified for each axis and activated separately by means of the → PLC.

Spline interpolation

With spline interpolation, the controller can generate a smooth curve characteristic from only a few specified interpolation points of a set contour.

Standard cycles

Standard cycles are provided for machining operations which are frequently repeated:

- For the drilling/milling technology
- For turning technology

The available cycles are listed in the "Cycle support" menu in the "Program" operating area. Once the desired machining cycle has been selected, the parameters required for assigning values are displayed in plain text.

Subblock

Block preceded by "N" containing information for a sequence, e.g. positional data.

Subprogram

The term "subprogram" has its origins during the time when part programs were split strictly into →main and subprograms. This strict division no longer exists with today's SINUMERIK NC language. In principle, any part program or any → cycle can be called up as a subprogram within another part program. It then runs through in the next → program level (x+1) (subprogram level (x+1)).
Synchronization
Statements in → part programs for coordination of sequences in different → channels at certain machining points.

Synchronized actions
1. Auxiliary function output
   During workpiece machining, technological functions (→ auxiliary functions) can be output from the CNC program to the PLC. For example, these auxiliary functions are used to control additional equipment for the machine tool, such as quills, grabbers, clamping chucks, etc.
2. 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.

Synchronized axes
Synchronized axes take the same time to traverse their path as the geometry axes take for their path.

Synchronized axis
A synchronized axis is the → gantry axis whose set position is continuously derived from the motion of the → leading axis and is, thus, moved synchronously with the leading axis. From the point of view of the programmer and operator, the synchronized axis "does not exist".

System memory
The system memory is a memory in the CPU in which the following data is stored:
- Data required by the operating system
- The operands timers, counters, markers

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 $. See → User-defined variable.

Tapping without compensating chuck
This function allows threads to be tapped without a compensating chuck. By using the interpolating method of the spindle as a rotary axis and the drilling axis, threads can be cut to a precise final drilling depth, e.g. for blind hole threads (requirement: spindles in axis operation).
Text editor

See → Editor

TOA area

The TOA area includes all tool and magazine data. By default, this area coincides with the channel area with regard to the access of the data. However, machine data can be used to specify that multiple channels share one TOA unit so that common tool management data is then available to these channels.

TOA unit

Each TOA area can have more than one TOA unit. The number of possible TOA units is limited by the maximum number of active channels. A TOA unit includes exactly one tool data block and one magazine data block. In addition, a TOA unit can also contain a toolholder data block (optional).

Tool

Active part on the machine tool that implements machining (e.g. turning tool, milling tool, drill, LASER beam, etc.).

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 controller which then takes it into account. The curvature center is maintained equidistantly around the contour, offset by the curvature radius.

Tool offset

Consideration of the tool dimensions in calculating the path.

Tool radius compensation

To directly program a desired workpiece contour, the control must traverse an equistant path to the programmed contour taking into account the radius of the tool that is being used (G41/G42).

Transformation

Additive or absolute zero offset of an axis.

Travel range

The maximum permissible travel range for linear axes is ± 9 decades. The absolute value depends on the selected input and position control resolution and the unit of measurement (inch or metric).
Glossary

User interface

The user interface (UI) is the display medium for a CNC in the form of a screen. It features horizontal and vertical softkeys.

User memory

All programs and data, such as part programs, subprograms, comments, tool offsets, and zero offsets/frames, as well as channel and program user data, can be stored in the shared CNC user memory.

User program

User programs for the S7-300 automation systems are created using the programming language STEP 7. The user program has a modular layout and consists of individual blocks.

The basic block types are:

- Code blocks
  These blocks contain the STEP 7 commands.
- Data blocks
  These blocks contain constants and variables for the STEP 7 program.

User-defined variable

Users can declare their own variables for any purpose in the part program or data block (global user data). A definition contains a data type specification and the variable name. See 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.

Velocity control

In order to achieve an acceptable traverse rate in the case of very slight motions per block, an anticipatory evaluation over several blocks (Look Ahead) can be specified.

WinSCP

WinSCP is a freely available open source program for Windows for the transfer of files.

Working area

Three-dimensional zone into which the tool tip can be moved on account of the physical design of the machine tool. See Protection zone.
**Working area limitation**

With the aid of the working area limitation, the traversing range of the axes can be further restricted in addition to the limit switches. One value pair per axis may be used to describe the protected working area.

**Working memory**

The working memory is a RAM in the → CPU that the processor accesses when processing the application program.

**Workpiece**

Part to be made/machined by the machine tool.

**Workpiece contour**

Set contour of the → workpiece to be created or machined.

**Workpiece coordinate system**

The workpiece coordinate system has its starting point in the → workpiece zero-point. In machining operations programmed in the workpiece coordinate system, the dimensions and directions refer to this system.

**Workpiece zero**

The workpiece zero is the starting point for the → workpiece coordinate system. It is defined in terms of distances to the → machine zero.

**Zero offset**

Specifies a new reference point for a coordinate system through reference to an existing zero point and a → frame.

1. **Settable**
   
   A configurable number of settable zero offsets are available for each CNC axis. The offsets - which are selected by means of G functions - take effect alternatively.

2. **External**
   
   In addition to all the offsets which define the position of the workpiece zero, an external zero offset can be overridden by means of the handwheel (DRF offset) or from the PLC.

3. **Programmable**
   
   Zero offsets can be programmed for all path and positioning axes using the TRANS statement.
Index

$ $AA_COUP_ACT, 201, 202 $AA_COUP_CORR, 344 $AA_COUP_CORR_DIST, 346 $AA_ESR_ENABLE, 390, 401 $AA_IN_SYNC, 239 $AA_JERK_COUNT, 712 $AA_JERK_TIME, 712 $AA_JERK_TOT, 712 $AA_LEAD_P, 239 $AA_LEAD_P_TURN, 239 $AA_LEAD_SP, 239 $AA_LEAD_SV, 239 $AA_LEAD_V, 239 $AA_MASL_STAT, 523 $AA_SYNC, 239, 314 $AA_SYNCDIFF, 313 $AA_TRAVEL_COUNT, 712 $AA_TRAVEL_COUNT_HS, 712 $AA_TRAVEL_DIST, 712 $AA_TRAVEL_DIST_HS, 712 $AA_TRAVEL_TIME, 712 $AA_TRAVEL_TIME_HS, 712 $AN_ESR_TRIGGER, 391 $SP_PROG_EVENT, 523 $SP_SEARCH_MASLC, 523 $SP_SEARCH_MASLD, 523 $VA_STOPSI, 397 $VA_SYNCDIFF, 313

2 2-axis swivel head, 79

3 3-axis and 4-axis transformation, 25 3-axis and 4-axis transformations, 45 3-axis kinematics, 77 3-axis to 5-axis transformation Call and application, 56 3-axis transformations, 59

4 4-axis kinematics, 77 4-axis Transformation, 25 4-axis transformations, 59

5 5-axis kinematics, 77 5-axis transformation Configuration of a machine, 33 Geometry of the machine, 34 Machine types, 31 Singular positions, 42 Tool orientation, 37

7 7-axis transformation, 70 Example, 138 Kinematics, 71

A aaJerkCount, 712 aaJerkTime, 712 aaJerkTotal, 712 aaTravelCount, 712 aaTravelCountHS, 712 aaTravelDist, 712 aaTravelDistHS, 712 aaTravelTime, 712 aaTravelTimeHS, 712 Acceleration Channel-specific, 660 Function-specific, 660 Acceleration mode, 353 Acceleration warning threshold, 724 Access rights, 671 ACCLIMA[FA], 352 Activate coupling, 737 Activate master-slave coupling, 732 Activate mirroring, 737 Activate monitor, 736 activate PTP traversal, 717

Special functions Function Manual, 07/2012, 6FC5397-2BP40-3BA0
Coupling module, 277
  Creating, 283
  Delete, 284
Coupling reference, 295
Coupling rule, 277, 292
Coupling type, 238
Coupling types, 331
CP-BASIC, 274
CPBC, 297
CP-COMFORT, 274
CPDEF, 283
CPDEF+CPLA, 285
CPDEL, 284
CPDEL+CPLA, 287
CP-EXPERT, 274
CPFMOF, 304
CPFMON, 303
CPFMONSON, 301
CPFPPOS, 299
CPFRRS, 296
CPLDEF, 285
CPLDEL, 286
CPLDEN, 293
CPLINSC, 311
CLLINTR, 310
CPLNUM, 293
CPLOF, 291
CPLON, 290
CPLOUTSC, 312
CPLOUTTR, 311
CPLPOS, 300
CPLSETVAL, 295
CPMALARM, 328
CPMBRAKE, 322
CPMPRT, 308
CPMRESET, 306
CPMSTART, 307
CPMVDI, 323
CPOF, 289
CPOF+CPFPOS, 305
CPON, 288
CPRECOF, 189
CPRECON, 189
CPRES, 336
CPSETTYPE, 331
CP-STATIC, 274
CPSYNCOP, 315
CPSYNCOP2, 318
CPSYNCOV, 315
CPSYNFLIP, 315
CPSYNFLIP2, 318
CPSYNFLIV, 316
CTOL, 188
Curve segment, 206
Curve table, 294
Curve tables, 205
  Activation, 222
  Behavior in operating modes, 224
  Deactivation, 223
  Delete, 208
  Insufficient memory, 208
  Interface signals, 224
  Memory optimization, 207
  Memory type, 207
  Multiple use, 223
  Overwrite, 208
  Programming, 211
  Standard memory type, 210
  Tool radius compensation, 206
  Transformations, 229
CUT3DC, 704
CUT3DF, 704
CUT3DF, 704
CUT3DFS, 704
Cutter shapes, 694

D

DB 31, ...
  DBX1.4, 159
  DBX101.3, 720
  DBX101.5, 158, 722
  DBX2.1, 159
  DBX24.3, 736
  DBX24.5, 732
  DBX29.5, 159, 175
  DBX4.3, 248, 249
  DBX66.0, 736
  DBX96.7, 733
  DBX97.0, 737
  DBX97.1, 737
  DBX97.2, 737
  DBX97.3, 737
DB10
  DBB0, 494
  DBB146, 494
DB11, ...
  DBX5.2, 156
DB11, ...
  DBX7.6, 614
DB21, ...
  DBB232, 717, 735
  DBX0.1, 738
  DBX0.2, 738
DBX0.3, 200
DBX1.4, 729
DBX1.5, 729
DBX29.4, 717, 734
DBX317.6, 735
DBX318.2, 124
DBX318.3, 124
DBX32.1, 739
DBX32.2, 739
DBX33.6, 717, 734
DBX37.3, 730
DBX37.4, 730, 731
DBX37.5, 730, 731
DBX6.7, 729
DBX7.3, 493
DBX7.4, 493

DB21, ...
DBB4, 729
DBX0.1, 605, 614, 618, 622
DBX0.2, 606, 614, 623
DBX1.0, 160
DBX1.4, 731
DBX317.6, 718
DBX318.0, 625
DBX318.2, 718
DBX318.3, 718
DBX32.1, 623
DBX32.2, 614
DBX32.6, 606
DBX33.0, 156
DBX33.4, 625
DBX33.6, 56, 579
DBX37.4, 730
DBX37.5, 730
DBX7.7, 614

DB31, ...
DBB101.4, 175, 176, 177
DBB101.6, 175, 176, 177
DBB101.7, 175, 176, 177
DBB102.2, 178
DBB102.3, 178
DBB102.4, 178
DBB102.5, 178
DBB102.6, 178
DBB102.7, 178
DBX 99.1, 251
DBX0.0, 200
DBX0.1, 200
DBX0.2, 200
DBX0.3, 200
DBX0.4, 200
DBX0.5, 200
DBX0.6, 200
DBX0.7, 200
DBX1.3, 200
DBX1.4, 169, 201
DBX1.5, 159, 169, 201, 224, 242
DBX1.6, 159, 169, 201, 224, 242
DBX101.2, 149, 168, 720
DBX101.3, 149, 168
DBX101.4, 156, 168, 721
DBX101.5, 150, 156, 168, 170
DBX101.6, 168, 722
DBX101.7, 168, 723
DBX103.0, 345
DBX103.4, 319
DBX103.5, 319
DBX103.6, 200, 201
DBX24.2, 736
DBX24.5, 727
DBX24.7, 527, 732
DBX26.4, 251, 255
DBX29.4, 157, 168, 175, 177, 719
DBX29.5, 157, 168, 719
DBX31.6, 343
DBX31.7, 346
DBX4.0, 200
DBX4.1, 200
DBX4.2, 200
DBX4.3, 200
DBX4.7/4.8, 153
DBX60.4, 169
DBX60.5, 169
DBX61.5, 527
DBX61.6, 527
DBX61.7, 527
DBX64.6, 169
DBX64.7, 169
DBX95.1, 726
DBX95.2, 726
DBX95.3, 726
DBX96.2, 733
DBX96.3, 733
DBX96.4, 733
DBX96.5, 727
DBX96.6, 527
DBX97.1, 592
DBX97.2, 592
DBX98.0, 250, 314, 341
DBX98.1, 250, 314, 341
DBX98.4, 251
DBX98.5, 252
DBX98.6, 252
DBX98.7, 384

Index
Index

H

Hardware and software limit switches
  Master-slave, speed coupling, 522
Hardware requirements, 277

I

I/Os, 635, 642
  Identification of axis sequence, 33
  Independent coupled motion axis, 195
INI program, 619
  Insertion depth (ISD), 684
Integrated outputs, 635
Interface signals
  ESR reaction is triggered, 726
  Generator operation – minimum speed fallen below, 726
  VDC link lower than alarm threshold, 726
Interface versions, 438
  Intermediate block, 417
  Intermediate blocks, 705
  Intermediate orientation, 118
  Interpolation, 242
  Interpolation of the angle of rotation, 114
  Interpolation of the rotation vector, 115
  Interpolation type and selection, 109
  Interruption point, 604
  Intersection procedure for 3D compensation, 689
  ISD, 684

K

Keywords, 279
  Kinematic transformation, 31
  Kinematics
    Swiveling linear axis, 26
    Kinematics of machines, 31

L

Large circle interpolation, 73
  Laser cutting
    Clearance control, 454
Leading axes
  Defining, 285
  Delete, 286
Leading axis
  Switch off, 291
  Switch on, 290
Limit angle for the fifth axis, 43
  Linear axis, 656
Logging, 243

M

Machine kinematics, 58
  Machine types, 31, 33, 62
    5-axis transformation, 58
    6-axis transformation, 66
Main program, 618
Master value
  -coupling, 232
  -object, 232
  Scaling, 310
  Translation, 310
Master value coupling
  Behavior in operating modes, 240
  Interface signals, 242
  Switch off, 238
  Switch on, 236
Master/slave coarse, 733
Master/slave compensatory cont. active, 733
Master/slave fine, 733
Master-slave coupling active, 733
Maximum axis acceleration, 352
Maximum axis velocity, 352
Maximum switching frequency, 639
MD10000, 23, 661
MD10002, 656, 661
MD10089, 397
MD10300, 467
MD10350, 467
MD10360, 642
MD10362, 467, 470
MD10366, 467, 470, 471
MD10620, 37, 573
MD10640, 37, 573
MD10674, 108
MD10700, 668, 670, 672
MD10712, 469
MD11410, 197, 328
MD11410 $MN_SUPPRESS_ALARM_MASK
  Bit31, 249
MD11450, 523
MD11602, 401, 523, 613
MD11604, 523, 613
MD11660 $MN_NUM_EG, 247
MD12701, 661
MD12750, 661
MD18114, 69
MD18170, 670
MD18180, 670
Index

MD18242, 671
MD18351, 612, 670
MD18400, 205, 209
MD18402, 205, 209
MD18403, 205, 209
MD18404, 205, 209
MD18406, 206, 209
MD18408, 206, 209
MD18409, 206, 209
MD18410, 206, 209
MD18450, 277
MD18452, 277
MD18860, 713
MD19610, 695
MD20050, 615, 644
MD20060, 23, 676
MD20080, 23, 676
MD20105, 401, 523
MD20110, 55, 189, 198, 235, 237, 240, 241, 579
MD20112, 189, 198, 235, 237, 240, 241
MD20115, 401
MD20140, 579
MD20150, 40, 574
MD20178, 90
MD20470, 187
MD20482, 86
MD20610, 123
MD20900, 210, 216
MD20905, 210, 213, 216, 225
MD21050, 185
MD21070, 185
MD21084, 697
MD21094, 92, 95, 97
MD21100, 36, 103, 113, 574
MD21102, 103
MD21104, 104
MD21106, 74, 75
MD21108, 44
MD21120, 76, 100
MD21130, 76
MD21150, 102
MD21155, 102
MD21160, 102
MD21165, 102
MD21180, 84, 138
MD21186, 106
MD21190, 123
MD21194, 123
MD21380, 367, 401
MD21380 $MC_DELAY_TIME1, 371
MD21380 $MC_ESR_DELAY_TIME1, 369, 371
MD21380 $MC_DELAY_TIME2, 369, 371
MD21381, 401
MD21381 $MC_ESR_DELAY_TIME2, 369, 371
MD22410, 354
MD24100, 33, 36, 62, 63, 64, 65, 66, 67, 536, 579, 582, 585
MD24110, 34, 62, 67, 101, 536, 585
MD24120, 537, 585, 615
MD24200, 62, 65, 67
MD24210, 62
MD24410, 101
MD24432, 101
MD24460, 579
MD24462, 101
MD24480, 33, 36
MD24482, 34
MD24500, 34, 64, 65, 67
MD24510, 34, 63
MD24520, 36, 63
MD24530, 42
MD24540, 42, 78
MD24550, 64, 65, 67
MD24558, 65, 67
MD24560, 34, 64, 65, 67
MD24561, 68
MD24567, 68
MD2457, 61
MD24570, 59, 63
MD24572, 59, 63
MD24573, 67
MD24574, 61, 62, 67
MD24576, 67
MD2458, 100
MD24582, 62
MD24585, 100, 102
MD24590, 106
MD24620, 36, 63
MD24630, 42
MD24640, 42, 78
MD24670, 61
MD24674, 62
MD24680, 100
MD24682, 62, 65
MD24690, 106
MD28010, 671
MD28020, 671
MD28040, 671
MD28090, 466, 593, 609, 642
MD28100, 466, 593, 609, 642
MD28105, 611
MD28590, 90
MD30110, 408
MD30130, 233, 236, 238
MD30140, 238

Special functions

Function Manual, 07/2012, 6FC5397-2BP40-3BA0

781
Index

MD30132, 233, 495
MD30230, 409
MD30450, 171
MD30455, 343
MD30552, 590
MD32000, 352, 468
MD32000 $MA_MAX_AX_VELO, 252
MD32100, 162
MD32200, 163, 468
MD32230, 468
MD32300, 352, 594
MD32300 $MA_MAX_AX_ACCEL, 252
MD32300 $MA_MAX_AX_ACCEL, 252
MD32400, 163
MD32410, 163, 458
MD32415, 189
MD32420, 163
MD32430, 163
MD32434, 599
MD32610, 163, 458
MD32620, 163
MD32650, 163
MD32800, 163
MD32810, 163, 164
MD32900, 164
MD32910, 164
MD33000, 163
MD33060, 713
MD33100, 85
MD34040, 154
MD34070, 154
MD34080, 158, 164, 176
MD34090, 158, 164, 176
MD34100, 152, 154, 156, 158, 160, 177
MD34110, 160
MD34330, 154, 158, 160
MD36000, 470
MD36010, 470
MD36012, 159
MD36020, 470
MD36030, 159
MD36040, 470
MD36060, 470
MD36100, 494
MD36110, 494
MD36120, 494
MD36130, 494
MD36500, 160, 185
MD36600, 340
MD36954, 397
MD36955, 397
MD37014, 520
MD37100, 164, 173, 494, 719
MD37110, 149, 154, 165, 173, 178, 179
MD37120, 149, 162, 173, 178, 179, 720
MD37130, 149, 162, 173, 179, 720
MD37135, 150
MD37140, 152, 157, 170, 171
MD37150, 150, 158
MD37160, 224, 235
MD37200, 314
MD37200 $MA_COUPLE_POS_TOL_COARSE, 250, 262
MD37200 $MN_COUPLE_POS_TOL_COARSE, 250
MD37202, 317
MD37210, 314
MD37210 $MA_COUPLE_POS_TOL_FINE, 250
MD37210 $MA_COUPLE_POS_TOL_FINE, 250, 262
MD37212, 317
MD37220, 314
MD37230, 314
MD37262, 526
MD37268, 519
MD37400, 417, 424
MD37402, 423
MD37500, 401
MD37500 $MA_ESR_REACTION, 370
MD37550 $MA_EQ_VEL_WARNING, 252
MD37560 $MA_EQ_ACC_TOL, 252
MD43108, 268
MD60900+i, 609
MD60940, 466
MD60948, 542, 644
MD60972, 655, 661
MD61516, 656, 661
MD61517, 656, 661
MD61518, 657
MD61519, 657
MD61532, 658
MD62500, 469
MD62502, 470
MD62504, 465, 470, 475
MD62505, 471, 481, 482, 730
MD62506, 471, 481, 482, 730
MD62508, 494
MD62510, 479, 486
MD62511, 472, 479, 486
MD62512, 479, 486
MD62513, 472, 479, 486
MD62516, 729
MD62520, 731
MD62521, 731
MD62522, 485
MD62523, 483

Special functions
Index

MD62524, 474
MD62528, 468
MD62560, 643
MD62571, 611
MD62572, 611
MD62574, 613
MD62575, 626
MD62580, 609
MD62600, 532, 550, 558, 564, 568, 585
MD62601, 548, 550, 558, 565
MD62602, 568, 569, 570, 571, 572, 585
MD62603, 538, 540, 541, 550, 558, 564, 586
MD62604, 539, 541, 550, 558, 564, 586
MD62605, 545, 550, 558, 564, 585
MD62606, 539, 558, 565, 586
MD62607, 539, 551, 559, 565, 586
MD62608, 539, 559, 565
MD62609, 539, 559, 565
MD62610, 539, 551, 554, 565
MD62611, 539, 551, 558, 564, 565
MD62612, 538, 551, 559, 565
MD62613, 538, 551, 559, 565
MD62614, 542, 565, 586
MD62615, 542, 586
MD62616, 539, 542, 565, 586
MD62617, 547, 548, 550, 559, 565, 586
MD62618, 547, 550, 559, 565, 586
MD62620, 545, 546, 550, 558, 564, 586
MD62629, 549
MD62630, 549
MD62631, 549
MD62632, 549
MD62636, 577
MD62637, 577
MD63514, 660
MD63540, 591
MD63541, 593
MD63542, 594
MD63543, 594
MD63544, 594
MD63545, 594
MD65520, 469
MD65530, 469
MELDW

Bit9, 392
Memory configuration
  Curve table, 205
  Generic coupling, 277
  Memory requirement, 671
  Minimum switching position clearance, 639
  Modulo display, 107
  Modulo leading axis, 223

Modulo rotary axis, 656

Monitoring
  Synchronous operation, 313

N
NC block compressor, 85
NC-controlled extended stop, 369
Number of active G function of G function group 25, 735
Number of active G function of G function group 25 (tool orientation reference), 717

O
Offset, 71
Offset after point of activation, 737
Offset vector, 657
Online tool length offset, 122
Opening angle, 118
Opening angle of the cone, 117
Operating modes
  JOG, 74
Option, 655
ORIC, 686, 701
ORICONCCW, 117, 120
ORICONCW, 117, 120
ORICONIO, 118, 120
ORICONTO, 119
ORICURVE, 119
ORID, 688
Orientation, 73, 81
Orientation axes, 28, 100
  Modulo display, 107
Orientation direction, 113
Orientation direction and rotation, 113
Orientation in TCS and MCS, 38
Orientation movements with axis limits, 83
Orientation of the machine axes, 658
Orientation programming, 37
Orientation relative to the path, 92
Orientation transformation, 31
  Programming, 103
Orientation transformation and orientable tool holders, 107
Orientation vector, 109
Orientation with axis interpolation, 73
ORIMKS, 38
ORIPLANE, 117, 120
ORISOF, 90
ORISON, 90
Index

ORIWKS, 38
ORIWKS, 38
Outside corners/inside corners, 685

P

p0888, 386, 387, 389
p0890, 399
p0891, 386, 387
p0892, 386, 388
p0893, 388
p1240, 389
p1248, 389
p1414, 458
p2161, 390
p9554, 397, 398
p9555, 397
p9580, 397, 398
p9697, 397, 398
PI controller, 353
Plane, 609
Pole, 42
Polynomial interpolation, 110
  Orientation vector, 108
Polynomial of the 5th degree, 114
Polynomials for 2 angles, 110
POLYPATH, 109
Position of the orientation coordinate system
  Generic transformation with 6 axes, 69
Program continuation point, 604
Program handling, 670
Programming
  Cartesian position, 72
  Orientation, 72
  Polynomial, 108
Protection window, 657
PTP traversal active, 718, 735

R

r9721, 399
r9723, 399
Rapid stop, 340
Reaction to Stop, 235
RESET response, 616
Restrictions, 216
Restrictions for kinematics and interpolation, 77
RESU ASUB, 622
RESU main program, 618
RESU working plane, 609
RESU-specific part programs
  Overview, 617
  Retrace mode active, 739
  Retrace support active, 739
  Retraceable contour range, 604
  Retract, 365, 366
  Retraction direction, 656
  Reverse/Forward, 738
  Rotary axis, 656
  Rotary axis of the cone, 117
  Rotation of the orientation vector, 112, 114
  RPY angles, 38
  Runtime optimization, 669

S

Scaling
  of a lead value, 310
SD42450, 188
SD42460, 188
SD42475, 86
SD42476, 86
SD42477, 86
SD42650, 74
SD42660, 75, 76
SD42670, 94
SD42672, 95
SD42678, 90
SD42680, 90
SD42970, 123
SD42974, 82
SD43100, 238
SD43102, 233
SD43104, 233
SD43106, 233
SD43108, 233
Series machine startup, 613
Shutting down, 365
  Extended, 366
Siemens compile cycles, 434
Simulated master value, 233
Singular positions, 42
Singularities, 78
Slot, 660
Smoothing
  Of the orientation characteristic, 89
Special functions, 434
Spindle
  in master value coupling, 236
Start, 674
Start gantry synchronization, 719
Start orientation, 117
Start retrace support, 738

Special functions

Function Manual, 07/2012, 6FC5397-2BP40-3BA0
Starting value, 217
Status of coupling, 202, 242
Status of setpoint exchange, 727
SW version, 440
Switch on collision protection, 736
Switching criteria, 636
Switching instants, 639
Switching position, 636
Switching position offset, 640, 641
Switch-over to axis interpolation, 84
swiveled linear axis
Pole, 48
Swiveling linear axis, 26
Synchronism
  Correction, 345
Synchronization, 353
    Following axis, 244
Synchronization difference, 249
    - scanning, 252
Synchronization mode, 301
Synchronous operation
  Deviation, 341
    - difference, 313
    - monitoring, 313
Synchronous operation monitoring
  stage 2, 317
Synchronous position
  Following axis, 299
    Leading axis, 300
Syntax check, 675
System variables, 281
  for orientation, 24

T
Tangential control, 417
  Applications, 418
  Examples, 525
TANGON, 423
Technology functions, 433
  Activate, 441
Temporary curve table, 208
TOFF active, 718
TOFF motion active, 718
Tool
  Orientation, smoothing, 89
Tool offsets, 633
Tool orientation, 28
Tool orientation using orientation vectors, 41, 77
Tool tip at a fixed point in space, 38
Toolholder, with orientation capability
  Programming, 77

Torque compensatory controller active, 733
TRAANG, 229
TRAILOF, 200
TRAILON, 199
Transformation, 632
Transformation active, 56, 717, 734
Transformation inactive, 56
Transformation types, 36
Translation
  of a lead value, 310
Two-axis swivel head, 31

U
Universal milling head, 27

V
V2 preprocessing
  Brief description, 667
Velocity threshold value, 637
Velocity warning threshold, 724
VELOLIMA[FA], 352
Versions, 274
Virtual leading axis, 233
Vocabulary, 676

W
WAITC, 298
