SIEMENS

SINUMERIK

SINUMERIK MC
Axes and spindles

Function Manual

Preface

Fundamental safety instructions 1

G2: Velocities, setpoint / actual value systems, closed-loop control 2

F1: Travel to fixed stop 3

P1: Transverse axes 4

V1: Feedrates 5

P2: Positioning axes 6

R2: Rotary axes 7

T1: Indexing axes 8

G1: Gantry axes 9

R1: Referencing 10

S9: Setpoint switchover 11

B2: Acceleration and jerk 12

N3: Software cams, position switching signals 13

M3: Coupled axes 14

P5: Oscillation 15

R3: Extended stop and retract 16

H1: Manual traversing 17

S1: Spindles 18

S3: Synchronous spindle 19

Appendix A

Valid for:

Control system
SINUMERIK MC

Software
CNC software version 1.12

06/2019
A5E47437747B AA

Legal information

Warning notice system

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

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

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

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

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

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

Qualified Personnel

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

Proper use of Siemens products

Note the following:

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

Trademarks

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

Disclaimer of Liability

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

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
GERMANY

Document order number: A5E47437747B AA
℗ 06/2019 Subject to change

Copyright © Siemens AG 2019.
All rights reserved

Preface

SINUMERIK documentation

The SINUMERIK documentation is organized into the following categories:

- General documentation/catalogs
- User documentation
- Manufacturer/service documentation

Additional information

You can find information on the following topics at the following address (https://support.industry.siemens.com/cs/de/en/view/108464614):

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

If you have any questions regarding the technical documentation (e.g. suggestions, corrections), please send an e-mail to the following address (mailto:docu.motioncontrol@siemens.com).

mySupport/Documentation

At the following address (https://support.industry.siemens.com/My/ww/en/documentation), you can find information on how to create your own individual documentation based on Siemens' content, and adapt it for your own machine documentation.

Training

At the following address (http://www.siemens.com/sitrain), you can find information about SITRAIN (Siemens training on products, systems and solutions for automation and drives).

FAQs

You can find Frequently Asked Questions in the Service&Support pages under Product Support (https://support.industry.siemens.com/cs/de/en/ps/faq).

SINUMERIK

You can find information about SINUMERIK at the following address (http://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.

Note regarding the General Data Protection Regulation

Siemens observes standard data protection principles, in particular the principle of privacy by design. That means that

this product does not process / store any personal data, only technical functional data (e.g. time stamps). If a user links this data with other data (e.g. a shift schedule) or stores personal data on the same storage medium (e.g. hard drive) and thus establishes a link to a person or persons, then the user is responsible for ensuring compliance with the relevant data protection regulations.

Technical Support

Country-specific telephone numbers for technical support are provided in the Internet at the following address (https://support.industry.siemens.com/sc/ww/en/sc/2090) in the "Contact" area.

Information on the structure and contents

Structure

This Function Manual is structured as follows:

- 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 machine data description
- For NC/PLC interface signals:
  NC Variables and Interface Signals List Manual
- 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 signals	... 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)
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)

Quantity structure

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

- Mode groups (DB11)
- Channels (DB21, etc.)
- Axes/spindles (DB31, etc.)

Data types

The control provides the following data types that can be used for programming in part programs:

Type	Meaning	Range of values
INT	Signed integers	-2.147.483.648 ... +2.147.483.647
REAL	Numbers with decimal point	≈ ±5.0*10^{-324} … ≈ ±1.7*10^{+308}
BOOL	Boolean values	TRUE (≠0) , FALSE (0)
CHAR	ASCII characters and bytes	0 ... 255 or -128 ... 127
STRING	Character string, null-terminated	Maximum of 400 characters + /0 (no special characters)
AXIS	Axis names	All axis names available in the control system
FRAME	Geometrical parameters for moving, rotating, scaling, and mirroring	---

Arrays

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

Example: `DEF INT ARRAY[2, 3, 4]`

Number systems

The following number systems are available:

- Decimal: `DEF INT number = 1234` or `DEF REAL number = 1234.56`
- Hexadecimal: `DEF INT number = 'H123ABC'`
- Binary: `DEF INT number = 'B10001010010'`

Querying REAL variables

We recommend that querying REAL or DOUBLE variables in NC programs and synchronized actions is programmed as limit value evaluation.

Example: Querying the actual value of an axis for a specific value

```
Program code                                                                    Comment
DEF REAL AXPOS = 123.456
IF ($VA_IM[<axis>] - 1ex-6) <= AXPOS <= ($VA_IM[<axis>] + 1ex-6)                ; actual position
   ...                                                                          == AXPOS
```

Program code	Comment
ELSE	
...	<> AXPOS
ENDIF	

Table of contents

Preface **3**

1 Fundamental safety instructions **27**

1.1 General safety instructions 27

1.2 Warranty and liability for application examples 27

1.3 Industrial security 28

2 G2: Velocities, setpoint / actual value systems, closed-loop control **31**

2.1 Brief description 31

2.2 Velocities, traversing ranges, accuracies 31
2.2.1 Velocities 31
2.2.2 Traversing ranges 33
2.2.3 Positioning accuracy of the control system 34
2.2.4 Input/display resolution, computational resolution 34
2.2.5 Scaling of physical quantities of machine and setting data 36

2.3 System of units, metric/inch 39
2.3.1 Function 39
2.3.1.1 Parameterized and programmed system of units 39
2.3.1.2 Extended system of units functionality 40
2.3.1.3 Extended system of units functionality 40
2.3.2 Commissioning 41
2.3.3 Programming 45
2.3.3.1 Switching over the system of units (G70/G71/G700/G710) 45

2.4 Setpoint/actual-value system 49
2.4.1 General information 49
2.4.2 Setpoint and encoder assignment 51
2.4.3 Adapting the motor/load ratios 55
2.4.4 Speed setpoint output 57
2.4.5 Machine data of the actual value system 59
2.4.6 Actual-value resolution 60
2.4.6.1 Machine data of the actual value resolution 60
2.4.6.2 Example: Linear axis with linear scale 63
2.4.6.3 Example: Linear axis with rotary encoder on motor 64
2.4.6.4 Example: Linear axis with rotary encoder on the machine 65
2.4.6.5 Example: Rotary axis with rotary encoder on motor 66
2.4.6.6 Example: Rotary axis with rotary encoder on the machine 67
2.4.6.7 Example: Intermediate gear with encoder on the tool 68

2.5 Closed-loop control 69
2.5.1 General information 69
2.5.2 Switchable position setpoint filter circuits 72
2.5.3 Commissioning 74
2.5.4 Parameter sets of the position controller 75
2.5.5 Relevant system data 76

2.6 Optimization of the control 77
2.6.1 Position controller, position setpoint filter: Balancing filter 77
2.6.2 Position controller, position setpoint filter: Phase filter 81
2.6.3 Position controller: injection of positional deviation 83
2.6.4 Position control with proportional-plus-integral-action controller 84
2.7 Data lists 87
2.7.1 Machine data 87
2.7.1.1 Displaying machine data 87
2.7.1.2 NC-specific machine data 87
2.7.1.3 Channelspecific machine data 87
2.7.1.4 Axis/spindlespecific machine data 88
3 F1: Travel to fixed stop 91
3.1 Brief description 91
3.2 Detailed description 92
3.2.1 Programming 92
3.2.2 Functional sequence 94
3.2.2.1 Selection 94
3.2.2.2 Fixed stop is reached 95
3.2.2.3 Fixed stop is not reached 97
3.2.2.4 Deselection 98
3.2.3 Behavior during block search 101
3.2.4 Behavior for reset and function abort 104
3.2.5 Behavior with regard to other functions 104
3.2.6 Setting data 105
3.2.7 System variables 107
3.2.8 Alarms 108
3.2.9 Travel with limited torque/force FOC 109
3.3 Examples 112
3.4 Data lists 113
3.4.1 Machine data 113
3.4.1.1 Axis/spindlespecific machine data 113
3.4.2 Setting data 114
3.4.2.1 Axis/spindle-specific setting data 114
4 P1: Transverse axes 115
4.1 Function 115
4.2 Parameterization 118
4.3 Programming 121
4.4 Supplementary conditions 122
4.5 Examples 123
4.6 Data lists 124
4.6.1 Machine data 124
4.6.1.1 Channelspecific machine data 124
4.6.1.2 Axis/spindlespecific machine data 124

5 V1: Feedrates 125

5.1 Brief description 125

5.2 Path feedrate F 126
5.2.1 Feedrate type G93, G94, G95 128
5.2.2 Type of feedrate G96, G961, G962, G97, G971 131
5.2.3 Feedrate for thread cutting (G33, G34, G35, G335, G336) 135
5.2.3.1 Feedrate with G33 135
5.2.3.2 Linear increasing/decreasing thread pitch change with G34 and G35 136
5.2.3.3 Acceleration behavior of the axis for G33, G34 and G35 138
5.2.3.4 Programmed run-in and run-out path for G33, G34 and G35 (DITS, DITE) 140
5.2.3.5 Fast retraction during thread cutting 142
5.2.3.6 Convex thread (G335, G336) 146
5.2.4 Feedrate for tapping without compensating chuck (G331, G332) 151
5.2.5 Feedrate for tapping with compensating chuck (G63) 153
5.2.6 FGROUP and FGREF 153

5.3 Feedrate for positioning axes (FA) 156

5.4 Feedrate control 157
5.4.1 Feedrate disable and feedrate/spindle stop 157
5.4.2 Feedrate override via machine control panel 159
5.4.3 Programmable feedrate override 163
5.4.4 Dry run feedrate 163
5.4.5 Multiple feedrate values in one block 165
5.4.6 Fixed feedrate values 170
5.4.7 Programmable feedrate characteristics 172
5.4.8 Feedrate for chamfer/rounding FRC, FRCM 173
5.4.9 Non-modal feedrate FB 175
5.4.10 Influencing the single axis dynamic response 176

5.5 Boundary conditions 181

5.6 Data lists 182
5.6.1 Machine data 182
5.6.1.1 NC-specific machine data 182
5.6.1.2 Channel-specific machine data 182
5.6.1.3 Axis/Spindle-specific machine data 183
5.6.2 Setting data 183
5.6.2.1 Channel-specific setting data 183
5.6.2.2 Axis/spindle-specific setting data 184

6 P2: Positioning axes 185

6.1 Brief description 185

6.2 Own channel, positioning axis or concurrent positioning axis 187
6.2.1 Own channel 188
6.2.2 Positioning axis 188
6.2.3 Concurrent positioning axis 191

6.3 Motion behavior and interpolation functions 192
6.3.1 Path interpolator and axis interpolator 192
6.3.2 Autonomous singleaxis operations 193
6.3.3 Autonomous single-axis functions with NC-controlled ESR 197

6.4 Positioning axis dynamic response 199
6.5 Programming 201
6.5.1 General 201
6.5.2 Revolutional feed rate in external programming 203
6.6 Block change 204
6.6.1 Settable block change time 206
6.6.2 End of motion criterion with block search 211
6.7 Control by the PLC 212
6.7.1 Starting concurrent positioning axes from the PLC 214
6.7.2 PLC-controlled axes 214
6.7.3 Control response of PLC-controlled axes 216
6.8 Response with special functions 217
6.8.1 Dry run (DRY RUN) 217
6.8.2 Single block 217
6.9 Examples 217
6.9.1 Motion behavior and interpolation functions 217
6.10 Data lists 218
6.10.1 Machine data 218
6.10.1.1 NC-specific machine data 218
6.10.1.2 Channelspecific machine data 219
6.10.1.3 Axis/spindlespecific machine data 219
6.10.2 Setting data 219
6.10.2.1 Axis/spindle-specific setting data 219
7 R2: Rotary axes 221
7.1 Brief description 221
7.2 Modulo 360 degrees 223
7.3 Programming of rotary axes 228
7.3.1 General information 228
7.3.2 Rotary axis with modulo conversion (continuously-turning rotary axis). 229
7.3.3 Rotary axis without modulo conversion 232
7.3.4 Other programming features relating to rotary axes 234
7.4 Activating rotary axes 234
7.5 Special features of rotary axes 236
7.6 Examples 237
7.7 Data lists 237
7.7.1 Machine data 237
7.7.1.1 General machine data 237
7.7.1.2 Axis/spindlespecific machine data 237
7.7.2 Setting data 238
7.7.2.1 General setting data 238
7.7.2.2 Axis/spindle-specific setting data 238
8 T1: Indexing axes 239
8.1 Brief description 239

8.2 Detailed description....239
8.2.1 Traversing of indexing axes in the AUTOMATIC mode....239
8.2.2 Traversing of indexing axes in the JOG mode....239
8.2.3 Traversing of indexing axes by PLC....241
8.3 Commissioning....242
8.3.1 Machine data for indexing axes....242
8.3.2 Machine data for equidistant indexing intervals....245
8.3.2.1 Features....245
8.3.2.2 Hirth axis....247
8.3.3 Axis-specific system variables....247
8.3.4 NC/PLC interface signals....248
8.4 Programming....248
8.5 Supplementary conditions....250
8.6 Examples....253
8.6.1 Example 1: Rotary axis as indexing axis....253
8.6.2 Example 2: Indexing axis as linear axis....254
8.6.3 Example 3: Rotary modulo axis as equidistant indexing axis....254
8.6.4 Example 4: Rotary axes as equidistant indexing axis with restricted traversing range....255
8.6.5 Example 5: Linear axis as equidistant indexing axis....255
8.6.6 Example 6: "Hirth axis"....256
8.7 Data lists....257
8.7.1 Machine data....257
8.7.1.1 General machine data....257
8.7.1.2 Axis/spindlespecific machine data....257
8.7.2 Setting data....257
8.7.2.1 General setting data....257
8.7.3 System variables....258
9 G1: Gantry axes....259
9.1 Brief description....259
9.2 "Gantry axes" function....260
9.2.1 Definition of a gantry grouping....260
9.2.2 Monitoring the synchronism difference....261
9.2.3 Extended monitoring of the synchronism difference....262
9.2.4 Referencing and synchronization of gantry axes....262
9.2.5 Control dynamics....263
9.2.6 Opening the gantry grouping....263
9.3 Referencing and synchronization of gantry axes....264
9.3.1 Introduction....264
9.3.2 Automatic synchronization....270
9.3.3 Points to note....271
9.4 Start-up of gantry axes....274
9.5 Parameter assignment: Response to faults....278
9.6 PLC interface signals for gantry axes....279
9.7 Miscellaneous points regarding gantry axes....281
9.8 Examples....283

9.8.1 Creating a gantry grouping ... 283
9.8.2 Setting the NC-PLC interface ... 285
9.8.3 Commencing start-up ... 285
9.8.4 Setting warning and trip limits ... 287
9.9 Data lists ... 288
9.9.1 Machine data ... 288
9.9.1.1 Axis/spindlespecific machine data ... 288
10 R1: Referencing ... 291
10.1 Brief Description ... 291
10.2 Axisspecific referencing ... 292
10.3 Channelspecific referencing ... 294
10.4 Reference point appraoch from part program (G74) ... 295
10.5 Referencing with incremental measurement systems ... 296
10.5.1 Hardware signals ... 296
10.5.2 Zero mark selection ... 298
10.5.3 Time sequence ... 299
10.5.4 Phase 1: Traversing to the reference cam ... 300
10.5.5 Phase 2: Synchronization with the zero mark ... 302
10.5.6 Phase 3: Traversing to the reference point ... 307
10.6 Referencing with distance-coded reference marks ... 309
10.6.1 General overview ... 309
10.6.2 Basic parameter assignment ... 310
10.6.3 Time sequence ... 312
10.6.4 Phase 1: Travel across the reference marks with synchronization ... 312
10.6.5 Phase 2: Traversing to the target point ... 314
10.7 Referencing by means of actual value adjustment ... 316
10.7.1 Actual value adjustment to the referencing measurement system ... 316
10.7.2 Actual value adjustment for measuring systems with distance-coded reference marks ... 317
10.8 Referencing in follow-up mode ... 318
10.9 Referencing with absolute encoders ... 321
10.9.1 Information about the adjustment ... 321
10.9.2 Calibration by entering a reference point offset ... 322
10.9.3 Adjustment by entering a reference point value ... 323
10.9.4 Automatic calibration with probe ... 325
10.9.5 Adjustment with BERO ... 327
10.9.6 Reference point approach with absolute encoders ... 328
10.9.7 Reference point approach for rotary absolute encoders with equivalent zero mark ... 328
10.9.8 Enabling the measurement system ... 330
10.9.9 Referencing variants not supported ... 332
10.10 Automatic restoration of the machine reference ... 332
10.10.1 Overview ... 332
10.10.2 Automatic referencing ... 333
10.10.3 Restoration of the actual position ... 334
10.11 Supplementary conditions ... 336
10.11.1 Large traverse range ... 336

10.12 Data lists 337
10.12.1 Machine data 337
10.12.1.1 NC-specific machine data 337
10.12.1.2 Channelspecific machine data 337
10.12.1.3 Axis/spindlespecific machine data 337

11 S9: Setpoint switchover 339

11.1 Brief description 339
11.2 Startup 341
11.3 Flow diagram 344
11.4 Boundary conditions 345
11.5 Data lists 346
11.5.1 Machine data 346
11.5.1.1 Axis/spindlespecific machine data 346

12 B2: Acceleration and jerk 347

12.1 Brief description 347
12.1.1 General information 347
12.1.2 Features 347
12.2 Functions 349
12.2.1 Acceleration without jerk limitation (BRISK/BRISKA) (channel/axis-specific) 349
12.2.1.1 General Information 349
12.2.1.2 Parameterization 350
12.2.1.3 Programming 351
12.2.2 Constant traversing time (channel-specific) 352
12.2.2.1 General Information 352
12.2.2.2 Parameterization 353
12.2.3 Acceleration adaptation (ACC) (axis-specific) 354
12.2.3.1 General Information 354
12.2.3.2 Programming 354
12.2.4 Acceleration reserve (channel-specific) 355
12.2.4.1 General Information 355
12.2.4.2 Parameterization 355
12.2.5 Limiting the path acceleration (channel-specific) 355
12.2.5.1 General Information 355
12.2.5.2 Parameterization 356
12.2.5.3 Programming 356
12.2.6 Path acceleration for real-time events (channel-specific) 357
12.2.6.1 General Information 357
12.2.6.2 Programming 358
12.2.7 Acceleration with programmed rapid traverse (G00) (axis-specific) 359
12.2.7.1 General Information 359
12.2.7.2 Parameterization 359
12.2.8 Acceleration with active jerk limitation (SOFT/SOFTA) (axis-specific) 360
12.2.8.1 General Information 360
12.2.8.2 Parameterization 360
12.2.9 Exceeding the acceleration with non-tangential block transitions (axis-specific) 360
12.2.9.1 General Information 360
12.2.9.2 Parameterization 361

12.2.10 Acceleration reserve for the radial acceleration (channel-specific) ... 361
12.2.10.1 General Information ... 361
12.2.10.2 Parameterization ... 362
12.2.11 Jerk limitation with path interpolation (SOFT) (channel-specific) ... 363
12.2.11.1 General Information ... 363
12.2.11.2 Parameterization ... 365
12.2.11.3 Programming ... 365
12.2.12 Jerk limitation with single-axis interpolation (SOFTA) (axis-specific) ... 366
12.2.12.1 Parameterization ... 366
12.2.12.2 Programming ... 367
12.2.13 Limiting the path jerking (channel-specific) ... 367
12.2.13.1 General Information ... 367
12.2.13.2 Parameterization ... 368
12.2.13.3 Programming ... 368
12.2.14 Path jerk for real-time events (channel-specific) ... 369
12.2.14.1 General Information ... 369
12.2.14.2 Programming ... 370
12.2.15 Jerk during programmed rapid traverse (G00) (axis-specific) ... 371
12.2.15.1 General Information ... 371
12.2.15.2 Parameterization ... 371
12.2.16 Excessive jerk for block transitions without constant curvature (axis-specific) ... 371
12.2.16.1 General Information ... 371
12.2.16.2 Parameterization ... 372
12.2.17 Velocity-dependent jerk adaptation (axis-specific) ... 372
12.2.18 Jerk filter (position setpoint filter, axis-specific) ... 374
12.2.18.1 General Information ... 374
12.2.18.2 Parameterization ... 377
12.2.19 Knee-shaped acceleration characteristic ... 379
12.2.19.1 Function: Adaptation to the motor characteristic curve ... 379
12.2.19.2 Function: Effects on the path acceleration ... 379
12.2.19.3 Function: Substitute curve ... 380
12.2.19.4 Parameterization ... 383
12.2.19.5 Programming: Channel-specific activation (DRIVE) ... 386
12.2.19.6 Programming: Axis-specific activation (DRIVEA) ... 386
12.2.19.7 Boundary conditions ... 387
12.2.20 Acceleration and jerk for JOG motions ... 388
12.2.20.1 Parameterization ... 388
12.2.20.2 Supplementary conditions ... 389

12.3 Examples ... 390
12.3.1 Acceleration ... 390
12.3.1.1 Path velocity characteristic ... 390
12.3.2 Jerk ... 392
12.3.2.1 Path velocity characteristic ... 392
12.3.3 Acceleration and jerk ... 393
12.3.4 Knee-shaped acceleration characteristic ... 395
12.3.4.1 Activation ... 395

12.4 Data lists ... 396
12.4.1 Machine data ... 396
12.4.1.1 NC-specific machine data ... 396
12.4.1.2 Channel-specific machine data ... 396
12.4.1.3 Axis/spindlespecific machine data ... 396

12.4.2 Setting data 397
12.4.2.1 Channelspecific setting data 397
12.4.3 System variables 397

13 N3: Software cams, position switching signals 399

13.1 Brief description 399

13.2 Cam signals and cam positions 400
13.2.1 Generation of cam signals for separate output 400
13.2.2 Generation of cam signals with gated output 404
13.2.3 Cam positions 408
13.2.4 Lead/delay times (dynamic cam) 410

13.3 Output of cam signals 411
13.3.1 Activating 411
13.3.2 Output of cam signals to PLC 411
13.3.3 Output of cam signals to NCK I/Os in position control cycle 412
13.3.4 Timer-controlled cam signal output 413
13.3.5 Independent, timer-controlled output of cam signals 415

13.4 Position-time cams 416

13.5 Boundary conditions 417

13.6 Data lists 417
13.6.1 Machine data 417
13.6.1.1 General machine data 417
13.6.2 Setting data 418
13.6.2.1 General setting data 418

14 M3: Coupled axes 419

14.1 Coupled motion 419
14.1.1 Brief description 419
14.1.1.1 Function 419
14.1.1.2 Preconditions 419
14.1.2 General functionality 420
14.1.3 Programming 424
14.1.3.1 Definition and switch on of a coupled axis grouping (TRAILON) 424
14.1.3.2 Switch off (TRAILOF) 424
14.1.4 Effectiveness of PLC interface signals 425
14.1.5 Status of coupling 426
14.1.6 Dynamics limit 427
14.1.7 Supplementary conditions 428
14.1.8 Examples 428

14.2 Curve tables 429
14.2.1 Product brief 429
14.2.1.1 Function 429
14.2.1.2 Preconditions 430
14.2.2 General functionality 430
14.2.3 Memory organization 431
14.2.4 Commissioning 433
14.2.4.1 Memory configuration 433
14.2.4.2 Tool radius compensation 434
14.2.4.3 Specification of memory type 434

14.2.5 Programming....435
14.2.6 Access to table positions and table segments....441
14.2.7 Activation/deactivation....445
14.2.8 Modulo-leading axis special case....446
14.2.9 Behavior in AUTOMATIC, MDA and JOG modes....446
14.2.10 Effectiveness of PLC interface signals....447
14.2.11 Diagnosing and optimizing utilization of resources....447
14.2.12 Supplementary conditions....451
14.2.13 Examples....451

14.3 Master value coupling....453
14.3.1 Product brief....453
14.3.1.1 Function....453
14.3.1.2 Preconditions....453
14.3.2 General functionality....453
14.3.3 Programming....457
14.3.4 Behavior in AUTOMATIC, MDA and JOG modes....461
14.3.5 Effectiveness of PLC interface signals....463
14.3.6 Special characteristics of the axis master value coupling function....463
14.3.7 Supplementary conditions....464

14.4 Electronic gear (EG)....464
14.4.1 Product brief....464
14.4.1.1 Function....464
14.4.1.2 Preconditions....465
14.4.2 Electronic gear (EG)....465
14.4.3 Definition of an EG axis group....473
14.4.4 Activating an EG axis group....474
14.4.5 Deactivating an EG axis group....478
14.4.6 Deleting an EG axis group....479
14.4.7 Interaction between rotation feedrate (G95) and electronic gearbox....479
14.4.8 Response to POWER ON, RESET, operating mode change, block search....480
14.4.9 System variables for electronic gearbox....481
14.4.10 Examples....482
14.4.10.1 Example using linear couplings....482
14.4.10.2 Extended example with non-linear components....486

14.5 Generic coupling....492
14.5.1 Brief description....492
14.5.1.1 Function....492
14.5.1.2 Requirements....492
14.5.2 Fundamentals....495
14.5.2.1 Coupling module....495
14.5.2.2 Keywords and coupling characteristics....497
14.5.2.3 System variables....499
14.5.3 Creating/deleting coupling modules....500
14.5.3.1 Creating a coupling module (CPDEF)....500
14.5.3.2 Delete coupling module (CPDEL)....501
14.5.3.3 Defining leading axes (CPLDEF or CPDEF+CPLA)....502
14.5.3.4 Delete leading axes (CPLDEL or CPDEL+CPLA)....503
14.5.4 Switching coupling on/off....504
14.5.4.1 Switching on a coupling module (CPON)....504
14.5.4.2 Switch off coupling module (CPOF)....505
14.5.4.3 Switching on leading axes of a coupling module (CPLON)....506

14.5.4.4 Switching off leading axes of a coupling module (CPLOF) 507
14.5.4.5 Implicit creation and deletion of coupling modules 507
14.5.5 Programming coupling characteristics 508
14.5.5.1 Coupling rule (CPLNUM, CPLDEN, CPLCTID) 508
14.5.5.2 Coupling relationship (CPLSETVAL) 510
14.5.5.3 Co-ordinate reference (CPFRS): 511
14.5.5.4 Block change behavior (CPBC) 512
14.5.5.5 Synchronized position of the following axis when switching on (CPFPOS+CPON) 514
14.5.5.6 Synchronized position of the leading axis when switching on (CPLPOS) 515
14.5.5.7 Synchronization mode (CPFMSON) 516
14.5.5.8 Behavior of the following axis at switch-on (CPFMON) 518
14.5.5.9 Behavior of the following axis at switch-off (CPFMOF) 519
14.5.5.10 Position of the following axis when switching off (CPFPOS+CPOF) 519
14.5.5.11 Condition at RESET (CPMRESET) 520
14.5.5.12 Condition at parts program start (CPMSTART) 521
14.5.5.13 Status during part program start in search run via program test (CPMPRT) 523
14.5.5.14 Offset / scaling (CPLINTR, CPLINSC, CPLOUTTR, CPLOUTSC) 524
14.5.5.15 Synchronism monitoring stage 1 (CPSYNCOP, CPSYNFIP, CPSYNCOV, CPSYNFIV) 527
14.5.5.16 Synchronous operation monitoring stage 2 (CPSYNCOP2, CPSYNFIP2) 531
14.5.5.17 Reaction to stop signals and commands (CPMBRAKE) 535
14.5.5.18 Response to certain NC/PLC interface signals (CPMVDI) 536
14.5.5.19 Alarm suppression (CPMALARM) 540
14.5.6 Coupling cascading 541
14.5.7 Compatibility 542
14.5.7.1 Adaptive cycles 542
14.5.7.2 Coupling types (CPSETTYPE) 543
14.5.7.3 Projected coupling (CPRES) 548
14.5.8 Cross-channel coupling, axis replacement 549
14.5.9 Behavior with rotary axes 550
14.5.10 Behavior during POWER ON, ... 551
14.5.11 CP SW limit monitoring 552
14.5.11.1 Function 552
14.5.11.2 Parameterization 554
14.5.11.3 Programming 555
14.5.11.4 Boundary conditions 555
14.5.11.5 Examples 556
14.5.12 Disturbance characteristic 557
14.5.12.1 Rapid stop 557
14.5.13 Tracking the deviation from synchronism 558
14.5.13.1 Fundamentals 558
14.5.13.2 Measuring the deviation from synchronism 559
14.5.13.3 Entering the deviation from synchronism directly 562
14.5.13.4 Synchronism correction 562
14.5.13.5 Diagnostics for synchronism correction 563
14.5.13.6 Resetting synchronism correction 564
14.5.13.7 Limitations and constraints 565
14.5.14 Examples 567
14.5.14.1 Programming examples 567
14.5.14.2 Adapt adaptive cycle 568

14.6 Dynamic response of following axis 569
14.6.1 Parameterized dynamic limits 569
14.6.2 Programmed dynamic limits 570

14.6.2.1 Programming (VELOLIMA, ACCLIMA) 570
14.6.2.2 Examples 572
14.6.2.3 System variables 573
14.7 General supplementary conditions 573
14.8 Data lists 574
14.8.1 Machine data 574
14.8.1.1 NC-specific machine data 574
14.8.1.2 Channelspecific machine data 574
14.8.1.3 Axis/spindlespecific machine data 575
14.8.2 Setting data 575
14.8.2.1 Axis/spindle-specific setting data 575
14.8.3 System variables 575
15 P5: Oscillation 579
15.1 Brief description 579
15.2 Asynchronous oscillation 580
15.2.1 Asynchronous oscillation 580
15.2.2 Influences on asynchronous oscillation 581
15.2.3 Asynchronous oscillation under PLC control 587
15.2.4 Special reactions during asynchronous oscillation 587
15.3 Oscillation controlled by synchronized actions 590
15.3.1 Infeed at reversal point 1 or 2 593
15.3.2 Infeed in reversal point range 594
15.3.3 Infeed at both reversal points 595
15.3.4 Stop oscillation movement at the reversal point 596
15.3.5 Oscillation movement restarting 597
15.3.6 Do not start partial infeed too early 598
15.3.7 Assignment of oscillation and infeed axes OSCILL 599
15.3.8 Definition of infeeds POSP 599
15.3.9 External oscillation reversal 600
15.4 Marginal conditions 601
15.5 Examples 601
15.5.1 Example of asynchronous oscillation 601
15.5.2 Example 1 of oscillation with synchronized actions 602
15.5.3 Example 2 of oscillation with synchronized actions 605
15.5.4 Examples for starting position 607
15.5.4.1 Define starting position via language command 607
15.5.4.2 Start oscillation via setting data 608
15.5.4.3 Non-modal oscillation (starting position = reversal point 1) 608
15.5.5 Example of external oscillation reversal 611
15.5.5.1 Change reversal position via synchronized action with "external oscillation reversal" 611
15.6 Data lists 611
15.6.1 Machine data 611
15.6.1.1 General machine data 611
15.6.2 Setting data 612
15.6.2.1 Axis/spindle-specific setting data 612
15.6.3 System variables 612
15.6.3.1 Main run variables for motion-synchronous actions 612

16 R3: Extended stop and retract 615
16.1 Brief description 615
16.2 ESR executed autonomously in the drive 615
16.2.1 Fundamentals 615
16.2.2 Configuring stopping in the drive 616
16.2.3 Configuring retraction in the drive 618
16.2.4 Configuring generator operation in the drive 619
16.2.5 ESR is enabled via system variable 621
16.2.6 Triggering ESR via system variable 621
16.2.7 Feedback of the ESR status 621
16.2.8 Acknowledge ESR reactions 622
16.2.9 Configuring ESR in the part program 623
16.2.9.1 Stopping (ESRS) 624
16.2.9.2 Retraction (ESRR) 624
16.2.9.3 Boundary conditions 626
16.3 Boundary conditions 627
16.4 Examples 628
16.4.1 NC-controlled reactions 628
16.4.2 Retraction while thread cutting 629
16.4.3 Rapid lift using ASUB and fast input 630
16.4.4 Rapid lift, absolute and incremental 630
16.4.5 Lift fast with linear relation of axes 631
16.5 Data lists 631
16.5.1 Machine data 631
16.5.1.1 Channelspecific machine data 631
16.5.1.2 Axis/spindlespecific machine data 632
16.5.2 System variables 632
17 H1: Manual traversing 633
17.1 Overview 633
17.2 Control via the PLC interface 636
17.3 Parameter assignment (general) 637
17.4 Continuous manual traversing 643
17.4.1 Function 643
17.4.2 Parameter assignment 646
17.4.3 Supplementary conditions 646
17.5 Incremental manual traversing 646
17.5.1 Function 646
17.5.2 Parameter assignment 650
17.5.3 Supplementary conditions 651
17.6 Manual traversing using a handwheel 651
17.6.1 Function 651
17.6.2 Parameter assignment 658
17.6.3 Travel request 664
17.7 Manual traversing of the spindle 667
17.8 Manual traversing of geometry axes/orientation axes 668

17.9 Approaching a fixed point in JOG 671
17.9.1 Function 671
17.9.2 Parameterization 674
17.9.3 Programming 676
17.9.4 Supplementary Conditions 676
17.9.5 Application example 677
17.10 Position travel in JOG 678
17.10.1 Function 678
17.10.2 Parameter setting 681
17.10.3 Supplementary Conditions 681
17.10.4 Application example 682
17.11 Circular travel in JOG 683
17.11.1 Function 683
17.11.2 Parameter setting 689
17.11.3 Supplementary Conditions 692
17.11.4 Application example 692
17.12 Retraction in the tool direction (JOG retract) 693
17.12.1 Overview 693
17.12.2 Parameterization 695
17.12.2.1 Automatic selection of JOG retract after Power On 695
17.12.2.2 Enable of the traversing direction 695
17.12.2.3 Measuring system status 695
17.12.3 Selection 696
17.12.4 Tool retraction 697
17.12.5 Deselection 699
17.12.6 Repeated selection 699
17.12.7 Continuing machining 700
17.12.8 State diagram 701
17.12.9 System data 701
17.12.10 Supplementary conditions 702
17.13 Use of handwheels in automatic mode 703
17.13.1 Handwheel override in automatic mode 703
17.13.1.1 General functionality 703
17.13.1.2 Programming and activating handwheel override 707
17.13.1.3 Special features of handwheel override in automatic mode 709
17.13.2 Contour handwheel/path input using handwheel (option) 710
17.13.3 DRF offset 713
17.13.3.1 DRF offset 713
17.13.3.2 Commissioning 716
17.13.3.3 Programming: Deselecting overlays axis-specifically (CORROF) 719
17.13.4 Double use of the handwheel 721
17.14 Monitoring functions 723
17.15 Start-up: Handwheels 724
17.15.1 General information 724
17.15.2 Connection via Ethernet 725
17.16 Data lists 728
17.16.1 Machine data 728
17.16.1.1 General machine data 728

17.16.1.2 Channel-specific machine data 728
17.16.1.3 Axis/spindlespecific machine data 729
17.16.2 Setting data 730
17.16.2.1 General setting data 730
17.16.2.2 Channel-specific setting data 730
17.16.2.3 Axis/spindle-specific setting data 730
17.16.3 System variable 731
17.16.3.1 System variable 731
17.16.4 OPI variable 731
17.16.4.1 OPI variable 731

18 S1: Spindles 733

18.1 Brief Description 733

18.2 Operating modes 734
18.2.1 Overview 734
18.2.2 Mode change 735
18.2.3 Control mode 736
18.2.4 Oscillation mode 739
18.2.5 Positioning mode 739
18.2.5.1 General functionality 739
18.2.5.2 Positioning from rotation 746
18.2.5.3 Positioning from standstill 751
18.2.5.4 "Spindle in position" signal for tool change 755
18.2.6 Axis mode 756
18.2.6.1 General functionality 756
18.2.6.2 Implicit transition to axis mode 759
18.2.7 Initial spindle state 762
18.2.8 Tapping without compensating chuck 763
18.2.8.1 Function 763
18.2.8.2 Programming 763
18.2.8.3 Example: Tapping with G331 / G332 765
18.2.8.4 Example: Output the programmed drilling speed in the current gear stage 765
18.2.8.5 Example: Application of the second gear-stage data block 766
18.2.8.6 Example: Speed is not programmed, the gearbox stage is monitored 766
18.2.8.7 Example: Gearbox stage cannot be changed, gearbox stage monitoring 766
18.2.8.8 Example: Programming without SPOS 767
18.2.8.9 Special case: Direction of rotation reversal via NC/PLC interface signal in the NC program 767
18.2.9 Tapping with compensating chuck 769
18.2.9.1 Function 769
18.2.9.2 Programming 769

18.3 Reference / synchronize 770

18.4 Configurable gear adaptations 775
18.4.1 Gear stages for spindles and gear change change 775
18.4.2 Spindle gear stage 0 786
18.4.3 Determining the spindle gear stage 788
18.4.4 Parameter set selection during gear step change 789
18.4.5 Intermediate gear 792
18.4.6 Nonacknowledged gear step change 793
18.4.7 Gear step change with oscillation mode 794
18.4.8 Gear stage change at fixed position 799

18.4.9 Configurable gear step in M70 805
18.4.10 Suppression of the gear stage change for DryRun, program test and SERUPRO 807
18.5 Additional adaptations to the spindle functionality that can be configured 809
18.6 Selectable spindles 811
18.7 Programming 815
18.7.1 Programming from the part program 815
18.7.2 Programming via synchronized actions 819
18.7.3 Programming spindle controls via PLC with FC18 819
18.7.4 Programming using NC/PLC interface signals 820
18.7.4.1 Function 820
18.7.4.2 Commissioning: Machine data 821
18.7.4.3 Commissioning: NC/PLC interface signals 822
18.7.4.4 Speed setpoint (SD43200) 822
18.7.4.5 Entering a constant cutting rate (SD43202) 823
18.7.4.6 Entering the spindle speed type for the master spindle (SD43206) 824
18.7.5 External programming (PLC, HMI) 825
18.8 Spindle monitoring 826
18.8.1 Permissible speed ranges 826
18.8.2 Axis/spindle stationary 827
18.8.3 Spindle in setpoint range 827
18.8.4 Minimum / maximum speed of the gear stage 828
18.8.5 Diagnosis of spindle speed limitation 829
18.8.6 Maximum spindle speed 831
18.8.7 Maximum encoder limit frequency 832
18.8.8 End point monitoring 834
18.8.9 M40: Automatic gear stage selection for speeds outside the configured switching thresholds 835
18.9 Spindle with SMI 24 (Weiss spindle) 837
18.9.1 General Information 837
18.9.2 Sensor data 838
18.9.3 Clamped state 840
18.9.4 Additional drive parameters 841
18.10 Boundary conditions 842
18.10.1 Changing control parameters 842
18.11 Examples 842
18.11.1 Automatic gear step selection (M40) 842
18.12 Data lists 843
18.12.1 Machine data 843
18.12.1.1 NC-specific machine data 843
18.12.1.2 Channelspecific machine data 843
18.12.1.3 Axis/spindlespecific machine data 844
18.12.2 Setting data 845
18.12.2.1 Channelspecific setting data 845
18.12.2.2 Axis/spindle-specific setting data 846
19 S3: Synchronous spindle 847
19.1 Brief description 847
19.1.1 Function 847

19.1.2 Synchronous mode 849
19.1.3 Prerequisites for synchronous mode 855
19.1.4 Selecting synchronous mode for a part program 856
19.1.5 Deselecting the synchronous mode for the part program 858
19.1.6 Controlling synchronous spindle coupling via PLC 859
19.1.7 Monitoring of synchronous operation 862

19.2 Programming 864
19.2.1 Definition (COUPDEF) 865
19.2.2 Switch the coupling (COUPON, COUPONC, COUPOF) on and off 868
19.2.3 Axial system variables for synchronous spindle 869
19.2.4 Automatic selection and deselection of position control 870

19.3 Configuration 871
19.3.1 Response of the synchronous-spindle coupling for NC Start 872
19.3.2 Behavior of the synchronous-spindle coupling for reset 873

19.4 Special features 873
19.4.1 Special features of synchronous mode in general 873
19.4.2 Restore synchronism of following spindle 875
19.4.3 Synchronous mode and NC/PLC interface signals 877
19.4.4 Differential speed between leading and following spindles 881
19.4.5 Behavior of synchronism signals during synchronism correction 886
19.4.6 Delete synchronism correction and NC reset 886
19.4.7 Special points regarding start-up of a synchronous spindle coupling 886

19.5 Boundary conditions 891

19.6 Examples 891

19.7 Data lists 892
19.7.1 Machine data 892
19.7.1.1 NC-specific machine data 892
19.7.1.2 Channelspecific machine data 892
19.7.1.3 Axis/spindlespecific machine data 893
19.7.2 Setting data 893
19.7.2.1 Channelspecific setting data 893
19.7.3 System variables 894

A Appendix 895

A.1 List of abbreviations 895

Index 905

Fundamental safety instructions

1

1.1 General safety instructions

⚠ WARNING

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

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

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

⚠ WARNING

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

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

- Protect the parameterization against unauthorized access.
- Handle possible malfunctions by taking suitable measures, e.g. emergency stop or emergency off.

1.2 Warranty and liability for application examples

Application examples are not binding and do not claim to be complete regarding configuration, equipment or any eventuality which may arise. Application examples do not represent specific customer solutions, but are only intended to provide support for typical tasks.

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

1.3 Industrial security

Note

Industrial security

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

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

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

For additional information on industrial security measures that can be implemented, please visit:

Industrial security (https://www.siemens.com/industrialsecurity)

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

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed at:

Industrial security (https://www.siemens.com/industrialsecurity)

Further information is provided on the Internet:

Industrial Security Configuration Manual (https://support.industry.siemens.com/cs/ww/en/view/108862708)

⚠ WARNING

Unsafe operating states resulting from software manipulation

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

- Keep the software up to date.
- Incorporate the automation and drive components into a holistic, state-of-the-art industrial security concept for the installation or machine.
- Make sure that you include all installed products into the holistic industrial security concept.
- Protect files stored on exchangeable storage media from malicious software by with suitable protection measures, e.g. virus scanners.
- On completion of commissioning, check all security-related settings.
- Protect the drive against unauthorized changes by activating the "Know-how protection" converter function.

G2: Velocities, setpoint / actual value systems, closed-loop control

2

2.1 Brief description

The description of functions explains how to parameterize a machine axis in relation to:

- Actual-value/measuring systems
- Setpoint system
- Operating accuracy
- Travel ranges
- Axis velocities
- Control parameters

2.2 Velocities, traversing ranges, accuracies

2.2.1 Velocities

Maximum path and axis velocities and spindle speed

The maximum path and axis velocities and spindle speed are influenced by the machine design, the dynamic response of the drive and the limit frequency of the actual-value acquisition (encoder limit frequency).

The maximum axis velocity is defined in machine data:

MD32000 $MA_MAX_AX_VELO (maximum axis velocity)

The maximum permissible spindle speed is specified using machine data:

MD35100 $MA_SPIND_VELO_LIMIT (maximum spindle speed)

For explanations, see Section "S1: Spindles (Page 733)".

With a higher feedrate (resulting from programmed feedrates and feedrate override), the velocity is limited to V_{max}.

This automatic feedrate limiting can lead to a drop in velocity over several blocks with programs generated by CAD systems with extremely short blocks.

Example:

Interpolation cycle = 12 ms

```
N10 G0 X0 Y0; [mm]
N20 G0 X100 Y100; [mm]
```

⇒ Path length programmed in block = 141.42 mm

⇒ V_{max} = (141.42 mm/12 ms) * 0.9 = 10606.6 mm/s = 636.39 m/min

Minimum path, axis velocity

The following restriction applies to the minimum path or axis velocity:

$$V_{min} \geq \frac{10^{-3}}{\text{Computational resolution } [\frac{\text{Incr.}}{\text{mm or degrees}}] * \text{IPO cycle [s]}}$$

The computational resolution is defined using machine data:
MD10200 $MN_INT_INCR_PER_MM (computational resolution for linear positions)

or
MD10210 $MN_INT_INCR_PER_DEG (computational resolution for angular positions)

If V_{min} is not reached, no traversing is carried out.

Example:

MD10200 $MN_INT_INCR_PER_MM = 1000 [incr/mm] ;

Interpolation cycle = 12 ms;

⇒ $V_{min} = 10^{-3}/(1000\ ^{incr}/_{mm} \times 12\ ms) = 0.005\ ^{mm}/_{min}$;

The value range of the feedrates depends on the computational resolution selected.

For the standard assignment of machine data:

MD10200 $MN_INT_INCR_PER_MM
(computational resolution for linear positions) (1000 incr./mm)

or

MD10210 $MN_INT_INCR_PER_DEG

(computational resolution for angular positions) (1000 incr./deg.)

The following value range can be programmed with the specified resolution:

Range of values for path feed F and geometry axes:	
Metric system:	Inch system:
0.001 ≤ F ≤ 999,999.999 [mm/min, mm/rev, degrees/min, degrees/rev]	0.001 ≤ F ≤ 399,999.999 [inch/min, inch/rev]

Range of values for feedrate for positioning axes:	
Metric system:	Inch system:
0.001 ≤ FA ≤ 999,999.999 [mm/min, mm/rev, degrees/min, degrees/rev]	0.001 ≤ FA ≤ 399,999.999 [inch/min, inch/rev]

Range of values for spindle speed S:	0.001 ≤ S ≤ 999,999.999 [rpm]

If the computational resolution is increased/decreased by a factor, then the value ranges change accordingly.

2.2.2 Traversing ranges

Range of values of the traversing ranges

The range of values of the traversing range depends on the computational resolution selected.

For the standard assignment of machine data:

MD10200 $MN_INT_INCR_PER_MM

(computational resolution for linear positions) (1000 incr./mm)

or

MD10210 $MN_INT_INCR_PER_DEG

(computational resolution for angular positions) (1000 incr./deg.)

The following value range can be programmed with the specified resolution:

Table 2-1 Traversing ranges of axes

	G71 [mm, degrees]	G70 [inch, degrees]
	Range	Range
Linear axes X, Y, Z, etc.	∓ 999,999.999	∓ 399,999.999
Rotary axes A, B, C, etc.	∓ 999,999.999	∓ 999,999.999
Interpolation parameters I, J, K	∓ 999,999.999	∓ 399,999.999

The unit of measurement of rotary axes is always degrees.

If the computational resolution is increased/decreased by a factor of 10, the ranges of values change accordingly.

The traversing range can be limited by software limit switches and work areas (see Function Manual *Monitoring and Compensations*, Section *Axis monitoring*).

For special features for a large traversing range for linear and rotary axes, see Section "R1: Referencing (Page 291)".

The traversing range for rotary axes can be limited via machine data.

Further information:

R2: Rotary axes (Page 221)

2.2.3 Positioning accuracy of the control system

Actual-value resolution and computational resolution

The positioning accuracy of the control depends on the actual-value resolution (=encoder increments/(mm or degrees)) and the computational resolution (=internal increments/(mm or degrees)).

The coarse resolution of these two values determines the positioning accuracy of the control.

The input resolution, interpolator and position-control cycle selections have no effect on this accuracy.

As well as limiting using MD32000, the control limits the maximum path velocity in relation to the situation and according to the following formula:

$$\frac{\text{Internal increments/mm}}{\text{Encoder increments/mm}} = \frac{1}{\text{ENC_RESOL [n]} * \text{ENC_PULSE_MULT[n]}} * \frac{\text{DRIVE_ENC_RATIO_NUMERA [n]}}{\text{DRIVE_ENC_RATIO_DENOM [n]}} * \frac{\text{DRIVE_AX_RATIO_DENOM [n]}}{\text{DRIVE_AX_RATIO_NUMERA [n]}} * \text{LEADSCREW_PITCH} * \text{INT_INCR_PER_MM}$$

2.2.4 Input/display resolution, computational resolution

Resolutions: Differences

Resolutions, e.g. resolutions of linear and angular positions, velocities, accelerations and jerk, must be differentiated as follows:

- Input resolution
  Data is input via the control panel or part programs.
- Display resolution
  Data is displayed via the control panel.
- Computational resolution
  Data input via the control panel or part program is displayed internally.

The input and display resolution is determined by the specified operator panel front used, whereby the display resolution for position values with the machine data:

MD9004 $MM_DISPLAY_RESOLUTION (display resolution)

can be changed.

The machine data:

MD9011 $MM_DISPLAY_RESOLUTION_INCH (display resolution for INCH measuring system)

can be used to configure the display resolution for position values with inch setting.

This allows you to display up to six decimal places with the inch setting.

For the programming of part programs, the input resolutions listed in the Programming Guide apply.

The desired computational resolution is defined using the machine data:

MD10200 $MN_INT_INCR_PER_MM (computational resolution for linear positions)

and

MD10210 $MN_INT_INCR_PER_ DEG (computational resolution for angular positions).

It is independent of the input/display resolution but should have at least the same resolution.

The maximum number of places after the decimal point for position values, velocities, etc., in the part program and the number of places after the decimal point for tool offsets, zero offsets, etc. (and therefore also for the maximum possible accuracy) is defined by the computational resolution.

The accuracy of angle and linear positions is limited to the computational resolution by rounding the product of the programmed value with the computational resolution to an integer number.

To make the rounding clear, powers of 10 should be used for the calculation resolution.

Example of rounding:

Computational resolution: 1000 incr./mm

Programmed path: 97.3786 mm

Effective value: 97.379 mm

Example of programming in the $^1/_{10}$ µm range:

All the linear axes of a machine are to be programmed and traversed within the range of values 0.1 to 1000 µm.

⇒ In order to position accurately to 0.1 µm, the computational resolution must be set to ≥ 10^4 incr./mm.

⇒ MD10200 $MN_INT_INCR_PER_MM = 10000 [incr./mm]:

⇒ Example of related part program:

```
Program code                          Comment
N20 G0 X 1.0000 Y 1.0000              ; Axes travel to the position
                                        X=1.0000 mm, Y=1.0000 mm;
N25 G0 X 5.0002 Y 2.0003              ; Axes travel to the position
                                        X=5.0002 mm, Y=2.0003 mm
```

2.2.5 Scaling of physical quantities of machine and setting data

Input/output units

Machine and setting data with a physical unit are interpreted in the input/output units stated below depending on whether the metric or inch system is selected:

Physical unit	Input/output units for standard basic system	
	Metric	Inch
Linear position	1 mm	1 inch
Angular position	1 degree	1 degree
Linear velocity	1 mm/min	1 inch/min
Angular velocity	1 rpm	1 rpm
Linear acceleration	1 m/s^2	1 $inch/s^2$
Angular acceleration	1 $rev./^{s2}$	1 $rev./s^2$
Linear jerk	1 m/s^3	1 $inch/s^3$
Angular jerk	1 $rev./s^3$	1 $rev./s^3$
Time	1 s	1 s
Position controller servo gain	1/s	1/s
Revolutional feedrate	1 mm/rev	inch/rev
Compensation value linear position	1 mm	1 inch
Compensation value angular position	1 degree	1 degree

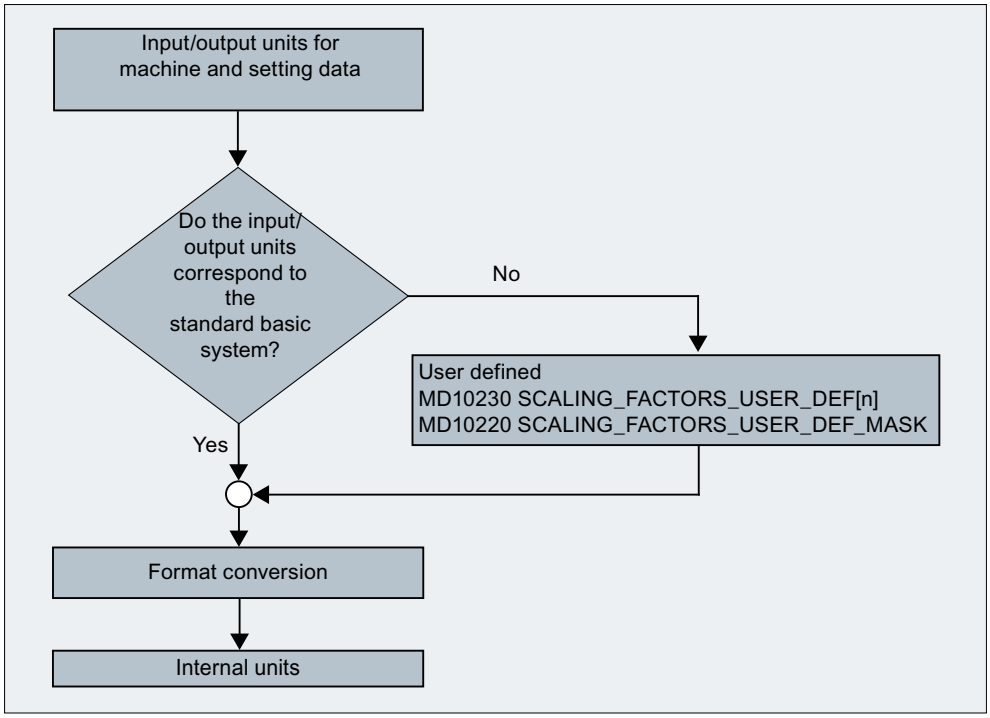
Internally in the control, the following units are used, whatever basic system is selected:

Physical unit	Unit
Linear position	1 mm
Angular position	1 degree
Linear velocity	1 mm/s
Angular velocity	1 deg./s
Linear acceleration	1 mm/s^2
Angular acceleration	1 $degree/s^2$
Linear jerk	1 mm/s^3
Angular jerk	1 $degree/s^3$
Time	1 s
Position controller servo gain	1/s

Physical unit	Unit
Revolutional feedrate	1 mm/degree
Compensation value linear position	1 mm
Compensation value angular position	1 degree

The user can define different input/output units for machine and setting data. This also requires an adjustment between the newly selected input/output units and the internal units via the following machine data:

- MD10220 $MN_SCALING_USER_DEF_MASK
- MD10230 $MN_SCALING_FACTORS_USER_DEF[n]



The following applies:

Selected I/O unit = (MD10230 $MN_SCALING_FACTORS_USER_DEF[n]) * internal unit

The selected I/O unit, expressed in the internal units 1 mm, 1 degree, and 1 s must therefore be entered in machine data MD10230 $MN_SCALING_FACTORS_USER_DEF[n].

Example 1:

Machine data input/output of the linear velocities is to be in m/min instead of mm/min (initial state).

(The internal unit is mm/s)

⇒ The scaling factor for the linear velocities is to differ from the standard setting. For this, in machine data:

MD10220 $MN_SCALING_USER_DEF_MASK

bit number 2 must be set.

⇒ MD10220 $MN_SCALING_USER_DEF_MASK = 'H4'; (bit no. 2 as hex value)

⇒ The scaling factor for the linear velocities is to differ from the standard setting. For this, in machine data:

MD10220 $MN_SCALING_USER_DEF_MASK

bit number 2 must be set.

⇒ MD10220 $MN_SCALING_USER_DEF_MASK = 'H4'; (bit no. 2 as hex value)

⇒ The scaling factor is calculated using the following formula:

$$\text{MD10230 SCALING_FACTORS_USER_DEF[n]} = \frac{\text{Selected input/output unit}}{\text{Internal unit}}$$

$$\text{MD10230 SCALING_FACTORS_USER_DEF[n]} = \frac{1\ \frac{m}{min}}{1\ \frac{mm}{s}} = \frac{\frac{1000\ mm}{60\ s}}{1\ \frac{mm}{s}} = \frac{1000}{60} = 16{,}667;$$

⇒ MD10230 SCALING_FACTORS_USER_DEF[n] = 16,667

Index n defines the "linear velocity" in the "Scaling factors of physical quantities" list.

Example 2:

In addition to the change in Example 1, the machine data input/output of linear accelerations must be in ft/s² instead of m/s² (initial state).
(The internal unit is mm/s².)

⇒ MD10220 SCALING_USER_DEF_MASK = H14 ; (Bit No. 4 and Bit No. 2 of example 1 as a Hex value)

$$\Rightarrow \text{MD10220 SCALING_FACTORS_USER_DEF[4]} = \frac{1\ \frac{ft}{s^2}}{1\ \frac{mm}{s^2}} = \frac{12*25{,}4\ \frac{mm}{s^2}}{1\ \frac{mm}{s^2}} = 304{,}8\ ;$$

⇒ MD10220 SCALING_FACTORS_USER_DEF[4] = 304,8

Index 4 defines the "linear acceleration" in the "Scaling factors of physical quantities" list.

2.3 System of units, metric/inch

2.3.1 Function

2.3.1.1 Parameterized and programmed system of units

SINUMERIK control systems can operate with a metric system of units as well as an inch system of units.

Parameterized system of units (basic system)

The basic setting of the system of units (basic system) is set in machine data MD10240 $MN_SCALING_SYSTEM_IS_METRIC (see "Commissioning (Page 41)").

Depending on the basic system, all length-related data is interpreted either as metric or inch system of units.

Metric system of units:	mm, mm/min, m/s2, m/s3, mm/rev.
Inch system of units:	inch, inch/min, inch/s2, inch/s3, inch/rev.

The basic system also defines the interpretation of the programmed F value for linear axes:

Feed type	Metric system of units	Inch system of units
G94	mm/min	inch/min
G95	mm/rev.	inch/rev.

Programmed system of units

Using the commands of G group 13 (system of units, inch/metric) within the part program, you can toggle between the metric and inch system of units (see "Programming (Page 45)").

The programmed system of units and the basic system may be identical or different at any time. Switchover of the system of units within a particular section of the part program could, for example, be used to machine an inch thread on a workpiece within a metric basic system.

The reset setting of the G group is defined in machine data MD20150 $MC_GCODE_RESET_VALUES (see "Commissioning (Page 41)").

HMI display

Data relating to length is displayed as follows on the user interface:

	Display	
	in the basic system	in the programmed system of units
Machine data	x	
Data in the machine coordinate system	x	

	Display	
	in the basic system	in the programmed system of units
Tool data	x	
Work offsets	x	
Data in the workpiece coordinate system		x

NC/PLC interface

In the case of NC/PLC interface signals containing dimension information, e.g. feedrate for path and positioning axes, data exchange is carried out with the PLC in the configured basic system.

Reading in external part programs

If part programs, including data sets (work offsets, tool offsets, etc.), programmed in a system of units other than the basic system are read in from an external source, then the basic system must first be changed.

2.3.1.2 Extended system of units functionality

From SW 5 and higher, the system of units functionality has been extended, which significantly simplifies toggling between systems of units.

The functions include:

- New G commands G700/G710 (see "Programming (Page 45)")
- Data backup with system of units identifier INCH/METRIC (see "Commissioning (Page 41)")
- Automatic data conversions when the system of units is changed, e.g. for:
  - Work offsets
  - Compensation data (EEC, QEC)
  - Tool offsets
  - ...

For compatibility reasons, the extended system of units functionality must be activated using a machine data (see "Commissioning (Page 41)").

2.3.1.3 Extended system of units functionality

From SW 5 and higher, the system of units functionality has been extended, which significantly simplifies toggling between systems of units.

The functions include:

- System of units switchover at the user interface
- New G commands G700/G710 (see "Programming (Page 45)")

- Data backup with system of units identifier INCH/METRIC (see "Commissioning (Page 41)")
- Automatic data conversions when the system of units is changed, e.g. for:
  - Work offsets
  - Compensation data (EEC, QEC)
  - Tool offsets
  - ...

For compatibility reasons, the extended system of units functionality must be activated using a machine data (see "Commissioning (Page 41)").

2.3.2 Commissioning

NC-specific machine data

Basic system

The basic system to be used by the controller for the scaling of length-dependent physical variables for data input/output is specified via the machine data:

MD10240 $MN_SCALING_SYSTEM_IS_METRIC

Value	Meaning
0 (= FALSE)	Data relating to length is interpreted as **inch** data.
1 (= TRUE)	Data relating to length is interpreted as **metric** data.

The system must be powered up again after changing this machine data as otherwise associated machine data, which have physical units, will be incorrectly scaled.

The following procedure must be complied with:

- Changing the MD using manual entry:
  - 1. Power up the system.
  - 2. Machine data with physical units: Enter a value.
- The MD is changed using machine data:
  - 1. Power up the system.
  - 2. Load the machine data file once again so that the new physical units are taken into account.

When changing MD10240, Alarm 4070 "Normalizing machine data has been changed" is output.

Note

When changing over the system of units at a control system, all length-related data must be converted consistently and completely into the other measuring system.

Conversion factor (NC-specific)

The factor for converting from metric into the inch system of units is set in machine data:

MD10250 $MN_SCALING_VALUE_INCH (conversion factor for inch)

Default value: 25.4

The conversion factor becomes active when selecting the non-metric basic system (MD10240 $MN_SCALING_SYSTEM_IS_METRIC = 0).

The following data are multiplied by the conversion factor:

- Length-related data for input/output (e.g. when uploading machine data, work offsets)
- Programmed F values for linear axes
- Programmed geometry of an axis (position, polynomial coefficients, radius for circle programming, ...), if the system of units programmed with G70/G71 differs from the basic system (MD10240 $MN_SCALING_SYSTEM_IS_METRIC).

Note

By changing the conversion factor, the control system can be adapted to customer-specific systems of units.

Extended system of units functionality

The extended system of units functionality can be activated using the following compatibility machine data:

MD10260 $MN_CONVERT_SCALING_SYSTEM

Value	Meaning
0 (= FALSE)	Extended system of units functionality **not** active (default setting; compatibility to previous software releases)
1 (= TRUE)	Extended system of units functionality active

When changing MD10240, Alarm 4070 "Normalizing machine data has been changed" is output.

Data backup

Individual data sets that are read from the control system and contain data affected by the system of units, are assigned an identifier during reading that indicates the current setting for the system of units:

- MD10260 $MN_CONVERT_SCALING_SYSTEM
- MD10240 $MN_SCALING_SYSTEM_IS_METRIC

The identifier records in which system of units the data have been read out. This ensures that no data sets are read into the control with a system of units other than that which is currently set.

Since the identifier is also evaluated in part programs, these can also be "protected" against operator errors as described above. You can therefore prevent part programs containing, e.g. only metric data, from running on an inch system of units.

Archive and machine data sets are downward-compatible with the following setting:

MD11220 $MN_INI_FILE_MODE = 2

Note

The INCH/METRIC identifier is only generated if the compatibility machine data:

MD10260 $MN_CONVERT_SCALING_SYSTEM = TRUE

Note

Rounding machine data

All length-related machine data is rounded to the nearest 1 pm when writing in the inch system of units (MD10240 $MN_SCALING_SYSTEM_IS_METRIC=0 and MD10260 $MN_CONVERT_SCALING_SYSTEM=1), to avoid rounding problems.

This corrects the loss of accuracy resulting from conversion to ASCII when reading out a data backup in the inch system of units when the data is read back into the system.

Input resolution and computational resolution

The input and calculation resolution is defined as an internal number of increments per millimeter.

MD10200 $MN_INT_INCR_PER_MM

The precision of entry of linear positions is limited to the calculation resolution. The product of the programmed position value and the calculation resolution is rounded up to the next integer. To make the rounding clear and understandable, powers of 10 should be used for the calculation resolution.

Example:

1 inch = 25.4 mm ⇒ 0.0001 inch = 0.00254 mm = 2.54 µm = 2540 nm

To be able to program and display the last 40 nm, a value of 100000 must be parameterized for the input and calculation resolution.

Only with this identical setting for both systems of units is it possible to change the system of units without a significant loss of accuracy. Once MD10200 has been set to this value, it will not need to be changed each time the measuring system is switched over.

System of units for positioning tables

The system of units for positional data of the indexing axis tables and switching points for software cams is configured in machine data:

MD10270 $MN_POS_TAB_SCALING_SYSTEM

Further information:
Function Manual Axis and Spindles; Software cams, position switching signals and indexing axes

User tool data

The physical units for user-defined tool and tool cutting edge data can be set in the following machine data:

- MD10290 $MN_CC_TDA_PARAM_UNIT
- MD10292 $MN_CC_TOA_PARAM_UNIT

Note

When the system of units is switched over, all length-related tool data is converted to the new system of units.

Channel-specific machine data

Reset position

For each channel, the reset position of G group 13 (system of units, inch/metric) is set in the machine data:

MD20150 $MC_GCODE_RESET_VALUES[12] (reset position of G group 13)

Axis/spindle-specific machine data

Conversion factor (axis-specific)

To make pure positioning axes independent of G70/G71, a factor for converting from metric into the inch system of units can be set in machine data:

MD31200 $MA_SCALING_FACTOR_G70_G71 (conversion factor when G70/G71 is active)

Default value: 25.4

Note

The conversion factor should be identical for all three geometry axes.

JOG and handwheel factor

The following machine data comprises two values containing axis-specific increment weighting factors for each of the two system of units:

MD31090 $MA_JOG_INCR_WEIGHT

Depending on the actual setting in MD10240 SCALING_SYSTEM_IS_METRIC, the control system automatically selects the appropriate value.

The user defines the two increment factors when the system is commissioned.

Example: Increment evaluations for the 1st axis

- Metric:
  MD31090 $MA_JOG_INCR_WEIGHT[0 ; AX1] = 0.001 mm
- Inch:
  MD31090 $MA_JOG_INCR_WEIGHT[1 ; AX1] = 0.00254 mm ≙ 0.0001 inch

In this way, MD31090 does not have to be written at every inch/metric switchover.

Remaining distances are not accumulated during incremental traversing with JOG when the system of units is changed, since all internal positions always refer to mm.

System of units for sag compensation

The system of units for sag compensation is configured using machine data:

MD32711 $MA_CEC_SCALING_SYSTEM_METRIC

Further information:
Function Manual Monitoring and Compensating; Compensations

2.3.3 Programming

2.3.3.1 Switching over the system of units (G70/G71/G700/G710)

Using the commands of G group 13 (inch/metric system of units) within a part program, you can switch over between the metric and inch system of units.

Activation

In order that commands G700 and G710 are available, the extended system of units functionality must be switched on (MD10260 $MN_CONVERT_SCALING_SYSTEM = 1).

Syntax

```
G70
G71
G700
G710
```

Meaning

G70:	Activating the inch system of units The **inch system of units** is used to read and write **geometrical data in units of length**. **Technological data in units of length** (e.g. feedrates, tool offsets, adjustable work offsets, machine data and system variables) is read and written using the **parameterized basic system**.	
	G group:	13
	Initial setting:	Settable via MD20150 $MC_GCODE_RESET_VALUES
	Effectiveness:	Modal

G71:	Activating the metric system of units The **metric system of units** is used to read and write **geometrical data in units of length**. **Technological data in units of length** (e.g. feedrates, tool offsets, adjustable work offsets, machine data and system variables) is read and written using the **parameterized basic system**.	
	G group:	13
	Initial setting:	Settable via MD20150 $MC_GCODE_RESET_VALUES
	Effectiveness:	Modal
G700:	Activating the inch system of units All geometrical and technological data in units of length is read and written using the inch system of units.	
	G group:	13
	Initial setting:	Settable via MD20150 $MC_GCODE_RESET_VALUES
	Effectiveness:	Modal
G710:	Activating the metric system of units All geometrical and technological data in units of length is read and written using the metric system of units.	
	G group:	13
	Initial setting:	Settable via MD20150 $MC_GCODE_RESET_VALUES
	Effectiveness:	Modal

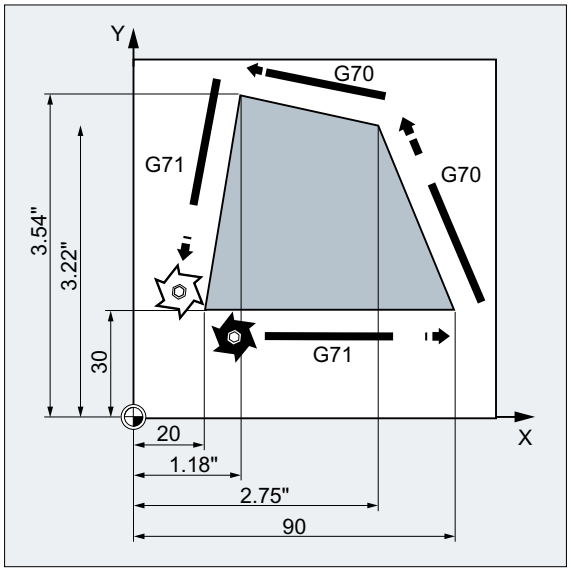
NOTICE

Axis-specific data of rotary axes

Axis-specific data of rotary axes is read and written using the parameterized basic system.

Example

The basic system is metric (MD10240 $MN_SCALING_SYSTEM_IS_METRIC = 1). However, the workpiece drawing has dimensions shown in inches. This is the reason why within the part program, the inch system of units is selected. After the inch dimensions have been processed, the metric system of units is again selected.



Program code	Comment
N10 G0 G90 X20 Y30 Z2 S2000 M3 T1	; X=20 mm, Y=30 mm, Z=2 mm, F=rapid traverse mm/min
N20 G1 Z-5 F500	; Z=-5 mm, F=500 mm/min
N30 X90	; X=90 mm
N40 **G70** X2.75 Y3.22	; programmed system of units: inch
	; X=2.75 inch, Y=3.22 inch, F=500 mm/min
N50 X1.18 Y3.54	; X=1.18 inch, Y=3.54 inch, F=500 mm/min
N60 **G71** X20 Y30	; programmed system of units: Metric
	; X=20 mm, Y=30 mm, F=500 mm/min
N70 G0 Z2	; Z=2 mm, F=rapid traverse mm/min
N80 M30	; end of program

Further information

Reading and writing data in the case of G70/G71 and G700/G710

Data area	G70 / G71		G700 / G710	
	Read	Write	Read	Write
Display, decimal places (WCS)	P	P	P	P
Display, decimal places (MCS)	G	G	G	G
Feedrates	G	G	P	P
Position data X, Y, Z	P	P	P	P

Data area	G70 / G71		G700 / G710	
	Read	Write	Read	Write
Interpolation parameters I, J, K	P	P	P	P
Circle radius (CR)	P	P	P	P
Polar radius (RP)	P	P	P	P
Thread pitch	P	P	P	P
Programmable FRAME	P	P	P	P
Settable FRAMES	G	G	P	P
Basic frames	G	G	P	P
External work offsets	G	G	P	P
Axial preset offset	G	G	P	P
Working area limits (G25/G26)	G	G	P	P
Protection areas	P	P	P	P
Tool offsets	G	G	P	P
Length-related machine data	G	G	P	P
Length-related setting data	G	G	P	P
Length-related system variables	G	G	P	P
GUDs	G	G	G	G
LUDs	G	G	G	G
PUDs	G	G	G	G
R parameters	G	G	G	G
Siemens cycles	P	P	P	P
Jog/handwheel increment factor	G	G	G	G
P: Writing/reading is performed in the programmed system of units. G: Writing/reading is performed in the configured basic system				

Synchronized actions

Note

Reading position data in synchronized actions

If a system of units has not been explicitly programmed in the synchronized action (condition component and/or action component) **length-related position data** in the synchronized action will always be read in the parameterized **basic system**.

Further information: Function Manual, Synchronized Actions

2.4 Setpoint/actual-value system

2.4.1 General information

Control loop

A control loop with the following structure can be configured for every closed-loop controlled axis/spindle:

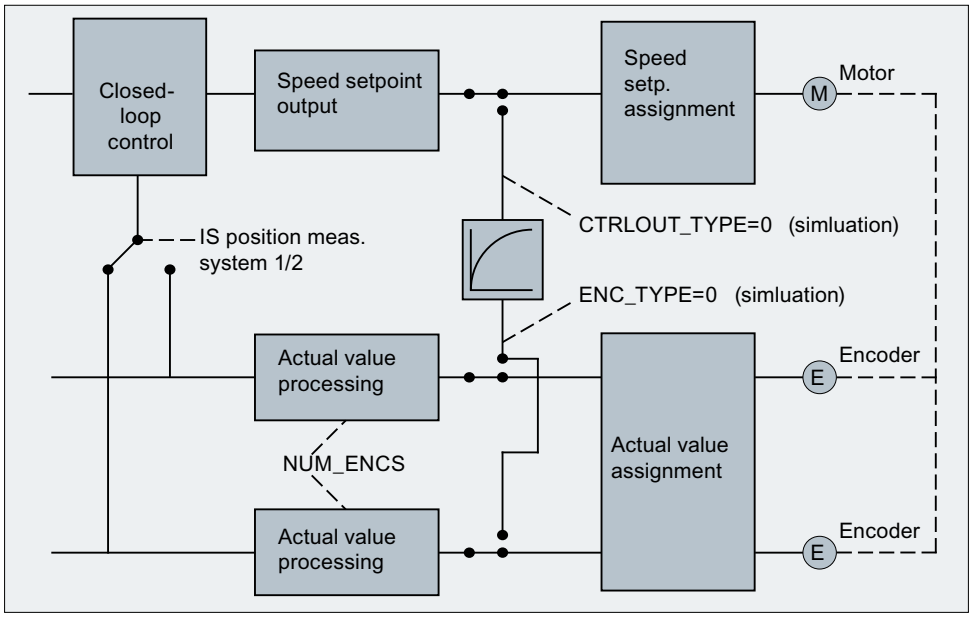


Figure 2-1 Block diagram of a control loop

Setpoint output

A setpoint telegram can be output for each axis/spindle. The setpoint output to the regulator is done on the controller.

Actual-value acquisition

A maximum of two measuring systems can be connected for each axis/spindle, e.g. a direct measuring system for machining processes with high accuracy requirements and an indirect measuring system for high-speed positioning tasks.

The number of encoders used is recorded in the machine data:

MD30200 $MA_NUM_ENCS (number of encoders)

In the case of two actual-value branches, the actual value is acquired for both branches.

The active measuring system is always used for position control, absolute value calculation and display. If both measuring systems are activated at the same time by the PLC interface, positioning measuring system 1 is chosen internally by the controller.

Reference point approach is executed by the selected measuring system.

Every position measuring system must be referenced separately.

For explanations of encoder monitoring, see Function Manual *Monitoring and Compensations*, Section *Axis monitoring*.

For explanations of compensation functions for actual value acquisition, see:
Further information:
Function Manual Monitoring and Compensating; Compensations

Switching between measuring systems

One can switch between the two measuring systems through the following NC/PLC interface signals:

DB31, ... DBX1.5 (position measuring system 1)

DB31, ... DBX1.6 (position measuring system 2)

Further information

See Function Manual Basic Functions; Various NC/PLC interface signals and functions.

It is possible to switch over measuring systems at any time, the axes do not have to be stationary to do this. Switchover only takes place if a permissible deviation between the actual values and the two measuring systems has not been violated.

The associated tolerance is entered in the machine data:

MD36500 $MA_ENC_CHANGE_TOL (max. tolerance on position actual value switchover)

On switchover, the current difference between position measuring system 1 and 2 is traversed immediately.

Monitoring

The permissible deviation between the actual values of the two measuring systems is to be entered in the machine data:

MD36510 $MA_ENC_DIFF_TOL

For the cyclic comparison of the two measuring systems used, this difference must not be exceeded, as otherwise Alarm 25105 "Measuring systems deviate" is generated.

If the axis is not referenced (at least in the current control measuring system), then the related monitoring is not active if MD36510 = 0 or if neither of the two measuring systems in the axis is active/available.

Types of actual-value acquisition

The used encoder type must be defined through the following machine data:

MD30240 $MA_ENC_TYPE (type of actual-value acquisition (actual position value))

Simulation axes

The speed control loop of an axis can be simulated for test purposes.

The axis "traverses" with a following error, similar to a real axis.

A simulation axis is defined by setting the two following machine data to "0":

MD30130 $MA_CTRLOUT_TYPE[n] (output value of setpoint)

MD30240 $MA_ENC_TYPE[n] (type of actual-value acquisition)

As soon as the standard machine data has been loaded, the axes become simulation axes.

The setpoint and actual value can be set to the reference point value with reference point approach.

The machine data:

MD30350 $MA_SIMU_AX_VDI_OUTPUT (output of axis signals with simulation axes)

can be used to define whether the axis-specific interface signals are to be output on the PLC during the simulation.

Actual-value correction

If actual-value corrections performed by the NC on the encoder selected for position control do not influence the actual value of another encoder defined in the same axis, then this encoder is to be declared as "independent" via the following machine data:

MD30242 $MA_ENC_IS_INDEPENDENT

Actual-value corrections include the following:

- Modulo treatment
- Reference point approach
- Measuring system comparison
- PRESET

2.4.2 Setpoint and encoder assignment

Setpoint marshalling

The following machine data is relevant for the setpoint assignment of a machine axis.

MD30100		$MA_CTRLOUT_SEGMENT_NR
Setpoint assignment: Bus segment		
System	Value	Meaning
	5	PROFIBUS DP / PROFINET (default)

MD30110		$MA_CTRLOUT_MODULE_NR
Setpoint assignment: Drive number / module number		
System	Value	Meaning
	x	The logical I/O address of the drive is assigned from MD13050 $MN_DRIVE_LOGIC_ADDRESS[n] via the drive number. The drive number (x) results from the index (n) of MD13050: x = n + 1 **Note** The machine data is of no significance if the drive is simulated (MD30130 $MA_CTRLOUT_TYPE = 0).

MD30120		$MA_CTRLOUT_NR
Setpoint assignment: Setpoint output on drive module/module		
System	Value	Meaning
	1	Modular drive at PROFIBUS / PROFINET with PROFIdrive profile (default)

MD30130		$MA_CTRLOUT_TYPE
Setpoint output type		
System	Value	Meaning
	0	Simulation (operation without drive)
	1	Setpoint output active

Encoder assignment

The following machine data is relevant for assigning the encoder information of the drive - transferred in the PROFIdrive telegram - to the encoder inputs of the machine axis:

MD30210		$MA_ENC_SEGMENT_NR[n]
Actual value assignment, bus segment		
System	Value	Meaning
	5	PROFIBUS DP / PROFINET

MD30220		$MA_ENC_MODULE_NR[n]
Actual value assignment: Drive number/measuring circuit number		
System	Value	Meaning
	x	The logical I/O address of the drive is assigned from MD13050 $MN_DRIVE_LOGIC_ADDRESS[n] via the drive number. The drive number (x) results from the index (n) of MD13050: x = n + 1

MD30230		$MA_ENC_INPUT_NR[n]
Actual value assignment: Input on drive module/measuring circuit module		
System	Value	Meaning
	x	Number of the encoder interface within the PROFIdrive telegram Examples PROFIdrive telegram 103 x = 1 → 1st encoder interface (G1_ZSW, G1_XIST1, G1_XIST2) x = 2 → 2nd encoder interface (G2_ZSW, G2_XIST1, G2_XIST2) PROFIdrive telegram 118 x = 1 → 1st encoder interface (G2_ZSW, G2_XIST1, G2_XIST2) x = 2 → 2nd encoder interface (G3_ZSW, G3_XIST1, G3_XIST2) **Note:** For SINAMICS S120: - Encoder 1 (G1_...): Motor encoder - Encoder 2 (G2_...): Direct measuring system - Encoder 3 (G3_...): Additional measuring system

MD30240		$MA_ENC_TYPE[n]
Encoder type of the actual value acquisition (actual position value)		
System	Value	Meaning
	0	Simulation (operation without encoder)
	1	Incremental encoder
	4	Absolute encoder
Note: Corresponds to PROFIdrive parameter p979		

MD30242		$MA_ENC_IS_INDEPENDENT[n, axis]
Encoder is independent		
System	Value	Meaning
	0	The encoder is not independent.
	1	The encoder is independent. If the actual-value corrections, which are made for the encoder selected for the position control, are not to influence the actual value of the second encoder defined in the same axis, then this should be declared as independent. Actual value corrections are: • - Modulo treatment • - Reference point approach • - Measuring system alignment • - PRESET Example: One axis, two encoders, the 2nd encoder is independent MD30200 $MA_NUM_ENCS[AX1] = 2 MD30242 $MA_ENC_IS_INDEPENDENT[0, AX1] = 0 MD30242 $MA_ENC_IS_INDEPENDENT[1, AX1] = 1 Selection, position measuring system 1 / 2: DB31.DBX1.5 / 1.6 If encoder 1 is selected for closed-loop position control, then the actual value corrections are only performed on this encoder, as encoder 2 is independent. If encoder 2 is selected for position control, then the actual value corrections are performed on both encoders, as encoder 1 is not independent. This means that the machine data only has an effect on the passive encoder of a machine axis.
	2	The passive encoder is dependent. The actual encoder value is changed by the active encoder. In combination with MD35102 $MA_REFP_SYNC_ENCS = 1, for reference point approach, the passive encoder is aligned to the active encoder - but is NOT referenced. In the referencing mode MD34200 $MA_ENC_REFP_MODE = 3 (distance-coded reference marks) the passive encoder is automatically referenced with the next traversing motion after passing the zero mark distance. This is done independent of the actual operating mode setting.
	3	The encoder is independent. For modulo rotary axes, modulo actual value corrections are also performed in the passive encoder.

Note

Machine data index [n]

The machine data index [n] for encoder assignment has the following meaning:

- n = 0: First encoder assigned to the machine axis
- n = 1: Second encoder assigned to the machine axis

The assignment is made using machine data:

- MD30220 $MA_ENC_MODULE_NR[n]
- MD30230 $MA_ENC_INPUT_NR[n]

Further information

Commissioning Manual, CNC Commissioning: NC, PLC, drive; communication between NC and drive > Drives: Assign axis

2.4.3 Adapting the motor/load ratios

Gear types

The following gear types are available for adapting the mechanical ratios:

Gear type	Activation	Adaptation	Installation location
Motor/load gear	Parameter set	Fixed configuration	Gear unit
Measuring gear encoder	Power On	Sensor-dependent	Sensor-side
Load intermediate gear unit	Warm restart	Load-dependent	Tool-side

Local position of gear unit / encoder

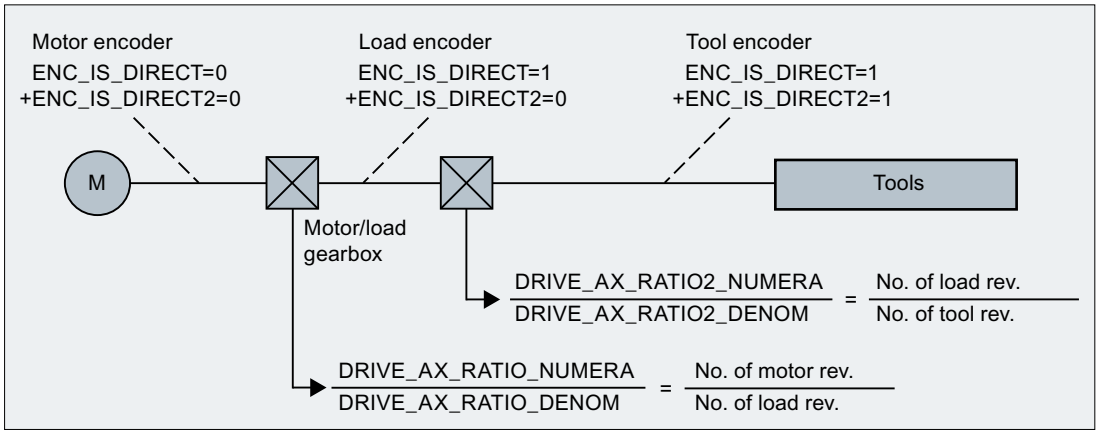


Figure 2-2 Gear unit types and encoder locations

Motor/load gear

The motor/load gear supported by SINUMERIK is configured via the following machine data:

- MD31060 $MA_DRIVE_AX_RATIO_NUMERA (numerator load gear)
- MD31050 $MA_DRIVE_AX_RATIO_DENOM (denominator load gear)

The transmission ratio is obtained from the numerator/denominator ratio of both machine data. The associated parameter sets are used automatically as default by the control system to synchronize the position controller with the relevant transmission ratios.

Since a gear stage change is not always carried out automatically, and there are also several ways to request a gear stage change, the position controller is not always incorporated via parameter sets.

Note

For further information about the parameter sets for gear stage change, see Section "S1: Spindles (Page 733)".

Intermediate gear

Additional, configurable load intermediate gears are also supported by the control system:

- MD31066 $MA_DRIVE_AX_RATIO2_NUMERA (intermediate gear numerator)
- MD31064 $MA_DRIVE_AX_RATIO2_DENOM (intermediate gear denominator)

Power tools generally have their "own" intermediate gear. Such variable mechanics can be configured by multiplying the active intermediate gear and the motor/load gear.

⚠ CAUTION

Different gear transmission ratios for switching

Unlike the motor/load gear, there is no parameter set for the intermediate gear and, therefore, no way of controlling the time-synchronized switchover to the part program or PLC (NC/PLC interface). Part machining during gear change is, therefore, ruled out. It remains the task of the user to match the synchronization of the relevant changed machine data to the corresponding mechanical switchover and activate it. On switchover during a motion, compensations **cannot** be ruled out due to jumps in the scaling factors. These are not monitored for violation of the maximum acceleration.

Encoder directly at the tool

Another connection option is possible for a "tool-side encoder" on the intermediate gear by configuring the following machine data:

MD31044 $MA_ENC_IS_DIRECT2

Encoder not directly at the tool

The following supplementary conditions apply to a gear change of the intermediate gear in position-control mode: The gear ratio to be changed is incorporated in a re-scaling of the encoder information in this case.

In this case, the following applies to axes/spindles in positioning mode:

- A non-abrupt gear change is **only possible at zero speed**.
  To do this, the tool-side position before and after a gear change are set equal for a change in the ratio, since the mechanical position does not (or hardly) change during a gear stage change.
  Recommendation:
  To avoid 21612 "Controller enable reset during motion", changeover should be carried out "only at zero speed". It is still permissible and expedient to switch the axis or spindle to speed-control or follow-up mode before or during a gear change.

Boundary conditions

If the encoder to be used for position control is **connected directly at the tool**, the gear stage change only affects the physical quantities at the speed interface between the NC and the drive of the motor/load gear. The internal parameter sets are not changed.

Reference point and machine reference

⚠ CAUTION
Loss of machine reference The control cannot detect all situations that can lead to loss of the machine reference. Therefore, it is the general responsibility of the commissioning engineer or user to initiate explicit referencing of zero marker synchronization in such cases.

In the case of gear changes, it is not possible to make a statement about the effect of the reference point or machine position reference on the encoder scaling. In such cases, the control partially cancels the status "Axis referenced/synchronized".

If the machine reference has been lost, it must first be restored through an adjustment or referencing of the lost machine reference.

See also

R1: Referencing (Page 291)

2.4.4 Speed setpoint output

Control direction and travel direction of the feed axes

You must determine the travel direction of the feed axis before starting work.

Control direction

Before the position control is started up, the speed controller and current controller of the drive must be started up and optimized.

Traversing direction

With the machine data:

MD32100 $MA_AX_MOTION_DIR (traversing direction)

the direction of motion of the axis can be reversed

without affecting the control direction of the position control.

Speed setpoint adjustment

In the case of speed setpoint comparison, the NC is informed which speed setpoint corresponds to which motor speed in the drive, for parameterizing the axial control and monitoring. This comparison is carried out automatically.

For PROFIBUS DP drives, alternatively, the manual speed setpoint comparison is also possible.

- **Manual adjustment**
  In the machine data:
  MD32250 $MA_RATED_OUTVAL
  a value not equal to zero is entered.

Note

Velocity adjustment and maximum speed setpoint

No speed compensation is required due to the automatic speed setpoint adjustment!

Maximum speed setpoint

The maximum speed setpoint is defined in percent. 100% means maximum speed setpoint or maximum speed for PROFIdrive drives (manufacturer-specific setting parameters in the drive, e.g. p1082 for SINAMICS).

The output of the spindle speed is implemented in the NC.

Data for five gear stages are realized in the controller.

These stages are defined by a minimum and maximum speed for the stage itself and by a minimum and maximum speed for the automatic gear stage changeover. A new set gear stage is output only if the new programmed speed cannot be traversed in the current gear stage.

With the machine data:

MD36210 $MA_CTRLOUT_LIMIT[n] (maximum speed setpoint)

the speed setpoint is restricted percentage-wise.

Values up to 200% are possible.

When the speed is exceeded, an alarm is generated.

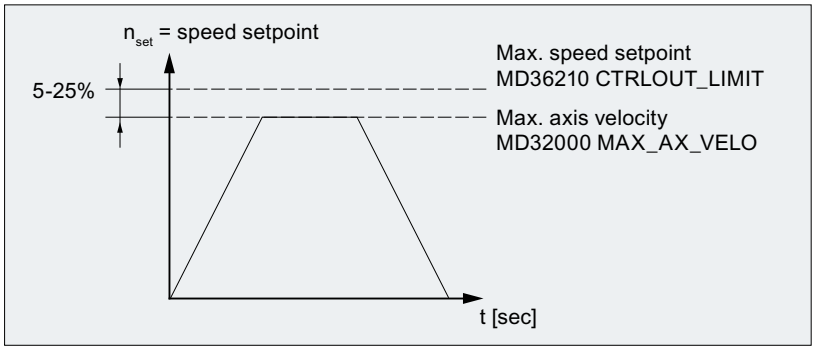


Figure 2-3 Maximum speed setpoint

However, due to control processes, the axes should not reach their maximum velocity (MD32000 $MA_MAX_AX_VELO) at 100% of the speed setpoint, but at 80% to 95%.

In case of axes, whose maximum speed is attained at around 80% of the speed setpoint range, the default value (80%) of the machine data:

MD32000 $MA_MAX_AX_VELO (maximum axis velocity)

can be adopted.

2.4.5 Machine data of the actual value system

Axis-specific machine data

To parameterize the actual value system, the following axis-specific machine data should be set:

Encoder and parameter set-independent machine data: $MA_	Meaning
MD30200 NUM_ENCS	Number of encoders
MD30300 IS_ROT_AX	Rotary axis / spindle
MD30310 ROT_IS_MODULO	Modulo conversion for rotary axis / spindle
MD30320 DISPLAY_IS_MODULO	360-degree modulo display for rotary axis or spindle
MD30330 MODULO_RANGE	Size of the modulo range
MD30340 MODULO_RANGE_START	Start position of the modulo range
MD31030 $MA_LEADSCREW_PITCH	Leadscrew pitch
MD31064 $MA_DRIVE_AX_RATIO2_DENOM	Intermediate gear denominator
MD31066 $MA_DRIVE_AX_RATIO2_NUMERA	Intermediate gear numerator
MD32000 $MA_MAX_AX_VELO	Maximum axis velocity

Encoder-dependent machine data: $MA_	Meaning
MD30210 ENC_SEGMENT_NR[n]	Actual value assignment: Number of bus segments
MD30220 ENC_MODULE_NR[n]	Actual value assignment: Drive number/ measuring circuit number
MD30230 ENC_INPUT_NR[n]	Actual value assignment: Input on drive module/measuring circuit module
MD30240 ENC_TYPE[n]	Encoder type of the actual value acquisition (actual position value)
MD30242 ENC_IS_INDEPENDENT[n]	Encoder is independent
MD30244 ENC_MEAS_TYPE[n]	Encoder measuring type
MD30250 ACT_POS_ABS[n]	Internal encoder position
MD30260 ABS_INC_RATIO[n]	Absolute encoder: Ratio between the absolute resolution and the incremental resolution
MD30270 ENC_ABS_BUFFERING[n]	Absolute encoder: Traversing range extension
MD34090 $MA_REFP_MOVE_DIST_CORR[n]	Reference point offset
MD34320 $MA_ENC_INVERS[n]	Length measuring system is in the opposite sense
n: Encoder index, with n = 0, 1, ... (1st encoder, 2nd encoder, ...)	

Note

The "Activate machine data" can be activated either in the part program with the command `NEWCONF` or via the user interface by pressing a softkey.

2.4.6 Actual-value resolution

2.4.6.1 Machine data of the actual value resolution

Depending where the measuring system (encoder) is mounted, the following measuring system types should be taken into account:

- Load-side encoder: Direct measuring system (DM)
- Motor-side encoder: Indirect measuring system (IM)

Parameterizing the actual value resolution depending on the axis type (linear/rotary axis)

The control system calculates the actual value resolution based on the following machine data.

Machine data for calculating the actual value resolution	Linear axis	Linear axis		Rotary axis	
	Linear scale / direct measuring system	Indirect measuring system	Direct measuring system: Machine/ tool	Indirect measuring system	Direct measuring system: Machine/tool
MD30300 $MA_IS_ROT_AX	0	0	0	1	1
MD31000 $MA_ENC_IS_LINEAR[n]	1	0	0	0	0
MD31010 $MA_ENC_GRID_POINT_DIST[n]	Spacing	-	-	-	-
MD34320 $MA_ENC_INVERS[n]	1)	-	-	-	-
MD31020 $MA_ENC_RESOL[n]	-	Pulses/ rev	Pulses/ rev	Pulses/ rev	Pulses/ rev
MD31025 $MA_ENC_PULSE_MULT[n]	Encoder multiplication				
MD31030 $MA_LEADSCREW_PITCH	-	mm/rev.	mm/rev.	-	-
MD31040 $MA_ENC_IS_DIRECT[n]	- / 1	0	1	0	1
MD31044 $MA_ENC_IS_DIRECT2[n]	- / 1	0	1	0	1
MD31050 $MA_DRIVE_AX_RATIO_DENOM[n]	-	Load rev.	- / 1	Load rev.	2)
MD31060 $MA_DRIVE_AX_RATIO_NUMERA[n]	-	Motor rev. if infeed gear available	- / 1	Motor rev.	2)
MD31070 $MA_DRIVE_ENC_RATIO_DENOM[n]	-	Encoder rev.	Encoder rev.	Encoder rev.	Encoder rev.
MD31080 $MA_DRIVE_ENC_RATIO_NUMERA[n]	-	Motor-side encoder 3)	Motor rev.	Motor rev.	Load rev.

- Not relevant

1) For distance-coded measuring systems

2) These machine data are not required for encoder matching (path evaluation). However, they must be entered correctly for the setpoint calculation! Otherwise the required servo gain factor (K_V) will not be set. The load revolutions are entered into machine data MD31050 $MA_DRIVE_AX_RATIO_DENOM and the motor revolutions in machine data MD31060 $MA_DRIVE_AX_RATIO_NUMERA.

3) The encoder on the motor side is a built-in encoder and, therefore, does **not** have a measuring gear unit.
The transmission ratio is always 1:1.

Machine data of the actual value resolution

The actual-value resolution results from the design of the machine, whether gearboxes are available and their gear ratio, the leadscrew pitch for linear axes and the resolution of the encoder being used. The following machine data must be set for this on the control system:

Encoder and parameter set-independent machine data: $MA_	Meaning
MD30300 IS_ROT_AX	Axis is a rotary axis / spindle
MD31010 ENC_GRID_POINT_DIST	Distance between reference marks of the linear scale
MD31030 LEADSCREW_PITCH	Leadscrew pitch
MD31064 DRIVE_AX_RATIO2_DENOM	Denominator of attached gearbox
MD31066 DRIVE_AX_RATIO2_NUMERA	Numerator of attached gearbox

Encoder-dependent machine data: $MA_	Meaning
MD31000 ENC_IS_LINEAR[n]	Measuring system is a linear scale
MD31020 ENC_RESOL[n]	Encoder pulses per revolution
MD31025 ENC_PULSE_MULT[n]	Encoder multiplication (high resolution)
MD31040 ENC_IS_DIRECT[n]	Direct or indirect measuring system
MD31044 ENC_IS_DIRECT2[n]	Encoders installed at the attached gearbox
MD31070 DRIVE_ENC_RATIO_DENOM[n]	Measuring gearbox denominator
MD31080 DRIVE_ENC_RATIO_NUMERA[n]	Measuring gearbox numerator
n: Encoder index, with n = 0, 1, ... (1st encoder, 2nd encoder, etc.)	

Parameter-set-dependent machine data: $MA_	Meaning
MD31050 $MA_DRIVE_AX_RATIO_DENOM[m]	Denominator load gearbox
MD31060 $MA_DRIVE_AX_RATIO_NUMERA[m]	Numerator load gearbox
m: Parameter set index, with m = 0, 1, ... (1st parameter set, 2nd parameter set, etc.)	

Parameterizing the computational resolution

The ratio of control-internal computational resolution to the actual-value resolution is an indication of how exactly the values calculated by the control system can be implemented on the machine.

Computational resolution: Linear axes

$$\frac{\text{Computational resolution}}{\text{Actual-value resolution}} = \frac{\text{Internal increments / (mm)}}{\text{Encoder increments / (mm)}}$$

Computational resolution: Rotary axes

$$\frac{\text{Computational resolution}}{\text{Actual-value resolution}} = \frac{\text{Internal increments / (degree)}}{\text{Encoder increments / (degree)}}$$

Machine data

General machine data: $MN_	Meaning
MD10200 INT_INCR_PER_MM	Computational resolution for linear positions
MD10210 INT_INCR_PER_DEG	Computational resolution for angular positions

Recommended setting

The above components and settings that are responsible for the actual-value resolution, should be selected so that the actual-value resolution is higher than the parameterized computational resolution.

$$\frac{\text{Computational resolution}}{\text{Actual-value resolution}} \leq 1$$

2.4.6.2 Example: Linear axis with linear scale

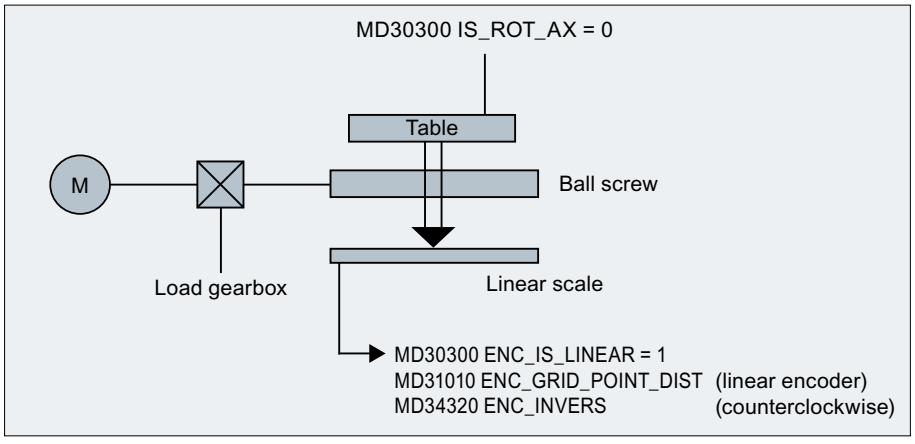


Figure 2-4 Linear axis with linear scale

The ratio of the internal increments to the encoder increments per mm is calculated as follows:

$$\frac{\text{Internal increments/mm}}{\text{Encoder increments/mm}} = \frac{\text{ENC_GRID_POINT_DIST [n]} * \text{INT_INCR_PER_MM}}{\text{ENC_PULSE_MULT[n]}}$$

2.4.6.3 Example: Linear axis with rotary encoder on motor

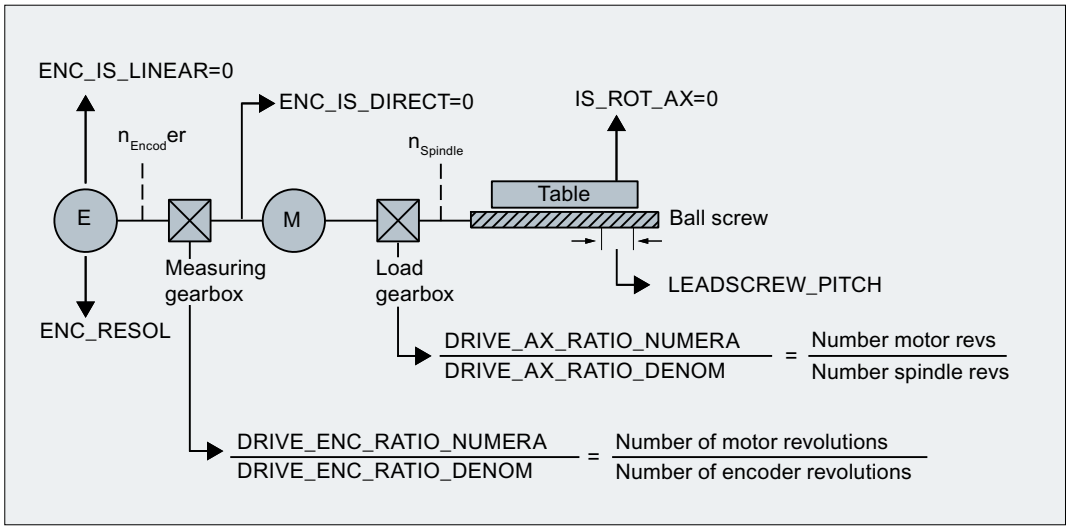


Figure 2-5 Linear axis with rotary encoder on motor

The ratio of the internal increments to the encoder increments per mm is calculated as follows:

$$\frac{\text{Internal increments/mm}}{\text{Encoder increments/mm}} = \frac{1}{\text{ENC_RESOL [n]} * \text{ENC_PULSE_MULT[n]}} * \frac{\text{DRIVE_ENC_RATIO_NUMERA [n]}}{\text{DRIVE_ENC_RATIO_DENOM [n]}} * \frac{\text{DRIVE_AX_RATIO_DENOM [n]}}{\text{DRIVE_AX_RATIO_NUMERA [n]}} * \text{LEADSCREW_PITCH} * \text{INT_INCR_PER_MM}$$

Example

Assumptions:

- Rotary encoder on the motor: 2048 pulses/revolution
- Internal pulse multiplication: 2048
- Gearbox, motor / ball screw: 5:1
- Leadscrew pitch: 10 mm/revolution
- Computational resolution: 10000 increments per mm

Machine data	Value
MD30300 $MA_IS_ROT_AX	0
MD31000 $MA_ENC_IS_LINEAR[0]	0
MD31040 $MA_ENC_IS_DIRECT[0]	0
MD31020 $MA_ENC_RESOL[0]	2048
MD31025 $MA_ENC_PULSE_MULT	2048
MD31030 $MA_LEADSCREW_PITCH	10
MD31080 $MA_DRIVE_ENC_RATIO_NUMERA[0]	1
MD31070 $MA_DRIVE_ENC_RATIO_DENOM[0]	1
MD31060 $MA_DRIVE_AX_RATIO_NUMERA[0]	5
MD31050 $MA_DRIVE_AX_RATIO_DENOM[0]	1
MD10210 $MN_INT_INCR_PER_DEG	10000

$$\rightarrow \frac{\text{Internal increments/mm}}{\text{Encoder increments/mm}} = \frac{1}{2048 * 2048} * \frac{1}{1} * \frac{1}{5} * 10\ \text{mm}$$

$$*10000\ \text{incr./mm} = 0.004768$$

An encoder increment corresponds to 0.004768 internal increments or 209.731543 encoder increments correspond to an internal increment.

2.4.6.4 Example: Linear axis with rotary encoder on the machine

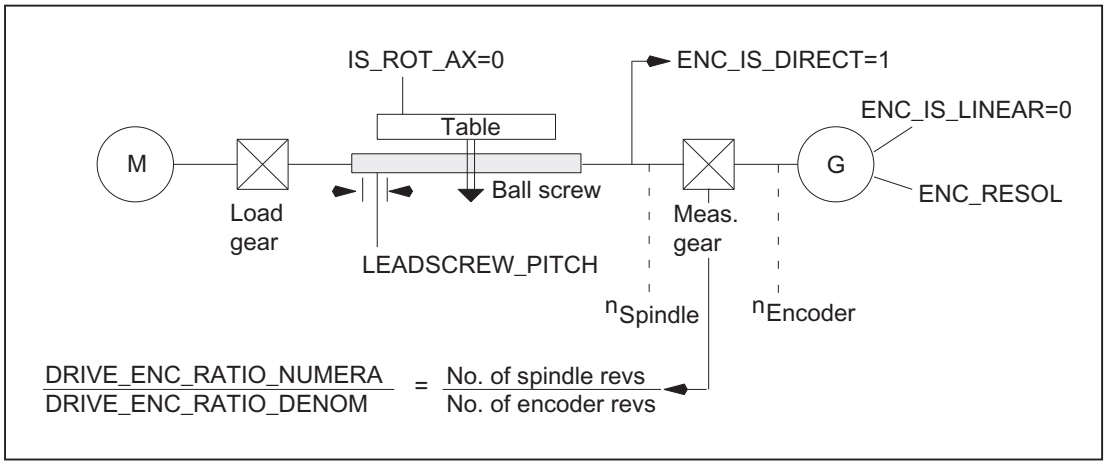


Figure 2-6 Linear axis with rotary encoder on the machine

The ratio of the internal increments to the encoder increments per mm is calculated as follows:

$$\frac{\text{Internal increments/mm}}{\text{Encoder increments/mm}} = \frac{1}{\text{ENC_RESOL [n] * ENC_PULSE_MULT[n]}} * \frac{\text{DRIVE_ENC_RATIO_NUMERA [n]}}{\text{DRIVE_ENC_RATIO_DENOM [n]}} * \text{LEADSCREW_PITCH} * \text{INT_INCR_PER_MM}$$

2.4.6.5 Example: Rotary axis with rotary encoder on motor

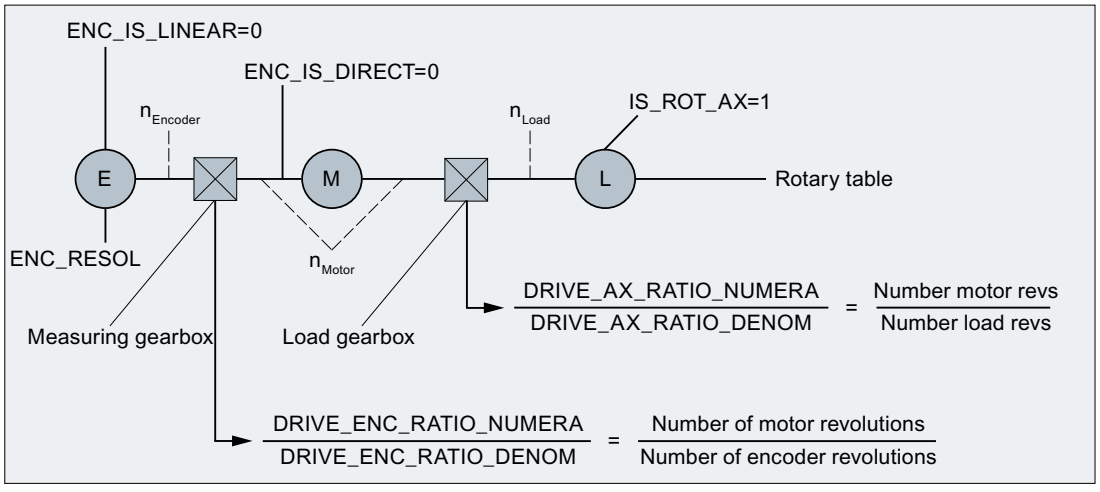


Figure 2-7 Rotary axis with rotary encoder on motor

The ratio of the internal increments to the encoder increments per degree is calculated as follows:

$$\frac{\text{Internal increments/degree}}{\text{Encoder increments/degree}} = \frac{360 \text{ degrees}}{\text{ENC_RESOL [n] * ENC_PULSE_MULT[n]}} * \frac{\text{DRIVE_ENC_RATIO_NUMERA [n]}}{\text{DRIVE_ENC_RATIO_DENOM [n]}} * \frac{\text{DRIVE_AX_RATIO_DENOM [n]}}{\text{DRIVE_AX_RATIO_NUMERA [n]}} * \text{INT_INCR_PER_DEG}$$

Example

Assumptions:

- Rotary encoder on the motor: 2048 pulses/revolution
- Internal pulse multiplication: 2048
- Gearbox, motor / rotary axis: 5:1
- Computational resolution: 1000 increments per degree

Machine data	Value
MD30300 $MA_IS_ROT_AX	1
MD31000 $MA_ENC_IS_LINEAR[0]	0
MD31040 $MA_ENC_IS_DIRECT[0]	0
MD31020 $MA_ENC_RESOL[0]	2048
MD31025 $MA_ENC_PULSE_MULT	2048
MD31080 $MA_DRIVE_ENC_RATIO_NUMERA[0]	1
MD31070 $MA_DRIVE_ENC_RATIO_DENOM[0]	1
MD31060 $MA_DRIVE_AX_RATIO_NUMERA[0]	5
MD31050 $MA_DRIVE_AX_RATIO_DENOM[0]	1
MD10210 $MN_INT_INCR_PER_DEG	1000

$$\rightarrow \frac{\text{Internal increments/degree}}{\text{Encoder increments/degree}} = \frac{360 \text{ degrees}}{2048 * 2048} * \frac{1}{1} * \frac{1}{5}$$

$$*1000 \text{ inc./degree} = 0{,}017166$$

An encoder increment corresponds to 0.017166 internal increments or 58.254689 encoder increments correspond to an internal increment.

2.4.6.6 Example: Rotary axis with rotary encoder on the machine

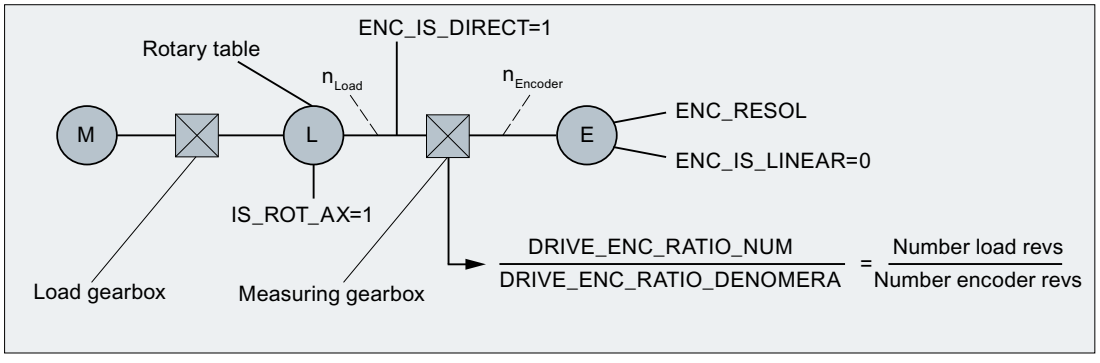


Figure 2-8 Rotary axis with rotary encoder on the machine

The ratio of the internal increments to the encoder increments per degree is calculated as follows:

$$\frac{\text{Internal increments/degree}}{\text{Encoder increments/degree}} = \frac{360 \text{ degrees}}{\text{ENC_RESOL [n]} * \text{ENC_PULSE_MULT[n]}} * \frac{\text{DRIVE_ENC_RATIO_NUMERA [n]}}{\text{DRIVE_ENC_RATIO_DENOM [n]}} * \text{INT_INCR_PER_DEG}$$

2.4.6.7 Example: Intermediate gear with encoder on the tool

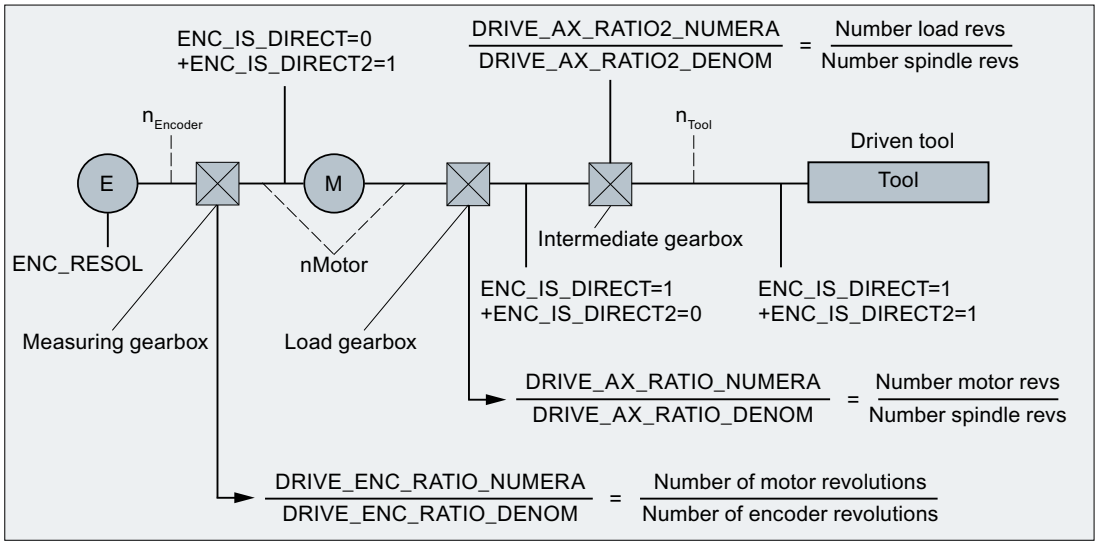


Figure 2-9 Intermediate gear with encoder directly on the rotating tool

The ratio of the internal increments to the encoder increments per degree is calculated as follows:

$$\frac{\text{Internal increments/degree}}{\text{Encoder increments/degree}} = \frac{360 \text{ degrees}}{\text{ENC_RESOL [n]} * \text{ENC_PULSE_MULT[n]}} * \frac{\text{DRIVE_ENC_RATIO_NUMERA [n]}}{\text{DRIVE_ENC_RATIO_DENOM [n]}} * \text{INT_INCR_PER_DEG}$$

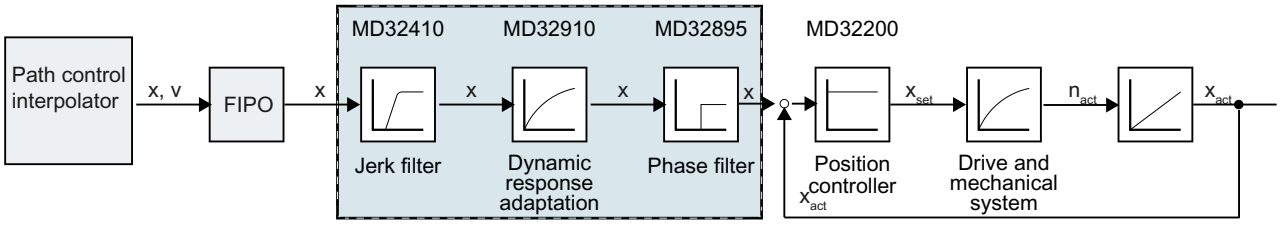
2.5 Closed-loop control

2.5.1 General information

Position control of an axis/spindle

The closed-loop control of an axis consists of the current and speed control loop of the drive plus a higher-level position control loop in the NC.

The basic structure of an axis/spindle position control is illustrated below:



MD32410	$MA_AX_JERK_TIME
MD32910	$MA_DYN_MATCH_TIME
MD32895	$MA_DESVAL_DELAY_TIME
MD32200	$MA_POSCTRL_GAIN
x	Position setpoint
v	Velocity setpoint
x_{act}	Position actual value
n_{act}	Speed actual value
n_{set}	Speed setpoint

Figure 2-10 Block diagram of the setpoint processing and closed-loop control

For information on the jerk limitation, see Section "B2: Acceleration and jerk (Page 347)".

For a description of the feedforward control, backlash, friction compensation, and leadscrew error compensation, see:

Further information
Function Manual Monitoring and Compensating; Compensations

Fine interpolation

Using the fine interpolator (FIPO), the contour precision can be further increased by reducing the staircase effect in the speed setpoint. You can set three different types of fine interpolation:

MD33000 $MA_FIPO_TYPE = <FIPO mode>

<FIPO mode>	Meaning
1	Differential fine interpolation with mean value generation (smoothing) over an IPO cycle
2	Cubic fine interpolation
3	Cubic fine interpolation optimized for use with the pre-control for the highest contour precision

K_V factor (servo gain)

In order that few contour deviations occur in the continuous-path mode, a high servo gain factor (K_V) is necessary:

MD32200 $MA_POSCTRL_GAIN[n]

However, if the servo gain factor (K_V) is too high, instability, overshoot and possibly impermissible high loads on the machine will result.

The maximum permissible servo gain factor (K_V) depends on the following:

- Design and dynamics of the drive
  (rise time, acceleration and braking capacity)
- Machine quality
  (elasticity, oscillation damping)
- Position control cycle or speed control cycle for active DSC

The servo gain factor (K_V) is defined as:

$$K_v = \frac{\text{Velocity}}{\text{Following error}} \; ; \; \frac{[\text{m/min}]}{[\text{mm}]}$$

Dynamic response adaptation

Axes that interpolate with one another, but with different servo gain factors (K_V) can be set to the same following error using the dynamic adaptation function. This allows an optimum contour accuracy to be achieved without loss of control quality by reducing the servo gain factors (K_V) to the dynamically weakest axis.

The function is activated via:

MD32900 $MA_DYN_MATCH_ENABLE = 1 (dynamic response adaptation)

The dynamic response adaptation is realized by entering a new equivalent time constant. It is calculated from the difference in the equivalent time constant of the dynamically weakest axis and the axis to be adapted:

MD32910 $MA_DYN_MATCH_TIME [n] = <difference in the equivalent time constant>

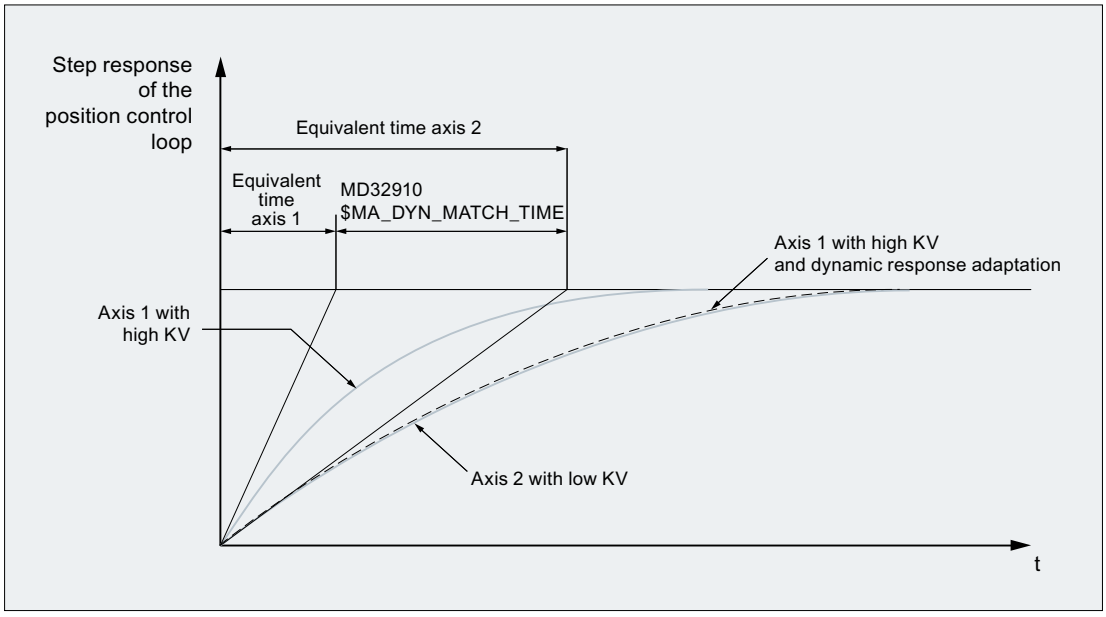


Figure 2-11 Dynamic response adaptation

Example of a dynamic response adaptation of three axes without speed feedforward control

The equivalent time constant of the position control loop is:	
Axis 1:	30 ms
Axis 2:	20 ms
Axis 3:	24 ms

With an equivalent time constant of 30 ms, axis 1 is the dynamically weakest axis.

This results in the following new equivalent time constants for the axes:

Axis 1: MD32910 $MA_DYN_MATCH_TIME = 0 ms
Axis 2: MD32910 $MA_DYN_MATCH_TIME = 30 ms - 20 ms = 10 ms
Axis 3: MD32910 $MA_DYN_MATCH_TIME = 30 ms - 24 ms = 6 ms

Approximation formulas for the equivalent time constant of the position control loop of an axis

The equivalent time constant T_{equiv} of the position control loop of an axis is approximately calculated depending on the type of feedforward control:

- Without feedforward control:

$$T_{Equiv} \approx \frac{1}{\text{MD32200 POSCTRL_GAIN [1/s]}}$$

- With speed feedforward control:

$$T_{Equiv} \approx \text{MD32810 EQUIV_SPEEDCTRL_TIME}$$

- For combined torque/speed feedforward control

$$T_{Equiv} \approx \text{MD32800 EQUIV_CURRCTRL_TIME}$$

Note

If dynamic response adaptation is realized for a geometry axis, then all other geometry axes must be set to the same dynamic response.

Further information
Commissioning Manual CNC: NC, PLC, drives

2.5.2 Switchable position setpoint filter circuits

Benefits

Two switchable position setpoint filter circuits allow the damping effect of the filters to be adapted to the specific machining type, and these can also be selected during machining. For example, filters that provide very high damping (Blackman filters) and very high axial jerk factors ($MA_MAX_AX_JERK, $MA_PATH_TRANS_JERK_LIM) can be used, e.g. for motion outside the workpiece or when roughing so that processing times can be significantly reduced. The relatively high contour deviation in conjunction with the powerful filters have an excessively high negative impact for these machining types.

It is advantageous to couple the filter switchover to the dynamic G code groups DYNNORM, DYNROUGH, DYNFINISH and DYNPREC and initiate it via a cycle (e.g. CUST_832). The status and values of the filter changeover can be queried using the following system variables.

- $VA_DESVAL_FILTERS_ACTIVE; status of the setpoint filter circuits
- $VA_DESVAL_FILTERS_DELAY_1/2; effective delay time of the 1st/2nd setpoint filter circuit
- $VA_DESVAL_FILTERS_DELTA_POS; position difference between the setpoint filter circuits.

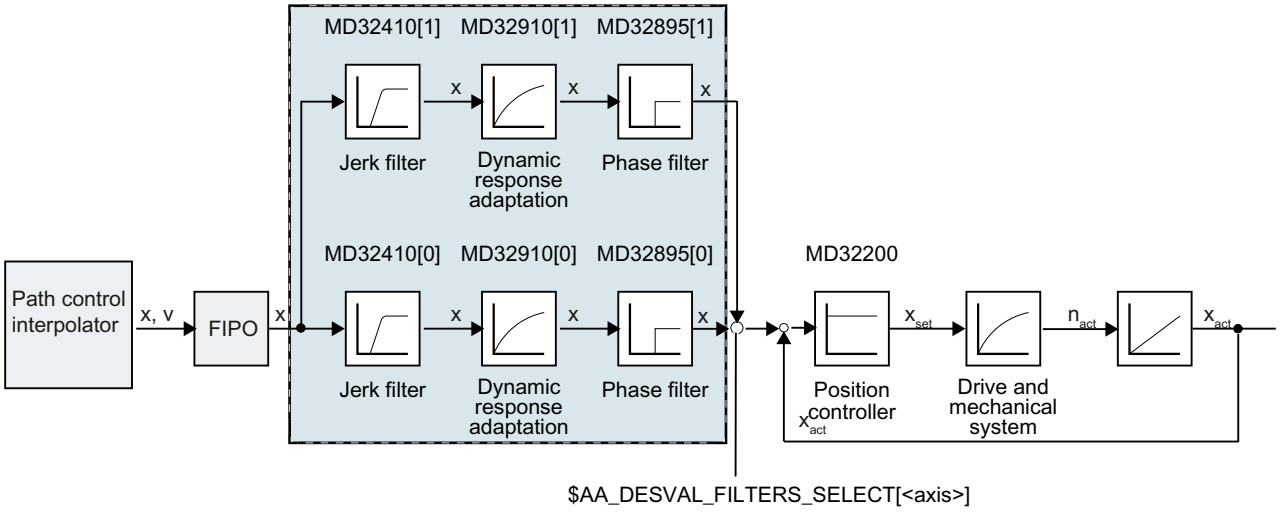
Option

You require the following options in order to use this function:

"Selectable filter circuits" (6FC5800-0xxy0-0YB0)

Function

Position control of an axis/spindle with switchable position setpoint filter circuit:



MD32410	$MA_AX_JERK_TIME
MD32910	$MA_DYN_MATCH_TIME
MD32895	$MA_DESVAL_DELAY_TIME
MD32200	$MA_POSCTRL_GAIN
x	Position setpoint
v	Velocity setpoint
x_{act}	Position actual value
n_{act}	Speed actual value
n_{set}	Speed setpoint

Figure 2-12 Switchable position setpoint filter circuits

Selection and/or preassignment of filter types for every filter circuit

The switchable position setpoint filter circuits are activated by entering a two-digit value in machine data MD32402 $MA_AX_JERK_MODE (filter type for axial jerk limiting). This allows two alternative jerk filters to be selected.

Parameterization

Select two filter types in order to switch between the two filter circuits from the part program:

- The jerk filter type of the first filter circuit is selected in the "Ones position" of machine data MD32402 AX_JERK_MODE, and the associated filter settings are set in index 0 of the machine data to parameterize the filter.
- The jerk filter type of the second filter circuit is selected in the "Tens position" of machine data MD32402 AX_JERK_MODE, and the associated filter settings are set in index 1 of the machine data to parameterize the filter.
- The following machine data has two indices:
  - MD32402 AX_JERK_FIR_FREQ
  - MD32408 AX_JERK_FIR_ORDER
  - MD32409 AX_JERK_FIR_WINDOW
  - MD32410 AX_JERK_TIME
  - MD32411 AX_JERK_TIME_ADD
  - MD32412 AX_JERK_FREQ
  - MD3414 AX_JERK_DAMP

Boundary conditions

Filter circuits can only be parameterized if the following conditions are satisfied:

- MD33000 $MA_FIPO_TYPE = 1 or 3
- MD32220 $MA_POSCTRL_INTEGR_ENABLE = 0
- MD32930 $MA_POSCTRL_OUT_FILTER_ENABLE = 0
- MD32610 $MA_VELO_FFW_WEIGHT = 1.0
- MD32620 $MA_FFW_MODE = 0, 3 or 4
- MD32648 $MA_SPLINES_CONTROL_CONFIG = 0

2.5.3 Commissioning

Switching over filter circuits

Select two filter types to preassigned filter types for the filter circuits. To switch over the parameterized filter circuits, use the system variable $AA_DESVAL_FILTERS_SELECT[<axis>].

Use the following system variables to interrogate the status of the filter circuits:

- $VA_DESVAL_FILTERS_ACTIVE - to determine the currently active filter circuit.
- $VA_DESVAL_FILTERS_DELTA_POS indicates the actual position difference between the filter circuits in order to determine a suitable switchover point. Suitable switchover points include motion with Constant velocity (acceleration = 0) - or when an axis is at a standstill, in which the position difference is small. If a switchover is made while an axis is accelerating, then an acceleration step (jump) can occur, which under certain circumstances can trigger the contour monitoring function. The switchover is made immediately when requested.
- You can determine the effective delay time of the first and second filter circuit using the following variables:
  $VA_DESVAL_FILTERS_DELAY_1
  $VA_DESVAL_FILTERS_DELAY_2

Example

Two switchover position setpoint filter circuits are parameterized with the following attributes using MD32402 $MA_AX_JERK_MODE = 25:

- With a value of 5, a position setpoint filter, filter type 5 = FIR lowpass is parameterized in the first position setpoint filter circuit.
- With a value of 20, a position setpoint filter, filter type 2 = moving average calculation parameterized in the second position setpoint filter circuit.
- To approximate the total delay times of the two filter circuits to each other, use MD32895 $MA_DESVAL_DELAY_TIME.
-

2.5.4 Parameter sets of the position controller

Six parameter sets per machine axis are available to quickly adapt the position control to the changed machine properties during operation, e.g. a gear change of the spindle, or to adjust the dynamic response to another axis, e.g. during tapping.

Machine data

A parameter set comprises the following machine data:

Number	Identifier $MA_	Meaning
31050	DRIVE_AX_RATIO_DENOM	Denominator load gearbox
31060	DRIVE_AX_RATIO_NUMERA	Numerator load gearbox
32200	POSCTRL_GAIN	Servo gain factor (K_V)
32452	BACKLASH_FACTOR	Backlash compensation
32610	VELO_FFW_WEIGHT	Feedforward control factor
36012	STOP_LIMIT_FACTOR	Exact stop coarse/fine factor and zero speed
32800	EQUIV_CURRCTRL_TIME	Equivalent time constant, current control loop for feed-forward control

Number	Identifier $MA_	Meaning
32810	EQUIV_SPEEDCTRL_TIME	Equivalent time constant, speed control loop for feed forward control
32910	DYN_MATCH_TIME	Time constant for dynamic response adaptation
36200	AX_VELO_LIMIT	Threshold value for velocity monitoring

Tapping, thread cutting

For tapping or thread cutting, the following applies with regard to the parameter sets of axes:

- For machine axes that are **not** involved in tapping or thread cutting, parameter set 1 (index = 0) is active. The other parameter sets do not have to be taken into account.
- For machine axes that are involved in tapping or thread cutting, the same parameter set number as that of the current gear stage of the spindle is active.
  All parameter sets correspond to the gear stages and must therefore be parameterized.

The current parameter set is displayed on the user interface at:

Operating Area "Diagnostics" > "Service axis"

Parameter sets during gear stage change

Each gear stage of a spindle is assigned a separate parameter set. The gear stage is selected via the following NC/PLC interface signal:

DB31, ... DBX16.0 - 2 = <actual gear stage>

Actual gear stage	DB31, ... DBX16.0 - 2	Parameter set	
1st gear stage	000	2	(Index=1)
1st gear stage	001	2	(Index=1)
2nd gear stage	010	3	(Index=2)
3rd gear stage	011	4	(Index=3)
4th gear stage	100	5	(Index=4)
5th gear stage	101 110 111	6	(Index=5)

For further information on gear stages for spindles, see Section "S1: Spindles (Page 733)".

2.5.5 Relevant system data

Axis machine data

Axis machine data for the jerk filter:

Number	Name	Effect
MD32400	$MA_AX_JERK_ENABLE	Axial jerk limitation
MD32402	$MA_AX_JERK_MODE	Filter type for axial jerk limitation

MD32407	$MA_AX_JERK_FIR_FREQ	Corner frequency axial jerk filter, type 5
MD32408	$MA_AX_JERK_FIR_ORDER	Filter order, axial jerk filter, type 5
MD32409	$MA_AX_JERK_FIR_WINDOW	Window type, axial jerk filter, type 5
MD32410	$MA_AX_JERK_TIME	Time constant for the axial jerk filter
MD32411	$MA_AX_JERK_TIME_ADD	Time constant for the second axial jerk filter, type 4
MD32412	$MA_AX_JERK_FREQ	Blocking frequency of the axial jerk filter
MD32414	$MA_AX_JERK_DAMP	Damping, axial jerk filter
MD32890	$MA_DESVAL_DELAY_ENABLE	Axial position setpoint phase filter
MD32895[<n>]	$MA_DESVAL_DELAY_TIME[<n>]	Time constant for the axial phase position setpoint phase filter

System variables

The following system variables are added with the function "Switchable position setpoint filter circuits":

Name	Effect	Access
$AA_DESVAL_FILTERS_SELECT	System variable that controls the switchover between the filter circuits.	r/w
$VA_DESVAL_FILTERS_ACTIVE	System variable in which the currently active filter circuit is located.	r
$VA_DESVAL_FILTERS_DELTA_POS	System variable that provides the position difference between the two filter circuits.	r
$VA_DESVAL_FILTERS_DELAY_1	System variable that provides the effective delay time between the 1st or 2nd filter circuit.	r
$VA_DESVAL_FILTERS_DELAY_2		r

2.6 Optimization of the control

2.6.1 Position controller, position setpoint filter: Balancing filter

Function

With feedforward control active, the position setpoint is sent through a so-called balancing filter before it reaches the controller itself. It is thus possible to control the speed setpoint to 100% in advance, without resulting in overshoots when positioning.

Activation

The feedforward control variant is selected and so also the filter activated using the axis-specific machine data:

MD32620 $MA_FFW_MODE (feedforward control mode)

Value	Meaning
3	Speed precontrol
4	Combined torque/speed precontrol

Activating and deactivating via the part program

Part programs can be used to activate and deactivate the feedforward control for all axes, using commands `FFWON` and `FFWOF`.

If the feedforward control of the individual axes should not be influenced by `FFWON`/`FFWOF`, the setting in the following machine data must be changed for these axes:

MD32630 $MA_FFW_ACTIVATION_MODE (activate feedforward control from program)

Parameterization

Recommended setting for recommissioning

If recommissioning, or if default values have been loaded (switch position 1 on commissioning switch and POWER ON), the following machine data default values apply:

- MD32620 $MA_FFW_MODE = 3
- MD32610 $MA_VELO_FFW_WEIGHT (feedforward control factor for the velocity feedforward control) = 1

The balancing time for the speed feedforward control then just has to be adjusted in the following machine data:

MD32810 $MA_EQUIV_SPEEDCTRL_TIME (equivalent time constant speed control loop for feedforward control)

Setting the equivalent time constant of the speed control loop (MD32810)

We recommend that the axis be allowed to move in and out in "AUTOMATIC" mode with a part program and that travel-in to the target position, i.e. the actual position value of the active measuring system, be monitored with the servo trace.

The initial value for setting is the time constant of the speed control loop. This can be read from the reference frequency characteristic of the speed control loop. In the frequent case of a PI controller with speed setpoint smoothing, an approximate equivalent time can be read from drive machine data p1414, p1415, p1416 and p1421.

This start value (e.g. 1.5 ms) is now entered:

MD32810 $MA_EQUIV_SPEEDCTRL_TIME = 0.0015

The axis then travels forward and backward and the operator monitors a greatly-magnified characteristic of the actual position value at the target position.

The following rules apply to making manual fine adjustments:

Monitoring	Measure	Effect
Overshoot	Increase MD32810	The position controller responds more slowly. ⇒ The tendency to overshoot decreases. Problem: The contour error (deviation from the programmed contour) increases at curves.
Excessively slow approach	Reduce MD32810	The position controller responds faster. ⇒ The contour error at curves decreases. Problem: The tendency to overshoot increases.

Note

The effects are similar to modification of the position controller gain (MD32200 $MA_POSCTRL_GAIN). They can be observed on the user interface in the operating section "Diagnostics" > "Axis service" using the computed K_V value.

MD32810 fine adjustment

Therefore, MD32810 should be assigned as small a value as possible, with the overshoot setting the limit during positioning. Furthermore, the initial value is only modified slightly during fine adjustment, typically by adding or deducting 0.25 ms.

For example, if the initial value is 1.5 ms, the optimum value calculated manually is usually within the range 1.25 ms to 1.75 ms.

In the case of axes equipped with direct measuring systems (load encoders) and strong elasticity, you may possibly accept small overshoots of some micrometers. These can be reduced with the help of the position setpoint filter for dynamic response adaptation (MD32910 $MA_DYN_MATCH_TIME) and for jerk (MD32410 $MA_AX_JERK_TIME), which also reduces the axis speed.

Identical axis data within an interpolation group

All the axes within an interpolation group should have **identical settings** in the following data:

- MD32200 $MA_POSCTRL_GAIN (adapted using MD32910)
- MD32620 $MA_FFW_MODE
- MD32610 $MA_VELO_FFW_WEIGHT
- MD32810 $MA_EQUIV_SPEEDCTRL_TIME (or MD32800 $MA_EQUIV_CURRCTRL_TIME) (dependent on the mechanical system and drive)
- MD32400 $MA_AX_JERK_ENABLE
- MD32402 $MA_AX_JERK_MODE
- MD32410 $MA_AX_JERK_TIME

The servo gain display (K_V) of the user interface in operating section "Diagnostics" > "Axis service" is used for checking.

Non-identical axis data within an interpolation group

If identical values are not possible for the above data, the following machine data can be used to make an adjustment:

MD32910 $MA_DYN_MATCH_TIME (time constant of dynamic response adaptation)

This allows the same servo gain value (K_V) to be displayed.

Different servo gain display values (K_V) usually point to the following:

- The gear ratios do not match in one or several axes.
- The feedforward control setting data does not match.

Setting the equivalent time constant of the current control loop (MD32800)

The activation of the torque feedforward control filter is performed with:

MD32620 $MA_FFW_MODE = 4

The same rules and recommendations apply to setting the time constant of the current control loop MD32800 $MA_EQUIV_CURRCTRL_TIME as to the speed feedforward control.

Limitation to stiff machines

Experience has shown that this expenditure is only worthwhile in the case of very stiff machines, and requires appropriate experience. The elasticities of the machine are often excited due to the injection of the torque so strongly that the existing vibrations neutralize the gain in contour accuracy.

In this case, it would be worth trying the function "Dynamic Stiffness Control" (DSC) as an alternative:

MD32640 $MA_STIFFNESS_CONTROL_ENABLE = 1

Note

The following conditions apply to use of DSC:

- In the NC, no actual value inversion (MD32110 $MA_ENC_FEEDBACK_POL = -1) must be parameterized. Actual-value inversion in the drive (SINAMICS parameter p0410) is permissible.
- The function must be activated for all axes that are in an interpolation relationship (path motion or coupling) (MD32640 = 1).
- If one of the axes is parameterized as a simulated axis (MD30130 $MA_CTRLOUT_TYPE = 0, MD30240 $MA_ENC_TYPE = 0), DSC cannot be applied. In this case, the function must be deactivated for all axes that are in an interpolation relationship (MD32640 = 0).

⚠ CAUTION

Exception error condition

An exception error condition may occur if p1192 is set for a non-existing measuring system and DSC is activated!

For service situations in which an additional measuring system has to be disabled and DSC is active, it must be assured that p1192 is also adapted.

Control response with POWER ON, RESET, REPOS, etc.

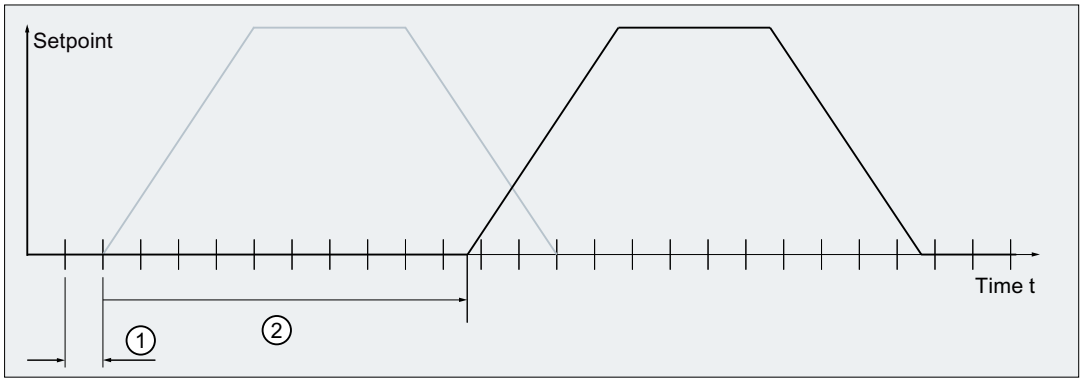
In the case of POWER ON and RESET, as well as with "Enable machine data", the setting data of the feedforward control is read in again (see the appropriate values of the relevant machine data).

Mode change, block search and REPOS have **no** influence on the feedforward control.

2.6.2 Position controller, position setpoint filter: Phase filter

Function

The axis-specific phase filter setpoint (dead time / delay) implements a pure phase shifter with which the setpoint phase response can be influenced. Together with the axis-specific jerk filter setpoint (MD32402_$MA_AX_JERK_MODE[<axis>] = 2; see Function Manual "Axes and Spindles", Section "Jerk filter") it is possible to adapt the amplitude response and the phase response independently of one another to the dynamically weakest of several axes, which should proceed along a programmed path together.



① Position controller cycle clock: 2 ms

② Time constant of the axis-specific phase filter setpoint: 19.2 ms

Figure 2-13 Effect of the axis-specific phase filter setpoint

Parameterization: Time constant

The time constant for the axis-specific phase filter setpoint can be set in the range:

(0 to 64) * position controller cycle clock (MD10061 $MN_POSCTRL_CYCLE_TIME)

The value of the time constants must be entered in seconds [s] in the machine data:

MD32895 $MA_DESVAL_DELAY_TIME[<Axis>]

Example: Position controller cycle clock: 2 ms ⇒ adjustable time constant: 0.0 to 0.128 s

Note

The time constant of the phase filter setpoint delays the axis' response characteristics for tapping, retractions, and exact stops / block changes, etc. It is therefore recommended to set the time constant as low as possible.

Limiting to a maximum value

A greater time constant than the permitted maximum value (64 * position controller cycle clock) is limited internally to the maximum value. No message / alarm is displayed.

Parameterization: Activation

The function of the axis-specific phase filter setpoint must be activated with the following machine data:

MD32890 $MA_DESVAL_DELAY_ENABLE[<Axis>] = "TRUE"

Examples

Assumption: Position controller cycle clock = 2 ms

1. MD32890 $MA_DESVAL_DELAY_ENABLE[<Axis>] = "FALSE"
   MD32895 $MA_DESVAL_DELAY_TIME[<Axis>] = <Time constant>
   - Phase filter setpoint: Not active
   - Time constant: Irrelevant
2. MD32890 $MA_DESVAL_DELAY_ENABLE[<Axis>] = TRUE
   MD32895 $MA_DESVAL_DELAY_TIME[<axis>] = 0.002
   - Phase filter setpoint: Active
   - Time constant: 2 ms ⇒ the setpoint output is delayed by a position controller cycle clock.
3. MD32890 $MA_DESVAL_DELAY_ENABLE[<Axis>] = TRUE
   MD32895 $MA_DESVAL_DELAY_TIME[<axis>] = 0.256
   - Phase filter setpoint: Active
   - Set time constant: 256 ms;
     maximum possible time constant: 64 * 2 ms = 128 ms
     ⇒ internally effective time constant: 128 ms

Boundary conditions

SINUMERIK Safety Integrated

The phase filter setpoint delays the output of axis-specific setpoints; e.g. for retractions (ESR for Stop E). However, the phase filter setpoint has no influence over shutdown processes; e.g. the SBH activation time.

2.6.3 Position controller: injection of positional deviation

Preconditions

- The function can only be used on axes with two encoders:
  MD30200 $MA_NUM_ENCS = 2
  One of the encoders must be parameterized as an indirect measuring system and the other as a direct measuring system:
  - Direct measuring system:
    MD31040 $MA_ENC_IS_DIRECT[1]=1
    The encoder for position actual-value acquisition is connected directly to the machine (load encoder).
  - Indirect measuring system:
    MD31040 $MA_ENC_IS_DIRECT[0]=0
    The encoder for position actual-value acquisition is located on the motor (motor encoder).
- Telegram type 136 or 138 must be configured as standard telegram type for PROFIdrive both in the drive and also in the NC (MD13060 $MN_DRIVE_TELEGRAM_TYPE).

Function

For active injected positional deviation, the difference position between the direct and the indirect measuring system of an axis is determined and in accordance with the weighting-factor setting is applied as additional current setpoint for the feedforward control in the position control cycle. The resulting oscillation damping improves the stability and positioning behavior of the axis.

Application

The function is used for axes with strong tendency to vibrate.

Effectiveness

The function acts only for axes with small natural frequency (to approximately 20 Hz).

Activation/parameterization

The function is activated by specifying the weighting factor:

MD32950 $MA_POSCTRL_DAMPING (damping of the speed control loop) = <value>

Value range: -100% ... +100%

An input value "100%" means: A supplementary torque in accordance with SINAMICS parameter p2003 is applied when the determined position difference between the two measuring systems reaches the following value:

- With linear motors: 1 mm
- With linear axis with rotary motor: MD31030 $MA_LEADSCREW_PITCH (leadscrew pitch)
- For rotary axis/spindle: 360 degrees

Standard setting is 0. In this case, the injection of positional deviation is inactive.

Note

The weighting factor MD32950 $MA_POSCTRL_DAMPING can be set on the basis of step responses, for example.

If the control approaches the stability limit (vibration inclination increases), the parameter is too large or the value has the incorrect sign.

2.6.4 Position control with proportional-plus-integral-action controller

Function

As standard, the core of the position controller is a P controller. It is possible to switch-in an integral component for special applications (such as an electronic gear). The resulting proportional-plus-integral-action controller then corrects the error between setpoint and actual positions down to zero in a finite, settable time period when the appropriate machine data is set accordingly.

⚠ CAUTION

Overshootings of the actual position for activated PI controller

In this instance, you must decide whether this effect is admissible for the application in question. Knowledge of the control technology and measurements with servo trace are an absolute prerequisite for using the function. If the appropriate machine data is incorrectly set, then machines could be damaged due to instability.

Procedure

1. First optimize the position control loop as a proportional-action controller using the tools described in the previous subsections.
2. Increase the tolerances of the following machine data while measurements are being taken to determine the quality of the position control with proportional-plus-integral-action controller:
   - MD36020 $MA_POSITIONING_TIME
   - MD36030 $MA_STANDSTILL_POS_TOL
   - MD36040 $MA_STANDSTILL_DELAY_TIME
   - MD36400 $MA_CONTOUR_TOL
3. Activate the position control loop as a proportional-plus-integral-action controller by setting the following machine data:
   MD32220 $MA_POSCTRL_INTEGR_ENABLE ; set value 1
   MD32210 $MA_POSCTRL_INTEGR_TIME ; integral time [sec.]
   Effect of integral time:
   - $T_n \rightarrow 0$:
     The control error is corrected quickly; however, the control loop can become instable.
   - $T_n \rightarrow \infty$:
     Effectiveness of the integral component is almost 0. Behavior of the controller like a pure proportional controller.
4. Find the right compromise for T_n between these two extreme cases for the application.

   Note

   T_n must not be chosen too near the stability limit because the occurrence of an instability can cause machine damage.

   It is therefore recommended to set T_n to no less than 1 s.

   Use servo trace to trace the travel-in of an automatic program traveling to and from a target position.
5. Set the servo trace to display the following:
   - Following error
   - Actual velocity
   - Actual position
   - Reference position
6. Reset the tolerance values in the following machine data to the required values, once the optimum value for T_n has been identified:
   - MD36020 $MA_POSITIONING_TIME
   - MD36030 $MA_STANDSTILL_POS_TOL
   - MD36040 $MA_STANDSTILL_DELAY_TIME
   - MD36400 $MA_CONTOUR_TOL

Supplementary conditions

DSC

If the integrator function is used, DSC (Dynamic Stiffness Control) must be switched off.

Example

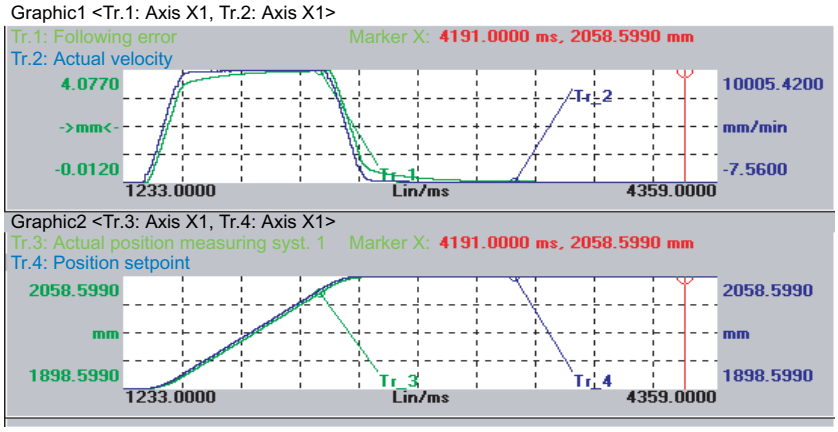
Setting result after several iterative processes for $_KR$ and T_n.

Machine data settings:

- MD32220 $MA_POSCTRL_INTEGR_ENABLE = 1
- MD32210 $MA_POSCTRL_INTEGR_TIME = 0.003
- MD32200 $MA_POSCTRL_GAIN[1] = 5.0

Parameter set selection 0

Each of the following quantities - following error, actual velocity, actual position, and position setpoint - has been recorded by servo trace. When traversing in JOG mode, the characteristic of the individual data shown in the following figure was then drawn.



2.7 Data lists

2.7.1 Machine data

2.7.1.1 Displaying machine data

Number	Identifier: $MM_	Description
9004	DISPLAY_RESOLUTION	Display resolution
9010	SPIND_DISPLAY_RESOLUTION	Display resolution for spindles
9011	DISPLAY_RESOLUTION_INCH	Display resolution for INCH system of measurement

2.7.1.2 NC-specific machine data

Number	Identifier: $MN_	Description
10000	AXCONF_MACHAX_NAME_TAB	Machine axis name
10050	SYSCLOCK_CYCLE_TIME	System basic cycle
10070	IPO_SYSCLOCK_TIME_RATIO	Factor for interpolator cycle
10060	POSCTRL_SYSCLOCK_TIME_RATIO	Factor for position-control cycle
10200	INT_INCR_PER_MM	Computational resolution for linear positions
10210	INT_INCR_PER_DEG	Computational resolution for angular positions
10220	SCALING_USER_DEF_MASK	Activation of scaling factors
10230	SCALING_FACTORS_USER_DEF	Scaling factors of physical quantities
10240	SCALING_SYSTEM_IS_METRIC	Basic system metric
10250	SCALING_VALUE_INCH	Conversion factor for switchover to inch system
10260	CONVERT_SCALING_SYSTEM	Basic system switchover active
10270	POS_TAB_SCALING_SYSTEM	Measuring system of position tables
10290	CC_TDA_PARAM_UNIT	Physical units of the tool data for CC
10292	CC_TOA_PARAM_UNIT	Physical units of the tool edge data for CC
13050	DRIVE_LOGIC_ADDRESS	Logical drive addresses
13060	DRIVE_TELEGRAM_TYPE	Standard message frame type for PROFIBUS DP
13070	DRIVE_FUNCTION_MASK	DP function used
13080	DRIVE_TYPE_DP	Drive type PROFIBUS DP

2.7.1.3 Channelspecific machine data

Number	Identifier: $MC_	Description
20150	GCODE_RESET_VALUES	Initial setting of the G groups

2.7.1.4 Axis/spindlespecific machine data

Number	Identifier: $MA_	Description
30110	CTRLOUT_MODULE_NR	Setpoint assignment: Drive number
30120	CTRLOUT_NR	Setpoint assignment: Setpoint output on drive module
30130	CTRLOUT_TYPE	Output type of setpoint
30200	NUM_ENCS	Number of encoders
30220	ENC_MODULE_NR	Actual value assignment: Drive module number
30230	ENC_INPUT_NR	Actual value assignment: Input on the drive module
30240	ENC_TYPE	Type of actual value acquisition (actual position value)
30242	ENC_IS_INDEPENDENT	Encoder is independent
30300	IS_ROT_AX	Rotary axis
31000	ENC_IS_LINEAR	Direct measuring system (linear scale)
31010	ENC_GRID_POINT_DIST	Distance between reference marks on linear scales
31020	ENC_RESOL	Encoder pulses per revolution
31030	LEADSCREW_PITCH	Leadscrew pitch
31040	ENC_IS_DIRECT	Encoder is connected directly to the machine
31044	ENC_IS_DIRECT2	Encoder on intermediate gear
31050	DRIVE_AX_RATIO_DENOM	Denominator load gearbox
31060	DRIVE_AX_RATIO_NUMERA	Load gear numerator
31064	DRIVE_AX_RATIO2_DENOM	Intermediate gear denominator
31066	DRIVE_AX_RATIO2_NUMERA	Intermediate gear numerator
31070	DRIVE_ENC_RATIO_DENOM	Measuring gear denominator
31080	DRIVE_ENC_RATIO_NUMERA	Measuring gear numerator
31090	JOG_INCR_WEIGHT	Weighting of increment for INC/handwheel
31200	SCALING_FACTOR_G70_G71	Factor for converting values when `G70`/`G71` is active
32000	MAX_AX_VELO	Maximum axis velocity
32100	AX_MOTION_DIR	Travel direction
32110	ENC_FEEDBACK_POL	Sign actual value (feedback polarity)
32200	POSCTRL_GAIN	Servo gain factor (K_V)
32210	POSCTRL_INTEGR_TIME	Integrator time position controller
32220	POSCTRL_INTEGR_ENABLE	Activation of integral component of position controller
32250	RATED_OUTVAL	Rated output voltage
32260	RATED_VELO	Rated motor speed
32400	AX_JERK_ENABLE	Axial jerk limitation
32402	AX_JERK_MODE	Filter type for axial jerk limitation
32407	AX_JERK_FIR_FREQ	Corner frequency axial jerk filter, type 5
32408	AX_JERK_FIR_ORDER	Filter order, axial jerk filter, type 5
32409	AX_JERK_FIR_WINDOW	Window type, axial jerk filter, type 5
32410	AX_JERK_TIME	Time constant for the axial jerk filter
32411	AX_JERK_TIME_ADD	Time constant for the second axial jerk filter, type 4
32412	AX_JERK_FREQ	Blocking frequency of the axial jerk filter

Number	Identifier: $MA_	Description
32414	AX_JERK_DAMP	Damping, axial jerk filter
32450	BACKLASH	Backlash
32500	FRICT_COMP_ENABLE	Friction compensation active
32610	VELO_FFW_WEIGHT	Feedforward control factor for speed feedforward control
32620	FFW_MODE	Feedforward control mode
32630	FFW_ACTIVATION_MODE	Activate feedforward control from program
32650	AX_INERTIA	Moment of inertia for torque feedforward control
32652	AX_MASS	Axis mass for torque precontrol
32711	CEC_SCALING_SYSTEM_METRIC	System of units for sag compensation
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
32890	DESVAL_DELAY_ENABLE	Axial position setpoint phase filter
32895	DESVAL_DELAY_TIME	Time constant for the axial phase position setpoint phase filter
32900	DYN_MATCH_ENABLE	Dynamics matching
32910	DYN_MATCH_TIME [n]	Time constant for dynamic response adaptation
32930	POSCTRL_OUT_FILTER_ENABLE	Activation of low-pass filter at position controller output
32950	POSCTRL_DAMPING	Damping of the speed control loop
33000	FIPO_TYPE	Fine interpolator type
34320	ENC_INVERS[n]	Length measuring system is inverse
35100	SPIND_VELO_LIMIT	Maximum spindle speed
36200	AX_VELO_LIMIT [n]	Threshold value for velocity monitoring
36210	CTRLOUT_LIMIT[n]	Maximum speed setpoint
36400	AX_JERK_ENABLE	Axis-specific jerk limitation
36410	AX_JERK_TIME	Time constant for the axis-specific jerk filter
36500	ENC_CHANGE_TOL	Max. tolerance for position actual-value switchover
36510	ENC_DIFF_TOL	Measuring system synchronism tolerance
36700	ENC_COMP_ENABLE[n]	Interpolatory compensation

F1: Travel to fixed stop

3

3.1 Brief description

Function

With the "Travel to fixed stop" function, moving machine parts, e.g. tailstock or sleeve, can be traversed so that they can apply a defined torque or force with respect to other machine parts over any time period.

Characteristics

- Programmability using commands in the part program or synchronized action:
  - Activating and deactivating the "Travel to fixed stop" function
  - Setting the clamping torque
  - Setting the monitoring window
- Manual intervention option using setting data:
  - Activating and deactivating the "Travel to fixed stop" function
  - Setting the clamping torque
  - Setting the monitoring window
- Presettings using machine data
  - Clamping torque
  - Monitoring window
- Identifying the function status via NC/PLC interface signals
- Enable or acknowledgment options via NC/PLC interface signals
- Reading the reference and actual status of the function via system variable
- "Travel to fixed stop" is possible for axes and spindles.
- "Travel to fixed stop" is possible simultaneously for several axes and parallel to the traversing of other axes.
- Multi-channel block search can be performed with calculation of all the required additional data (SERUPRO).
- Simulated axis traversal in conjunction with "travel to fixed stop" and "torque reduction" possible.
- "Vertical" axes can also be moved with FXS to a fixed stop.

3.2 Detailed description

3.2.1 Programming

Function

Travel to fixed stop

The "Travel to fixed stop" function is controlled via the FXS, FXST and FXSW commands.

The activation can also be performed without traversing motion of the relevant axis. The torque is immediately limited. The fixed stop is monitored as soon as the axis is traversed.

Note

Synchronized actions

The "Travel to fixed stop" function can also be controlled via synchronized actions.

Further information
Function Manual Synchronized Actions

Travel with limited torque/force

Travel with limited torque/force can be controlled via the FOCON, FOCOF and FOC commands (see Section "Traveling with limited torque/force FOC").

Syntax

```
FXS[<Axis>]=<Request>
FXST[<Axis>]=<Clamping torque>
FXSW[<Axis>]=<Window width>
```

Meaning

Parameter	Meaning
FXS:	"Travel to fixed stop" function, effectiveness: modal
<Request>:	0 = switch off 1 = switch on
FXST:	Set clamping torque
<Clamping torque>:	Clamping torque in % of the maximum drive torque. SINAMICS S120: p2003
FXSW:	Set monitoring window
<Window width>:	Width of the tolerance window around the fixed stop Unit: mm, inch or degrees
<Axis>:	Name of the channel axis, type: AXIS

Changes to the torque limiting (FXST)

The torque limit value can be changed in each block. The change becomes effective before executing the traversing motion programmed in the block. The torque limitation acts in addition to the acceleration limitation (ACC).

Ramp-shaped change

A time can be set using the machine data, within which the torque limit value is linearly changed:

MD37012 $MA_FIXED_STOP_TORQUE_RAMP_TIME (time until the new torque limit is reached)

Changes to the monitoring window (FXSW)

The monitoring window can be changed in each block. The change becomes effective before executing the traversing motion programmed in the block.

When the monitoring window is changed, not only does the window width change, but also the reference point of the window to the actual axis position.

"Travel to fixed stop" when the continuous-path mode is active (G64)

The following machine data can be used to set that with the selection of the (FXS) function during active continuous-path mode (G64), no exact stop is triggered at the block change (G60):

MD37060 $MA_FIXED_STOP_ACKN_MASK (monitoring PLC acknowledgements for travel to fixed stop)

Bit	Value	Meaning
0	0	Start of the traversing motion without acknowledgement by the PLC
	1	Start of the traversing motion after acknowledgement by the PLC

Boundary conditions

- The traversing motion to the fixed stop can be programmed as a path- or block-related or modal positioning axis motion.
- Travel to fixed stop can be selected for several machine axes simultaneously.
- For a machine axis, which is traversed to a fixed stop, it is not permissible that transformation, coupling and frame functions are active:
- The travel path and the activation of the function must be programmed in one block in the part program.
- If "Travel to fixed stop" is activated via synchronized actions, the travel path and the activation of the function can be programmed in separate blocks.

3.2.2 Functional sequence

3.2.2.1 Selection

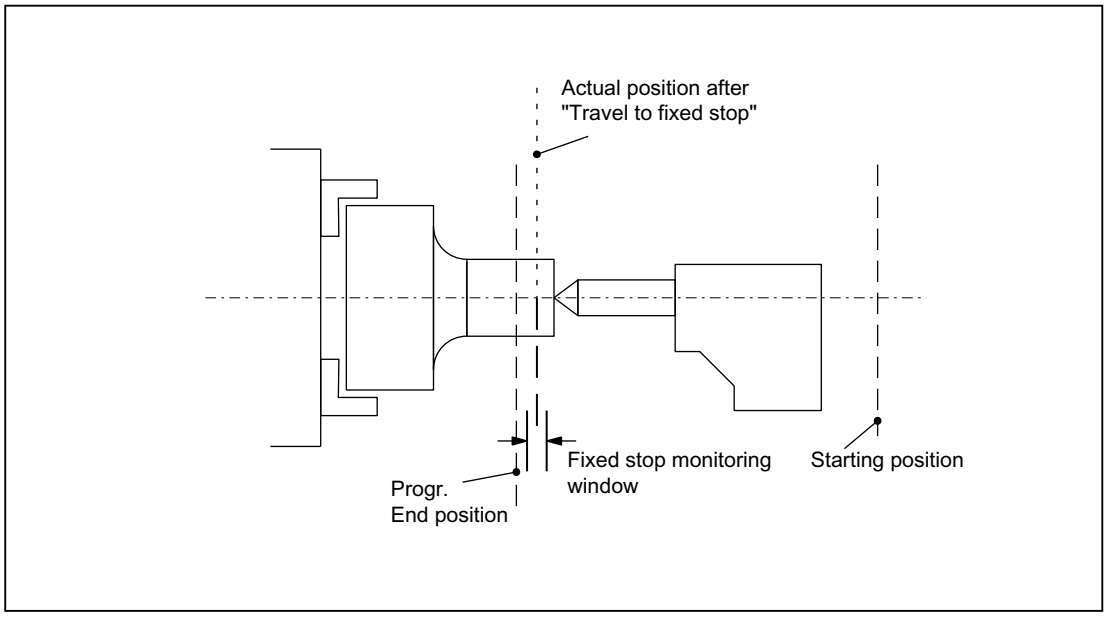


Figure 3-1 Example of travel to fixed stop

Procedure

The NC detects that the function "Travel to fixed stop" is selected via the command `FXS[x]=1` and signals the PLC using the IS DB31, ... DBX62.4 ("Activate travel to fixed stop") that the function has been selected.

If the machine data:

MD37060 $MA_FIXED_STOP_ACKN_MASK (monitoring PLC acknowledgements for travel to fixed stop)

is set correspondingly, the system waits for the acknowledgement of the PLC using the IS DB31, ... DBX3.1 ("Enable travel to fixed stop").

The programmed target position is then approached from the start position at the programmed velocity. The fixed stop must be located between the start and target positions of the axis/spindle. A programmed torque limit (clamping torque specified via `FXST[<axis>]`) is effective from the start of the block, i.e. the fixed stop is also approached with reduced torque. Allowance for this limitation is made in the NC through an automatic reduction in the acceleration rate.

If no torque has been programmed in the block or since the start of the program, then the value is valid in the axis-specific machine data:

MD37010 $MA_FIXED_STOP_TORQUE_DEF (default for fixed stop clamping torque)

is entered.

3.2.2.2 Fixed stop is reached

Detecting the fixed stop

Detecting the fixed stop or identifying that the machine axis has reached the fixed stop can be set using the following machine data:

MD37040 $MA_FIXED_STOP_BY_SENSOR = <value> (fixed stop detection via sensor)

<value>	Meaning
0	The fixed stop has been reached when the contour deviation of the machine axis has exceeded the value set in the machine data: MD37030 $MA_FIXED_STOP_THRESHOLD (threshold for fixed stop detection)
1	An external sensor detects when the fixed stop has been reached and informs the control via the following axial NC/PLC interface signal: DB31, ... DBX1.2 == 1 (sensor for fixed stop)
2	The fixed stop has been reached, if one of the conditions, specified under <value> == 0 OR <value> == 1 applies.

Ineffective NC/PLC interface signals

When the axis is in the "Fixed stop reached" state, the following NC/PLC interface signals have no effect:

- DB31, ... DBX1.3 (axis/spindle disable)
- DB31, ... DBX2.1 (controller enable)

Actions when the fixed stop is reached

The following actions are executed when the fixed stop is reached:

- The torque in the drive is increased up to the programmed clamping torque (`FXST`)
- The remaining distance to go is deleted
- The position setpoint is tracked
- The NC/PLC interface signal is set: DB31, ... DBX62.5 = 1 ("fixed stop reached")
- Executing a block change:
  - Executing a block change when the fixed stop is reached: MD37060 $MA_FIXED_STOP_ACKN_MASK, Bit 1 = 0
    The block change is immediately executed after reaching the fixed stop.
  - Block change is only executed after the acknowledgment by the PLC user program: MD37060 $MA_FIXED_STOP_ACKN_MASK, Bit 1 = 1
    After reaching the fixed stop, the block is not changed until acknowledgment by the PLC user program:
    DB31, ... DBX1.1 == 1 (acknowledge fixed stop reached)

  In order to also maintain the clamping torque after the block change, the NC still outputs a setpoint for the machine axis.
- Activating the fixed stop monitoring or the monitoring window

Monitoring window

If, in the traversing block to the fixed stop or since the beginning of the program, no specific value for the monitoring window is programmed with `FXSW` then the value set in the machine data is active:

MD37020 $MA_FIXED_STOP_WINDOW_DEF (default for fixed stop monitoring window)

If the axis leaves the position it was in when the fixed stop was detected by more than the specified window, then alarm 20093 "Fixed stop monitoring has responded" is displayed and the "Travel to fixed stop" function is deselected.

The monitoring window must be selected by the user such that the alarm is activated only when the axis leaves the fixed stop position.

NOTICE
"Fixed stop reached" and when the fixed stop breaks
As soon as the "fixed stop reached" state is identified, then a speed setpoint derived from the axis-specific machine data servo gain factor (K_V) (MD32200) and the threshold for the fixed stop detection (MD37030) is entered for the drive. If the fixed stop breaks in this state, then the axis accelerates until it reaches the monitoring window limit. The velocity that is then reached is proportional to the values set in the specified machine data. For appropriately high values, it is possible that the drive accelerates up to the maximum motor speed.

Overview

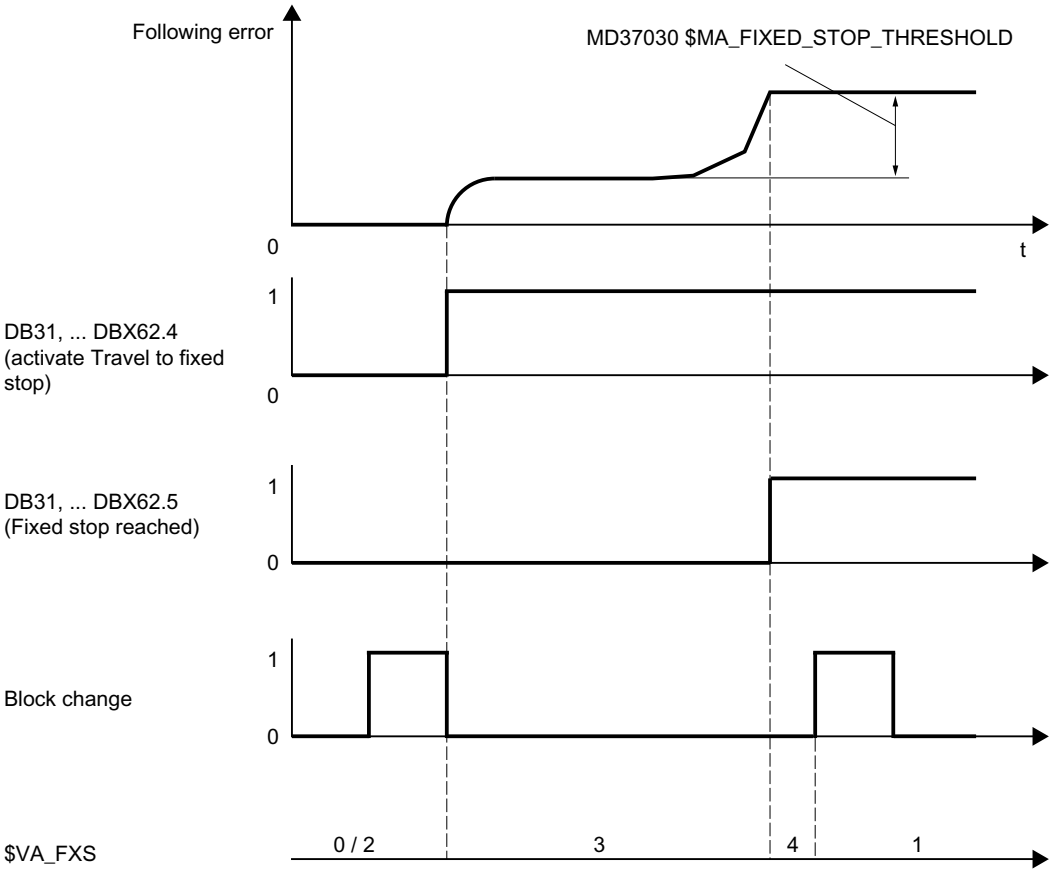


Figure 3-2 Fixed stop is reached

3.2.2.3 Fixed stop is not reached

Alarm suppression

Alarms for various causes of breakage can be suppressed using the machine data:

MD37050 $MA_FIXED_STOP_ALARM_MASK = <value>

Value	Description: Suppressed alarms
0	Alarm 20091 "Fixed stop not reached"
2	Alarm 20091 "Fixed stop not reached" Alarm 20094 "Fixed stop aborted"
3	Alarm 20094 "Fixed stop aborted"

Actions in the case of a fault or breakage

The following actions are executed when a fault occurs or for a breakage:

- The NC/PLC interface signal is reset: DB31, ... DBX62.4 = 0 (activate travel to fixed stop)
- Depending on the setting in the machine data, the system waits for acknowledgment:
  - MD37060 $MA_FIXED_STOP_ACKN_MASK
  - DB31, ... DBX3.1 == 0 (enable travel to fixed stop)
- Withdrawing torque limits
- Executing a block change

Overview

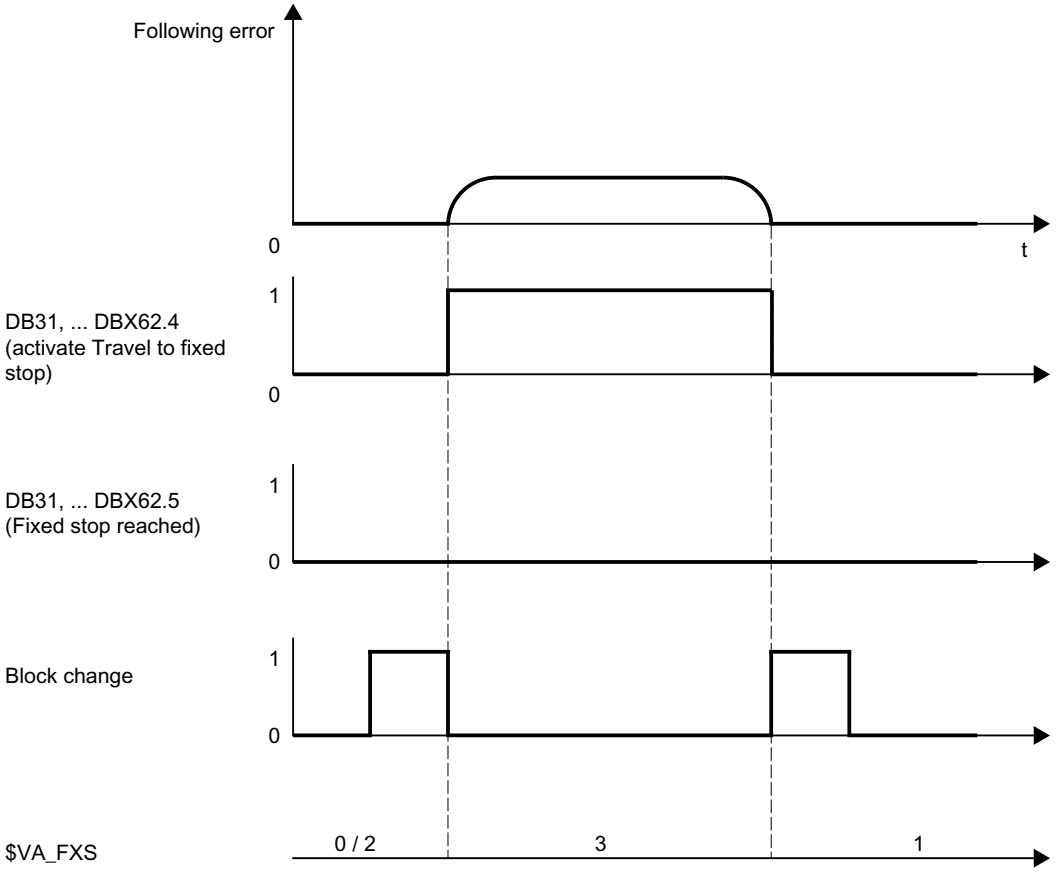


Figure 3-3 Fixed stop is not reached

3.2.2.4 Deselection

The "travel to fixed stop" function is deselected using the command `FXS[<axis>] = 0` in a block of an NC program.

Actions when deselecting the function

When deselecting the function, the following actions are executed:

- A preprocessing stop is initiated (STOPRE)
- The NC/PLC interface signals are reset
  - DB31, ... DBX62.4 = 0 (activate travel to fixed stop)
  - DB31, ... DBX62.5 = 0 (fixed stop reached)
- Depending on the machine data, the system waits for acknowledgment from the PLC user program:
  - MD37060 $MA_FIXED_STOP_ACKN_MASK
  - DB31, ... DBX3.1 == 0 (enable travel to fixed stop)
  - DB31, ... DBX1.1 == 0 (acknowledge fixed stop reached)
- Exiting the follow-up mode
- Axis resumes closed-loop position control

Pulse enable

The pulse enable or pulse inhibit can be canceled via:

- Drive: Via terminal EP (enable pulses)
- NC/PLC interface signal: DB31, ... DBX21.7 ("pulse enable")

The behavior at the fixed stop can be set via the following machine data:

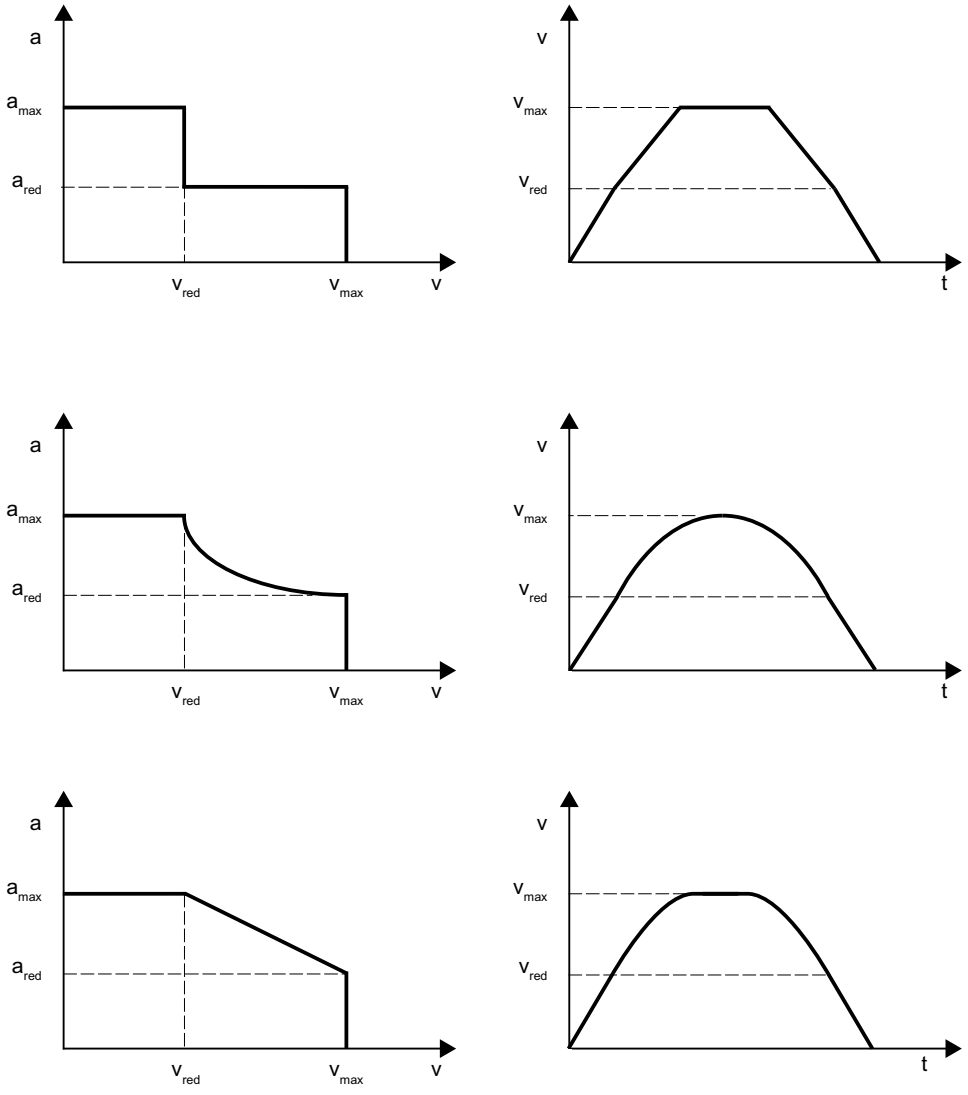
MD37002 $MA_FIXED_STOP_CONTROL, bit 0 and bit 1 (sequence control for traversing to fix stop)

Bit	Value	Meaning
0	Behavior for pulse**inhibit** at the fixed stop	
	0	"Travel to fixed stop" is aborted
	1	"Travel to fixed stop" is interrupted, i.e. the drive is without power.
1	Behavior for pulse**enable** at the fixed stop	
	0	The torque is injected suddenly.
	1	The torque is linearly ramped-up over the time specified in the machine data: MD37012 $MA_FIXED_STOP_TORQUE_RAMP_TIME

Special case: Deleting the pulse enable during deselection

If, during deselection, the function in state: "Waiting for PLC acknowledgements" the pulse enable is deleted, the torque limit is reduced to 0. In this phase, if pulse enable is set again torque is no longer established in the drive. Once the deselection has been completed the axis can be traversed normally again.

Overview



① Traversing block with deselection `FXS[<axis>]=0`

Figure 3-4 Fixed stop deselection

3.2.3 Behavior during block search

Function

Block search with calculation

- If the target block is located in a program section in which the axis must stop at a fixed limit, then the fixed stop is approached if it has not yet been reached.
- If the target block is located in the program section in which the axis must not stop at a fixed limit, then the axis leaves the fixed stop if it is still positioned there.
- If the axis is in the "Fixed stop reached" state, message 10208 "Press NC start to continue the program" is displayed. The program can be continued with NC start.
- Clamping torque `FXST` and monitoring window `FXSW` have the value that they have for normal program processing at the start of the target block.

Block search without calculation

The `FXS`, `FXST` and `FXSW` commands are ignored during the block search.

Effectiveness of FOCON/FOCOF

The state of the modal-acting torque/force reduction `FOCON`/`FOCOF` is maintained during the block search and is effective in the approach block.

Block search with FXS or FOC

The user selects `FXS` or `FOC` in a program area of the target block in order to acquire all states and functions of the machining last valid. The NC starts the selected program in Program test mode automatically. After the target block has been found, the NC stops at the beginning of the target block, deselects Program test internally again and displays the Stop condition "Search target found" in its block display.

⚠ CAUTION

SERUPRO approach does not really take the FXS command into account

The approach to the programmed end position of the `FXS` block is only simulated **without torque limitation**.

If `FXS` is located between the beginning of the program and the search target, the command is not executed by the NC. The motion is only simulated up to the programmed end point.

The user can log the turning on and turning off of `FXS` in the part program. If necessary, the user can start an ASUP in order to activate or deactivate `FXS` in this SERUPRO-ASUP.

System variable

The reference and actual state of the "travel to fixed stop" function can be read using the following system variable:

- $AA_FXS (reference state)
- $VA_FXS (actual state)

SERUPRO: $AA_FXS (reference state)

During SERUPRO, $AA_FXS supplies the following values depending on the activation status of the "Travel to fixed stop" function:

Activation status of the "travel to fixed stop" function	System variable $AA_FXS ==
"Deactivated"	0 (axis not at fixed stop)
"Activated"	3 ("Travel to fixed stop" selection is active)

Note

During SERUPRO, the system variable $AA_FXS only supplies the values 0 and 3. As a result, based on $AA_FXS, the program sequence can be changed with SERUPRO compared to the normal program execution for program branches.

SERUPRO: $VA_FXS (actual state)

During SERUPRO, the variable $VA_FXS always supplies the real state of axis on the machine.

Example

The current state of the "Travel to fixed stop" function can be determined in the SERUPRO ASUP via the system variables $AA_FXS and $VA_FXS, and the appropriate response initiated:

```
Program code: FXS_SERUPRO_ASUP.MPF                    Comment
N100 WHEN ($AA_FXS[X]==3) AND ($VA_FXS[X]==0) DO      ; Reference=="Selection not active" AND
FXS[X]=1                                              ; Actual=="Axis not at fixed stop"
                                                      ; => "Switch on"
N200 WHEN ($AA_FXS[X]==0) AND ($VA_FXS[X]==1) DO      ; Reference=="Axis not at fixed stop" AND
FXS[X]=0                                              ; Actual=="Fixed stop successfully approached"
                                                      ; => "Switch on"
N1020 REPOSA                                          ; Reapproaching the contour linearly with all axes
```

Displaying the REPOS offset

Once the search target has been found, for each axis, the actual state regarding the "Travel to fixed stop" is displayed using the following NC/PLC interface signals:

- DB31, ... DBX62.4 (activate travel to fixed stop)
- DB31, ... DBX62.5 (fixed stop reached)

Example:

If the axis is at the fixed stop and the target block is available after deselection of `FXS`, the new target position is displayed via DB31, ... DBX62.5 (fixed stop reached) as the `REPOS` offset.

REPOS and FXS

With REPOS, the functionality of FXS is repeated automatically and called FXS-REPOS in the following. This sequence is comparable to the FXS_SERUPRO_ASUP.MPF program. Every axis is taken into account and the torque last programmed before the search target is applied.

The user can treat FXS separately in a SERUPRO ASUP.

The following then applies:

Every FXS action executed in the SERUPRO ASUP automatically takes care of

$AA_FXS[<axis>] = $VA_FXS[<axis>].

This deactivates FXS-REPOS for axis X.

Deactivating FXS-REPOS

FXS-REPOS is deactivated by:

- An FXS synchronized action which refers to REPOSA
- $AA_FXS[X] = $VA_FXS[X] in the SERUPRO_ASUP

Note

A SERUPRO ASUP **without** FXS treatment or no SERUPRO ASUP results automatically in FXS-REPOS.

⚠ CAUTION

Speed too high for FXS-REPOS

FXS-REPOS traverse all path axes together to the target position. Axes with and without FXS treatment thus traverse together with the G command and feedrate valid in the target block. As a result, the fixed stop may be approached in rapid traverse(G0) or higher velocity.

FOC-REPOS

FOC-REPOS behaves in the same way as FXS-REPOS.

A changing torque characteristic during the program preprocessing cannot be implemented with FOC-REPOS.

Example

Axis X is traversed from position 0 to 100. FOC is switched on every 20 millimeters for 10 millimeters. The resulting torque characteristic is generated with non-modal FOC and cannot be traced by FOC-REPOS. Axis X is traversed by FOC-REPOS with or without FOC in accordance with the last programming before the target block.

Programming examples for FXS "Travel to fixed stop", see Function Manual "Basic Functions", Section "Program test".

3.2.4 Behavior for reset and function abort

NC reset

As long as the function is still not in the "Successful travel to fixed stop" state, the travel to fixed stop can be aborted with NC reset.

Even when the fixed stop has already has been approached, but the specified stop torque not yet fully reached, then the function can still be aborted with NC reset. The position setpoint of the axis is synchronized with the current actual position.

As soon as the function is in the "Successful travel to fixed stop" state, the function also remains active after the NC reset.

Function abort

A function abort can be triggered by the following events:

- Emergency stop

  ⚠ CAUTION

  Dangerous machine situations possible for travel to limit stop

  It must be ensured that no dangerous machine situations occur while travel to fixed stop is active when an "Emergency stop" is triggered or reset.

  For example, behavior when setting and canceling the pulse enable:
  MD37002 $MA_FIXED_STOP_CONTROL, bit 0 (behavior for pulse disable at the stop)
  - Bit 0 = 0: Travel to fixed stop is aborted
  - Bit 0 = 1: Travel to fixed stop is interrupted, i.e. the drive is without power
    Once the pulse disable is canceled again, the drive presses with the stop torque again.

  Note

  NC and drive have no power during "Emergency stop", i.e. the PLC must react.

- Functional state: "Fixed stop not reached"
- Functional state: "Fixed stop aborted"
- Aborted by the PLC user program:
  DB31, ... DBX62.4 = 0 ("Activate travel to fixed stop")
- Cancellation of the pulse enable and machine data parameterization:
  MD37002 $MA_FIXED_STOP_CONTROL, bit 0 = 0 (see above)

3.2.5 Behavior with regard to other functions

Measurement with delete distance-to-go

"Travel to fixed stop" (FXS) cannot be programmed in a block with "Measurement with delete distance-to-go" (MEAS). Except when on function acts on a path axis while the other acts on a positioning axis or both functions act on positioning axes.

Contour monitoring

The axis contour monitoring function is inoperative while "Travel to fixed stop" is active.

Positioning axes

For "Travel to fixed stop" with positioning axes `POSA`, the block change is executed even when the positioning axis has not yet reached the fixed stop by this time.

Vertical axes

The "Travel to fixed stop" function can be used for vertical axes even when alarms are active.

If a function-specific alarm occurs for a vertical axis when traveling to fixed stop, the NC/PLC interface signal DB11, DBX6.3 (mode group ready) is not reset. This means that the corresponding drive is not de-energized.

This corresponds to an electronic weight compensation for the vertical axis and can be configured via the following machine data:

MD37052 $MA_FIXED_STOP_ALARM_REACTION

Further information

Further information on vertical axes can be found in:

- Function Manual Drive Functions SINAMICS S120
- Function Manual Monitoring and Compensating; Compensation, electronic weight counterbalance

3.2.6 Setting data

The values programmed via the function-specific `FXS`, `FXST` and `FXSW` commands are written block-synchronously to the following, immediately effective, axis-specific setting data:

Switching the function on/off

SD43500 $SA_FIXED_STOP_SWITCH (selection/deselection of travel to fixed stop)

Clamping torque

SD43510 $SA_FIXED_STOP_TORQUE (clamping torque)

Note

Clamping torque greater than 100%

A value for the clamping torque in SD43510 greater than 100% of the maximum motor torque is only advisable for a short time. In addition, the maximum motor torque is limited by the drive. For example, the following drive parameters have a limiting effect:

- p1520/p1521 upper torque limit/force limit / lower torque limit/force limit
- p1522/p1523 upper torque limit/force limit / lower torque limit/force limit
- p1530/p1531 power limit motorized / power limit regenerative
- p0640 current limit
- p0326 motor stall force correction factor

Detailed information on the drive parameters and the functions can be found in:

Further information

- SINAMICS S120/S150 List Manual
- Function Manual Drive Functions SINAMICS S120

Monitoring window

SD43520 $SA_FIXED_STOP_WINDOW (monitoring window)

Default setting

The defaults for the setting data are set via the following machine data:

- Clamping torque:
  MD37010 $MA_FIXED_STOP_TORQUE_DEF (default clamping torque)
- Monitoring window:
  MD37020 $MA_FIXED_STOP_WINDOW_DEF (default monitoring window)

Effectiveness

The setting data for the clamping torque and monitoring window takes effect immediately. In this way, the clamping state can be adapted to the machining situation at any time by the operator or via the PLC user program.

Further information

Further detailed information on the machine and setting data can be found in:

List Manual, Detailed Machine Data Description

3.2.7 System variables

Reference/actual state

The reference and actual state of the "travel to fixed stop" function can be read using the following system variables:

- $AA_FXS = <value> (status, "Travel to fixed stop" reference state)
- $VA_FXS = <value> (status, "Travel to fixed stop" actual state)

<value>	Description
0	Axis is not at fixed stop
1	Fixed stop was successfully approached
2	Approach to fixed stop failed
3	Selection "Travel to fixed stop" is active.
4	Fixed stop was detected.
5	Travel to fixed stop deselection active

Additional information

If errors occurred when traversing to the fixed stop ($VA_FXS == 2), then additional information is displayed in the following system variables:

- $VA_FXS_INFO = <value> (additional information when "traveling to fixed stop")

<value>	Description
0	No additional information available
1	No approach motion programmed
2	Programmed end position reached, motion stopped
3	Cancellation using a reset, DB21, ... DBX7.7 == 1
4	Monitoring window was exited
5	Torque reduction rejected by drive
6	PLC has withdrawn the enable, DB31, ... DBX3.1 == 0 (travel to fixed stop enabled)

Application example for $AA_FXS

In order that a block change is executed, no alarm should be triggered when a fault occurs. The cause is then determined using the system variable $VA_FXS_INFO and a specific response initiated.

Requirement: MD37050 $MA_FIXED_STOP_ALARM_MASK = 0 (when traveling to fixed stop, no alarm initiated)

Program code	Comment
`X300 Y500 F200 FXS[X1]=1 FXST[X1]=25 FXSW[X1]=5`	`; Travel to fixed stop`
`IF $AA_FXS[X1] == 2 GOTOF FXS_ERROR`	`; IF fixed stop == reached`

```
Program code                                                          Comment
   R100=$AA_IM[X1]                                                    ; THEN (normal case)
   IF R100 ...                                                        ; Evaluation of the
   ...                                                                ; actual position
   GOTOF PROG_END
FXS_ERROR:                                                            ; ELSE (error case)
   CASE($VA_FXS_INFO[X1]) OF 0 GOTOF LABEL_0 OF 1 GOTOF LABEL_1 ...   ; Error handling
LABEL_0: ...
      GOTOF CASE_END
LABEL_1: ...
      GOTOF CASE_END
...
CASE_END:
...
PROG_END: M30                                                         ; ENDIF end of program
```

3.2.8 Alarms

Alarm 20091 "Fixed stop not reached"

If the fixed stop position is not reached during travel to fixed stop, alarm 20091 "Fixed stop not reached" is displayed and a block change executed.

Alarm 20092 "Travel to fixed stop is still active"

If there is a travel request or renewed function selection for the axis after the fixed stop has been reached, alarm 20092 "Travel to fixed stop is still active" is displayed.

Alarm 20093 "Standstill monitoring at fixed stop has triggered"

If an axis has reached the fixed stop and is then moved out of this position by more than the value specified in the setting data

SD43520 FIXED_STOP_WINDOW (fixed stop monitoring window)

alarm 20093 "Standstill monitoring at fixed stop has triggered" is displayed, travel to fixed stop for this axis is deselected and the following system variable set:
$AA_FXS[x] = 2

Alarm 20094 "Function has been aborted"

The travel to fixed stop is aborted if the clamping torque can no longer be applied due to the cancellation of the pulse enable, or the requested acknowledgement signal at the NC/PLC interface has been reset:

- Acknowledgement signal required: MD37060 $MA_FIXED _STOP_ACKN_MASK, bit 0 = 1
- Acknowledgement signal: DB31, ... DBX3.1 == 0 (enable travel to fixed stop)

Enabling the fixed stop alarms

The following machine data can be use to set whether the fixed stop alarms

- Alarm 20091 "Fixed stop not reached",
- Alarm 20094 "Fixed stop aborted"

are displayed:

MD37050 $MA_FIXED_STOP_ALARM_MASK (enable of the fixed stop alarms)

Settable functional behavior for fixed stop alarms

The following machine data can be used to set that the function is not aborted when function-specific alarms occur:

- Alarm 20090 Travel to fixed stop not possible
- Alarm 20091 Fixed stop not reached
- Alarm 20092 Travel to fixed stop is still active
- Alarm 20093 Standstill monitoring at fixed stop has triggered
- Alarm 20094 Travel to fixed stop aborted

MD37052 $MA_FIXED_STOP_ALARM_REACTION (reaction to fixed stop alarms)

Alarm suppression after new programming

Travel to fixed stop can be used for simple measuring processes.

For example, it is possible to carry out a check for tool breakage by measuring the tool length by traversing onto a defined obstacle. To do so, the fixed stop alarm must be suppressed. When the function for clamping workpieces is then used "normally," the alarm can be activated using part program commands.

3.2.9 Travel with limited torque/force FOC

Function

Using the "Force Control" function to traverse with limited torque/force, when traversing, the maximum permissible torque/force can be limited to a percentage of the maximum possible axis torque. The limit value can be changed at any time, also while the axis is traversing. The changes can be realized in the interpolator clock cycle, dependent on distance, time or on any other variable. The "Force Control" function can also be programmed in synchronized actions. The function can be permanently activated, or block related.

Programming

Syntax

```
FOCON[<axis>]
FOCOF[<axis>]
```

FOC[<axis>]

Meaning

Parameter	Meaning
FOCON:	Activate torque/force limiting
FOCOF:	Deactivate torque/force limiting
FOC:	Activate non-modal torque/force limiting
<Axis>:	Channel axis name, type: AXIS

Example

Program code	Comment
N10 FOCON[X]	; Modal activation of the torque limit
N20 X100 Y200 FXST[X]=15	; X travels with reduced torque (15%)
N30 FXST[X]=75 X20	; Changing the torque to 75%,
N40 FOCOF[X]	; Disable torque limit

Parameterization

Machine data

- MD37010 $MA_FIXED_STOP_TORQUE_DEF (default for fixed stop clamping torque)
  The value specified in the machine data is effective after activating the function, as long as no explicit value is programmed using FXST.
- MD36042 $MA_FOC_STANDSTILL_DELAY_TIME (delay time standstill monitoring for active torque/force limiting)

Permanent activation (FOCON/FOCOF)

Permanent activation of the "Force Control" function after POWER ON or RESET can be set using the machine data:

MD37080 $MA_FOC_ACTIVATION_MODE, bit 0 and bit 1 (basic setting of the modal force/ torque limiting)

Bit	Value	Meaning
0	0	The "Force Control" function is **not** active after POWER ON.
	1	The "Force Control" function is active after POWER ON.
1	0	The "Force Control" function is **not** active after RESET.
	1	The "Force Control" function is active after RESET.

If a limiting torque or force is programmed using FXST = <value> before activation, then this is effective from activation onwards.

If a limiting torque or force was not programmed using FXST = <value> before activation, then the value specified in the machine data is effective as standard.

MD37010 $MA_FIXED_STOP_TORQUE_DEF (default for fixed stop clamping torque)

Block-related activation (FOC)

The function for the actual block is activated using the FOC command.

Activating the function using a synchronized action takes effect up to the end of the current part program block.

Priority of FXS before FOC

Activating the "Travel to fixed stop" function FXS has a higher priority than the "Force Control" function. If "Travel to fixed stop" is active at the same time as "Force Control", then the first is executed.

Deselecting the "Travel to fixed stop" function FXS cancels the clamping. A "Force Control" function that is at same time permanently active, remains active.

System variable

Status of the "Force Control" function

The status of the "Force Control" function can be read using system variable $AA_FOC.

Changing the status of the "Travel to fixed stop" function FXS does not change the status of the "Force Control" function.

$AA_FOC	
Value	**Meaning**
0	FOC not active
1	FOC modal active
2	FOC non-modal active

Status of torque limiting

The actual status of the torque limiting can be read using system variable $VA_TORQUE_AT_LIMIT.

$VA_TORQUE_AT_LIMIT	
Value	**Meaning**
0	actually effective torque < torque limit value
1	actually effective torque == torque limit value

Restrictions

the "Force Control" function has the following restrictions:

- The change of the torque/force limitation representing itself as an acceleration limitation is **only taken into account in the traversing movement at block limits** (see command ACC).
- Only FOC: **No monitoring** is possible from the NC/PLC interface to check that the active torque limit has been reached.

- If the acceleration limitation is not adapted accordingly, an increase in the following error occurs during the traversing motion.
- If the acceleration limitation is not adapted accordingly, the end-of-block position is possibly reached later than specified in the machine data:
  MD36040 $MA_STANDSTILL_DELAY_TIME
  The machine data:
  MD36042 $MA_FOC_STANDSTILL_DELAY_TIME
  is introduced for this and monitored in this status.

3.3 Examples

Example 1: Travel to fixed stop with static synchronized actions

Travel to fixed stop (FXS) is initiated when requested via R parameter ($R1) in a static synchronized action.

```
Program code                        Comment
N10 IDS=1 WHENEVER                  ; Static synchronized action 1:
(($R1==1) AND                       ; R1==1 (FXS for Y requested) AND
($AA_FXS[Y]==0)) DO                 ; Avoidance of multiple selection
                                    ; $AA_FXS[Y]==0 (axis not at limit): =>
$R1=0 FXS[Y]=1                      ; reset $R1, activate FXS for Y
FXST[Y]=10                          ; Limit torque: 10%
FA[Y]=200                           ; Axial feedrate Y: 200
POS[Y]=150                          ; Positioning movement Y

N11 IDS=2 WHENEVER                  ; Static synchronized action 2:
($AA_FXS[Y]==4) DO                  ; $AA_FXS[Y]==4 (limit detected): =>
FXST[Y]=30                          ; Limit torque:  30%

N12 IDS=3 WHENEVER                  ; Static synchronized action 3:
($AA_FXS[Y]==1) DO                  ; $AA_FXS[Y]==1 (limit reached successfully): =>
FXST[Y]=$R0                         ; Limit torque: Value from R parameter $R0

N13 IDS=4 WHENEVER                  ; Static synchronized action 4:
(($R3==1) AND                       ; R3==1: Deselection of FXS for Y requested
($AA_FXS[Y]==1)) DO                 ; $AA_FXS[Y]==1 (limit reached successfully): =>
FXS[Y]=0                            ; Deselect FXS
FA[Y]=1000 POS[Y]=0                 ; Positioning movement Y

N20 FXS[Y]=0 G0 G90 X0 Y0           ; Set initial settings: FXS deselected,
                                    X and Y at initial position
N30 RELEASE(Y)                      ; Release Y for movements in synchronized actions
...
```

Program code	Comment
N60 GET(Y)	Include axis Y back into the path group
...	

Note

Avoidance of multiple selection for FXS

To avoid a multiple selection, we recommend that prior to activating FXS, query either the $AA_FXS==0 system variable or a user-specific flag. See above, program example N10

Example 2: Traveling to fixed stop with block-related synchronized actions

"Travel to fixed stop" is activated from a specific position of the traversing motion of the following block

Program code	Comment
N10 G0 G90 X0	; Starting position
N20 WHEN $AA_IW[X]>17 DO FXS[X]=1	; Synchronized action: Actual position of X axis > 17
	; => activate FXS for axis X
N30 G1 F200 X100	; Traversing motion for axis X

Note

A block-related synchronized action is processed in the following main program block.

3.4 Data lists

3.4.1 Machine data

3.4.1.1 Axis/spindlespecific machine data

Number	Identifier: $MA_	Description
36042	FOC_STANDSTILL_DELAY_TIME	Delay time, standstill monitoring for `active torque/force limiting`
37000	FIXED_STOP_MODE	Travel to fixed stop mode
37002	FIXED_STOP_CONTROL	Sequence monitoring for travel to fixed stop
37010	FIXED_STOP_TORQUE_DEF	Fixed stop clamping torque default setting
37012	FIXED_STOP_TORQUE_RAMP_TIME	Time until the modified torque limit is reached
37020	FIXED_STOP_WINDOW_DEF	Default for fixed stop monitoring window
37030	FIXED_STOP_THRESHOLD	Threshold for fixed stop detection

Number	Identifier: $MA_	Description
37040	FIXED_STOP_BY_SENSOR	Fixed stop detection via sensor
37050	FIXED_STOP_ALARM_MASK	Enabling the fixed stop alarms
37052	FIXED_STOP_ALARM_REACTION	Reaction to fixed stop alarms
37060	FIXED_STOP_ACKN_MASK	Monitoring of PLC acknowledgments for travel to fixed stop
37070	FIXED_STOP_ANA_TORQUE	Torque limit on fixed stop approach for analog drives
37080	FOC_ACTIVATION_MODE.	Initial setting of the modal torque/force limiting

3.4.2 Setting data

3.4.2.1 Axis/spindle-specific setting data

Number	Identifier: $SA_	Description
43500	FIXED_STOP_SWITCH	Selection of travel to fixed stop
43510	FIXED_STOP_WINDOW	Fixed stop clamping torque
43520	FIXED_STOP_TORQUE	Fixed stop monitoring window

P1: Transverse axes

4

4.1 Function

Transverse axis

Within the framework of "turning" technology, the transverse axis refers to the machine axis that travels perpendicular to the axis of symmetry of the spindle, in other words, to longitudinal axis Z.

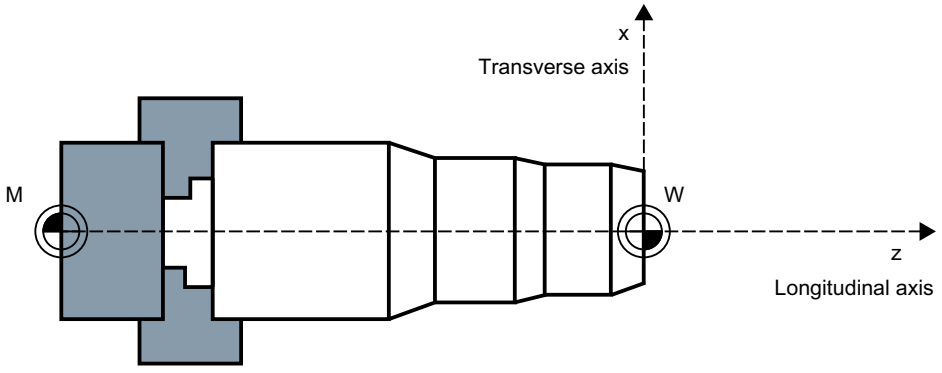


Figure 4-1 Position of the transverse axis in the machine coordinate system

Geometry axis as transverse axis

Every geometry axis of a channel can be defined as a transverse axis.

For the geometry axis as a transverse axis, the following functions can be simultaneously or separately permitted and activated:

- Programming and display in the diameter
- Reference axis for constant cutting speed G96/G961/G962

Several transverse axes in the channel

The introduction several transverse axes in the channel involves a functional decoupling of diameter programming and reference axis for G96/G961/G962. Diameter programming and reference axis for G96/G961/G962 can be active for different transverse axes (see table below).

	Programming and display in the diameter			**Reference axis for G96 / G961 / G962**	
Permissible axis type:	Geometry axis		Linear channel axes	Geometry axis	
Selection in the channel:	one	m of 3	m of n	one	one of 3
Specific effect: **Machine data:**	Channel MD20100	Axis MD30460		Channel MD20100	

	Programming and display in the diameter	Reference axis for G96 / G961 / G962
Programming:	DIAM* channel-specific modal G group 29	SCC[<Axis>] channel-specific modal Reference axis for G96/G961/G962
Acceptance during axis replacement:	DIAM*A[<Axis>] axis-specific modal	
Axis-specific non-modal diametral/radius programming:	DAC, DIC; RAC, RIC blockwise axis-specific only programming	
DIAM*: DIAMOF, DIAMON, DIAM90, DIAMCYCOF DIAM*A[<Axis>]: DIAMOFA[<Axis>], DIAMONA[<Axis>], DIAM90A[<Axis>], DIACYCOFA[<Axis>], DIAMCHANA[<Axis>] <Axis>: Axis name for geometry, channel or machine axis name		

Note

Rotary axes are not permitted to serve as transverse axes.

Diameter-related data

DIAMON/DIAMONA[<Axis>]

After activation of the diameter programming with DIAMON/DIAMONA[<Axis>] (see "Programming (Page 121)"), the following data refer to diameter dimensions:

- Display data of transverse axis in the workpiece coordinate system:
  - Setpoint and actual position
  - Distance-to-go
  - REPOS offset
- "JOG" mode:
  - Increment (INC) through incremental manual handwheel traversing (depending on the active MD)
- Part program programming:
  - End positions, independent of reference mode (G90 / G91)
  - Interpolation parameters of circular-path programming (G2 / G3) if these are programmed with part program instruction AC absolute.
- Actual values read with reference to the workpiece coordinate system (WCS):
  - $AA_MW[<transverse axis>]
    System variable of the measuring functions MEAS (measuring with delete distance-to-go) and MEAW (measuring without delete distance-to-go)
  - $P_EP[<transverse axis>]
  - $AA_IW[<transverse axis>]

DIAM90/DIAM90A[<Axis>]

After activation of the reference-mode-dependent diameter programming with `DIAM90/DIAM90A[<Axis>]`, the following data is always displayed in relation to diameter regardless of the operating mode (`G90`/`G91`):

- Actual value
- Actual values read with reference to the workpiece coordinate system (WCS):
  - $AA_MW[<transverse axis>]
    System variable of the measuring functions `MEAS` (measuring with delete distance-to-go) and `MEAW` (measuring without delete distance-to-go)
  - $P_EP[<transverse axis>]
  - $AA_IW[<transverse axis>]

DIAMCYCOF/DIACYCOFA[<Axis>]

With `DIAMCYCOF` / `DIACYCOFA[<Axis>]`, a changeover to radius programming takes place within the controller. The diameter programming status that was active before `DIAMCYCOF` or `DIACYCOFA[AX]` continues to be displayed to the HMI.

Permanently radius-related data

For transverse axes, the following data is **always** entered, programmed and displayed in relation to radius:

- Part program programming
  - Interpolation parameters of circular-path programming with CIP
- Offsets:
  - Tool offsets
  - Programmable and configurable frames
  - External work offset
  - DRF and preset offset
  - etc.
- Working area limitation
- Software limit switch
- Feedrate
- Display data with reference to the machine coordinate system
- Display data of the service images for axis, FSD and MSD

Extended functions for data that is always radius-related:

The following applies for PLC axes, via FC18 or axes controlled exclusively from the PLC:

- The dimension for PLC axes in the radius also applies to several transverse axes with diameter function and is independent of channel-specific or axis-specific diameter programming.
- In the JOG mode (Inc) a PLC axis is subordinate to the channel status. If diameter programming is active and MD20624 $MC_HANDWH_CHAN_STOP_COND bit 15 = 0, only half the path of the specified increment is traversed.

4.2 Parameterization

Defining a geometry axis as a transverse axis

Defining a geometry axis as a transverse axis in the channel is realized using machine data:

MD20100 $MC_DIAMETER_AX_DEF (geometry axis with transverse axis function)

Example

MD20100 $MC_DIAMETER_AX_DEF="X" ; geometry axis X is the transverse axis in the channel.

For this axis, diameter programming and assigning a constant cutting speed with G96 / G961 / G962 are both permitted.

Several transverse axes in the channel

Defining additional transverse axes in the channel, for which the functionality of the axis-specific diameter programming is permitted, is performed using the machine data:

MD30460 BASE_FUNCTION_MASK.Bit 2 = <Value>

Bit	Value	Meaning
2	0	Axis-specific diameter programming is **not** permitted.
	1	Axis-specific diameter programming is permitted.

Note

The setting MD30460 bit 2 = 1 is only possible for linear axes.

Boundary conditions

In a channel, a transverse axis can be defined as channel-specific (MD20100) and axis-specific (MD30460, bit 2) axis at the same time. The channel-specific machine data has the higher priority.

With MD20100, the following channel-specific functions are assigned to the transverse axis during the run-up

The following axis-specific basic position is assigned to the transverse axis during the power-up of the NC: "Transfer of the diameter programming channel status" DIAMCHANA[<Axis>]

With the release of the axis-specific diameter programming (MD30460, bit 2 = 1), the following axis-specific operations can be used at the user's end:

- DIAMONA (axis-specific modal diameter programming: ON)
- DIAMOFA (axis-specific modal diameter programming: OFF)
- DIAM90A (axis-specific modal diameter programming for G90 and AC, radius programming for G91 and IC)
- DIACYCOFA (axis-specific modal diameter programming: OFF in cycles)
- DIAMCHANA (transfer of the diameter programming channel status)

Dimensions for tool parameters

With the following channel-specific machine data, the following tool parameters can be activated as diameter values for all of the transverse axes defined in the channel:

MD20360 $MC_TOOL_PARAMETER_DEF_MASK.<Bit> = <value>

MD20360 $MC_TOOL_PARAMETER_DEF_MASK		
Bit	**Value**	**Meaning**
1	1	Transverse axis tool length as a diameter
2	1	Alarm for wear or tool length as a diameter and plane change
3 [1)]	0	Work offset $P_EXTFRAME and frames For transverse axes, work offsets in frames are always calculated as radius values.
	1	The work offset (WO) in frames in the transverse axis as a diameter The frame stores the work offsets internally as a radius value. There is no conversion during a change of diameter to radius programming or vice versa.
4	1	Preset value as a diameter
5 [1)]	0	External work office (axis overlay) For transverse axes, the external work offset is always calculated as a radius value.
	1	External WO of the transverse axis as a diameter There is no conversion during a change of diameter to radius programming or vice versa.
6	1	Actual values of the transverse axis as a diameter
7	1	Display of actual values of the transverse axis as a diameter value
8 [1)]	1	Display of remaining path in WCS always as a radius
9 [1)]	For all of the transverse axes for which it is set that the specifications of the handwheel are path specifications (MD11346 $MN_HANDWH_TRUE_DISTANCE == 1), the following applies:	
	0	Half of the path of the specified handwheel increment is traveled if channel-specific or axis-specific diameter programming is active for this axis.
	1	Half of the path of the specified handwheel increment is always traveled.
10	1	Tool portion of an active tool carrier that can be oriented if no tool is active
11	1	Evaluation of $TC_DP6 as a diameter
12	1	Evaluation of $TC_DP15 as wear of the tool diameter

MD20360 $MC_TOOL_PARAMETER_DEF_MASK		
Bit	**Value**	**Meaning**
13 [1)]	1	When jogging around circles, the circle center point coordinate is always a radius value, see SD42690 $SC_JOG_CIRCLE_CENTRE
14 [1)]	1	For cycle masks, the absolute values of the transverse axis are always meant as the radius.
15	1	Incremental values of the transverse axis for cycle masks as diameters
1) The function is dependent upon the settings in MD20100 $MC_DIAMETER_AX_DEF and MD30460 $MA_BASE_FUNCTION_MASK, bit 2		

Behavior during manual handwheel traversing

The behavior during manual handwheel traversing (MD11346 $MN_HANDWH_TRUE_DISTANCE == 1 or 3) of a transverse axis while diameter programming is active can be adjusted with the following machine data:

MD20624 $MC_HANDWH_CHAN_STOP_COND.Bit 15 = <value>

Bit	Value	Meaning
15	0	Only the half path of the specified increment is traveled.
	1	The specified increment is traveled completely.

Displaying position values in the diameter

Position values of transverse axes are always displayed as diameter values, if:

MD27100 $MC_ABSBLOCK_FUNCTION_MASK, bit 0 = 1

Channel-specific basic position after power up, reset

The channel-specific basic position after power up or channel reset or the part program end of G group 29 (`DIAMON`, `DIAM90`, `DIAMOF`, `DIAMCYCOF`) is defined by the machine data MD20110 $MC_RESET_MODE_MASK.

MD20110 $MC_RESET_MODE_MASK, bit 0 = 0:

- Power-up: Basic position according to MD20150 $MC_GCODE_RESET_VALUE
- Channel reset or part program end: Basic position according to MD20150 $MC_GCODE_RESET_VALUES

MD20110 $MC_RESET_MODE_MASK, bit 0 = 1:

- After power-up: Basic position according to MD20150 $MC_GCODE_RESET_VALUE
- After channel reset or part program end: Basic position according to MD20152 $MC_GCODE_RESET_MODE

The user can set the respective desired status via an event-driven program call (ProgEvent).

If the basic position after the power-up is `G96` / `G961` / `G962`, a transverse axis must be defined with MD20100 $MC_DIAMETER_AX_DEF, otherwise the alarm 10870 is displayed.

To ensure that the reference axis for G96 / G961 / G962 is retained during a reset, end of part program or start of part program, the following setting must be made:

- Channel reset or part program end: MD20110 $MC_RESET_MODE_MASK, bit 18 = 1
- Part program start: MD20112 $MC_START_MODE_MASK, bit 18 = 1

A reference axis for G96 / G961 / G962 can also be assigned without the parameterization of a transverse axis in MD20100 via SCC[AX]. For this scenario, the constant cutting speed must not already be activated with G96. For additional information, see the following:

Further information

Programming Manual Fundamentals, Feedrate Control and Spindle Motion
"Constant cutting rate (G96/G961/G962, G97/G971/G972, G973, LIMS, SCC)

4.3 Programming

Transverse axes can be programmed with respect to both diameter and radius. Generally, they are diameter-related, i.e. programmed with doubled path dimension so that the corresponding dimensional information can be transferred to the part program directly from the technical drawings.

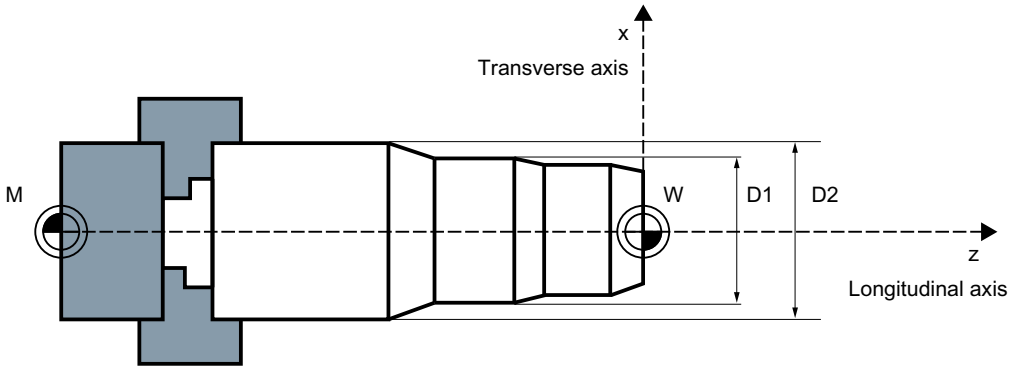


Figure 4-2 Transverse axis with diameter information (D1, D2)

Switching the diameter programming on/off

Channel-specific diameter programming

The activating or deactivating of the diameter programming is done via the modally active part program statements of the G group 29:

- DIAMON: Diameter programming ON
- DIAMOF: Diameter programming OFF, in other words, radius programming ON
- DIAM90: Diameter or radius programming depending on the reference mode:
  - Diameter programming ON in connection with absolute dimensioning G90
  - Radius programming ON in connection with incremental dimensioning G91
- DIAMCYCOF: Radius programming for G90 and G91 ON, for the HMI, the last active G command of this group remains active

Reference is made exclusively to the transverse axis of the channel.

Axis-specific diameter programming for several transverse axes in one channel

Note

The additionally specified axis must be activated via MD30460 $MA_BASE_FUNCTION_MASK with bit 2 = 1.

The axis specified must be a known axis in the channel. Geometry, channel or machine axes are permitted.

Programming is not permitted in synchronized actions.

The following axis-specific modal statements can be programmed several times in a part program block:

- `DIAMONA[<Axis>]`: Diameter programming for `G90`, `G91`, `AC` and `IC` ON
- `DIAMOFA[<Axis>]`: Diameter programming OFF, in other words, radius programming ON
- `DIAM90A[<Axis>]`: Diameter or radius programming depending on the reference mode:
  - Diameter programming ON in connection with absolute dimensioning `G90` and `AC`
  - Radius programming ON in connection with incremental dimensioning `G91` and `IC`
- `DIACYCOFA[Axis]`: Radius programming for `G90` and `G91` ON, for the HMI, the last active G command of this group remains active
- `DIAMCHANA[Axis]`: Acceptance of diameter programming channel status
- `DIAMCHAN`: All axes with MD30460, bit 2 = 1 accept the diameter programming channel status

Axis-specific modal statements have priority over the channel setting.

4.4 Supplementary conditions

Axis interchange

Due to a `GET` request from the parts program, the diameter programming status for an additional transverse axis is accepted in the new channel during axis replacement using `RELEASE[<Axis>]`.

Axis interchange in synchronized actions

For axis replacement in synchronized actions, a transverse axis takes the status of the axis-specific diameter programming with it into the new channel if:

- with MD30460, bit 2 = 1 axis-specific diameter programming is permitted for the transverse axis.
- the transverse axis is not subordinated to the channel-specific diameter programming in the releasing channel.

The active dimension can be queried via the system variable $AA_DIAM_STAT[<Axis>].

4.5 Examples

Example 1

X is a transverse axis defined via MD20100 $MC_DIAMETER_AX_DEF.

Y is a geometry axis and U is an additional axis. These two axes are additional transverse axes with diameter specifications defined in MD30460 $MA_BASE_FUNCTION_MASK with bit 2 = 1.

DIAMON is not active after power up.

Program code	Comment
N10 G0 G90 X100 Y50	;no diameter programming is active
N20 DIAMON	;Channel-specific diameter programming, in effect for X
N30 Y200 X200	;Dimensions: X in the diameter, Y in the radius
N40 DIAMONA[Y]	;axis-specific modal diameter programming, ;in effect for Y
N50 Y250 X300	;Dimensions: X and Y in diameter
N60 DIAM90	;Dimensions: X G90/AC in the diameter, G91/IC in the radius
N70 Y200	;Y: continuing, axis-specific modal diameter programming
N75 G91 Y20 U=DIC(40)	;Dimensions: Y in the diameter, U non-modally IC in the diameter
N80 X50 Y100	;Dimensions: X in the radius (G91), Y in the diameter
N85 G90 X100 U200	;Dimensions: X in the diameter, U in the radius
N90 DIAMCHANA[Y]	;Y accepts the channel status DIAM90
N95 G91 X100 Y100	;Dimensions: X and Y in the radius(G91)
N100 G90 X200 Y200	;Dimensions: X and Y in diameter

Example 2

Transverse axes with diameter specification applied as in the previous example.

X and Y are located in channel 1 and are also known in channel 2; i.e. permitted for axis replacement.

Program code	Comment
Channel 1	
N10 G0 G90 X100 Y50	;no diameter programming is active
N20 DIAMON	;Channel-specific diameter programming for X
N30 Y200 X200	;Dimensions: X in the diameter, Y in the radius
N40 DIAMONA[Y]	;Y axis-specific modal diameter programming
N50 Y250 X300	;Dimensions: X and Y in diameter
N60 SETM(1)	;Synchronous marker 1
N70 WAIT(1,2)	;wait for synchronous marker 1 in channel 2
Channel 2	
...	
N50 DIAMOF	;channel 2 no diameter programming active

Program code	Comment
...	
N100 WAIT(1,1)	;wait for synchronous marker 1 in channel 1
N110 GETD(Y)	;Axis replacement direct Y
N120 Y100	;Y is the channel-specific diameter programming ;subordinated in channel 2; i.e. dimension in the radius

4.6 Data lists

4.6.1 Machine data

4.6.1.1 Channelspecific machine data

Number	Identifier: $MC_	Description
20050	AXCONF_GEOAX_ASSIGN_TAB[n]	Assignment of geometry axis to channel axis
20060	AXCONF_GEOAX_NAME_TAB[n]	Channel axis name in the channel
20100	DIAMETER_AX_DEF	Geometry axis with transverse axis function
20110	RESET_MODE_MASK	Definition of the control basic setting after power-up and RESET / part program end
20112	START_MODE_MASK	Definition of the control basic setting for NC start
20150	GCODE_RESET_VALUES[n]	Basic setting of the G groups
20152	GCODE_RESET_MODE[n]	G command basic setting at RESET / part program end
20360	TOOL_PARAMETER_DEF_MASK	Definition of tool parameters
20624	HANDWH_CHAN_STOP_COND	Definition of the behavior of the manual handwheel traversing
27100	ABSBLOCK_FUNCTION_MASK	Parameterize block display with absolute values

4.6.1.2 Axis/spindlespecific machine data

Number	Identifier: $MA_	Description
30460	BASE_FUNCTION_MASK	Axis functions

V1: Feedrates

5

5.1 Brief description

Types of feedrate

The feedrate determines the machining speed (axis or path velocity) and is observed in every type of interpolation, even where allowance is made for tool offsets on the contour or on the tool center point path (depending on G commands).

The following types of feedrate allow optimum adaptation to the various technological applications (turning, milling, drilling, etc.):

- Rapid traverse feedrate (G0)
- Inverse-time feedrate (G93)
- Linear feedrate (G94)
- Revolutional feedrate (G95)
- Constant cutting speed (G96, G961)
- Constant speed (G97, G971)
- Feedrate for thread cutting (G33, G34, G35)
- Feedrate for tapping without compensating chuck (G331, G332)
- Feedrate for tapping with compensating chuck (G63)
- Feedrate for chamfer/rounding FRC, FRCM
- Non-modal feedrate FB

Axis assignment of the feedrates

Feedrates can be assigned to the axes variably to adjust to the different technological requirements.

The following variants are possible:

- Separate feedrates for the working plane and the infeed axis
- Variable axis assignment for path feedrate
- Feedrate for positioning axes

Feedrate control

The programmed feedrate can be changed during the machining or for test purposes to enable adjustment to the changed technological conditions.

- Via the machine control panel
- Via the operator panel front

- Via the PLC
- Per program command

5.2 Path feedrate F

Path feedrate F

The path feedrate represents the geometrical total of the speed components in the participating axes. It is therefore generated from the individual motions of the interpolating axes.

The default uses the axial speeds of the geometry axes which have been programmed. The `FGROUP` command can be used to include other geometry and/or synchronized axes in the calculation of the path feedrate.

The path feedrate F determines the machining speed and is observed in every type of interpolation even where allowance is made for tool offsets. The value programmed under the address F remains in a program until a new F value or a new type of feedrate is programmed.

Range of values for path feedrate F

See Description of Functions G2: "Speeds, Setpoint / Actual Value Systems, Closed-Loop Control", Section: "Velocities (Page 31)".

F value at PLC interface

The F value of the current path feedrate is always entered in the channel-specific PLC interface for auxiliary functions (DB21, ... DBB158 to 193).

For explanations of the associated interface signals (change signal, F value), see Function Manual "Basic Functions" "H2: Auxiliary function outputs to PLC".

Feedrate with transition circle

Further information
Programming Manual Fundamentals

Feedrate for internal radius and external radius path sections

For circular blocks or spline blocks with curvature in the same direction and tool radius offset activated (G41/G42), the programmed feedrate can act on the center point path or on the contour (depending on the internal radius or external radius path sections).

A group of G commands is provided for this purpose:

- CFTCP
  Programmed feedrate acting on the center point path.
- CFC
  Programmed feedrate acting on the contour.
- CFCIN
  Programmed feedrate acting only on the contour with a concave spline.

Further information
Programming Manual Fundamentals

Maximum tool path velocity

The maximum path velocity results from the maximum velocities of the linear or rotary axes involved (MD32000 $MA_MAX_AX_VELO), i.e. the axis with the lowest maximum velocity determines the maximum path velocity. This cannot be exceeded.

If `G0` is programmed, traversing is at the path velocity resulting from the MD32000 $MA_MAX_AX_VELO limitation.

Limit velocity for path axes

In addition, the `FL[<axis>]` statement can be used to program a limit velocity for path axes (geometry and synchronized axes).

This enables separate feedrates to be programmed for the working plane and infeed axis. This means that a feedrate is specified for the path-related interpolation and for the infeed axis. The axis perpendicular to the selected machining plane is designated as the infeed axis. The infeed axis-specific feedrate can be programmed to limit the axis velocity and therefore the path velocity. No coordinate rotations through frames should be included, i.e. the infeed axis must be an axis of the standard coordinate system. This function can be used to compensate for the fact that a cutter has a lower cutting performance on the face side than across the cutter circumference.

Programming example:

Program code	Comment
`... G94 ...`	`; Selection of feedrate type (mm/min)`
`X30 Y20 F200`	`; Path feedrate = 200 mm/s`
`FL[Z]=50 Z-30`	`; Max. feedrate for Z axis: 50 mm/s`

Low-resolution encoders

When using low-resolution encoders, more continuous path or axis motions can be achieved with smoothed actual values. The larger the time constant, the better the smoothing of the actual values, and the longer the overtravel.

MD34990 $MA_ENC_ACTVAL_SMOOTH_TIME[<axis>] (smoothing time constant for actual values)

Smoothed actual values are used for:

- Thread cutting (G33, G34, G35)
- Feedrate per revolution (G95, G96, G97, FPRAON)
- Display of speed, actual position and velocity

5.2.1 Feedrate type G93, G94, G95

Effectiveness

The feedrate types G93, G94, G95 are active for the G commands of group 1 (except G0) in the automatic modes.

G94 or G95 can be used for traversing in JOG mode.

Further information
Manual traversing and manual traversing using a handwheel (Page 633)

Inverse-time feedrate (G93)

The inverse-time feedrate is used when it is easier to program the duration, rather than the feedrate, for retraction of a block.

The inverse-time feedrate is calculated from the following formula:

```
F = v / s
```

with `F`: Inverse-time feedrate in rpm
`v`: Required path velocity in mm/min or inch/min
`s`: Path length in mm/inch

Programming example

Program code	Comment
`N10 G1 G93 X100 Y200 F2`	`; The programmed path is traversed in 0.5 min.`
`...`	

Note

`G93` may not be used when `G41` / `G42` is active. If the block length varies greatly from block to block, a new F value should be programmed in each block for `G93`.

Linear feedrate (G94)

The linear feedrate is programmed in the following units relative to a linear or rotary axis:

- [mm/min, degrees/min] on standard metric systems
- [inch/min, degrees/min] on standard imperial systems

Revolutional feedrate (G95)

The revolutional feedrate is programmed in the following units relative to a master spindle:

- [mm/rev] on standard metric systems
- [inch/rev] on standard imperial systems
- [degrees/rev] on a rotary axis

The path velocity is calculated from the actual speed of the spindle according to the following formula:

```
V = n * F
```

with V: Path velocity in mm/min or inch/min
n: Speed of the master spindle in rpm
F: Programmed revolutional feedrate in mm/rev or inch/rev

Note

The programmed F value is deleted when the system switches between the feedrate types G93, G94 and G95.

Tooth feedrate

Primarily for milling operations, the tooth feedrate FZ... (feed distance per tooth), which is more commonly used in practice, can be programmed instead of the revolutional feedrate F...:

The control system uses the $TC_DPNT (number of teeth per revolution) tool parameter associated with the active tool offset data record to calculate the effective revolutional feedrate for each traversing block from the programmed tooth feedrate.

```
F = FZ * $TC_DPNT
```

with F: Revolutional feedrate in mm/rev or inch/rev
FZ: Tooth feedrate in mm/tooth or inch/tooth
$TC_DPNT: Tool parameter: Number of teeth/rev

Example: Milling cutter with 5 teeth ($TC_DPNT = 5)

```
Program code                Comment
N10 G0 X100 Y50
N20 G1 G95 FZ=0.02          ; Tooth feedrate 0.02 mm/tooth
N30 T3 D1                   ; Load tool and activate tool offset data block.
M40 M3 S200                 ; Spindle speed 200 rpm
N50 X20                     ; Milling with FZ = 0.02 mm/tooth
                            ; effective revolutional feedrate:
                            ; F = 0.02 mm/tooth * 5 teeth/rev = 0.1 mm/rev
                            ; or:
                            ; F = 0.1 mm/rev * 200 rpm = 20 mm/min
```

```
Program code                Comment
...
```

Setting data

Revolutional feedrate in JOG mode

The behavior of an axis in terms of its revolutional feedrate relative to the master spindle of the channel to which the axis is currently assigned in JOG mode depends on the settings in the NC-specific setting data:

SD41100 $SN_JOG_REV_IS_ACTIVE, Bit <x>

Bit	Value	Meaning
0	0	The behavior depends on further setting data, depending on the function of the axis: • **Axis** / **spindle**: SD43300 $SA_ASSIGN_FEED_PER_REV_SOURCE • **Geometry axis** on which a **frame with rotation** acts: SD42600 $SC_JOG_FEED_PER_REV_SOURCE • **Orientation axis**: SD42600 $SC_JOG_FEED_PER_REV_SOURCE
	1	The axis is traversed with the revolutional feedrate. **Boundary condition** Traversing is performed without the revolutional feedrate if the following applies: • **Axis** / **spindle**: The master spindle is stationary **AND** SD43300 $SA_ASSIGN_FEED_PER_REV_SOURCE == -3 • **Geometry axis** on which a **frame with rotation** acts: The master spindle is stationary **AND** SD42600 $SC_JOG_FEED_PER_REV_SOURCE == -3 • **Orientation axis**: The master spindle is stationary AND SD42600 $SC_JOG_FEED_PER_REV_SOURCE == -3
1	0	In **rapid traverse**, traversing is performed **with** the revolutional feedrate.
	1	In **rapid traverse**, traversing is performed **without** the revolutional feedrate.
2 1)	0	In **handwheel jogging**, traversing is performed **with** the revolutional feedrate.
	1	In **handwheel jogging**, traversing is performed **without** the revolutional feedrate.
3 1)	0	In **DRF handwheel traversing**, traversing is performed **with** the revolutional feedrate.
	1	In **DRF handwheel traversing**, traversing is performed **without** the revolutional feedrate.
1) The function is active in addition to the setting in bit 1.		

NC/PLC interface signals

- Revolutional feedrate active (**channel-specific**)
  The interface signal indicates that axes are traversed with revolutional feedrate in the channel:
  - AUTOMATIC mode: Path or synchronized axes
  - JOG mode: Geometry axes

  DB31, ... DBX62.2 == 1 (revolutional feedrate active)
- Revolutional feedrate active (**axis-specific**)
  The interface signal indicates that the axis is traversed with revolutional feedrate:
  DB31, ... DBX33.2 == 1 (revolutional feedrate active)

5.2.2 Type of feedrate G96, G961, G962, G97, G971

Constant cutting rate (G96, G961)

The constant cutting rate is used on turning machines to keep the cutting conditions constant, independently of the work diameter of the workpiece. This allows the tool to be operated in the optimum cutting performance range and therefore increases its service life.

Selection of G96, G961:

When programming G96, G961, the corresponding S value is interpreted as the cutting rate in m/min or ft/min along the transverse axis. If the workpiece diameter decreases during machining, the speed is increased until the constant cutting speed is reached.

When G96, G961 is first selected in the part program, a constant cutting rate must be entered in mm/min or ft/min.

With G96, the control system will automatically switch to revolutional feedrate (as with G95), i.e. the programmed feedrate F is interpreted in mm/rev or inch/rev.

When programming G961, linear feedrate is selected automatically (as with G94). A programmed feedrate F is interpreted in mm/min or inch/min.

Determining the spindle speed

Based on the programmed cutting rate (either S_{G96} or S_{G961}) and the actual cartesian position of the transverse axis (radius), the control system calculates the spindle speed at the TCP using the following formula:

$$n = \frac{S_{Speed}}{2 * \pi * r}$$

n:	Spindle speed
S_{Speed}:	Programmed cutting rate
π	Circle constant
r:	Radius (distance, center of rotation to TCP)

The following is assumed when determining the radius:

- The transverse axis position 0 in the WCS represents the center of rotation.
- Position offsets (such as online tool offset, external zero offset, $AA_OFF, DRF offset and compile cycles) and position components through couplings (e.g. following axis for TRAIL) are not taken into account when determining the radius.

Frames (e.g. programmable frames such as SCALE, TRANS or ROT) are taken into account in the calculation of the spindle speed and can bring about a change in speed, if the effective diameter at the TCP changes.

Diameter programming and reference axis for several transverse axes in one channel:

One or more transverse axes are permitted and can be activated simultaneously or separately:

- Programming and displaying in the user interface in the diameter
- Assignment of the specified reference axis with `SCC[<axis>]` for a constant cutting rate G96, G961, G962

For more information, see Description of Functions "P1: Transverse axes (Page 115)".

Example

S_{G96} = 230 m/min

- Where r = 0.2 m → n = 183.12 rpm
- Where r = 0.1 m → n = 366.24 rpm

⇒ The smaller the workpiece diameter, the higher the speed.

For G96, G961 or G962, a geometry axis must be defined as the transverse axis.

The transverse axis, whose position affects the speed of the mater spindle, is defined using channel-specific machine data:

MD20100 $MC_DIAMETER_AX_DEF (geometry axis with transverse axis function)

The function G96, G961 or G962 requires that the machine zero and the workpiece zero of the transverse axis are in the turning center of the spindle.

Constant speed (G97, G971)

G97, G971 deactivates the "Constant cutting rate function" (G96, G961) and saves the last calculated spindle speed. With G97, the feedrate is interpreted as a revolutional feedrate (as with G95). When programming G971, linear feedrate is selected (as with G94). The feedrate F is interpreted in mm/min or inch/min.

When G97, G971 is active, an S value can be reprogrammed to define a new spindle speed. This will not modify the cutting rate programmed in G96, G961.

G97, G971 can be used to avoid speed variations in motions along the transverse axis without machining (e.g. cutting tool).

Note

G96, G961 is only active during workpiece machining (G1, G2, G3, spline interpolation, etc., where feedrate F is active).

The response of the spindle speed for active G96, G961 and G0 blocks can be defined in the channel-specific machine data:

MD20750 ALLOW_G0_IN_G96 (G0 logic for G96, G961)

When constant cutting rate G96, G961 is selected, no gear stage change can take place.

The spindle override switch acts on the calculated spindle speed.

A DRF offset in the transverse axis does not affect the spindle speed setpoint calculation.

At the start of machining (after G0) and after NC stop, G60, G09, ... the path start waits for "nAct= nSet".

The interface signals "nAct = nSet" and "Set speed limited" are not modified by internal speed settings.

When the speed falls below the minimum speed or if the signal "Axis/spindle stationary" is recognized, "nAct =nSet" is reset.

A path operation which has started (G64, rounding), is not interrupted.

Spindle speed limitation with G96, G961

A maximum spindle speed can be specified for the "Constant cutting rate" function:

- In the setting data:
  SD43230 $SA_SPIND_MAX_VELO_LIMS (spindle speed limitation for G96/G961)
- In the part program (for the master spindle) with the programming command LIMS

The most recently changed value (LIMS or SD) is active.

LIMS is effective with G96, G961, G97 and can be specified on up to four speed limitations in the part program in one block. Spindle number <Sn> = 1, 2, 3, or 4 of the master spindle that is possible in the particular instance can be programmed in part program command LM[<Sn>].

When the block is loaded in the main run, all programmed values are transferred to the setting data SD43230 $SA_SPIND_MAX_VELO_LIMS.

Depending on the machine data:
MD10710 PROG_SD_RESET_SAVE_TAB[n] (setting data to be updated),
the speed limit set with LIMS remains stored after the control is switched off.
When G96, G961, G97 are reactivated, this speed limitation is also activated.

The maximum permissible spindle speed defined via G26 or via the setting data:
SD43220 $SA_SPIND_MAX_VELO_G26 (maximum spindle speed)
cannot be exceeded.

In the event of incorrect programming that would cause one of the speed limits (G26 or SD43220 $SA_SPIND_MAX_VELO_G26) to be exceeded, the following interface signal is set.

DB31, ... DBX83.1 (programmed speed too high)

In order to ensure smooth rotation with large part diameters, the spindle speed is not permitted to fall below a minimum level.
This speed can be set via the setting data:
SD43210 $SA_SPIND_MIN_VELO_G25 (minimum spindle speed)
and, depending on the gear stage, with the machine data:
MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT (minimum speed of the gear stage)
.

The minimum spindle speed can be changed in the part program with G25. In the event of incorrect programming that would cause one of the speed limits (G25 or SD43210 $SA_SPIND_MIN_VELO_G25) to be undershot, the following interface signal is set.

DB31, ... DBX83.2 (speed setpoint too low)

For more information on the spindle-speed limitations, see function description S1: "Spindles", Section: "S1: Spindles (Page 733)".

Note

The speed limits changed with G25/G26/LIMS in the part program are taken into the setting data and therefore remain saved after the end of program.

However, if the speed limits changed with G25/G26/LIMS are no longer to apply after the end of program, the following definitions must be inserted in the GUD block of the machine manufacturer:

REDEF $SA_SPIND_MIN_VELO_G25 PRLOC

REDEF $SA_SPIND_MAX_VELO_G26 PRLOC

REDEF $SA_SPIND_MAX_VELO_LIMS PRLOC

Master spindle changeover with G96, G961

If the master spindle is switched over when G96, G961 are active, the speed of the former master spindle is retained. This corresponds to a transition from G96 to G97. The master spindle newly defined with SETMS executes the "Constant cutting rate" function generated in this way.

Alarms

Constant cutting rate G96, G961, G962

- If no F value is programmed, alarm 10860 "No feedrate programmed" is output. The alarm is not generated with G0 blocks.
- Alarm 14800 "Programmed path velocity smaller than or equal to zero" is output while programming a negative path velocity.
- If, with an active G96, G961 or G962, no transverse axis is defined in the machine data: MD20100 $MC_DIAMETER_AX_DEF (geometry axis with transverse axis function), alarm 10870 "No transverse axis defined" is output.

- If a negative maximum spindle speed is programmed with the LIMS program command when G96, G961 are active, alarm 14820 "Negative maximum spindle speed programmed for G96, G961" is output.
- If no constant cutting rate is programmed when G96, G961 is selected for the first time, alarm 10900 "No S value programmed for constant cutting rate" is output.

5.2.3 Feedrate for thread cutting (G33, G34, G35, G335, G336)

5.2.3.1 Feedrate with G33

G33

The function G33 can be used to machine threads with constant pitch of the following type:

Speed S, feedrate F, thread pitch

A revolutional feedrate [mm/revolution] is used for G33 threads. The revolutional feedrate is defined by programming the thread pitch [mm/revolution].

The speed of the axes for the thread length is calculated from the programmed spindle speed S and the thread pitch.

Feedrate F [mm/min] = speed S [rev/min] * pitch [mm/rev]

At the end of the acceleration ramp, the position coupling between the spindle actual value (spindle setpoint with SPCON on master spindle) and the axis setpoint is established. At this moment, the position of the axis in relation to the zero mark of the spindle (including zero mark offsets) is as if the axis had accelerated abruptly at the start of the block when the thread start position (zero mark plus SF) was crossed. Compensation is made for the following error of the axis.

Minimum spindle speed

In order to ensure smooth rotation at low speeds, the spindle speed is not permitted to fall below a minimum level.

This speed can be set:

- With the setting data:
  SD43210 $SA_SPIND_MIN_VELO_G25 (minimum spindle speed)
- For each gear stage with the machine data:
  MD35140 $MA_GEAR _STEP_MIN_VELO_LIMIT (minimum speed for gear stage change)

The minimum spindle speed can be changed in the part program with G25.

NC stop, single block

NC stop and single block (even at the block boundary) are only active after completion of thread chaining. All successive G33 blocks and the first following non-G33 block are traversed as a block.

Premature abortion without destruction

Thread cutting can be aborted without destruction before the end point is reached. This can be done by activating a retraction motion.

Thread cutting with ROT frame

With ROT frame and G33, G34, G35, alarm 10607 "Thread with frame not executable" is activated if the rotation causes a change in the thread length and thus the pitch. Rotation around the thread axis is permissible.

Alarm 10607 "Thread with frame not executable" can be suppressed by setting bit 12 in machine data MD11410 $MN_SUPPRESS_ALARM_MASK, if the ROT instruction is used intentionally in the application.

All other frames are accepted by the NC without alarm. Attention is drawn to the pitch-changing effect of SCALE.

5.2.3.2 Linear increasing/decreasing thread pitch change with G34 and G35

Function

The thread pitch increase (G34) defines the numerical increase in the pitch value. A larger pitch results in a larger distance between the threads on the workpieces. The velocity of the thread axis therefore increases with assumed constant spindle speed.

The opposite therefore applies for the decrease in thread pitch (G35).

The following definitions are made for the thread pitch change:

- G34: Increase in thread pitch corresponds to progressive change
- G35: Decrease in thread pitch corresponds to degressive change

Both G34 and G35 functions imply the functionality of G33 and also provide the option of programming an absolute pitch change value for the thread under F. If the start and end pitch of a thread is known, the thread pitch change can be determined using the following equation:

$$F = \frac{|k^2_e - k^2_a|}{2 * l_G}$$

The meaning is as follows:

F: The thread pitch change to be programmed [mm/rev^2]

k_e: Thread pitch of axis target point coordinate, thread axis [mm/rev]

k_a: Initial thread pitch (programmed under I, J or K) [mm/rev]

l_G: Thread length [mm]

The absolute value of F must be applied to G34 or G35 depending on the required pitch increase of decrease.

When the thread length l_G, pitch change F and initial pitch k_a are known, the pitch increase at the end of block k_e can be determined as follows by modifying the formula:

- For G34 (increasing pitch):

$$k_e = \sqrt{k^2_a + F * 2 * l_G}$$

- For G35 (decreasing pitch):

$$k_e = \sqrt{k^2_a - F * 2 * l_G}$$

Note

If the formula results in a negative root expression, the thread cannot be machined!

In this case, the NC signals alarm 10605 or alarm 22275.

Application

The G34 and G35 functions can be used to produce self-shearing threads.

Example

Thread cutting G33 with decreasing thread pitch G35

Program code	Comment
N1608 M3 S10	; Spindle speed
N1609 G0 G64 Z40 X216	; Approach starting point
N1610 G33 Z0 K100 SF=R14	; Thread with constant pitch 100 mm/rev
N1611 G35 Z-220 K100 F17.045455	; Thread pitch decrease 17.045455 mm/rev2
	; Thread pitch at end of block 50 mm/rev
N1612 G33 Z-240 K50	; Traverse thread block without jerk
N1613 G0 X218	

Program code	Comment
N1614 G0 Z40	
N1616 M17	

Monitoring during the block preparation

Any pitch changes that would overload the thread axis when G34 is active or would result in an axis standstill when G35 is active, are detected in advance during block preparation. Alarm 10604 "Thread pitch increase too high" or 10605 "Thread pitch decrease too high" is signaled.

During thread cutting, certain practical applications require a correction of the spindle speed. In this case, the operator will base his correction on the permissible velocity of the thread axis.

To do this, it is possible to suppress the output of alarms 10604 and 10605 as follows:

MD11410 $MN_SUPPRESS_ALARM_MASK bit 10 = 1

Block preparation is then continued normally.

Monitoring during the execution

The following situations are monitored cyclically when the thread is machined (interpolation):

- Exceeding of maximum velocity of thread axis
- Reaching of axis standstill with G35

The following alarm is signaled when the monitoring function responds:

- Alarm 22269 "Maximum velocity of thread axis reached" or
- Alarm 22275 "Zero velocity of thread axis reached"

5.2.3.3 Acceleration behavior of the axis for G33, G34 and G35

The acceleration behavior of the feed axis when thread cutting/tapping with G33, G34 or G35 can be set via the channel-specific setting data:

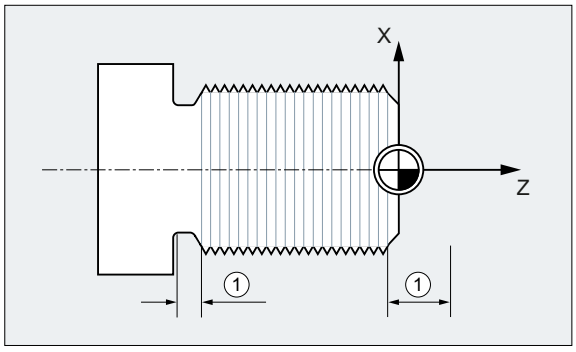
SD42010 $SC_THREAD_RAMP_DISP[<n>] = <Value>

<n>	Meaning	<Value>	Meaning
0	Acceleration behavior at the thread run-in:	< 0 (Default: -1)	The acceleration of the thread axis when running-in the thread is as programmed in BRISK/SOFT
		= 0	The acceleration of the thread axis when running-in the thread is sudden (step function) (≙ BRISK).
		> 0	The maximum thread run-in distance is specified. Specifying the maximum run-in thread path can also be specified via address DITS also in the part program (see "Programmed run-in and run-out path for G33, G34 and G35 (DITS, DITE) (Page 140)"). When a block is inserted in the main run, the programmed run-in path is transferred into the setting data. **Note** Too short a distance can result in the axis being overloaded when accelerating.
1	Acceleration behavior at thread run-out:	< 0 (Default: -1)	The braking of the thread axis when running-ou the thread is as programmed in BRISK/SOFT
		= 0	The braking of the thread axis when running-out the thread is sudden (step function) (≙ BRISK).
		> 0	The maximum thread run-out distance is specified. Specifying the maximum run-out thread path can also be specified via address DITE also in the part program (see "Programmed run-in and run-out path for G33, G34 and G35 (DITS, DITE) (Page 140)"). When a block is inserted in the main run, the programmed run-out path is transferred into the setting data. **Note** Too short a distance can result in an acceleration overload of the axis.
2	Acceleration behavior for thread transitions within a thread chain	= -1 (default)	Permits smoothing corners between two thread blocks in order to maintain dynamic response limits. Rounding using smoothing is not carried out if the dynamic performance of the machine allows a hard transition, e.g. as a result of the effectiveness of the overload factor for axial velocity step/jumps (MD32310 $MA_MAX_ACCEL_OVL_FACTOR).
		= 0	Corners between two thread blocks are executed precisely without rounding - the axes follow the control loop commands.
		> 0	Reserved

5.2.3.4 Programmed run-in and run-out path for G33, G34 and G35 (DITS, DITE)

The run-in and run-out path of the thread can be specified in the part program with the DITS and DITE addresses.

The thread axis is accelerated or braked along the specified path.



① Run-in/run-out path, depending on the machining direction

Short run-in path

Due to the collar on the thread runin, little room is left for the tool start ramp.
This must therefore be specified shorter via DITS.

Short run-out path

Because of the shoulder at the thread run-out, there is not much room for the tool braking ramp, introducing a risk of collision between the workpiece and the tool cutting edge. The deceleration ramp can be specified shorter using DITE. Due to the inertia of the mechanical system, however, a collision can still occur.

Remedy: Program a shorter thread, reduce the spindle speed.

Note

DITE acts at the end of the thread as a rounding clearance. This achieves a smooth change in the axis motion.

Effects

The programmed run-in and run-out path only increases the rate of acceleration on the path. If one of the two paths is set larger than the thread axis needs with active acceleration, the thread axis is accelerated or decelerated with maximum acceleration.

Syntax

```
DITS=<Value> DITE=<Value>
```

Meaning

DITS:	Define thread run-in path
DITE:	Define thread run-out path
<value>:	Only paths, and not positions, are programmed with DITS and DITE. The programmed run-in/run-out path is handled according to the current dimension setting (inches, metric).

Example

```
Program code                                    Comment
...
N40 G90 G0 Z100 X10 SOFT M3 S500
N50 G33 Z50 K5 SF=180 DITS=1 DITE=3          ; Start of smoothing with Z=53.
N60 G0 X20
```

Further information

SD42010 $SC_THREAD_RAMP_DISP

When a block containing DITS and/or DITE is inserted in the main run, the programmed run-in/run-out path is transferred into the setting data SD42010 $SC_THREAD_RAMP_DISP:

- SD42010 $SC_THREAD_RAMP_DISP[**0**] = programmed value of DITS
- SD42010 $SC_THREAD_RAMP_DISP[**1**] = programmed value of DITE

If no run-in/run-out path is programmed before or in the first thread block, the current value of the setting data is used.

Behavior following channel / mode group / program end reset

SD 42010 values which have been overwritten by DITS and/or DITE remain active even following a channel / mode group / program end reset.

Behavior following warm start

In case of a warm start, the setting data is reset to the values which were active before overwriting by DITS and/or DITE (standard behavior).

If, however, the values programmed with DITS and DITE shall also be active following a warm restart, the setting data SD42010 $SC_THREAD_RAMP_DISP must be listed in the machine data MD10710 $MN_PROG_SD_RESET_SAVE_TAB:

MD10710 $MN_PROG_SD_RESET_SAVE_TAB[<n>] = 42010

Behavior if the run-in and/or run-out path is very short

If the run-in and/or run-out path is very short, the acceleration of the thread axis is higher than the configured value. This causes an acceleration overload on the axis.

Alarm 22280 "Programmed run-in path too short" is then issued for the thread run-in (with the appropriate configuration in MD11411 $MN_ENABLE_ALARM_MASK). The alarm is purely for information and has no effect on part program execution.

See also

Acceleration behavior of the axis for G33, G34 and G35 (Page 138)

5.2.3.5 Fast retraction during thread cutting

Function

The "Rapid retraction during thread cutting" function can be used to interrupt thread cutting without causing irreparable damage in the following circumstances:

- NC stop (NC/PLC interface signal)
- Alarms that implicitly trigger NC stop
- Switching of a rapid input
  Further information
  Programming Manual Advanced; Fast retraction from the contour

The retraction motion can be programmed via:

- Retraction path and retraction direction (relative)
- Retraction position (absolute)

Note

Tapping

The "Fast retraction" function **cannot** be used with tapping (`G331` / `G332`).

Programming

Syntax

Enable fast retraction, retraction motion via retraction path and retraction direction:

```
G33 ... LFON DILF=<Value> LFTXT/LFWP ALF=<Value>
```

Enable fast retraction, retraction motion via retraction position:

```
POLF[<Axis name>]=<Value> LFPOS
POLFMASK/POLFMLIN(<Axis 1 name>,<Axis 2 name>, etc.)
G33 ... LFON
```

Disable fast retraction for thread cutting:

```
LFOF
```

Meaning

`LFON`:	Enable fast retraction for thread cutting (`G33`).
`LFOF`:	Disable fast retraction for thread cutting (`G33`)

DILF= : Define length of retraction path

The value preset during MD configuration (MD21200 $MC_LIFTFAST_DIST) can be modified in the part program by programming DILF.

Note:
The configured MD value is always active following NC-RESET.

LFTXT
LFWP: The retraction direction is controlled in conjunction with ALF with G commands LFTXT and LFWP.

- LFTXT: The plane in which the retraction motion is executed is calculated from the path tangent and the tool direction (default setting).
- LFWP: The plane in which the retraction motion is executed is the active working plane.

ALF= : The direction is programmed in discrete degree increments with ALF in the plane of the retraction motion.

With LFTXT, retraction in the tool direction is defined for ALF=1.

With LFWP, the direction in the working plane is derived from the following assignment:

- G17 (X/Y plane)
  ALF=1 ; Retraction in the X direction
  ALF=3 ; Retraction in the Y direction
- G18 (Z/X plane)
  ALF=1 ; Retraction in the Z direction
  ALF=3 ; Retraction in the X direction
- G19 (Y/Z plane)
  ALF=1 ; Retraction in the Y direction

ALF=3 ; Retraction in the Z direction

Further information:
Programming options with ALF are also described in "Traverse direction for rapid retraction from the contour" in the Programming Manual Advanced.

LFPOS: Retraction of the axis declared with POLFMASK or POLFMLIN to the absolute axis position programmed with POLF.

POLFMASK: Release of axes (<Axis 1 name>,<Axis 1 name>, etc.) for independent retraction to absolute position.

POLFMLIN: Release of axes for retraction to absolute position in linear relation

Note:
Depending on the dynamic response of all the axes involved, the linear relation cannot always be established before the lift position is reached.

`POLF[]`: Define absolute retraction position for the geometry axis or machine axis in the index

Effectiveness: modal

`=<Value>`: In the case of geometry axes, the assigned value is interpreted as a position in the workpiece coordinate system. In the case of machine axes, it is interpreted as a position in the machine coordinate system.

The values assigned can also be programmed as incremental dimensions:

`=IC<Value>`

`<Axis name>`: Name of a geometry axis or machine axis

Note

`LFON` or `LFOF` can always be programmed, but the evaluation is performed exclusively during thread cutting (`G33`).

Note

`POLF` with `POLFMASK`/`POLFMLIN` are not restricted to thread cutting applications.

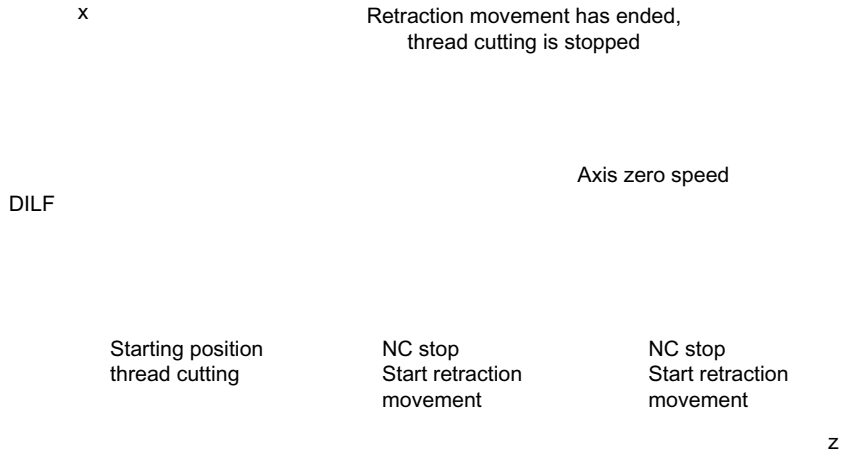


Figure 5-1 Interruption of G33 through retraction motion

Dynamic response of the retraction motion

The retraction motion is executed with maximum axis dynamic response:

- MD32000 $MA_MAX_AX_VELO[<Axis>] (velocity)
- MD32300 $MA_MAX_AX_ACCEL[<Axis>] (acceleration)
- MD32431 $MA_MAX_AX_JERK[<Axis>] (jerk)

Example

```
Program code                          Comment
N55 M3 S500 G90 G18                   ; Set active machining plane.
...
N65 MSG ("thread cutting")
MM_THREAD:
N67 $AC_LIFTFAST=0                    ; Reset before starting the thread.
N68 G0 Z5
N69 X10
N70 G33 Z30 K5 LFON DILF=10 LFWP      ; Enable fast retraction for thread
ALF=7                                   cutting.
                                      ; Retraction path = 10 mm
                                      ; Retraction plane Z/X (because of G18).
                                      ; Retraction direction -X (with ALF=3;
                                        retraction direction +X).
N71 G33 Z55 X15
N72 G1                                ; Deselect thread cutting.
N69 IF $AC_LIFTFAST GOTOB MM_THREAD   ; If thread cutting has been
                                        interrupted.
N90 MSG ("")
...
N70 M30
N55 M3 S500 G90 G0 X0 Z0
...
N87 MSG ("tapping")
N88 LFOF                              ; Deactivate fast retraction before
                                        tapping.
N89 CYCLE...                          ; Tapping cycle with G33.
N90 MSG ("")
...
N99 M30
```

Behavior at power on and reset

After power on and reset, the following settings are activated:

- Initial settings for the retraction motion (LFON / LFOF) and retraction direction (LFTXT / LFWP): MD20150 $MC_GCODE_RESET_VALUES
- Retraction path: MD21200 $MC_LIFTFAST_DIST

5.2.3.6 Convex thread (G335, G336)

Function

The G commands G335 and G336 can be used to turn convex threads (= differing to the cylindrical form). Application is the machining of extremely large components that sag in the machine because of their self-weight. Paraxial thread would result in the thread being too small in the middle of the component. This can be compensated with convex threads.

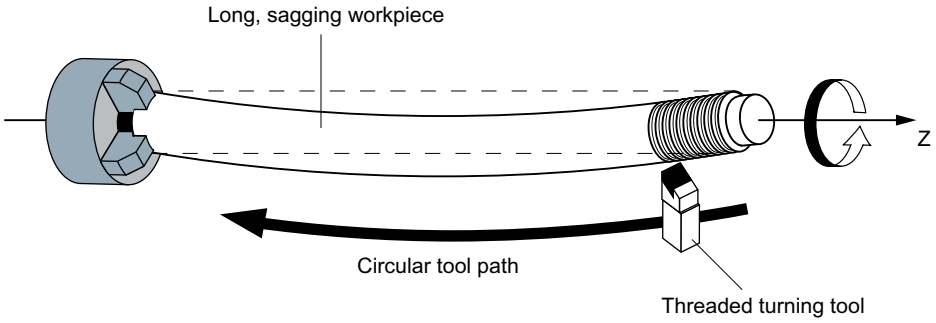


Figure 5-2 Turning a convex thread

Programming

The turning of a convex thread is programmed with `G335` or `G336`:

`G335`:	Turning of a convex thread on a circular tool path **in a clockwise direction**
`G336`:	Turning of a convex thread on a circular tool path **in a counter-clockwise direction**

The programming is performed first as for a linear thread by specifying the axial block end points and the pitch via parameters `I`, `J` and `K`.

An arc is also specified. As for `G2`/`G3`, this can be programmed via the center point, radius, opening angle or intermediate point specification. When programming the convex thread with center point programming, the following must be taken into account: Since `I`, `J` and `K` are used for the pitch in thread cutting, the circle parameters in the center point programming must be programmed with `IR=...`, `JR=...` and `KR=...`.

`IR=...`:	Cartesian coordinate for the circle center point in the X direction
`JR=...`:	Cartesian coordinate for the circle center point in the Y direction
`KR=...`:	Cartesian coordinate for the circle center point in the Z direction

Note

`IR`, `JR` and `KR` are the default values of the interpolation parameter names for a convex thread that can be set via machine data MD10651 $MN_IPO_PARAM_THREAD_NAME_TAB.

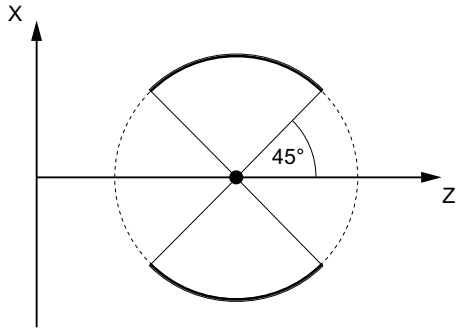
Optionally, a starting point offset `SF` can also be specified.

Syntax

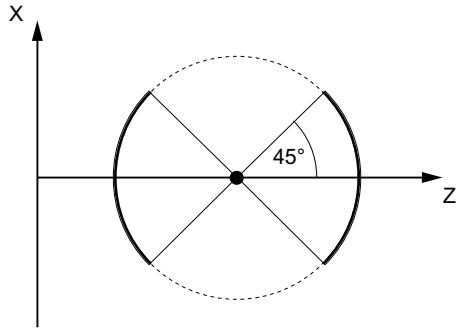
The syntax for the programming of a convex thread therefore has the following general form:
`G335/G336 <axis target point coordinate(s)> <pitch> <arc> [<starting point offset>]`

Permissible arc areas

The arc programmed at `G335`/`G336` must be in an area in which the specified thread main axis (`I`, `J` or `K`) has the main axis share on the arc over the entire arc:



Permissible areas for the **Z** axis (pitch programmed with `K`)



Permissible areas for the **X** axis (pitch programmed with `I`)

A change of the thread main axis as shown in the following figure is **not** permitted:

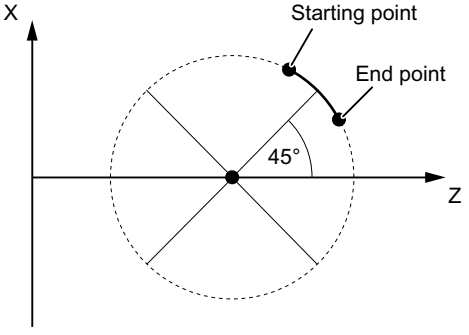


Figure 5-3 Convex thread: Area that is not permissible

Boundary conditions

Frames

G335 and G336 are also possible with active frames. However, you must ensure that the permissible arc areas are maintained in the basic coordinate system (BCS).

Response for …

The G335/G336 response for:

- Power On / Power Off
- Mode change
- NCK / mode group / channel / part program end reset

- Block search / REPOS / ASUP
- Alarms / emergency stop / malfuncations

corresponds to the behavior for G33/G34/G35.
There are no specific restrictions.

Examples

Example 1: Convex thread in the clockwise direction with end and center point programming

Program code	Comment
N5 G0 G18 X50 Z50	; Approach starting point.
N10 **G335 Z100** K=3.5 **KR=25 IR=-20** SF=90	; Turn convex thread in the clockwise direction.

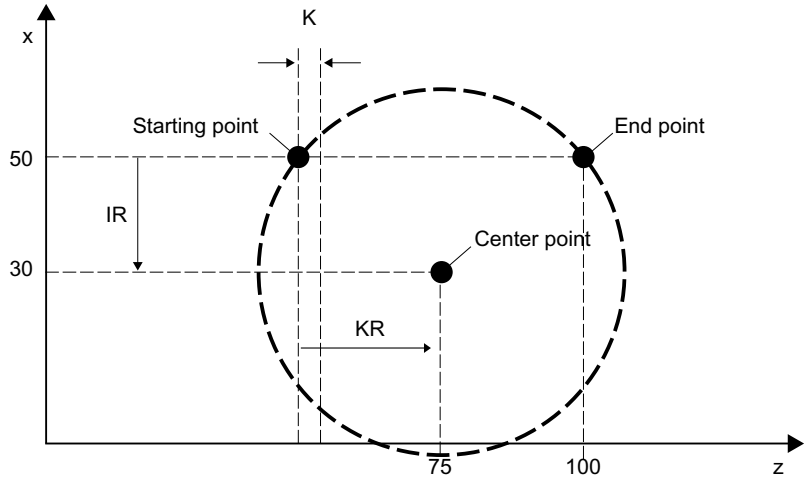


Figure 5-4 Convex thread in the clockwise direction with end and center point programming

Example 2: Convex thread in the counter-clockwise direction with end and center point programming

Program code	Comment
N5 G0 G18 X50 Z50	; Approach starting point.
N10 **G336 Z100** K=3.5 **KR=25 IR=20** SF=90	; Turn convex thread in the counter-clockwise direction.

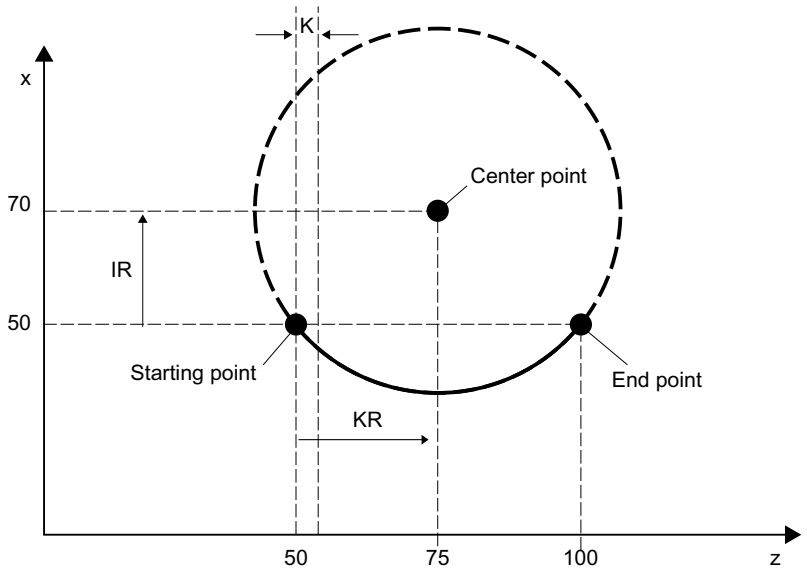


Figure 5-5 Convex thread in the counter-clockwise direction with end and center point programming

Example 3: Convex thread in the clockwise direction with end point and radius programming

```
Program code
N5 G0 G18 X50 Z50
N10 G335 Z100 K=3.5 CR=32 SF=90
```

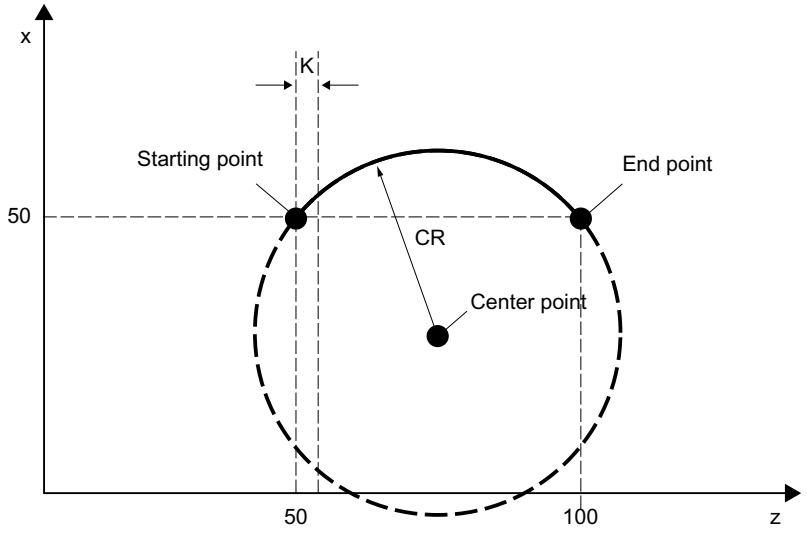


Figure 5-6 Convex thread in the clockwise direction with end point and radius programming

Example 4: Convex thread in the clockwise direction with end point and opening angle programming

```
Program code
N5 G0 G18 X50 Z50
N10 G335 Z100 K=3.5 AR=102.75 SF=90
```

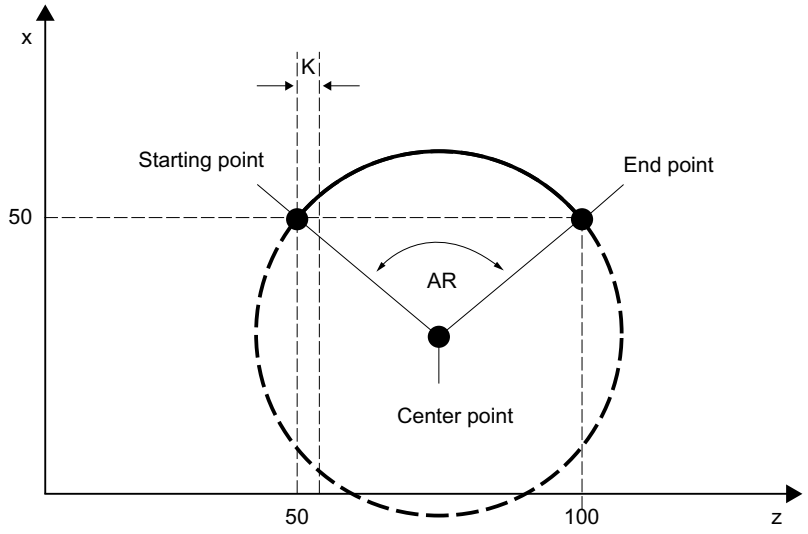


Figure 5-7 Convex thread in the clockwise direction with end point and opening angle programming

Example 5: Convex thread in the clockwise direction with center point and opening angle programming

Program code

```
N5 G0 G18 X50 Z50
N10 G335 K=3.5 KR=25 IR=-20 AR=102.75 SF=90
```

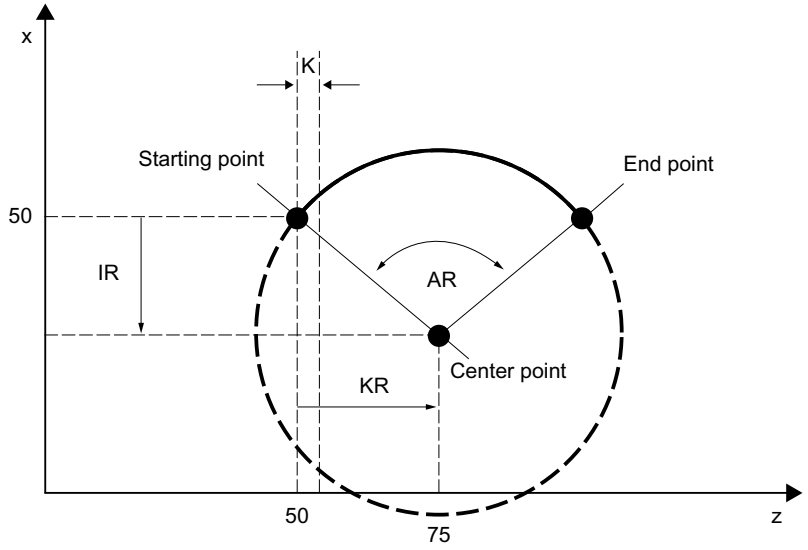


Figure 5-8 Convex thread in the clockwise direction with center point and opening angle programming

Example 6: Convex thread in the clockwise direction with end and intermediate point programming

Program code

```
N5 G0 G18 X50 Z50
N10 G335 Z100 K=3.5 I1=60 K1=64
```

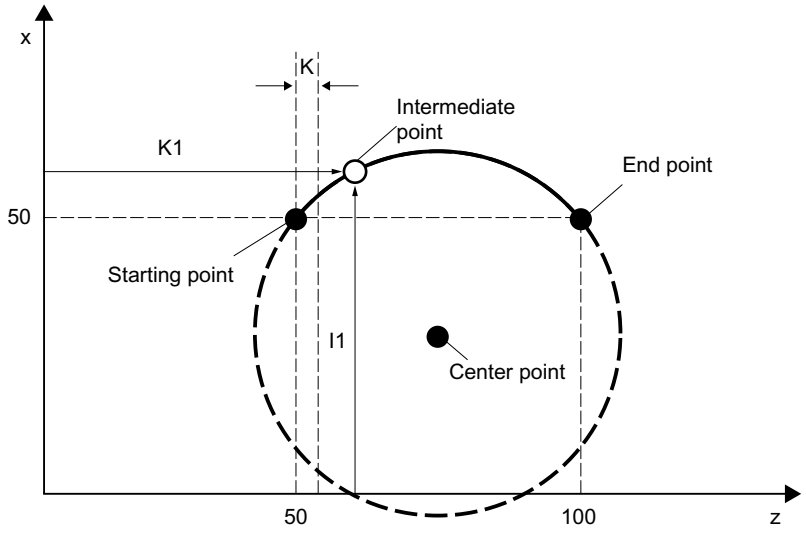


Figure 5-9 Convex thread in the clockwise direction with end and intermediate point programming

5.2.4 Feedrate for tapping without compensating chuck (G331, G332)

Function

A thread can be tapped by rigid tapping with the functions `G331` (tapping) and `G332` (tapping retraction).

Path feedrate

With `G331` / `G332`, the path feedrate F for the axes involved in tapping is derived from the effective spindle speed S and the programmed pitch:

```
F [mm/min] = S [rpm] * thread pitch [mm/U]
```

Requirement

The requirement for rigid tapping is a **position-controlled spindle** with position measuring system.

Machine data

- Preventing stop events
  In the machine data, the stopping response when G331 / G332 is active is defined:
  MD11550 $MN_STOP_MODE_MASK, Bit <x> = <value>

Bit	Value	Meaning
0	0	When G331 / G332 is active, stopping is **not** performed during a path motion or dwell time (G4).
	1	When G331 / G332 is active, stopping is performed during interruption of path control mode, for example, by G60 or G4.

If you always want to prevent stopping the traverse movement when G331 / G332 is active, this area must be declared a stop delay area with the commands DELAYFSTON and DELAYFSTOF.

Further information

Function Manual Basic Functions; Program operation, Influencing the stop events by stop delay areas

---

Note

Single block

If the single block is activated in the stop delay area, the NC stops at the end of the first block outside the stop delay area. If the single block is already selected before the stop delay area, the NC stops at each block limit, i.e. also in the stop delay area! This deselects the stop delay area.

Override changes

If the override is changed **before** a stop delay area, the override takes effect **in** the stop delay area.

If the override is changed **in** the stop delay area, the change takes effect **after** the stop delay area.

- Override
  Which override is active during rigid tapping is set in:
  MD12090 $MN_OVR_FUNCTION_MASK, Bit <x> = <value>

Bit	Value	Meaning
0	0	The **spindle override** is active for G331/G332 Depending on the setting in the following machine data, the override is related either to the programmed spindle speed or to the configured spindle speed limitation: MD12080 $MN_OVR_REFERENCE_IS_PROG_FEED
	1	The **path override** is active for G331/G332 Depending on the setting in the following machine data, the override is related either to the programmed path feedrate or to the configured path feedrate limitation: MD12082 $MN_OVR_REFERENCE_IS_MIN_FEED **Note** This only applies if the limitation value is lower than the programmed path feedrate.

Boundary conditions

Further overrides

The following overrides are inactive during rigid tapping

- Programmable path feedrate override OVR
- Rapid traverse override

5.2.5 Feedrate for tapping with compensating chuck (G63)

Function

G63 is a subfunction for tapping threads using a tap with compensating chuck. An encoder (position encoder) is not required.

Speed S, feedrate F, thread pitch

With G63, a speed S must be programmed for the spindle and a feedrate F for the infeed axis (axis for thread length).

The feedrate F must be calculated by the programmer on the basis of the speed S and the thread pitch.

Feedrate F [mm/min] = speed S [rev/min] * pitch [mm/rev]

Further information

For more information on G63, see Programming Manual NC Programming.

5.2.6 FGROUP and FGREF

Programming

It should be possible to program the effective machining feedrate in the usual way as a path feedrate via the F value in processing procedures where the tool, the workpiece or both are moved by a rotary axis (e.g. laser machining of rotating tubes).

In order to achieve this, it is necessary to specify an effective radius (reference radius) for each of the rotary axes involved. You can do this by programming the modal NC address:
FGREF[<rotary axis>]=<reference radius>

The unit of the reference radius depends on the G70/G71/G700/G710 setting.

In order to include the axes in the calculation of the path feedrate, they must all be specified in the FGROUP command.

In order to ensure compatibility with the behavior with no FGREF programming, the evaluation 1 degree = 1 mm is activated on system powerup and RESET.

This corresponds to a reference radius of:

FGREF = 360 mm / (2π) = 57.296 mm

This default is independent of the active basic system (MD10240 $MN_SCALING_SYSTEM_IS_METRIC) and the currently active G70/G71/G700/G710 setting.

Special features of the feedrate weighting for rotary axes in FGROUP:

```
Program code
N100 FGROUP(X,Y,Z,A)
N110 G1 G91 A10 F100
N120 G1 G91 A10 X0.0001 F100
```

The programmed F value in block N110 is evaluated as a rotary axis feedrate in degrees/min, while the feedrate weighting in block N120 is either 100 inch/min or 100 mm/min, depending on the current inch/metric setting.

NOTICE

Different systems of units

The FGREF factor also works if only rotary axes are programmed in the block. The normal F value interpretation as degree/min applies in this case only if the radius reference corresponds to the FGREF default:

- For G71/G710: FGREF[A]=57.296
- For G70/G700: FGREF[A]=57.296/25.4

Example

The following example is intended to demonstrate the effect of FGROUP on the path and path feedrate. The variable $AC_TIME contains the time of the block start in seconds. It can only be used in synchronized actions.

```
Program code                    Comment
N100 G0 X0 A0
N110 FGROUP(X,A)
N120 G91 G1 G710 F100           ; Feedrate = 100 mm/min or 100 degrees/min
N130 DO $R1=$AC_TIME
N140 X10                        ; Feedrate = 100 mm/min, path = 10 mm, R1 = approx. 6 s
N150 DO $R2=$AC_TIME
N160 X10 A10                    ; Feedrate = 100 mm/min, path = 14.14 mm, R2 = approx.
                                8 s
N170 DO $R3=$AC_TIME
N180 A10                        ; Feedrate = 100 degrees/min, path = 10 degrees, R3
                                = approx. 6 s
N190 DO $R4=$AC_TIME
N200 X0.001 A10                 ; Feedrate = 100 mm/min, path = 10 mm, R4 = approx. 6 s
N210 G700 F100                  ; Feedrate = 2540 mm/min or 100 degrees/min
N220 DO $R5=$AC_TIME
```

```
Program code                    Comment
N230 X10                        ; Feedrate = 2540 mm/min, path = 254 mm, R5 = approx. 6
                                s
N240 DO $R6=$AC_TIME
N250 X10 A10                    ; Feedrate = 2540 mm/min, path = 254.2 mm, R6 = approx.
                                6 s
N260 DO $R7=$AC_TIME
N270 A10                        ; Feedrate = 100 degrees/min, path = 10 degrees, R7
                                = approx. 6 s
N280 DO $R8=$AC_TIME
N290 X0.001 A10                 ; Feedrate = 2540 mm/min, path = 10 mm, R8 = approx.
                                0.288 s
N300 FGREF[A]=360/(2*$PI)       ; Set 1 degree = 1 inch via the effective radius
N310 DO $R9=$AC_TIME
N320 X0.001 A10                 ; Feedrate = 2540 mm/min, path = 254 mm, R9 = approx. 6
                                s
N330 M30
```

Diagnostics

Read reference radius

The value of the reference radius of a rotary axis can be read using system variables:

- For the display in the user interface, in synchronized actions or with a preprocessing stop in the part program via the system variable:

  $AA_FGREF[<axis>] Current main run value

- Without preprocessing stop in the part program via the system variable:

  $PA_FGREF[<axis>] Programmed value

If no values are programmed, the default 360 mm / (2π) = 57.296 mm (corresponding to 1 mm per degree) will be read in both variables.

For linear axes, the value in both variables is always 1 mm.

Read path axes affecting velocity

The axes involved in path interpolation can be read using system variables:

- For the display in the user interface, in synchronized actions or with a preprocessing stop in the part program via the system variables:

  $AA_FGROUP[<axis>] Returns the value "1" if the specified axis affects the path velocity in the current main run record by means of the basic setting or through FGROUP programming. Otherwise, the variable returns the value "0".

  $AC_FGROUP_MASK Returns a bit key of the channel axes programmed with FGROUP which are to affect the path velocity.

- Without preprocessing stop in the part program via system variables:

$PA_FGROUP[<axis>]	Returns the value "1" if the specified axis affects the path velocity by means of the basic setting or through FGROUP programming. Otherwise, the variable returns the value "0".
$P_FGROUP_MASK	Returns a bit key of the channel axes programmed with FGROUP which are to affect the path velocity.

5.3 Feedrate for positioning axes (FA)

Function

The velocity of a positioning axis is programmed with axis-specific feedrate FA.

FA is modal.

The feedrate is always G94.

Note

The maximum axis velocity (MD32000 $MA_MAX_AX_VELO) is not exceeded.

Programming

No more than five axis-specific feedrates can be programmed in each part program block.

Syntax:

```
FA[<positioning axis>] = <feedrate value>
```

<positioning axis>:	Name of the channel axis (MD20080 $MC_AXCONF_CHANAX_NAME_TAB)	
<feedrate value>:	Feedrate	
	Value range:	0.001…999 999.999 mm/min, deg/min or 0.001…39 999.9999 inch/min

Default setting

If no axial feedrate FA is programmed, the axial default setting is applied:

MD32060 $MA_POS_AX_VELO (initial setting for positioning axis velocity)

Output to PLC

The feedrate value can be output to the PLC:

- To the channel-specific NC/PLC interface via:
  DB21, ... DBB158 - DBB193
- To the axis-specific NC/PLC interface via:
  DB31, ... DBB78 - DBB81

The output time is specified with the machine data:

MD22240 $MC_AUXFU_F_SYNC_TYPE (output time of F functions)

The output is suppressed in the default setting (MD22240 = 3), because drops in velocity can occur through the output of F functions to the NC/PLC interface in continuous-path mode.

For further information see Function Manual "Basic Functions", Section "H2: Auxiliary function outputs to PLC".

Reset behavior

The behavior after the end of program or NC reset is specified by the machine data:

MD22410 $MC_F_VALUES_ACTIVE_AFTER_RESET (F function is active even after reset)

Value	Meaning
0	The default values are effective after NC reset.
1	The last programmed FA values are effective after NC reset.

5.4 Feedrate control

5.4.1 Feedrate disable and feedrate/spindle stop

Function

The "Feed disable" or "Feed/spindle stop" brings the axes to a standstill with adherence to the braking characteristics With the exception for thread cutting `G33`, the programmed contour is maintained.

Channel-specific feedrate disable

All axes (geometry and special axes) of a channel are stopped in all modes using the channel-specific NC/PLC interface signal:

DB21, ... DBX6.0 (feedrate disable)

Channel-specific "Feed stop" for geometry axes in the JOG mode

The traversing motion of a specific geometry axis of the channel is stopped using the channel-specific NC/PLC interface signals:

- DB21, ... DBX12.3 (feedrate stop, geometry axis 1)
- DB21, ... DBX16.3 (feedrate stop, geometry axis 2)
- DB21, ... DBX20.3 (feedrate stop, geometry axis 3)

The interface signals are only active in the **JOG** mode.

Axis-specific "Feedrate stop/spindle stop" for machine axes

Traversing motion of the machine axis is stopped using the axis-specific NC/PLC interface signal:

DB31, ... DBX4.3 (feedrate stop / spindle stop)

AUTOMATIC and MDI modes

The following applies in the AUTOMATIC and MDI modes:

- If the "Feed stop" is performed for a **path axis**, all the axes traversed in the current block and all axes participating in the axis group are stopped.
- If the "Feed stop" is performed for a **positioning axis**, only this axis is stopped.

Thread cutting

Thread cutting	Effectiveness
G33, G34, G35	Effective **Notice** Contour deviation
G331, G332	Effective
G63	Axis: Effective Spindle: **Not** effective

Axis-specific axis/spindle disable for machine axes

Traversing motion of the machine axis is stopped using the axis-specific NC/PLC interface signal:

DB31, ... DBX1.3 (axis/spindle disable)

5.4.2 Feedrate override via machine control panel

Function

With the "Feedrate override via machine control panel", the operator can locally increase or decrease the path feedrate at the machine as a percentage with immediate effect. To achieve this, the programmed feedrates are multiplied with the override values available at the NC/PLC interface.

The feedrate can be changed axis-specifically for positioning axes.

The "Spindle override" can be used to change the spindle speed and the cutting rate (G96, G961).

With a feedrate change, the axial acceleration and velocity limits are maintained. There are no contour errors along the path.

The feedrate override can be changed separately for path and position axes.

The overrides influence the programmed values or the limits (e.g. G26, LIMS for spindle speed).

Channel-specific feedrate and rapid traverse override

For feedrate and rapid traverse override, dedicated enable signals and correction/offset factors are available in the NC/PLC interface:

DB21, ... DBX6.7 (feedrate override active)

DB21, ... DBB4 (feedrate override)

DB21, ... DBX6.6 (rapid traverse override active)

DB21, ... DBB5 (rapid traverse override)

The override factors can be specified from the PLC either in the binary or Gray-coded format. The format is communicated to the NC via the following machine data:

MD12020 $MN_OVR_FEED_IS_GRAY_CODE (path feedrate override switch Gray-coded)

MD12040 $MN_OVR_RAPID_IS_GRAY_CODE (rapid traverse override switch Gray-coded)

The following permanent assignment applies to binary code:

Binary code	Decimal	Override factor
00000000	0	0.00 ≙ 0%
00000001	1	0.01 ≙ 1%
00000010	2	0.02 ≙ 2%
00000011	3	0.03 ≙ 3%
00000100	4	0.04 ≙ 4%
...	...	...
01100100	100	1.00 ≙ 100%
...	...	...
11001000	200	2.00 ≙ 200%

With Gray coding, the override factors corresponding to the switch position must be entered in the following machine data:

MD12030 $MN_OVR_FACTOR_FEEDRATE [<n>] (evaluation of the path feedrate override switch)

MD12050 $MN_OVR_FACTOR_RAPID_TRA [<n>] (evaluation of the rapid traverse override switch)

An active feedrate override acts on all path axes that are assigned to the current channel. An active rapid traverse override has an effect on all the axes that are traversed with rapid traverse and that are assigned to the current channel.

No rapid traverse override switch available

If there is no dedicated rapid traverse override switch, you can switch between rapid traverse override and feedrate override. The override to be active can be selected via the PLC or operator panel front. When rapid traverse override is active, the feedrate override values are limited to 100%.

- When the rapid traverse override is activated via the operator panel front, the basic PLC program:
  - Transfers the selection of the feedrate override for rapid traverse on the activation signal for the rapid traverse override:
    DB21, ... DBX6.6 = DB21, ... DBX25.3
  - Transfers the feedrate override value in the rapid traverse value
    DB21, ... DBB5 = DB21, ... DBB4
- When the rapid traverse override is selected via the PLC, the PLC user program:
  - Must set the activation signal for the rapid traverse override:
    DB21, ... DBX6.6 = 1
  - Must transfer the feedrate override value in the rapid traverse override value:
    DB21, ... DBB5 = DB21, ... DBB4

Effectiveness of the channel-specific feedrate and rapid traverse override:	
• With active G33, G34, G35:	**Not** effective
• With active G63:	**Not** effective
• With active G331, G332:	**Not** effective

Reference velocity for path feedrate override

The reference velocity for the "Path feedrate override via machine control panel" can be set differently to the standard feedrate (= programmed feedrate).

MD12082 $MN_OVR_REFERENCE_IS_MIN_FEED

Axis-specific feedrate override

An enable signal and a byte for the feedrate override factor are available in the NC/PLC interface for each positioning axis:

DB31, ... DBX1.7 (override effective)

DB31, ... DBB0 (feedrate override)

The override factor can be specified from the PLC either in the binary or Gray-coded format. The format is communicated to the NC via the following machine data:

MD12000 $MN_OVR_AX_IS_GRAY_CODE (axis feedrate override switch Gray-coded)

The following permanent assignment applies to binary code:

Binary code	Decimal	Override factor
00000000	0	0.00 ≙ 0%
00000001	1	0.01 ≙ 1%
00000010	2	0.02 ≙ 2%
00000011	3	0.03 ≙ 3%
00000100	4	0.04 ≙ 4%
...	...	...
01100100	100	1.00 ≙ 100%
...	...	...
11001000	200	2.00 ≙ 200%

With Gray coding, the override factors corresponding to the switch position must be entered in the following machine data:

MD12010 $MN_OVR_ FACTOR_AX_ SPEED [<n>] (evaluation of the axis feedrate override switch)

Effectiveness of the axis-specific feedrate override:	
• With active G33, G34, G35:	**Not** effective
• With active G63:	**Not** effective (the override is set in the NC permanently to 100%)
• With active G331, G332:	**Not** effective (the override is set in the NC permanently to 100%)

Spindle override

An enable signal and a byte for the spindle override factor are available in the NC/PLC interface for each spindle:

DB31, ... DBX1.7 (override effective)

DB31, ... DBB19 (spindle override)

The override factor can be specified from the PLC either in the binary or Gray-coded format. The format is communicated to the NC via the following machine data:

MD12060 $MN_OVR_SPIND_IS_GRAY_CODE (spindle override switch Gray-coded)

The following permanent assignment applies to binary code:

Binary code	Decimal	Override factor
00000000	0	0.00 ≙ 0%
00000001	1	0.01 ≙ 1%
00000010	2	0.02 ≙ 2%
00000011	3	0.03 ≙ 3%

Binary code	Decimal	Override factor
00000100	4	0.04 ≙ 4%
...	...	...
01100100	100	1.00 ≙ 100%
...	...	...
11001000	200	2.00 ≙ 200%

With Gray coding, the override factors corresponding to the switch position must be entered in the following machine data:

MD12070 $MN_OVR_FACTOR_SPIND_SPEED [<n>] (evaluation of the spindle override switch)

Effectiveness of the "spindle override":	
• With active G33, G34, G35:	Effective
• With active G63:	**Not** effective
• With active G331, G332:	Effective

Reference of the spindle override

The spindle override can refer to the speed or the programmed speed limited by the machine or setting data. The setting is realized via:

MD12080 $MN_OVR_REFERENCE_IS_PROG_FEED (override reference velocity)

Limiting the override factor

For binary-coded override factors, the maximum possible overrides for path feedrate, axis feedrate and spindle speed can be limited:

MD12100 $MN_OVR_FACTOR_LIMIT_BIN (limit for binary coded override switch)

Override active

For overrides that have been enabled, the specified override values entered via the machine control panel become immediately active in all operating modes and machine functions.

Override inactive

An override factor of 100% is internally effective if an override is not activated. The override factor at the NC/PLC interface is not evaluated.

An exception is the zero setting for a binary interface and the 1st switch setting for a Gray-coded interface. In these cases, the override factors entered at the NC/PLC interface are evaluated. For a binary interface, the override factor is always 0%. For a Gray-coded interface, the value entered in machine data for the 1st switch position value is output as override value. It should be assigned the value "0".

5.4.3 Programmable feedrate override

Function

The "Programmable feedrate override" function can be used to change the velocity level of path and positioning axes via the part program.

Programming

Syntax	Meaning
`OVR=<value>`	Feedrate change for path feedrate F
`OVRA[<axis>=<value>`	Feedrate change for positioning feedrate FA

The programmable range is between 0 and 200%.

Default setting: 100%

Effectiveness

The NC/PLC interface signals DB21, ... DBB6 (rapid traverse or feedrate override active) and DB31, ... DBX1.7 (axis-specific override active) do **not** refer to the programmable feedrate override. The programmable feedrate override remains active when these signals are deactivated.

The effective override is calculated from the product of the "Programmable feedrate override" and the "Feedrate override via machine control panel (Page 159)".

The default setting for the "Programmable feedrate override" is 100%.

The default setting is effective:

- If no feedrate override is programmed or
- After reset if the machine data:
  MD22410 $MC_F_VALUES_ACTIVE_AFTER_RESET (F function is active even after reset) is not set.

Note

OVR is not effective with G33, G34, G35.

5.4.4 Dry run feedrate

Function

The dry run feedrate is used when testing part programs without machining the workpiece in order to allow the program or program sections to execute with an increased path feedrate, for example.

Activation

The dry run feedrate can be selected in the automatic modes and activated from the PLC or the operator panel front.

When activated from the operator panel front, the interface signal:
DB21, ... DBX24.6 (dry run feedrate selected)
is set and transferred from the basic PLC program to the interface signal:
DB21, ... DBX0.6 (activate dry run feedrate).

When selected on the PLC, the interface signal DB21, ... DBX0.6 (activate dry run feedrate) must be set from the PLC user program.

Effectiveness

As long as the "Activate dry run feedrate" interface signal is set, instead of the programmed feedrate, the feedrate value set via SD42100 DRY_RUN_FEED is effective in the way specified via SD42101 $SC_DRY_RUN_FEED_MODE (see parameterization):

The dry run feedrate is always interpreted as linear feedrate (G94).

Parameterization

Activation of dry run feedrate

The time of activation depends on the setting in the machine data:

MD10704 $MN_DRYRUN_MASK (activation of dry run feedrate)

Value	Meaning
0	The dry run feedrate may only be switched on and off at the end of the block (default setting).
1	The dry run feedrate can also be activated during the program processing (in the part program block). **Notice:** Activation during processing triggers an internal reorganization operation on the controller which causes the axes to be stopped for a short time. This can affect the surface finish of the workpiece being machined.
2	The dry run feedrate can be activated/deactivated at any time without the axes being stopped. The function only takes effect with a block "later" in the program run.

Changing the dry run feedrate

The feedrate for the dry run is entered in the setting data:

SD42100 $SC_DRY_RUN_FEED (dry run feedrate)

The setting data can be changed via the operator panel front in the "Parameters" operating area.

If the selection has been accepted by the NC, the following NC/PLC interface signal is set:

DB21, ... DBX318.6 (dry run feedrate active)

"DRY" is displayed in the operator panel front status bar to indicate an active dry run feedrate if:

- Selection took place during the program stop at the end of a block or
- The machine data MD10704 $MN_DRYRUN_MASK was set to "1" during the program execution

Mode of operation of the dry run feedrate

The mode of operation of the dry run feedrate entered in SD42100 can be set via the setting data:

SD42101 $SC_DRY_RUN_FEED_MODE

Value	Meaning
0	The programmed feedrate is compared to the dry run feedrate in SD42100 and then traversing is performed with the higher of the two feedrates (default setting).
1	The programmed feedrate is compared to the dry run feedrate in SD42100 and then traversing is performed with the lower of the two feedrates.
2	The dry run feedrate entered in SD42100 takes effect directly, irrespective of the programmed velocity.
3 - 9	Reserved
10	As for configuration 0, except for thread cutting (G33, G34, G35) and tapping (G331, G332, G63). These functions are executed as programmed.
11	As for configuration 1, except for thread cutting (G33, G34, G35) and tapping (G331, G332, G63). These functions are executed as programmed.
12	As for configuration 2, except for thread cutting (G33, G34, G35) and tapping (G331, G332, G63). These functions are executed as programmed.

5.4.5 Multiple feedrate values in one block

Function

The function "Multiple feedrate values in one block" can be used to activate six different feedrate values of an NC block, a dwell time or a retraction motion-synchronously, depending on the external digital and/or analog inputs.

When the input for the sparking out time or retraction path is activated, the distance-to-go for the path axes or the particular single axis is deleted and the dwell time or retraction is started.

The retraction is started within an IPO cycle.

Signals

The input signals are combined in one input byte for the function. A fixed functional assignment applies within the byte.

Table 5-1 Input byte for the "Multiple feedrates in one block" function

	Bit							
	7	6	5	4	3	2	1	0
Input no.	I7	I6	I5	I4	I3	I2	I1	I0
Feedrate address	F7	F6	F5	F4	F3	F2	ST	SR

I7 to I2: Activation of feedrates F7 to F2

E1: Activation of the dwell time ST/STA (in seconds)

I0: Activation of the retraction motion SR/SRA

Priority of the signals

The signals are scanned in ascending order starting at I0. Therefore, the retraction motion (SR) has the highest priority and the feedrate F7 the lowest priority.

SR and ST end the feedrate motions that were activated with F2 to F7.
SR also ends ST, i.e. the complete function.

The signal with the highest priority determines the current feedrate.

The response to loss of the respective highest-priority input (F2 - F7) can be defined with the machine data:

MD21230 $MC_MULTFEED_STORE_MASK (storage behavior for the "Multiple feedrate values in one block" function)

Bit	Value	Meaning
2 ... 7	0	With the loss of the respective highest-priority input, the associated feedrate is not retained (default setting).
	1	Set bit 2 to 7 ensures that the associated feedrate (F2 to F7) that was selected by the respective highest-priority input signal is also retained when there is a loss of the input signal and a lower-priority input is active.

The end-of-block criterion is satisfied when:

- The programmed end position is reached
- The retraction motion ends (SR)
- The dwell time elapses (ST)

Hardware assignment

The input byte for the "Multiple feedrate values in one block" function can be assigned a maximum of two digital input bytes or comparator input bytes of the NC I/O:

MD21220 $MC_MULTFEED_ASSIGN_FASTIN (assignment of the input bytes of the NC I/O for "Multiple feedrate values in one block"), bits 0 ... 15

The input bits can also be inverted:

MD21220 $MC_MULTFEED_ASSIGN_FASTIN, bit 16 ... 31

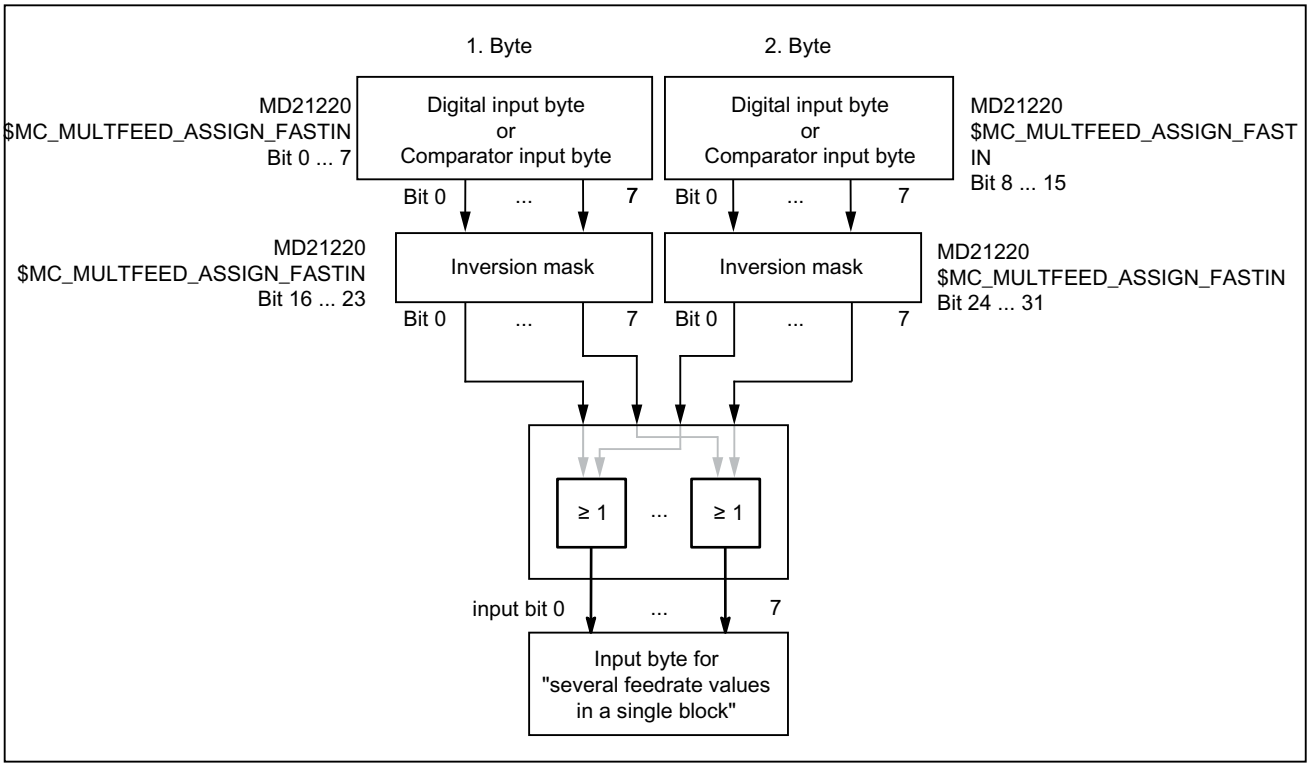


Figure 5-10 Signal assignment for the "Multiple feedrate values in one block" function

The assignment of the digital input bytes and parameterization of the comparators are described in:
Further information:
Function Manual Basic Functions; Digital and analog NC I/O

Programming

Path motion

The path feedrate is programmed under the address `F` and remains valid until an input signal is present. This value acts modally.

`F2=...` to `F7=...` can be used in addition to the path feedrate to program up to six further feedrates in the block. The numerical expansion indicates the bit number of the input that activates the feedrate when changed:

Example:

```
F7=1000          ;7 corresponds to input bit 7
```

The programmed values act non-modally. The path feedrate programmed under `F` applies in the next block.

Dwell (sparking out time) and retraction path are programmed under separate addresses in the block:

`ST=...`	Dwell time (for grinding sparking out time)
`SR=...`	Retraction path

These addresses apply non-modally.

Axial motion

The axial feedrates are programmed under address `FA` and remain valid until an input signal is present. They act modally.

`FMA[2,<axis>]=...` to `FMA[7,<axis>]=...` can be used to program up to six further feedrates per axis in the block.

The first expression in square brackets indicates the bit number of the input that activates the feedrate when changed. The second expression indicates the axis to which the feedrate applies.

Example:

```
FMA[3,Y]=1000          ; Axial feedrate for Y axis, corresponds to input bit 3
```

The values programmed under `FMA` act non-modally. The feedrate programmed under `FA` applies to the next block.

Dwell (sparking out time) and retraction path can also be defined for a single axis:

`STA[<axis>]=...`	Axial dwell time (sparking out time)
`SRA[<axis>]=...`	Axial retraction path

The expression in square brackets indicates the axis for which the sparking out time and retraction path apply.

Examples:

```
STA[X]=2.5             ; The sparking out time for the X axis is 2.5 seconds.
SRA[X]=3.5             ; The retraction path for the X axis is 3.5 (unit e.g. mm).
```

These addresses apply non-modally.

Note

Retraction path

The unit for the retraction path refers to the current valid unit of measurement (mm or inch).

The reverse stroke is always made in the opposite direction to the current motion. `SR`/`SRA` always programs the value for the reverse stroke. No sign is programmed.

Note

POS instead of POSA

If feedrates, sparking out time (dwell time) or return path are programmed for an axis on account of an external input, this axis must not be programmed as POSA axis (positioning axis over multiple blocks) in this block.

Note

Status query

It is also possible to poll the status of an input for synchronous commands of various axes.

Note

LookAhead

Look Ahead is also active for multiple feedrates in one block. In this way, the current feedrate can be restricted by the Look Ahead value.

Application

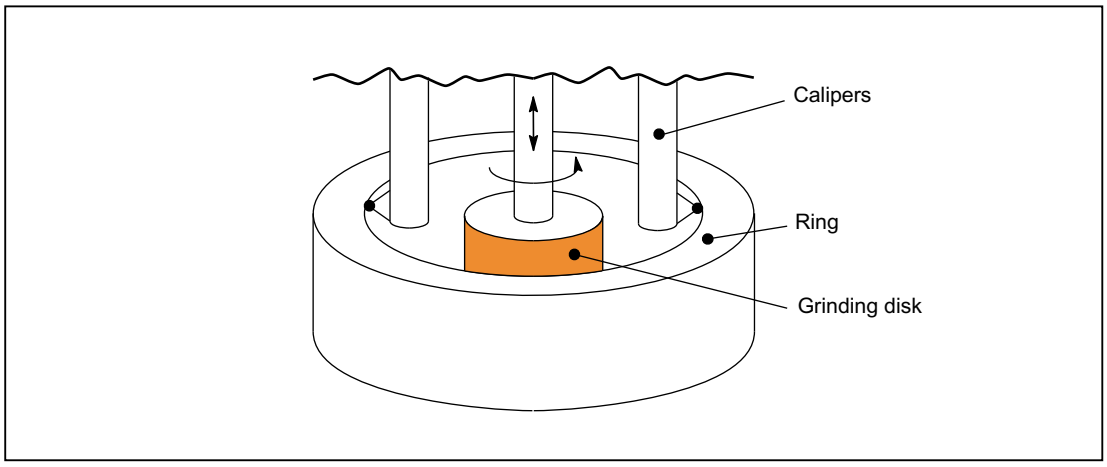
The "Multiple feedrate values in one block" function is used primarily for grinding, but is not restricted to it.

Typical applications are, for example:

- Analog or digital calipers
  Depending on whether the external inputs are analog or digital, various feedrate values, a dwell time and a retraction path can be activated. The limit values are defined via the setting data.
- Switching from infeed to working feedrate via proximity switch

Example

Internal grinding of a conical ring, where the actual diameter is determined using calipers and, depending on the limits, the feedrate value required for roughing, finishing or fine finishing is activated. The position of the calipers also provides the end position. Thus, the block end criterion is determined not only by the programmed axis position of the infeed axis but also by the calipers.




```
G00 X40 F=1000                                          ; Initial setting
G01 X48 F=20 F7=5 F3=2.5 F2=0.5 ST=1.5 SR=0.5
```

- SR=0.5: Retraction
- ST=1.5: Sparking-out time
- F2=0.5: Feedrate for fine finishing
- F3=2.5: Feedrate for finishing
- F7=5: Feedrate for roughing
- F=20: Feedrate for grinding in air

5.4.6 Fixed feedrate values

Function

The "Fixed feedrate values" function can be used to activate fixed feedrates (max. four) defined via the machine data instead of the programmed feedrate or the configured JOG velocities.

The function is available in AUTOMATIC and JOG mode.

Behavior in AUTOMATIC mode

The contour travels at the activated fixed feedrate, instead of using the programmed feedrate.

Behavior in JOG mode

The axis is traversed with the activated fixed feedrate instead of the configured JOG velocity / JOG rapid traverse velocity. The travel direction is specified via the interface signal.

Parameterization

The setting of the fixed feedrates is performed:

- For linear axes with the machine data:
  MD12202 $MN_PERMANENT_FEED[<n>]
- For rotary axes with the machine data:
  MD12204 $MN_PERMANENT_ROT_AX_FEED[<n>]

where <n> = 0, 1, 2, 3 (for fixed feedrate 1, 2, 3, 4)

Note

The fixed feedrates are always linear feedrate values. Switchover to linear feedrate is conducted internally even in case of revolutional feedrate.

Activation

The fixed feedrates are activated via NC/PLC interface signals:

- In AUTOMATIC mode for path/geometry axes using the channel-specific interface signals:
  DB21, ... DBX29.0 (activate fixed feedrate 1)
  DB21, ... DBX29.1 (activate fixed feedrate 2)
  DB21, ... DBX29.2 (activate fixed feedrate 3)
  DB21, ... DBX29.3 (activate fixed feedrate 4)
- In JOG mode for machine axes using the axis-specific interface signals:
  DB31, ... DBX3.2 (activate fixed feedrate 1)
  DB31, ... DBX3.3 (activate fixed feedrate 2)
  DB31, ... DBX3.4 (activate fixed feedrate 3)
  DB31, ... DBX3.5 (activate fixed feedrate 4)

Supplementary conditions

Effectiveness

The function "Fixed feedrate values" is **not** active:

- For spindles
- For positioning axes
- When tapping

Override = 0

The traversing behavior for override = 0 depends on the setting in machine data:

MD12200 $MN_RUN_OVERRIDE_0

DRF offset

The DRF offset cannot be activated for a selected fixed feedrate.

5.4.7 Programmable feedrate characteristics

Function

To permit flexible definition of the feedrate characteristic, the feedrate programming according to DIN 66025 has been extended by linear and cubic characteristics.

The cubic profiles can be programmed directly or as an interpolating spline.

Programming

You can program the following feedrate profiles:

- FNORM
  Behavior in accordance with DIN 66025 (default setting).
  An F-value programmed in the block is applied over the entire path of the block, and is subsequently regarded as a fixed modal value.
- FLIN
  An F value programmed in the block is traversed linearly over the path from the current value at the beginning of the block to the end of the block, and is subsequently regarded as modal value.
- FCUB
  The non-modal programmed F values (relative to the end of the block) are connected by a spline. The spline starts and ends tangentially to the previous or following feedrate setting. If the F address is missing in one block, then the last programmed F value is used.
- FPO
  The F address [syntax: `F=FPO(...,...,...)`] designates the characteristic of the feedrate via a polynomial from the current value to the end of the block in which it was programmed. The end value is treated as modal from there onwards.

Parameterization

If FLIN and FCUB are used in connection with compression COMPON, a tolerance can be defined for the path feedrate:

MD20172 $MC_COMPRESS_VELO_TOL (max. permissible deviation of the path feedrate with compression)

Boundary conditions

FLIN/FCUB

The path velocity profile programmed with FLIN or FCUB is not active together with revolutional feedrate for G95 as well as with constant cutting rate with G96/G961 and G97/G971.

Further information

For further information on the programmable feedrate characteristics, see Programming Manual NC Programming.

5.4.8 Feedrate for chamfer/rounding FRC, FRCM

The machining conditions can change significantly during surface transitions to chamfer/rounding. Hence, the chamfer/rounding contour elements require dedicated, optimized feedrate values to achieve the desired surface quality.

Function

The feedrate for chamfer/rounding can be programmed via NC addresses.

Programming

Syntax:
```
... FRC/FRCM=<value>
```

Meaning:

`FRC`:	**Non-modal** feedrate for chamfer/rounding
`FRCM`:	**Modal** feedrate for chamfer/rounding
`<value>`:	The feedrate is interpreted according to the active feedrate type: • G94, G961, G971: Feedrate in mm/min or inch/min or °/min • G95, G96, G97: Revolutional feedrate in mm/rev or inch/rev

Note

FRC is only effective if a chamfer/rounding is programmed in the block or if RNDM has been activated.

FRC overwrites the F or FRCM value in the current block.

The feedrate programmed under FRC must be greater than zero.

FRCM=0 activates the feedrate programmed under F for chamfering/rounding.

Parameterization

Assignment of the chamfer/rounding to the previous or following block

The feedrate type (G94, G95, G96, G961 ...) and therefore the conversion to the internal format must be consistent within the block for F and FRC/FRCM. In this context, the following machine data must be taken into account:

MD20201 $MC_CHFRND_MODE_MASK (chamfer/rounding behavior)

Bit	Value	Meaning
0	0	The technology of the chamfer/rounding (feedrate, feedrate type, M commands, etc.) is determined by the following block (default setting).
	1	The technology of the chamfer/rounding is determined by the previous block (recommended setting).

Maximum number of empty blocks

The number of blocks without traversing information in the compensation plane (empty blocks) permitted between two blocks with traversing information during active chamfer/rounding, is limited. The maximum number is specified in the machine data:

MD20200 $MC_CHFRND_MAXNUM_DUMMY_BLOCKS (empty blocks for chamfer/radii)

Supplementary conditions

FLIN/FCUB

Feedrate interpolation FLIN and FCUB is **not** possible for chamfer/rounding.

G0

FRC/FRCM is not active when a chamfer is traversed with G0. The programming is possible in accordance with the F value without error message.

Change G94 ↔ G95

If FRCM is programmed, the FRCM value will need to be reprogrammed like F on change G94 ↔ G95, etc. If only F is reprogrammed and if the feedrate type FRCM > 0 before the change, an error message will be output.

Example

Example 1: MD20201 bit 0 = 0; take feedrate from following block (default setting!)

```
Program code                        Comment
N10 G0 X0 Y0 G17 F100 G94
N20 G1 X10 CHF=2                    ; Chamfer N20-N30 with F=100 mm/min
N30 Y10 CHF=4                       ; Chamfer N30-N40 with FRC=200 mm/min
N40 X20 CHF=3 FRC=200               ; Chamfer N40-N60 with FRCM=50 mm/min
N50 RNDM=2 FRCM=50
N60 Y20                             ; Modal rounding N60-N70 with FRCM=50 mm/min
N70 X30                             ; Modal rounding N70-N80 with FRCM=50 mm/min
N80 Y30 CHF=3 FRC=100               ; Chamfer N80-N90 with FRC=100 mm/min
N90 X40                             ; Modal rounding N90-N100 with F=100 mm/min (dese-
                                      lection of FRCM)
N100 Y40 FRCM=0                     ; Modal rounding N100-N120 with G95 FRC=1 mm/rev
N110 S1000 M3
```

Program code	Comment
N120 X50 G95 F3 FRC=1	
...	
M02	

Example 2: MD20201 bit 0 = 1; take feedrate from previous block (recommended setting!)

Program code	Comment
N10 G0 X0 Y0 G17 F100 G94	
N20 G1 X10 CHF=2	; Chamfer N20-N30 with F=100 mm/min
N30 Y10 CHF=4 FRC=120	; Chamfer N30-N40 with FRC=120 mm/min
N40 X20 CHF=3 FRC=200	; Chamfer N40-N60 with FRC=200 mm/min
N50 RNDM=2 FRCM=50	
N60 Y20	; Modal rounding N60-N70 with FRCM=50 mm/min
N70 X30	; Modal rounding N70-N80 with FRCM=50 mm/min
N80 Y30 CHF=3 FRC=100	; Chamfer N80-N90 with FRC=100 mm/min
N90 X40	; Modal rounding N90-N100 with FRCM=50 mm/min
N100 Y40 FRCM=0	; Modal rounding N100-N120 with F=100 mm/min
N110 S1000 M3	
N120 X50 CHF=4 G95 F3 FRC=1	; Chamfer N120-N130 with G95 FRC=1 mm/rev
N130 Y50	; Modal rounding N130-N140 with F=3 mm/rev
N140 X60	
...	
M02	

5.4.9 Non-modal feedrate FB

Function

The "Non-modal feedrate" function can be used to define a separate feedrate for a single part program block. After this block, the previous modal path feedrate is active again.

Programming

Syntax:

```
... FB=<value>
```

Meaning:

FB: Separate feedrate for the current block

<value>: The feedrate is interpreted according to the active feedrate type:

- G94, G961, G971: Feedrate in mm/min or inch/min or °/min
- G95, G96, G97: Revolutional feedrate in mm/rev or inch/rev

Note

The feedrate programmed under FB must be greater than zero.

If no traversing motion is programmed in the block (e.g. computation block), the FB has no effect.

If no explicit feed for chamfering/rounding is programmed, then the value of FB also applies for any contour element chamfering/rounding in this block.

Simultaneous programming of FB and FD (manual handwheel traversing with feedrate override) or F (modal path feedrate) is not possible.

5.4.10 Influencing the single axis dynamic response

Single axes

Single axes can be programmed in the part program, in synchronized actions and via the PLC:

- Part program:
  ```
  POS[<axis>]=...
  POSA[<axis>]=...
  SPOS[<axis>]=...
  SPOSA[<axis>]=...
  OS[<axis>]=...
  OSCILL[<axis>]=...
  ```
- Synchronized actions:
  ```
  EVERY ... DO
  POS[<axis>]=...
  SPOS[<spindle>]=...
  MOV[<axis>]=...
  ```
- PLC: FC18

Dynamic response

The dynamic response of an axis is influenced by:

- MD32060 $MA_POS_AX_VELO (positioning axis velocity)
  The effective positioning axis velocity can be changed:
  - Part program / synchronized action: Axial feedrate FA or percentage feedrate override OVRA
  - PLC: Specification of FRate or overwriting the axial override
- MD32300 $MA_MAX_AX_ACCEL (maximum axis acceleration)
  The effective maximum axis acceleration can be changed:
  - Part program indirectly: Writes the machine data with subsequent "Activate machine data"
  - Part program directly: Percentage acceleration override ACC
  - Synchronized actions indirectly: Writes the machine data and triggers an ASUP for the activation of "Activate machine data" function
  - Synchronized actions directly: Percentage acceleration override `ACC` (cannot be preset by the PLC).

  Via the PLC, the same options apply as in synchronized actions.
- Part program commands: BRISKA, SOFTA, DRIVEA, JERKA
  Cannot be programmed in synchronized actions (only indirectly via ASUP).
  Cannot be specified by the PLC (only indirectly via ASUP).
- Active servo parameter set
  The active parameter set can be changed:
  - Part program / synchronized action: SCPARA
  - PLC: DB31, … DBX9.0-2 (controller parameter set)

  For detailed information on the servo parameter sets, see "Parameter sets of the position controller (Page 75)".

Note

Dynamic response changes

Dynamic response changes made in the part program do not affect command or PLC axis motion. Dynamic response changes made in synchronized actions have no effect on traversing motion programmed in the part program.

Feedforward control

The type of feedforward control and the path axes that should be traversed with feedforward control can be directly programmed in the part program using FFWON/FFWOF. In synchronized actions and from the PLC, programming is only possible indirectly via an ASUP.

Percentage acceleration override (ACC)

In a part program or synchronized action, the acceleration specified in machine data:
MD32300 $MA_MAX_AX_ACCEL (maximum axis acceleration)
can be changed in a range from 0% – 200% using the ACC command.

Syntax:
```
ACC[<axis>]=<value>
```

Meaning:

ACC:	Keyword for the programming of the percentage acceleration override
<Axis>:	Name of the channel axis or spindle
<value>:	Acceleration change in percent relative to MD32300
	Value range: 0 ... 200

The actual axial acceleration value can be read via the system variable $AA_ACC. It is determined by:

$AA_ACC[<axis>] = (MD32300 $MA_MAX_AX_ACCEL[<axis>]) * ACC[<axis>] / 100

MD32320 $MA_DYN_LIMIT_RESET_MASK can be used to specify the basic setting of the value programmed with ACC for a channel reset or end of part program M30.

Note

The acceleration override programmed with ACC can be read using the system variable $AA_ACC. However, $AA_ACC is read in the part program at a different time than when reading in a synchronized action.

The system variables $AA_ACC only contain the value programmed in the part program with ACC if, in the meantime, the acceleration override was not changed by programming ACC in a synchronized action. The same applies for the reverse situation.

Percentage acceleration override and main run axes

Depending on whether the system variable $AA_ACC is read in the part program or synchronized action, the value for the acceleration override programmed with ACC is output for the NC axes or main run axes (command axes, PLC axes, asynchronous oscillating axes, etc.).

For correct results, system variable $AA_ACC must therefore always be read at the same location (part program or synchronized action) from where the acceleration override was programmed with ACC.

Examples:

Writing ACC in a part program:
```
N80 G01 POS[X]=100 FA[X]=1000 ACC[X]=90 IPOENDA[X]
```

Writing ACC in a synchronized action:
```
N100 EVERY $A_IN[1] DO POS[X]=50 FA[X]=2000 ACC[X]=140 IPOENDA[X]
```

Writing ACC and reading $AA_ACC in a part program:

```
ACC[X]=50                                          ; writing
RO=$AA_ACC[X]                                      ; reading

IF (RO <> $MA_MAX_AX_ACCEL[X] * 0.5)               ; checking
   SETAL(61000)
ENDIF
```

Writing ACC and reading $AA_ACC in a synchronized action:

```
WHEN TRUE DO ACC[X]=25 R0=$AA_ACC[X]               ; writing and reading
G4 F1

IF (RO <> $MA_ MAX_AX_ACCEL[X] * 0.25)             ; checking
   SETAL(61001)
ENDIF
```

end-of-motion criterion for single axes

Similar to the block change criterion for path interpolation (G601, G602, G603) the end-of-motion criterion for traversing motion of individual axes can be programmed in part programs / synchronized actions:

Program command	End-of-motion criterion
FINEA[<axis>]	"Exact stop fine"
COARSEA[<axis>]	"Exact stop coarse"
IPOENDA[<axis>]	"Interpolator stop" (IPO stop)

The most recently programmed value is kept after the end of program or NC reset.

The effective end-of-motion criterion can be read using the axis-specific system variable $AA_MOTEND.

Note

Depending on whether the system variable $AA_MOTEND is read in the part program or synchronized action, it contains the value for the NC axes or the main run axes.

Example:

Part program:

```
N80 G01 POS[X]=100 FA[X]=1000 ACC[X]=90 COARSEA[X]
```

Synchronized action:

```
N100 EVERY $A_IN[1] DO POS[X]=50 FA[X]=2000 ACC[X]=140 IPOEN-
DA[X]
```

Further information

For further information on the block change and movement end criteria for FINEA, COARSEA and IPOENDA, see:
Block change (Page 204)

Programmable servo parameter set (SCPARA)

In the part program / synchronized action, the servo parameter set can be specified using SCPARA.

Syntax

```
SCPARA[<axis>] = <parameter set number>
```

Meaning

SCPARA:	Keyword for the activation of the specified servo parameter set
<Axis>:	Name of the channel axis
<parameter set number>:	Number of the servo parameter set to be activated

Note

The activation of the parameter set specified using SCPARA can be suppressed from the PLC user program:

DB31,… DBX9.3 = 1 (parameter set specification disabled by SCPARA)

In this case, **no** message is displayed.

The number of the active parameter set can be read using the system variable $AA_SCPAR.

Boundary conditions

Different end-of-motion criteria

Different end-of-motion criteria will affect how quickly or slowly part program blocks are completed. This can have side effects for technology cycles and PLC user parts.

Parameter set change

The PLC user program must be expanded if the servo parameter set is to be changed both inside a part program or synchronized action and the PLC.

Power On

After POWER ON, the following basic settings are made:

- Percentage acceleration override for all single-axis interpolations: 100%
- End-of-motion criterion for all single-axis interpolations: FINEA
- Servo parameter set: 1

Mode change

When the operating mode is changed from AUTOMATIC to JOG, the programmed dynamic response changes remain valid.

Reset

With reset, the last programmed value remains for the part program specifications. The settings for main-run interpolations do not change.

Block search

The last end-of-motion criterion programmed for an axis is collected and output in an action block. The last block with a programmed end-of-motion criterion that was processed in the block search run serves as a container for all programmed end-of-motion criteria for all axes.

5.5 Boundary conditions

Unit of measurement

The valid unit of measurement of the feedrates depends on the set measuring system and the entered axis type:

MD10240 $MN_SCALING_SYSTEM_IS_METRIC (basic system of the control metric/inch)

MD30300 $MA_IS_ROT_AX (rotary or linear axis)

Initial setting for the feedrate type

The initial setting for the feedrate type is specified in the machine data:

MD20150 $MC_GCODE_RESET_VALUES (initial setting of the G groups)

The default setting is G94.

The initial setting of the feedrate type is only displayed when a part program is started.

Effectiveness after reset

Whether the last programmed F, FA, OVR, OVRA values are also active after reset depends on the setting in the machine data:

MD22410 $MC_F_VALUES_ACTIVE_AFTER_RESET (F function is active even after reset)

Spindle positioning

With active G95, G96, G961, G97, G971, G33, G34, G35 spindle positioning should not be performed, because the derived path feedrate following spindle positioning = 0. If the programmed axis position has not then been reached, the block cannot be completed.

5.6 Data lists

5.6.1 Machine data

5.6.1.1 NC-specific machine data

Number	Identifier: $MN_	Description
10651	IPO_PARAM_THREAD_NAME_TAB	Name of the interpolation parameters for convex threads
10704	DRYRUN_MASK	Activation of dry run feedrate
10710	PROG_SD_RESET_SAVE_TAB	Setting data to be updated
11410	SUPPRESS_ALARM_MASK	Mask for suppressing special alarms
11550	STOP_MODE_MASK	Defines the stop behavior
12000	OVR_AX_IS_GRAY_CODE	Axis feedrate override switch, Gray-coded
12010	OVR_FACTOR_AX_SPEED	Evaluation of the axis feedrate override switch
12020	OVR_FEED_IS_GRAY_CODE	Path feedrate override switch, Gray-coded
12030	OVR_FACTOR_FEEDRATE	Evaluation of the path feedrate override switch
12040	OVR_RAPID_IS_GRAY_CODE	Rapid traverse override switch, Gray-coded
12050	OVR_FACTOR_RAPID_TRA	Evaluation of the rapid traverse override switch
12060	OVR_SPIND_IS_GRAY_CODE	Spindle override switch, Gray-coded
12070	OVR_FACTOR_SPIND_SPEED	Evaluation of the spindle override switch
12080	OVR_REFERENCE_IS_PROG_FEED	Override reference velocity
12082	OVR_REFERENCE_IS_MIN_FEED	Defines the reference of the path override
12090	OVR_FUNCTION_MASK	Selection of override specifications
12100	OVR_FACTOR_LIMIT_BIN	Limit for binary-coded override switch
12200	RUN_OVERRIDE_0	Traversing with override 0
12202	PERMANENT_FEED	Fixed feedrates for linear axes
12204	PERMANENT_ROT_AX_FEED	Fixed feedrates for rotary axes

5.6.1.2 Channel-specific machine data

Number	Identifier: $MC_	Description
20100	DIAMETER_AX_DEF	Geometry axes with transverse axis functions
20150	GCODE_RESET_VALUES	Basic setting of the G groups
20172	COMPRESS_VELO_TOL	Maximum permissible deviation from path feed for compression
20200	CHFRND_MAXNUM_DUMMY_BLOCKS	Empty blocks with phase/radii
20201	CHFRND_MODE_MASK	Behavior for chamfer/rounding
20750	ALLOW_GO_IN_G96	G0 logic for G96, G961
21200	LIFTFAST_DIST	Traversing path for fast retraction from the contour

Number	Identifier: $MC_	Description
21220	MULTFEED_ASSIGN_FASTIN	Assignment of input bytes of NC I/O for "Multiple feedrate values in one block"
21230	MULTFEED_STORE_MASK	Storage behavior for the "Multiple feedrate values in one block" function
22240	AUXFU_F_SYNC_TYPE	Output timing of F functions
22410	F_VALUES_ACTIVE_AFTER_RESET	F function active after reset

5.6.1.3 Axis/Spindle-specific machine data

Number	Identifier: $MA_	Description
30300	IS_ROT_AX	Rotary axis/spindle
32000	MAX_AX_VELO	Maximum axis velocity
32060	POS_AX_VELO	Initial setting for positioning axis velocity
32300	MAX_AX_ACCEL	Axis acceleration
32320	DYN_LIMIT_RESET_MASK	Reset behavior of dynamic limits
34990	ENC_ACTIVAL_SMOOTH_TIME	Smoothing time constant for actual values
35100	SPIND_VELO_LIMIT	Maximum spindle speed
35130	GEAR_STEP_MAX_VELO_LIMIT	Maximum speed of gear stage
35140	GEAR_STEP_MIN_VELO_LIMIT	Minimum speed of gear stage
35160	SPIND_EXTERN_VELO_LIMIT	Spindle-speed limitation via PLC

5.6.2 Setting data

5.6.2.1 Channel-specific setting data

Number	Identifier: $SC_	Description
42000	THREAD_START_ANGLE	Start angle for thread
42010	THREAD_RAMP_DISP	Acceleration behavior of axis when thread cutting
42100	DRY_RUN_FEED	Dry run feedrate
42101	DRY_RUN_FEED_MODE	Dry run feed mode
42110	DEFAULT_FEED	Default value for path feed
42600	JOG_FEED_PER_REV_SOURCE	Revolutional feedrate control in the JOG mode
43300	ASSIGN_FEED_PER_RES_SOURCE	Revolutional feedrate for positioning axes/spindles

5.6.2.2 Axis/spindle-specific setting data

Number	Identifier: $SA_	Description
43210	SPIND_MIN_VELO_G25	Programmed spindle speed limiting G25
43220	SPIND_MAX_VELO_G26	Programmed spindle speed limiting G26
43230	SPIND_MAX_VELO_LIMS	Spindle speed limiting with G96

P2: Positioning axes

6

6.1 Brief description

Axes for auxiliary movements

In addition to axes for machining a workpiece, modern machine tools can also be equipped with axes for auxiliary movements, e.g.:

- Axis for tool magazine
- Axis for tool turret
- Axis for workpiece transport
- Axis for pallet transport
- Axis for loader (also multi-axis)
- Axis for tool changer
- Axis for sleeve assembly / end support

The axes for the workpiece machining are called path axes. Within the channel they are guided by the interpolator such that they start simultaneously, accelerate, reach the end point and stop together.

Axes for auxiliary movements are traversed independently of the path axes at a separate, axis-specific feedrate. In the past, many of these axes were moved hydraulically and started by an auxiliary function in the part program. With the closed-loop axis control implemented in the NC,

the axis can be addressed by name in the part program and its actual position displayed on the screen.

Note

"Positioning axis/Auxiliary spindle" option

Axes for auxiliary movements must not be interpolating ("full-value") NC axes. Auxiliary movements may also be carried out using special axes, which can be obtained using the "Positioning axis/Auxiliary spindle" option.

Functional restrictions

Optional positioning axes/auxiliary spindles have fewer functions. The following functions are **not** possible:

- Using the axis as a path axis, geometry axis, or special path axis
- Incorporating the axis into the geometry axis grouping (GEOAX)
- Rigid thread cutting and tapping

Commissioning

As standard, axes are defined as interpolating axes:

MD30460 $MA_BASE_FUNCTION_MASK bit 8 = 0

If an axis is to be operated as a positioning axis/auxiliary spindle with reduced functionality, the value for bit 8 must be set to "1":

MD30460 $MA_BASE_FUNCTION_MASK bit 8 = **1**

Function

The "positioning axes" function makes it easier to integrate axes for auxiliary movement into the control system:

- during programming:
  The axes are programmed together with the axes for workpiece machining in the same part program, without having to sacrifice valuable machining time.
  There are special (`POS`, `POSA`) traversing instructions.
- during program testing/start-up:
  Program testing and start-up are performed simultaneously for all axes.
- during operation:
  Operation and monitoring of the machining process commence simultaneously for all axes.
- during PLC configuring/commissioning:
  No allowance has to be made on PLC or external computers (PCs) for synchronization between axes for machining and axes for auxiliary movements.
- during system configuring:
  A second channel is not required.

Motions and interpolations

Each channel has one path interpolator and at least one axis interpolator with the following interpolation functions:

- for path interpolator:
  Linear interpolation (G1), circular interpolation (G2 / G3), spline interpolation, etc.
- for axis interpolator:
  If a positioning axis is programmed, an axis interpolator starts in the control (with linear interpolation G1).
- End-of-motion criterion:
  The programmed end position of a positioning axis has been reached when the end-of-motion criterion FINEA, COARSA or IPOENDA is fulfilled.
- Path axes with rapid traverse movement:
  Path axes can be traversed in linear or non-linear interpolation mode with rapid traverse movement (G0).
  Further information
  Function Manual Basic Functions; Continuous-path mode, exact stop, LookAhead
- Autonomous singleaxis operations:
  Single PLC axes, command axes started via static synchronized actions or asynchronous reciprocating axes can be interpolated independently of the NC.
  An axis/spindle interpolated by the main run then reacts independently of the NC program. The channel response triggered by the program run is decoupled to transfer the control of a certain axis / spindle to the PLC.
- Control by PLC:
  All channel-specific signals normally act to the same extent on path and positioning axes. Positioning axes can be controlled via additional, axis-specific signals.
  PLC axes are traversed by the PLC via special function blocks in the basic program; their movements can be asynchronous to all other axes. The travel motions are executed separate from the path and synchronized actions.

6.2 Own channel, positioning axis or concurrent positioning axis

When axes are provided for auxiliary movements on a machine tool, the required properties will decide whether the axis is to be:

- programmed in an internal part program (see Section "Own channel (Page 188)").
- programmed in the same part program as the machining operation (see Section "Positioning axis (Page 188)").
- started exclusively by the PLC during machining (see Section "Concurrent positioning axis (Page 191)").

6.2.1 Own channel

A channel represents a self-contained NC which, with the aid of a part program, can be used to control the movement of axes, spindles and machine functions independently of other channels.

Non-dependence between channels

Independence between channels is assured by means of the following provisions:

- An active part program per channel
- Channel-specific interface signals such as
  - DB21, ... DBX7.1 (NC start)
  - DB21, ... DBX7.3 (NC stop)
  - DB21, ... DBX7.7 (reset)
- one feedrate override per channel
- One rapid traverse override per channel
- Channel-specific evaluation and display of alarms
- Channel-specific display, e.g. for
  - Actual axis positions
  - Active G commands
  - Active auxiliary functions
  - Current program block
- Channel-specific testing and channel-specific control of programs:
  - Single block
  - Dry run (DRY RUN)
  - Block search
  - Program test

Further information

For more information on the channel functionality, please refer to:
Function Manual Basic Functions; Mode group, channel, program mode, reset response

6.2.2 Positioning axis

Positioning axes are programmed together with path axes, i.e. with the axes that are responsible for workpiece machining.

Commands for positioning axes and path axes can be included in the same NC block. Although they are programmed in the same NC block, the path and positioning axes are not interpolated

together and do not reach their end point simultaneously (no direct time relationship, see also Section "Motion behavior and interpolation functions (Page 192)").

Positioning axis types and block change

The block change time depends on the programmed positioning axis type (refer also to Section "Block change (Page 204)"):

Type	Description
1	The block change occurs when all path and positioning axes have reached their programmed end point.
2	The block change occurs when all path axes have reached their programmed end point. With positioning axis type 2, it is possible to approach the programmed end point across several block limits.
3	It is possible to set the block change within the braking ramp of the single axis interpolation if the criteria for the motion end and the block change are fulfilled for the path interpolation.

Motion synchronization

Positioning axes permit movements to be activated from the same machining program and such movements to be synchronized at block limits (type 1) or at explicit points by means of a `WAITP` command (type 2).

Motion end criterion for block change in the brake ramp

For single-axis interpolation, it is also possible to set another end-of-motion criterion for the block change in the braking ramp.

Traverse path axes in G0 as positioning axis

Each path axis can be traversed as positioning axis in rapid traverse movement (`G0`). Thus all axes travel to their end point independently.

In this way, two sequentially programmed X and Z axes are treated like positioning axes in conjunction with `G0`. The block change to axis Z can be initiated by axis X as a function of the braking ramp time setting (100-0%). Axis Z starts to move while axis X is still in motion. Both axes approach their end point independently of one another.

Axis types

Positioning axes can be linear axes and rotary axes.

Positioning axes can also be configured as indexing axes.

Independence of positioning axes and path axes

The mutual independence of positioning and path axes is ensured by the following measures:

- No shared interpolation
- Each positioning axis has a dedicated axis interpolator

- Dedicated feedrate override for each positioning axis
- Dedicated programmable feedrate
- Dedicated "axis-specific delete distance-to-go" interface signal

Dependencies

Positioning axes are dependent in the following respects:

- A shared part program
- Starting of positioning axes only at block boundaries in the part program
- With rapid traverse movement `G0` path axes traverse as positioning axes in one of two different modes
- No rapid traverse override
- The following interface signals act on the entire channel and therefore on positioning axes:
  - DB21, ... DBX7.1 (NC start)
  - DB21, ... DBX7.3 (NC stop)
  - DB21, ... DBX7.7 (reset)
  - DB21, ... DBX6.1 (read-in disable)
- Alarms specific to program and channel also deactivate positioning axes.
- Program control (dry run feed, program test, DRF, ... etc.) also act on positioning axes
- Block search and single block also act on positioning axes.
- The last block with a programmed end-of-motion criterion that was processed in the search run serves as a container for setting all axes.
- Group 1 (modal movement commands) of the G commands (i.e. `G0`, `G1`, `G2`, ...) does not apply to positioning axes.

Further information

Programming Manual NC Programming

Applications

The following are typical applications for positioning axes:

- Single-axis loaders
- Multi-axis loaders without interpolation (PTP → point-to-point traversing)
- Workpiece feed and transport

Other applications are also possible:

- With G0 workpiece delivery and workpiece transport can travel to their end points independently of one another.
- On machines with several machining processes in sequence: Significant reduction in individual machining steps due to block change in the braking ramp of the single-axis interpolation.

Note

Positioning axes are not suitable for multi-axis loaders that require interpolation between the axes (path interpolator).

6.2.3 Concurrent positioning axis

Concurrent positioning axes are positioning axes with the following properties:

- Activation from the PLC need not take place at block limits, but can be implemented at any time in any operating mode (even when a part program is already being processed in the channel).
- Program command WAITP is required to move a concurrent positioning axis from the part program immediately after power ON.
- The part program continues to run uninhibited, even if the concurrent positioning axis has not reached the position defined by the PLC.
- An automatic axis change is possible, depending on the setting in the machine data MD30552 $MA_AUTO_GET_TYPE.
- With programming commands:
  - GET(<axis>) or WAITP(<axis>) becomes a concurrent positioning axis of the channel axis again.
  - "RELEASE (axis)" or WAITP(<axis>) is a channel axis that becomes a concurrent axis under PLC control.

Activation from PLC

For SINUMERIK 840D sl, the concurrent positioning axis is activated via FC 18 from the PLC.

- Feedrate
  For feedrate = 0, the feedrate is determined from the following machine data:
  MD32060 $MA_POS_AX_VELO (initial setting for positioning axis velocity)
- Absolute dimensions (G90), incremental dimensions (G91)
  Absolute dimensions along shortest path for rotary axes (<rotary axis name>=DC(<value>))

The following functions are defined:

- Linear interpolation (G1)
- Feedrate in mm/min or degrees/min (G94)

- Exact stop (G9)
- Settable zero offsets currently selected are valid

Applications

Typical applications for concurrent positioning axes include:

- Tool magazines with manual loading and unloading during machining
- Tool magazines with tool preparation during machining

6.3 Motion behavior and interpolation functions

6.3.1 Path interpolator and axis interpolator

Path interpolator

Every channel has a path interpolator for a wide range of interpolation modes such as linear interpolation (G1), circular interpolation (G2/G3), spline interpolation etc.

Axis interpolator

Each channel has axis interpolators in addition to path interpolators. The maximum number corresponds to the maximum number of existing channel axes.

If a positioning axis is programmed, an axis interpolator starts in the control with straight line interpolation G1. This axis interpolator runs independently of the path interpolator until the programmed end position of the positioning axis has been reached.

There is no time relationship between the path interpolator and the axis interpolator, nor between the axis interpolators.

Path control mode (G64) is not possible with positioning axes.

The programmed end position of a positioning axis has been reached when the end-of-motion criterion FINEA, COARSA or IPOENDA is fulfilled.

6.3.2 Autonomous singleaxis operations

Functionality

Single PLC axes, command axes started via static synchronized actions or asynchronous reciprocating axes can be interpolated independently of the NC. An axis/spindle interpolated by the main run then reacts independently of the NC program with respect to:

- NC stop
- Alarm handling
- Program control
- End of program
- RESET

Boundary conditions

Axes/spindles currently operating according to the NC program are not controlled by the PLC.

Command axis movements **cannot** be started via non-modal or modal synchronized actions for PLC-controlled axes/spindles. Alarm 20143 is signaled.

Transfer axis control to the PLC

Description of the sequence

1. PLC → NC: Request to control the axis
   DB31, ... DBX28.7 = 1 (PLC controls axis)
2. NC: Checks whether the axis is a main run axis or a neutral axis.
3. NC: Checks whether an additional axis may be controlled from the PLC.
4. NC confirms the transfer:
   - DB31, ... DBX63.1 = 1 (PLC controls the axis)
   - System variable $AA_SNGLAX_STAT = 1

Result: The PLC controls the axis/spindle.

Alternatives

Initial state: The axis is controlled by the PLC. As a result of a channel stop, the channel is in the "interrupted" state.

- Axis state "inactive" ⇒
  - The stop state is canceled.
  - If the axis is started, this directly results in axis motion.
- Axis state "active" ⇒
  - The stop state is **not** canceled.
  - Generate the axis state according to **Use case 2 "Stop axis"**.
  - Resume axis motion according to **Use case 3 "Continue axis motion"**.
- A reset is performed in the channel ⇒
  This process is asynchronous to control acceptance by the PLC. The two previously mentioned alternatives can occur or the axis is assigned to the channel and is reset.

Boundary conditions

Axes/spindles, traversed by an NC program, cannot be transferred to the PLC. Axes/spindles, which are traversed by static synchronized actions or as oscillating axis, as neutral axis, concurrent positioning axis or command axis, can be transferred.

Relinquish axis control by the PLC

Description of the sequence:

1. PLC → NC: The PLC returns axis control to the NC
   DB31, ... DBX28.7 = 0 (PLC controls axis)
2. NC: Checks whether an axial alarm is present.
3. NC: Checks whether a movement has been activated that has still not been completed? If yes, then the movement is stopped with an axial stop according to **Use case 2 "Stop axis/ spindle"**.
4. NC: Carries out an axial reset corresponding to **Use case 4 "Reset axis/spindle"**.
5. NC confirms the acceptance:
   DB31, ... DBX63.0 = 0 (reset executed)
   DB31, ... DBX63.1 = 0 (PLC controls the axis)
   DB31, ... DBX63.2 = 0 (axis stop active)
   System variable $AA_SNGLAX_STAT = 0

Result: The NC has now taken over control of the axis/spindle.

Alternatives

In the following cases the NC confirms the transfer - but internally sets the "stopped" channel state for the axis/spindle:

- The channel is in the "interrupted" state
- A stop alarm is active for the channel
- A stop alarm is active for the mode group

Boundary conditions

The axis/spindle must be operating under PLC control.

The NC confirms acceptance of an axis/spindle only if an axial alarm is not active.

Description of the sequence based on use cases

Requirement

The axis/spindle is controlled by the PLC

Relevant NC/PLC interface signals

One of the axes/spindles controlled by the PLC can be influenced by the following NC/PLC interface signals independent of the NC program:

- DB21, ... DBX6.2 (delete distance-to-go)
- DB31, ... DBX28.1 (reset)
- DB31, ... DBX28.2 (continue)
- DB31, ... DBX28.6 (stop along braking ramp)

For signal flow between the NC and PLC at the NC/PLC interface during autonomous single operations, see Section "Control by the PLC (Page 212)".

Use case 1: Cancel axis/spindle

The behavior when canceling the axis/spindle function is the same as for "delete distance-to-go":

DB21, ... DBX6.2 = 1 (delete distance-to-go)

Use case 2: Stop axis/spindle

The following traversing motion of the axis/spindle controlled from the main run is stopped:

- PLC axis
- Asynchronous oscillating axis
- Command axis by static synchronized action
- Superimposed motion $AA_OFF,DRF handwheel traversal, online tool offset and external work offset.

Following axis movements of the axis/spindle are not stopped.

Description of the sequence:

- PLC → NC: Request to stop the axis/spindle
  DB31, ... DBX28.6 = 1 (stop along braking ramp)
- NC: Brakes the axis along a ramp.
- NC confirms the execution:
  - DB31, ... DBX60.6 = 1 (exact stop coarse)
  - DB31, ... DBX60.7 = 1 (exact stop fine)
  - DB31, ... DBX63.2 = 0 (axis stop active)
  - DB31, ... DBX64.6 / 7 = 0 (traversing command minus/plus)
  - Axis status interrupted with system variable $AA_SNGLAX_STAT == 3.

Result: The axis/spindle is stopped.

Note

Following axis movements

Following axis movements can only be suppressed when the leading axis stops.

Retraction motion

Retraction motion triggered by the "Extended stop and retract" function cannot be stopped.

Further information

R3: Extended stop and retract (Page 615)

Use case 3: Continue axis/spindle

Traversing motion interrupted after **Use case 2 "Stop axis"** should be continued.

Description of the sequence:

- PLC → NC: Continue axis
  DB31, ... DBX28.2 = 1 (continue)
- NC: Checks whether for the axis/spindle an axial alarm with delete criterion "CANCELCLEAR" or "NCSTARTCLEAR" is present? If yes, then this is deleted.
- NC: Checks whether axis motion can be resumed? If yes, then the axis/spindle makes the transition into the "active" state.
- NC confirms the execution:
  - DB31, ... DBX60.6 = 0 (exact stop coarse)
  - DB31, ... DBX60.7 = 0 (exact stop fine)
  - DB31, ... DBX63.2 = 0 (axis stop active)
  - DB31, ... DBX64.6 / 7 = 1 (traversing command minus/plus)
  - Axis state active with system variable $AA_SNGLAX_STAT == 4.

Result: Traversing motion of the axis/spindle is continued.

Boundary conditions

In the following cases, the request to continue is ignored:

- The axis/spindle is not controlled by the PLC.
- The axis/spindle is not in the stopped state.
- An alarm is pending for the axis/spindle.

Use case 4: Reset axis/spindle (reset)

Description of the sequence:

- PLC → NC: Reset request for this axis/spindle
  DB31, ... DBX28.1 = 1 (reset)
- NC: Transitions the axis/spindle into the "stopped" state.
- NC: Interrupts the stopped sequences and signals to the PLC the interruption - essentially the same as for "Delete distance-to-go".
- NC: The internal states for the axis/spindle are reset.
- NC: The axis-specific machine data, which become effective at reset, become active.

  Note

  In conjunction with a channel reset, no axis-specific machine data is active for axes controlled from the PLC.

- NC confirms the execution:
  - DB31, ... DBX63.0 = 1 (reset executed)
  - DB31, ... DBX63.2 = 0 (axis stop active)
  - System variable $AA_SNGLAX_STAT = 1
- NC exits this operation.

6.3.3 Autonomous single-axis functions with NC-controlled ESR

Extended stop numerically controlled

The numerically controlled extended stop and retract function is also available for single axes and is configurable with axial machine data:

Delay time for ESR single axis with

MD37510 $MA_AX_ESR_DELAY_TIME1

ESR time for interpolatory braking of the single axis with

MD37511 $MA_AX_ESR_DELAY_TIME2

The values of these axial machine data are however effective only if the axis/spindle is a single axis.

The NC-controlled extended stop and retract is activated by the axial trigger $AA_ESR_TRIGGER[axis]. It works analogously to $AC_ESR_TRIGGER and applies exclusively to single axes.

Further information

M3: Coupled axes (Page 419)

Extended retract numerically controlled

For retracting single axes, the value must have been programmed via POLFA(axis, type, value) and the following conditions must be met:

- The axis must be a single axis at the time of triggering
- $AA_ESR_ENABLE[axis]=1
- POLFA(axis, type, value) with type=1 or type=2 only
  POLFA(axis, value, axis, type, axis type).

Note

NC-controlled extended stop for single axes:

The trigger is only effective if the axis is a single axis at the time of triggering, otherwise the trigger is ignored and the axial stop for this axis is **not** executed.

NC-controlled extended retract for single axes:

The channel-specific NC-controlled extended retract function is **not** effective for single axes. All axes that are single axes at the time of triggering $AC_ESR_TRIGGER will be ignored for channel-specific retraction.

This also applies when all the parameters for retraction are set, such as:

MD37500 $MA_ESR_REACTION

$AA_ESR_ENABLE for the axis, etc.

Examples

Extended **stopping** of a single axis:

MD37500 $MA_ESR_REACTION[AX1]=22

MD37510 $MA_AX_ESR_DELAY_TIME1[AX1]=0.3

MD37511 $MA_AX_ESR_DELAY_TIME2[AX1]=0.06

...

$AA_ESR_ENABLE[AX1] = 1

$AA_ESR_TRIGGER[AX1]=1 ; axis begins to stop the process here

Extended **retraction** of a single axis:

MD37500 $MA_ESR_REACTION[AX1]=21

...

$AA_ESR_ENABLE[AX1] = 1

POLFA(AX1, 1, 20.0); AX1 is assigned the axial retraction position 20.0 ; (absolute)

$AA_ESR_TRIGGER[AX1]=1 ; AX1 begins to retract here

POLFA(axis, type): permissible programming abbreviation

POLFA(axis, 0/1/2) ; quick deactivation/activation

⚠ WARNING

No preprocessing limitation

If abbreviated notation is used and only the type is changed, make sure that the value for the retraction position or retraction path in the application is meaningful!

The abbreviated notation should only be used in exceptional circumstances.

This particularly applies after:

A power on, the retraction path or the retraction position must be reset.

POLFA(axis, 1, $AA_POLFA[axis]) ; causes a preprocessing stop

POLFA(axis, 1); does **not** cause a preprocessing stop

6.4 Positioning axis dynamic response

Velocity

The positioning axes traverse at the axis-specific feedrate programmed for them. As described under "Motion behavior and interpolation functions (Page 192)", this feedrate is not influenced by the path axes.

The feedrate is programmed as an axis-specific velocity in units of min/mm, inch/min or degrees/min.

The axis-specific feedrate is always permanently assigned to a positioning axis by the axis name.

If a positioning axis has no programmed feedrate, the control system automatically applies the rate set in axis-specific machine data:

MD32060 $MA_POS_AX_VELO (initial setting for positioning axis velocity)

The programmed axis-specific feedrate is active until the end of the program.

Feedrate override

The path and positioning axes have separate feedrate overrides. Each positioning axis can be influenced by its own axis-specific feedrate override.

Rapid traverse override

Rapid traverse override applies only to path axes. Positioning axes have no rapid traverse interpolation (only axial linear interpolation G01) and therefore no rapid traverse override.

Revolutional feedrate

In JOG mode the behavior of the axis/spindle also depends on the setting of SD41100 JOG_REV_IS_ACTIVE (revolutional feedrate when JOG active).

- If this setting data is active, an axis/spindle is always moved with revolutional feedrate MD32050 $MA_JOG_REV_VELO (revolutional feedrate with JOG) or MD32040 $MA_JOG_REV_VELO_RAPID (revolutional feedrate with JOG with rapid traverse overlay) as a function of the master spindle.
- If the setting data is not active, the behavior of the axis/spindle depends on SD43300 $SA_ASSIG_FEED_PER_REV_SOURCE (revolutional feedrate for positioning axes/ spindles)
- If the setting data is not active, the behavior of a geometry axis on which a frame with rotation is effective depends on the channel-specific setting data SD42600 $SC_JOG_FEED_PER_REV_SOURCE. (In the operating mode JOG, revolutional feedrate for geometry axes on which a frame with rotation is effective.)

Maximum axial acceleration

With positioning axis motions, one of the two following maximum values is effective depending on the set positioning axis dynamic response mode:

- MD32300 $MA_MAX_AX_ACCEL [0] (maximum axial acceleration for path motions in the dynamic response mode DYNNORM)
- MD32300 $MA_MAX_AX_ACCEL [1] (maximum axial acceleration for path motions in the dynamic response mode DYNPOS)

The positioning axis dynamic response mode is set in the NC-specific machine data:

MD18960 $MN_POS_DYN_MODE = <mode>

<mode>	Meaning
0	Effective maximum axial acceleration: MD32300 $MA_MAX_AX_ACCEL[0]
1	Effective maximum axial acceleration: MD32300 $MA_MAX_AX_ACCEL[1]

Maximum axial jerk

When traversing positioning axes with active jerk limitation, the value from one of the following machine data takes effect as maximum axial jerk:

- MD32430 $MA_JOG_AND_POS_MAX_JERK (maximum axial jerk for positioning axis motions)
- MD32431 $MA_MAX_AX_JERK [0] (maximum axial jerk for path motions in the dynamic response mode DYNNORM)
- MD32431 $MA_MAX_AX_JERK [1] (maximum axial jerk for path motions in the dynamic response mode DYNPOS)

The machine data to be used is determined by the set positioning axis dynamic response mode:

MD18960 $MN_POS_DYN_MODE = <mode>

<mode>	Meaning
	The following is effective as maximum axial jerk for positioning axis motions:
0	MD32430 $MA_JOG_AND_POS_MAX_JERK With active G75 (fixed-point approach): MD32431 $MA_MAX_AX_JERK[0]
1	MD32431 $MA_MAX_AX_JERK[1]

6.5 Programming

6.5.1 General

Note

For programming positioning axes, please observe the following documentation:
Further information
See V1: Feedrates (Page 125) and S1: Spindles (Page 733)

Note

The maximum number of positioning axes that can be programmed in a block is limited to the maximum number of available channel axes.

Definition

Positioning axes are defined using the following parameters:

- Axis type: Positioning axis type 1, type 2 or type 3
- End point coordinates (in absolute dimensions or in incremental dimensions)
- Feedrate for linear axes in [mm/min], for rotary axes in [degrees/min]

Example: Positioning axis type 1

Program code	Comment
`POS[Q1]=200 FA[Q1]=1000`	`;Axis Q1 with feedrate 1000 mm/min at Position 200.`

Example: Positioning axis type 2

Program code	Comment
`POSA[Q2]=300 FA[Q2]=1500`	`;Axis Q2 with feedrate 1500 mm/min at Position 300.`

Note

Within a part program, an axis can be a path axis or a positioning axis. Within a movement block, however, each axis must be assigned a unique axis type.

Programming in synchronized action

Axes can be positioned completely asynchronous to the part program from synchronized actions.

Example:

Program code	Comment
ID=1 WHENEVER $R==1 DO POS[Q4]=10 FA[Q3]=990	;The axial feedrate is specified permanently.

Further information
Function Manual Synchronized Actions

Block change

The block change can be adjusted for positioning axis types 1 and 2 with:

```
FINEA=<axis name> or
FINEA[<axis name>]

COARSEA=<axis name> or
COARSEA[<axis name>]

IPOENDA=<axis name> or
IPOENDA[<axis name>]
```

In Type 3 positioning axis, the block change within the brake ramp of the single interpolation can be set with:

```
IPOBRKA=<axis name> or
IPOBRKA(<axis name>[,<instant in time*>])
```

* Instant in time of the block change, referred to the braking ramp as a %

Absolute dimension / incremental dimension

The programming of the end point coordinates takes place in absolute dimension (`G90`) or in incremental dimension (`G91`).

Example	Meaning
	Programming the end point coordinates:
`G90 POS[Q1]=200`	In absolute dimension
`G91 POS[Q1]=AC(200)`	In absolute dimension
`G91 POS[Q1]=200`	In incremental dimension
`G90 POS[Q1]=IC(200)`	In incremental dimension

Reprogram type 2 positioning axes

With type 2 positioning axes (motion across block limits), you need to be able to detect in the part program whether the positioning axis has reached its end position. Only then is it possible to reprogram this positioning axis (otherwise an alarm is issued).

If POSA and then POSA again with IPOBRKA (block change in the braking ramp) is programmed, an alarm is not issued. For more information, please refer to NC command IPOBKA in Section "Settable block change time".

Coordination (WAITP)

The coordination command WAITP enables you to designate a position in the NC program where the program is to wait until an axis programmed with POSA in a previous NC block has reached its end position.

WAITP exists in an internal block.

An explicit reference must be made to any axis for which the program is to wait.

Example:

Program code	Comment
N10 G01 G90 X200 F1000 POSA[Q1]=200 FA[Q1]=500	
N15 X400	
N20 WAITP(Q1)	;The program processing is stopped automatically until Q1 is at position.
N25 X600 POS[Q1]=300	;Q1 is a positioning axis type 1 (feedrate FA[Q1] from block N10).
N30 X800 Q1=500	;Q1 is the path axis (path feedrate F1000 from set N10).

Tool offset

A tool length compensation for positioning axes can be implemented by means of an axial zero offset, allowing, for example, the positioning path of a loader to be altered. An example where the axial zero offset might be used in place of the tool length compensation is where a loader containing tools of various dimensions has to bypass an obstacle.

End of program

The program end (program status selected) is delayed until all axes (path axes + positioning axes) have reached their programmed end points.

6.5.2 Revolutional feed rate in external programming

The two following setting data can be used to specify that the revolutional feed rate of a positioning axis should be derived from another rotary axis/spindle:

SD43300 $SA_ASSIGN_FEED_PER_REV_SOURCE(revolutional feed rate for position axes/ spindles)

SD42600 JOG_FEED_PER_REV_SOURCE (control of revolutional feed rate in JOG)

The following settings are possible:

Value	Description
0	No revolutional feed rate selected
>0	The revolutional feed rate is derived from the round axis/spindle with the machine axis index specified here.
-1	The revolutional feed rate is derived from the master spindle of the channel in which the axis/spindle is active.
-2	The revolutional feed rate is derived from the rotary axis/spindle with the machine axis index 0.
-3	The revolutional feed rate is derived from the master spindle of the channel in which the axis/spindle is active. No revolutional feed rate is selected if the master spindle is at a standstill.

6.6 Block change

Since path and positioning axes are interpolated separately, they reach their programmed end positions at different instants in time. If path and positioning axes are programmed in a block together, then the block change behavior depends on the programmable type of positioning axes.

Type 1: Block-related positioning axis

Properties:

- The block change is performed as soon as **all path and positioning axes** have reached their respective programmed end-of-motion criterion:
  - Path axes: `G601`, `G602`, `G603`
  - Positioning axes: `FINEA`, `COARSA`, `IPOENDA`
- Programming the positioning axis: `POS[<axis>]`

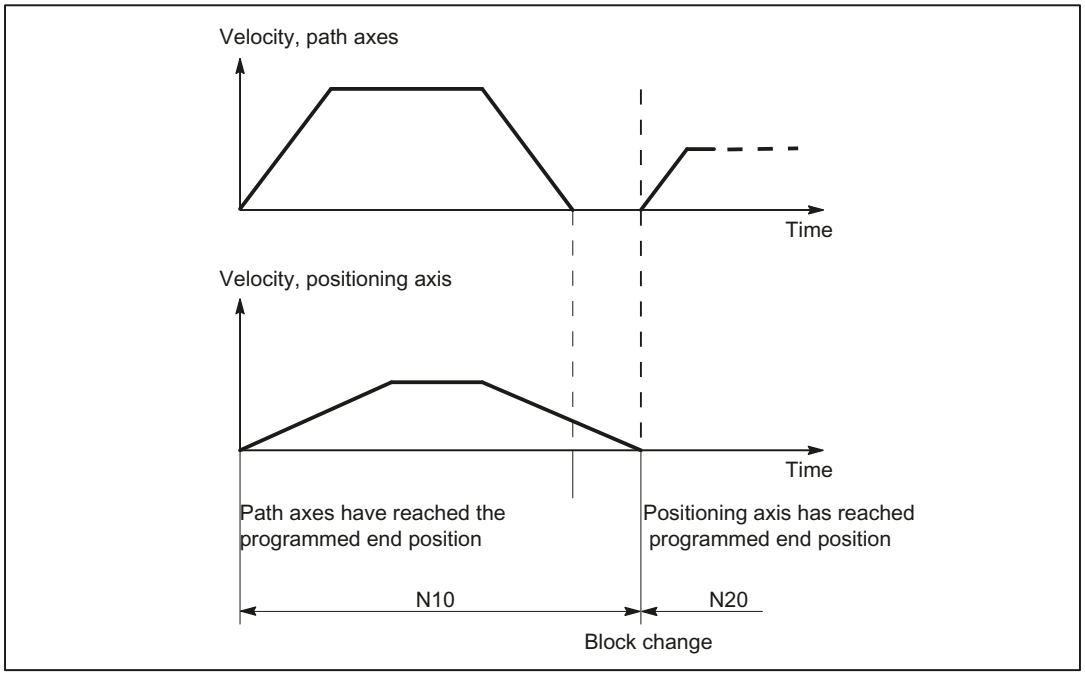


Figure 6-1 Block change for path axis and positioning axis type 1

Note

Continuous path mode

Continuous path mode across block limits (G64) is only possible if the positioning axes reach their end-of-motion criterion before the path axes (in the diagram above, this is not the case).

Type 2: Modal positioning axis (across blocks)

Properties:

- The block change is performed as soon as **all path axes** have reached their programmed end-of-motion criterion (G601, G602, G603)
- Programming the positioning axis: POSA[<axis>]
- The positioning axis traverses beyond the block limits to its programmed end position. It is not permissible that the positioning axis is programmed again before reaching its end-of-motion criterion.

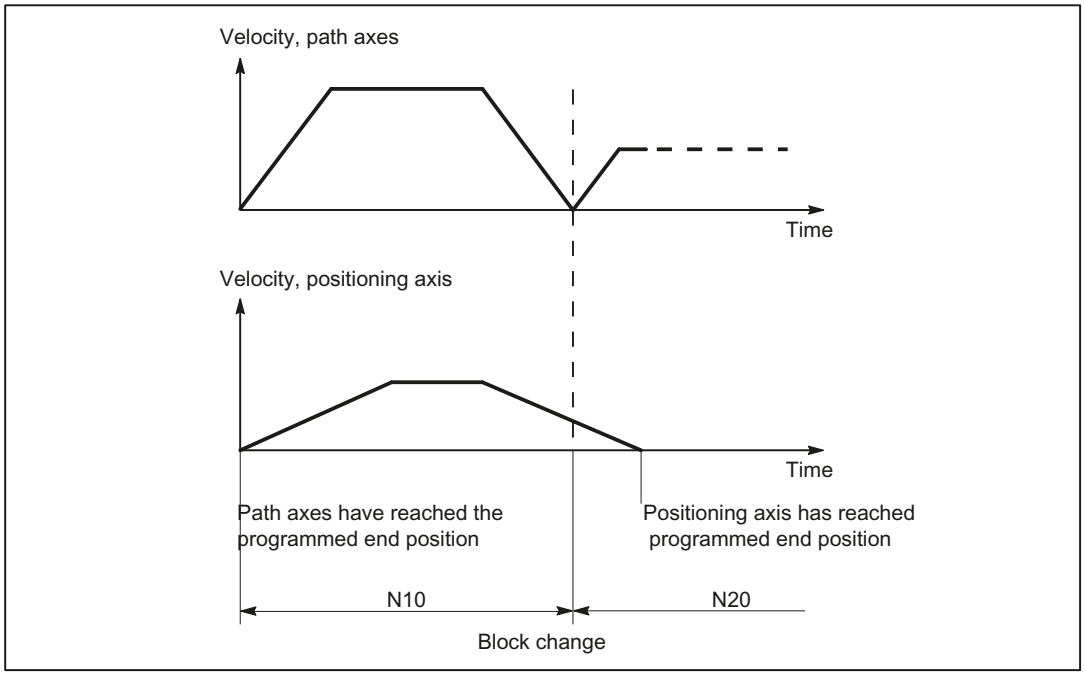


Figure 6-2 Block change for path axis and positioning axis type 2

6.6.1 Settable block change time

Type 3: Conditional block-related positioning axis

Properties:

- The block change is performed as soon as **all path and positioning axes** have reached their respective programmed end-of-motion criterion:
  - Path axes: `G601`, `G602`, `G603`
  - Positioning axes: `IPOBRKA`
- Programming the positioning axis:
  - `N(x) IPOBRK(<axis>[,<instant in time>]) ;own block`
  - `N(x+1) POS[<axis>]`

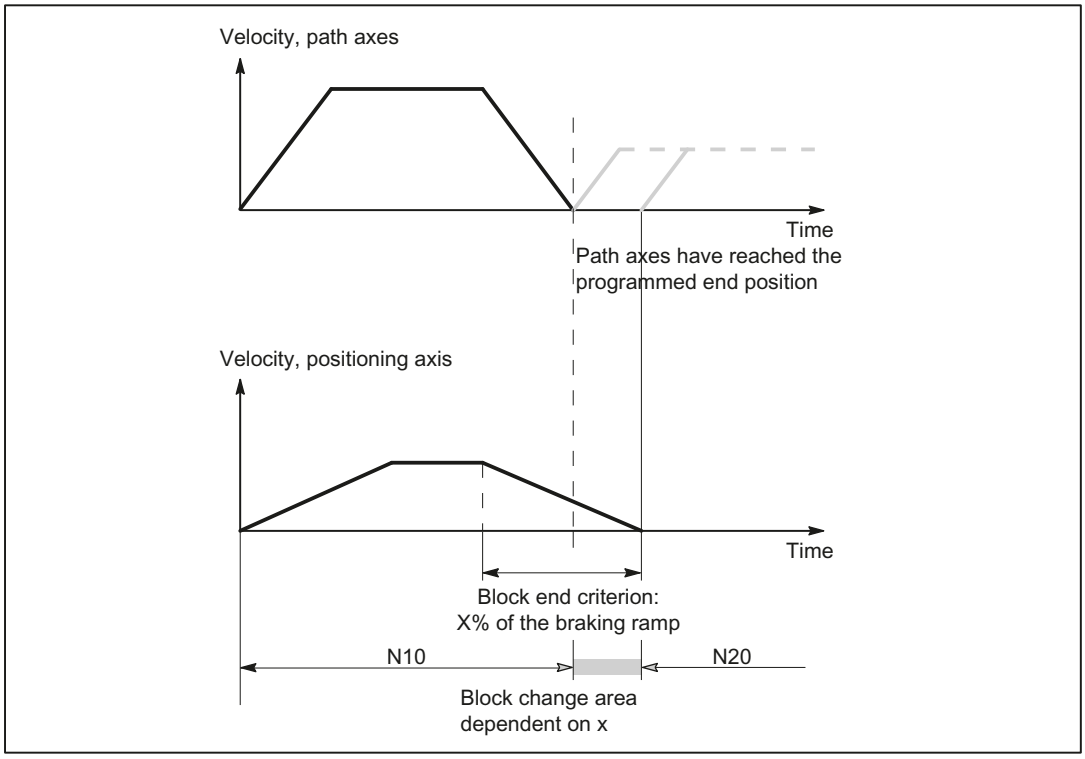


Figure 6-3 Block change for path axis and positioning axis type 3

Block change criterion: "Braking ramp" (IPOBRKA)

If, when activating the block change criterion "brake ramp", a value is programmed for the optional parameter <instant in time>, then this becomes effective for the next positioning motion and is written into the setting data synchronized to the main run. If no value is specified for the block change instant in time, then the actual value of the setting data is effective.

SD43600 $SA_IPOBRAKE_BLOCK_EXCHANGE

The time at which the block change can be realized is specified as a percentage of the braking ramp:

- 100% = start of the braking ramp
- 0% = end of the braking ramp, the same significance as block change criterion IPOENDA

Programming

```
IPOBRKA(<axis>[,<instant in time>])
```

IPOBRKA:	Block change criterion: Deceleration ramp Effectiveness: modal
<Axis>:	Channel axis name (X, Y, ...)

<instant in time>: Time of the block change, referred to the braking ramp as a %:
- 100% = start of the braking ramp
- 0% = end of the braking ramp, the same significance as IPOENDA

Type: REAL

IPOBRKA is deactivated for the corresponding access when an axis end-of-motion criterion (FINEA, COARSEA , IPOENDA) is next programmed for the axis.

Additional block change criterion: "Tolerance window" (ADISPOSA)

A tolerance window around the end of block (either as actual or setpoint position) can be defined as additional block change criterion. Then, two conditions must be fulfilled for the block change:

- Block change criterion: "Braking ramp"
- Block change criterion: "Tolerance window"

Programming

```
ADISPOSA(<axis>[,<mode>,<window size>])
```

ADISPOSA: Tolerance window for end-of-motion criterion

Effectiveness: modal

<Axis>: Channel axis name (X, Y, ...)

<mode>: Reference of the tolerance window

Type: INT

Value range:
- 0 Tolerance window not active
- 1 Tolerance window with respect to the set-point position
- 2 Tolerance window with respect to actual position

<window size>: Size of the tolerance window

Type: REAL

System variable for end-of-motion criterion

The effective end-of-motion criterion can be read using the system variable $AA_MOTEND.

Further information: Parameter Manual, Book 2

Note

Information about other programmable end-of-motion criteria `FINEA`, `COARESA`, `IPOENDA` can be found in:

Further information: Function Manual, Basic Functions

- Operating modes (Page 734)
- Influencing the single axis dynamic response (Page 176)

Boundary conditions

Premature block change

A premature block change is not possible in the following cases:

- Oscillating axis
  During oscillation with partial infeed, the block-specific oscillation motion must remain active until the axis with partial infeed has reached its final position.
- Handwheel
  For handwheel input, the last set end-of-motion criterion applies.

Changing the axis state

The axis for which a block change occurred within the braking ramp can only be programmed in the following block in the same axis state.

When the axis state changes, e.g. to `POS` followed by `SPOS`, the last programmed end-of-motion criterion `FINEA`, `COARSEA`, `IPOENDA` is active. This also applies in the following cases:

- a positioning axis becomes a path axis
- if the program waits for the end of the positioning movement: `WAITP`, `M30`, end of the technology cycle, preprocessing stop
- Velocity override is deactivated or activated

Note

For further information about programming positioning axes, see:

Further information

Programming Manual Fundamentals; Feedrate control and spindle motion

Programming Manual Advanced, Special motion commands

Examples

Block change criteria "braking ramp" in the part program

Program code	Comment
	;Default setting effective.

Program code	Comment
N10 POS[X]=100	;Positioning motion from X to position 100.
	;Block change criterion: "Exact stop fine"
N20 IPOBRKA(X,100)	;Block change criterion: "Braking ramp", 100% = start of the braking ramp.
N30 POS[X]=200	;The block is changed as soon as axis X starts to brake.
N40 POS[X]=250	;Axis X no longer brakes at position 200, but rather continues to traverse to position 250. As soon as the axis starts to brake, the block changes.
N50 POS[X]=0	;Axis X brakes and returns to position 0, the block change is realized at position 0 and "exact stop fine".
N60 X10 F100	;Axis X traverses as path axis to position 10.
N70 M30	

Block change criterion "braking ramp" in synchronized action

In the technology cycle:

Program code	Comment
FINEA	;End-of-motion criterion: "Exact stop fine"
N10 POS[X]=100	;The technology cycle block change is realized if the X axis has reached position 100 and "exact stop fine" is reached.
N20 IPOBRKA(X,100)	;Activate block change criterion "braking ramp", 100% = start of the braking ramp.
N30 POS[X]=200	;Technology cycle block is changed as soon as axis X starts to brake.
N40 POS[X]=250	;X axis does not brake at position 200, but rather continues to traverse to position 250. As soon as the X axis starts to brake, the block change takes place in the technology cycle.
N50 POS[X]=0	;X axis brakes and returns to position 0, the block change is realized at position 0 and "exact stop fine".
N60 M17	

Block change criterion "braking ramp" and "tolerance window" in the part program

Program code	Comment
	;Default setting effective.
N10 POS[X]=100	;Positioning motion from X to position 100.
	;Block change criterion: "Exact stop fine"
N20 IPOBRKA(X,100)	;Block change criterion: "Braking ramp", 100% = start of the braking ramp.
N21 ADISPOSA(X,1,0.5)	;Tolerance window: 1 = setpoint position, tolerance = 0.5
N30 POS[X]=200	;The block is changed as soon as axis X starts to brake.
N40 POS[X]=250	;Axis X no longer brakes at position 200, but rather continues to traverse to position 250. As soon as the axis starts to brake, the block changes.

Program code	Comment
N50 POS[X]=0	;Axis X brakes and returns to position 0, the block change is realized at position 0 and "exact stop fine".
N60 X10 F100	
N70 M30	

Block change criterion "braking ramp" and "tolerance window" in synchronized action

In the technology cycle:

Program code	Comment
FINEA	;End-of-motion criterion: "Exact stop fine"
N10 POS[X]=100	;The technology cycle block change is realized if the X axis has reached position 100 and "exact stop fine" is reached.
N20 IPOBRKA(X,100)	;Activate block change criterion "braking ramp", 100% = start of the braking ramp.
N21 ADISPOSA(X,2,0.3)	;Tolerance window: 2 = actual position, tolerance = 0.3
N30 POS[X]=200	;Technology cycle block change is realized as soon as the X axis starts to brake and the actual position of the X axis >= 199.7.
N40 POS[X]=250	;X axis does not brake at position 200, but rather continues to traverse to position 250. As soon as the X axis starts to brake and the position of the X axis >= 249.7, the block change takes place in the technology cycle.
N50 POS[X]=0	;X axis brakes and returns to position 0, the block change is realized at position 0 and "exact stop fine".
N60 M17	

`IPOBRKA(X)` could also be written into the `N20` blocks without specifying the instant in time, if the corresponding value has already been entered into the setting data:

SD43600 $SA_IPOBRAKE_BLOCK_EXCHANGE[X] == 100

See also

Control by the PLC (Page 212)

6.6.2 End of motion criterion with block search

Last block serves as container

The last end-of-motion criterion programmed for an axis is collected and output in an action block. The last block with a programmed motion end condition that was processed in the search run serves as a container for setting all axes.

Example

For two action blocks with end-of-motion criteria for three axes:

Program code	Comment
N01 G01 POS[X]=20 POS[Y]=30	
IPOENDA[X]	;Last programmed IPOENDA end-of-motion criterion
N02 IPOBRKA(Y, 50)	;Second action block for Y axis IPOENDA
N03 POS[Z]=55 FINEA[Z]	;Second action block for the Z axis FINEA
N04 $A_OUT[1]=1	;First action block for output as digital output
N05 POS[X]=100	
N06 IPOBRKA(X, 100)	;Second action block for X axis IPOBRKA container
...	;For all programmed end-of-motion criteria
TARGET:	;Block search target

The first action block contains the digital output:

$A_OUT[1] = 1.

The second action block contains the settings for the end-of-motion criteria:

for the X axis IPOBRKA / $SA_IPOBRAKE_BLOCK_EXCHANGE[AX1]=100

for the Y axis IPOBRKA / $SA_IPOBRAKE_BLOCK_EXCHANGE[AX2]=50

for the Z axis FINEA. The motion end condition IPOENDA last programmed is also stored for the X axis.

6.7 Control by the PLC

PLC axes

PLC axes are traversed from the PLC and can move asynchronously to all other axes. The travel motions are executed separate from the path and synchronized actions.

Further information
Function Manual PLC

Concurrent positioning axes

Concurrent positioning axes can be started from the PLC using the FC18 function block.

Channel-specific signals

All channel-specific signals act to the same extent on path and positioning axes.

Only the following signals are an exception:

- NST DB21, ... DBB4 ("feedrate override")
- IS DB21, ... DBX6.2 ("Delete distance to go")

Axisspecific signals

Positioning axes have the following additional signals:

- IS DB31, ... DBX76.5 ("Positioning axis")
- Feedrate for positioning axes/spindles (FA)
- NST DB31, ... DBB0 ("feedrate override") - axis-specific
- IS DB31, ... DBX2.2 ("Distance to go/Spindle reset") Axis-specific deletion of distance to go

Single-axis functions of PLC-controlled axes

The behavior of individual PLC axes can be manipulated in the following way with machine data:
MD30460 $MA_BASE_FUNCTION_MASK:

- Bit 4 = 1
  The axis is exclusively controlled by the PLC.
- Bit 5 = 1
  The axis is a permanently assigned PLC axis.
- Bit 6 = 1
  The channel-specific NC/PLC interface signal:
  DB21, ... DBX6.0 ("feed disable")
  is not active for the axis if this is a PLC-controlled axis.
- Bit 7= 1
  The channel-specific NC/PLC interface signal:
  DB21, ... DBX36.3 ("all axes stationary")
  is set**independently** of an axis if this is a PLC-controlled axis.

For a PLC-controlled axis:

- The channel-specific NC/PLC interface signal DB21, ... DBX6.0 ("feed disable"), is active if bit 6 = 0 in machine data MD30460 $MA_BASE_FUNCTION_MASK.
- The channel-specific NC/PLC interface signal DB21, ... DBX36.3 ("all axes stationary") is only set if bit 7 = 0 in machine data MD30460 $MA_BASE_FUNCTION_MASK.

If an attempt is made to assign an **exclusively PLC-controlled axis** to the NC program or to request the axis for the NC program, then this is rejected with Alarm 26075. Alarm 26076 is generated in the same way for a PLC axis with fixed assignment.

A **PLC axis with fixed assignment** is a "neutral axis" on power up. For a travel request via the NC/PLC interface, a concurrent positioning axis automatically changes to a PLC axis without being interchanged beforehand.

Axis interchange via PLC

The type of an axis for axis interchange is transferred to the PLC with axial interface byte NC→PLC NST DB31, ... DBB68 (see also Function Manual "Basic Functions" Section "K5: Cross-channel program coordination and channel-by-channel running-in"):

- IS DB31, ... DBX68.0-68.3 ("NC axis/spindle in channel") channels 1 to 10
- IS DB31, ... DBX68.4 ("new type requested by PLC")
- IS DB31, ... DBX68.5 ("axis can be replaced")
- IS DB31, ... DBX68.6 ("neutral axis/spindle")
- IS DB31, ... DBX68.7 ("PLC axis/spindle")

If IS DB31, ... DBX68.5 ("axis can be replaced") = 1, an axis replacement request can be issued from the PLC.

6.7.1 Starting concurrent positioning axes from the PLC

Traversing an axis in the NC as competing positioning axes from the PLC user program is realized using block FC18:

The following functions are defined:

- Linear interpolation (G01)
- Feedrate in mm/min or degrees/min (G94)
- Exact stop (G09)
- The active adjustable work offsets are taken into account

Since each axis is assigned to exactly one channel, the control can select the correct channel from the axis name/axis number and start the concurrent positioning axis on this channel.

Further information

Function Manual PLC; Block descriptions > FC18: SpinCtrl, spindle control

6.7.2 PLC-controlled axes

PLC actions

The table below compares the following PLC actions as response of the NC for a machine axis:

- Start machine axis as PLC axis using FC18
- Initiate `NC start` or `NC stop`
- Axis-specific `STOP`, `RESUME` or `RESET`
- Trigger `NC-RESET`
- Cancel or set controller enable for the machine axis
- Relinquish control of machine axis to NC

Examples of NC responses

PLC actions as response of the NC are listed in the table below.

PLC actions	NC response
Machine axis AX1 is the channel axis in channel 1, start as PLC axis using FC18	
Initiate NC stop axes and spindles DB21, ... DBX7.4 = 1 (NC stop axes plus spindle)	AX1 is stopped.
DB21, ... DBX7.1 = 1 initiate (NC start)	AX1 continues to traverse.
The PLC wants to control AX1, DB31, ... DBX28.7 = 1 (PLC controls axis)	Checking AX1 is relinquished to the PLC. DB31, ... DBX63.1 = 1 (PLC controls the axis)
Initiate NC stop for axes and spindles DB21, ... DBX7.4 = 1 ("NC stop axes plus spindle")	AX1 is not stopped.
Initiate axial stop DB31, ... DBX28.6 = 1 (stop along braking ramp)	AX1 is stopped DB31, ... DBX63.2 ==1 (axis stop active)
Initiate axial continuation DB31, ... DBX28.2 = 1 (continue)	AX1 traverses further DB31, ... DBX63.2 == 0 (axis stop active)
Initiate NC RESET DB21, ... DBX7.7 = 1 initiate (reset)	No effect on AX1
Initiate axial reset DB31, ... DBX28.1 = 1 (reset)	AX1 is stopped and the traversing motion is interrupted: • DB31, ... DBX63.2 = 0 (axis stop active) • Read-in axial machine data • DB31, ... DBX63.0 = 1 (reset executed) • DB31, ... DBX63.2 = 0 (axis stop active)
Start machine axis AX1 as PLC axis via FC 18	DB31, ... DBX63.0 = 0 (reset executed)
Withdraw controller enable for AX1: DB31, ... DBX2.1 = 0 (controller enable)	Alarm 21612 "Axis %1 measuring system change" is displayed
Initiate axial continuation DB31, ... DBX28.2 = 1 (continue)	• Alarm 21612 "Axis %1 measuring system change" is deleted • DB21, ... DBX40.7 = 1(traversing command plus) • AX1 does not traverse due to a missing controller enable signal.
Set controller enable for AX1 DB31, ... DBX2.1 = 1 (controller enable)	AX1 moves to the programmed end point.
Initiate axial reset DB31, ... DBX28.1 = 1 (reset)	• Stop AX1 • Read-in axial machine data • DB31, ... DBX63.0 = 0 (reset)
PLC relinquishes control of AX1 to the NC from DB31, ... DBX28.7 = 0 (PLC controls axis)	• NC accepts control of machine axis • DB31, ... DBX63.1 = 0 (PLC controls the axis) • DB31, ... DBX63.0 = 0 (reset)

6.7.3 Control response of PLC-controlled axes

Response for a channel reset, activate machine data, block search and MD30460

Table 6-1

Control response to	PLC-controlled axis
Mode change and NC program control	work independently of axis.
Channel RESET	No axial machine data are effective and a traversing movement is not aborted.
NEWCONF	No axial machine data are effective.
Block search Type 5 SERUPRO	are processed during SERUPRO to simulate the normal procedure, e.g. PLC takes over or relinquishes control of this axis which is also traversing via the PLC.
All block search variants of types 1, 2, 4 and 5	The PLC takes over control of the axis **before** the approach block and is responsible for positioning this axis.
NC-controlled retraction activated with $AC_ESR_TRIGGER.	has no effect and acts only on the specified PLC-controlled axis.
machine data: MD30460 $MA_BASE_FUNKTION_MASK	**which is not controlled exclusively by the PLC**
Bit 4 = 0	**cannot** be changed directly with axis replacement command GET (axis) or AXTOCHAN(axis, channel) to an axis controlled by the NC program, see * **Note on axis replacement**.
Bit 4 = 1	cannot be requested for the NC program. GET or AXTOCHAN from the NC program or a synchronized action, or programming the axis in the NC program, are rejected with alarm 26075.
MD30460 $MA_BASE_FUNKTION_MASK	**For the PLC-controlled axis**
Bit 6 = 1	channel-specific IS DB 21, ... DBX6.0 ("feed disable") is not effective. This axis is not stopped when feed disable is activated, but continues to move.
Bit 7 = 1	the axis is not taken into account when IS DB 21, ... DBX36.3 ("all axes stationary") is generated. This signal is output with 1 even if all other axes in the channel are stationary and only the PLC-controlled axis is active.

*** Note on axis replacement**

This replacement of a "neutral axis" by an "NC program axis" does not take place until the PLC has really relinquished control of the axis in accordance with use case "Relinquish control of axis". Waiting for this axis interchange is displayed on the HMI operator panel front.

6.8 Response with special functions

6.8.1 Dry run (DRY RUN)

The dry run feedrate is also effective for positioning axes unless the programmed feedrate is larger than the dry run feedrate.

Activation of the dry run feed entered in SD42100 $SA_DRY_RUN_FEED can be controlled with SD42101 $SA_DRY_RUN_FEED_MODE. See

Further information

V1: Feedrates (Page 125)

6.8.2 Single block

Positioning axis type 1

Single-block mode is effective with positioning axes of type 1.

Positioning axis type 2

Positioning axes of type 2 also continue across block limits in single block mode.

Positioning axis type 3

Positioning axes of type 3 also continue across block limits in single block mode.

6.9 Examples

6.9.1 Motion behavior and interpolation functions

In the following example, the two positioning axes Q1 and Q2 represent two separate units of movement. There is no interpolation relationship between the two axes. In the example, the positioning axes are programmed as type 1 (e.g. in N20) and type 2 (e.g. in N40).

Program example

Program code

```
N10 G90 G01 G40 T0 D0 M3 S1000
N20 X100 F1000 POS[Q1]=200 POS[Q2]=50 FA[Q1]=500
FA[Q2]=2000
```

```
Program code
N30 POS[Q2]=80
N40 X200 POSA[Q1] = 300 POSA[Q2]=200] FA[Q1]=1500
N45 WAITP[Q2]
N50 X300 POSA[Q2]=300
N55 WAITP[Q1]
N60 POS[Q1]=350
N70 X400
N75 WAITP[Q1, Q2]
N80 G91 X100 POS[Q1]=150 POS[Q2]=80
N90 M30
```

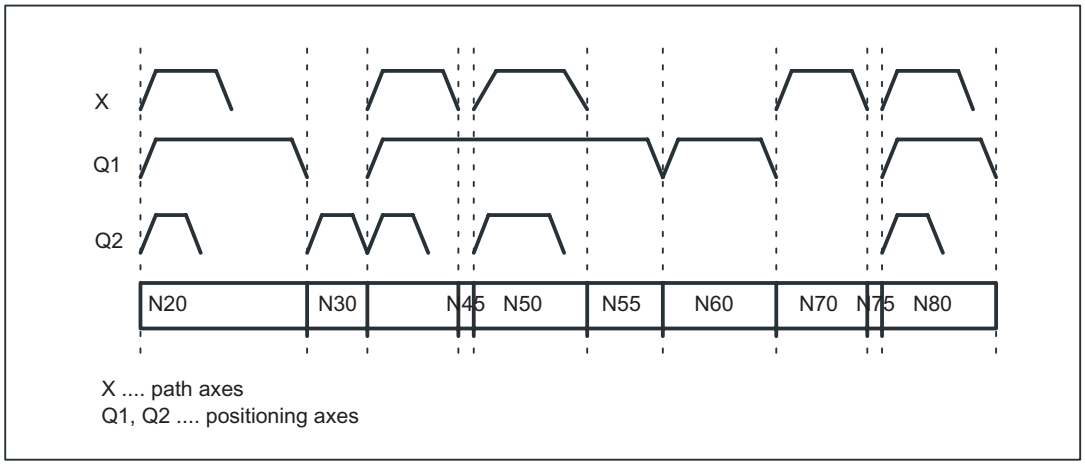


Figure 6-4 Timing of path axes and positioning axes

6.10 Data lists

6.10.1 Machine data

6.10.1.1 NC-specific machine data

Number	Identifier: $MN_	Description
18960	POS_DYN_MODE	Type of positioning axis dynamic response

6.10.1.2 Channelspecific machine data

Number	Identifier: $MC_	Description
20730	G0_LINEAR_MODE	Interpolation behavior with G0
20732	EXTERN_G0_LINEAR_MODE	Interpolation behavior with G00
22240	AUXFU_F_SYNC_TYPE	Output timing of F functions

6.10.1.3 Axis/spindlespecific machine data

Number	Identifier: $MA_	Description
30450	IS_CONCURRENT_POS_AX	Concurrent positioning axis
30460	BASE_FUNCTION_MASK	Axis functions
32060	POS_AX_VELO	Feedrate for positioning axis
32300	MAX_AX_ACCEL	Maximum axis acceleration
32430	JOG_AND_POS_MAX_JERK	Maximum axial jerk for positioning axis movements
32431	MAX_AX_JERK	Maximum axial jerk for path movements
37510	AX_ESR_DELAY_TIME1	Delay time for ESR single axis
37511	AX_ESR_DELAY_TIME2	ESR time for interpolatory braking of single axis

6.10.2 Setting data

6.10.2.1 Axis/spindle-specific setting data

Number	Identifier: $SA_	Description
43600	IPOBRAKE_BLOCK_EXCHANGE	Braking ramp block change condition
43610	ADISPOSA_VALUE	Braking ramp tolerance window

R2: Rotary axes

7

7.1 Brief description

The following are typical applications for positioning axes:

- 5-axis machining (operating range limited or unlimited)
- Rotary axis for eccentric machining (unlimited operating range)
- Rotary axis for cylindrical or form grinding (unlimited operating range)
- C axis with `TRANSMIT` (unlimited operating range)
- Rotary axis on winding machines (unlimited operating range)
- Rotary workpiece axis (C) on hobbing machines (unlimited operating range)
- Round tool magazines and tool turrets (unlimited operating range)
- Rotary axis for peripheral surface transformation (limited operating range)
- Swivel axes for gripping (operating range 360°)
- Rotary axes for swiveling (operating range < 360°; e.g., 60°)
- Milling swivel axis (A) on hobbing machines (operating range, e.g., 90°)

Axis addresses

Linear axes, rotary axes and directions of movement of numerically-controlled machine tools are designated according to DIN 66025 as follows:

- Linear axes: X, Y, Z
- Rotary axes: A, B, C

- Assignment: A rotates about X, B rotates about Y and C rotates about Z
- Direction of rotation: The positive rotary-axis direction of a rotation corresponds to a clockwise rotation when looking in the positive axis direction of the corresponding linear axis.

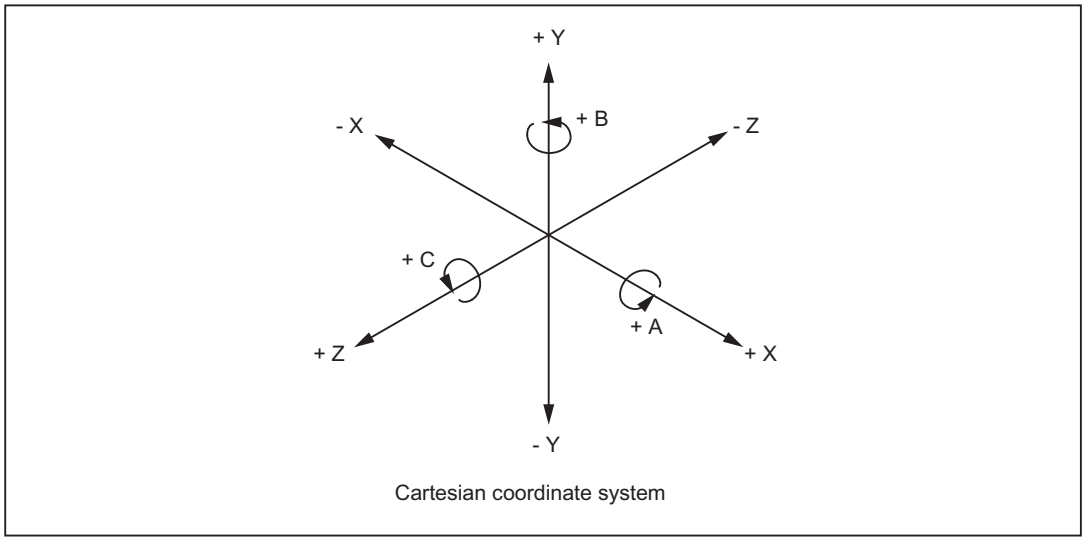


Figure 7-1 Axis identifiers and directions of movement for rotary axes

Units of measurement

The following units of measurement apply as standard to inputs and outputs for rotary axes:

Units of measurement for rotary axes	
Physical quantity	Unit
Angular position	Degrees
Programmed angular velocity	Degrees/min
Angular velocity	1) rev/min
Angular acceleration	1) rev/sec^2
Angular jerk limit	1)rev/sec^3
1) Unit of the axis-specific machine data for rotary axes.	

Further information

G2: Velocities, setpoint / actual value systems, closed-loop control (Page 31)

Feedrate

The programmed feedrate F corresponds to an angular velocity (degrees/min) in the case of rotary axes.

If rotary axes and linear axes traverse a common path with `G94` or `G95`, the feedrate should be interpreted in the linear-axis unit of measurement (e. g. mm/min, inch/min].

The tangential velocity of the rotary axis refers to diameter D_{unit} (unit diameter D_{unit}=360/π). In the case of unit diameter $D=D_{unit}$, the programmed angular velocity in degrees/min and the tangential velocity in mm/min (or inch/min) are numerically identical.

In general, the following applies for tangential velocity:

$F = F_{angle} * D/D_{unit}$

With $D_{unit} = 360/\pi$

F = Tangential velocity [mm/min]
F_{angle} = Angular velocity [degrees/min]
D = Diameter acted on by F [mm]
D_E = Unit diameter [mm]
π = Circle constant Pi

JOG mode: Revolutional feedrate

In the JOG mode, the response of the rotary axis spindle also depends on the setting data:

SD41100 $SN_JOG_REV_IS_ACTIVE (revolutional feed rate for JOG active)

7.2 Modulo 360 degrees

A 360° modulo rotary axis is a rotary axis whose position is mapped to the range from 0° to 359.999°. As a consequence, in principle, the traversing range of a 360° modulo rotary axis is infinity. The traversing range limits (software limit switch and working area limits) of 360° modulo rotary axes are not effective.

In the diagram below, the absolute position of a rotary axis in the positive direction of rotation is represented as a spiral. The pointer to the spiral indicates the absolute position of the rotary axis (420°). The point where the pointer intersects with the modulo circle marks the corresponding 360° modulo position (60°).

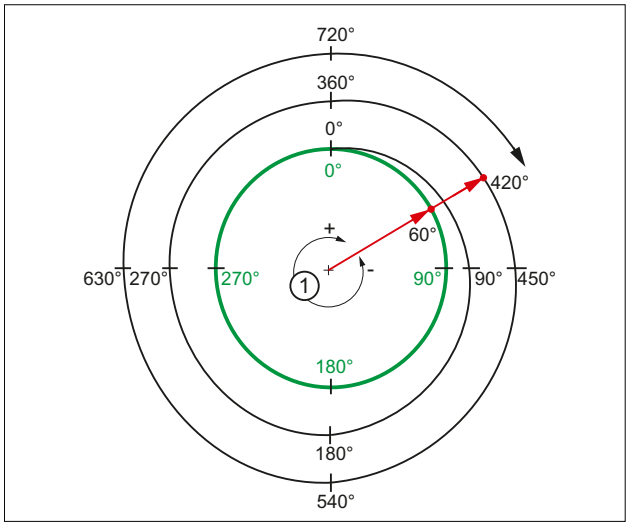


Figure 7-2 Modulo_mapping

Parameter assignment

Modulo function

The modulo function for a rotary axis is activated using the axis-specific machine data:

MD30310 $MA_ROT_IS_MODULO = 1

Note

For a 360° modulo rotary axis, it is recommended that the position display is also set to 360° at the user interface.

Modulo range

For a 360° modulo rotary axis, the modulo range **must** be set to 360° using the axis-specific machine data:

MD30330 $MA_MODULO_RANGE = 360.0

Starting position of the 360° modulo range

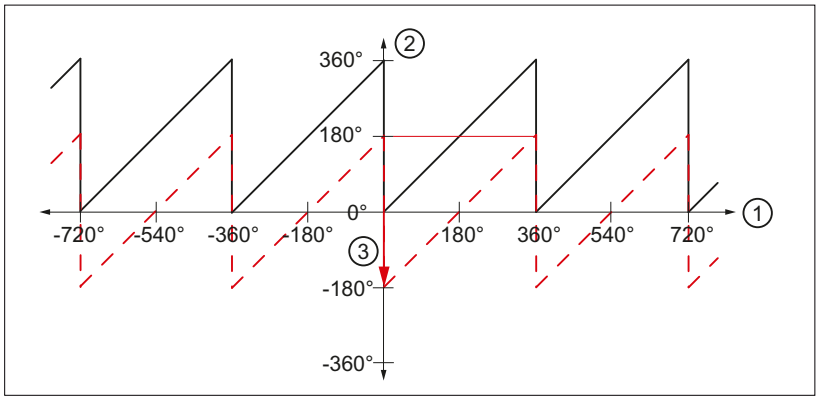
The starting position of the 360° modulo range can be set using the axis-specific machine data:

MD30340 $MA_MODULO_RANGE_START = <Starting position>

As a consequence, the parameterized starting position represents an offset of the modulo function in the vertical axis. The modulo range still remains 360°.

Examples

- Standard starting position: 0° ⇒ modulo range: 0° - 360°
- Shifted starting position: -180° ⇒ modulo range: -180° to +180°



① Absolute position
② Modulo position
③ Shifted starting position (-180°)

Figure 7-3 Standard and shifted starting position (red)

Note

Modulo partition axes

By aligning the two following machine data, indexing positions of 360° modulo indexing axes can be implemented in the same way as for the 360° modulo range:

- MD30503 $MA_INDEX_AX_OFFSET
- MD30340 $MA_MODULO_RANGE_START

See the examples in Chapter "Indexing axes, "Example 3: Rotary modulo axis as equidistant indexing axis (Page 254)" and the following chapters.

360° modulo rotary axis: Positioning behavior for G90 (modal absolute programming)

The standard behavior for G90 is set via the machine data:

MD30455 $MA_MISC_FUNCTION_MASK, bit 2 = <value>

<value>	Meaning
0	The 360° modulo rotary axis positions corresponding to AC as standard for G90 programming: • Target position > Current position: Move in positive traversing direction • Target position < Current position: Move in negative traversing direction
1	For G90 programming, the 360° modulo rotary axis positions as standard corresponding to DC, i.e. the shortest path.

Position display in the 360° modulo representation

Activating the position display at the user interface in the 360° modulo representation is realized using the axis-specific machine data:

MD30320 $MA_DISPLAY_IS_MODULO = 1

NC/PLC interface signals

Activating the traversing range limits

In principle, the traversing range of a 360° modulo rotary axis is infinity. This is the reason that axis-specific traversing range limits to not apply as standard, such as software limit switches and working error limits. The traversing range limits of a 360° modulo rotary axis can be activated using the subsequent absolute axis-specific NC/PLC interface signal, possibly initiated via an M/H function in the NC program:

DB31, ... DBX12.4 = 1 (360° modulo rotary axes: Activate traversing range limits)

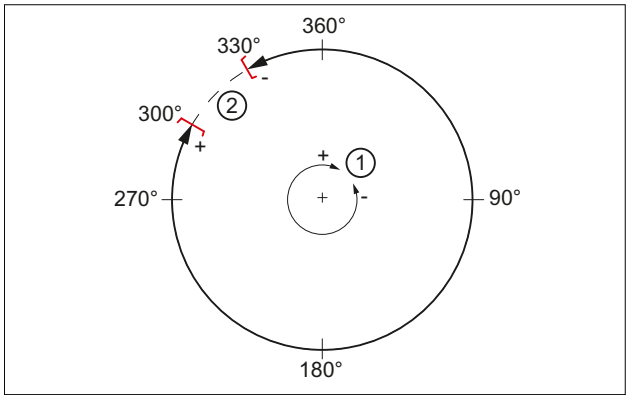
Preconditions:

- The 360° modulo rotary axis must be referenced.
- Software limit switch
  Software limit switch 1 or 2 to be monitored must be active:
  - DB31, ...DBX12.2 / 3 (activation)
  - MD36100 $MA_POS_LIMIT_MINUS / MD36110 $MA_POS_LIMIT_PLUS (software limit switch 1)
  - MD36120 $MA_POS_LIMIT_MINUS2 / MD36130 $MA_POS_LIMIT_PLUS2 (software limit switch 2)
- Working area limitation
  Both limits must be active, either via G26/G25 or setting data:
  - SD43400 $SA_ WORKAREA_PLUS_ENABLE / SD43410 $SA_WORKAREA_MINUS_ENABLE (activation)
  - SD43420 $SA_WORKAREA_LIMIT_PLUS / SD43430 $SA_WORKAREA_LIMIT_MINUS (limiting)
- Synchronization with STOPRE
  If the traversing range limits are activated using the NC/PLC interface signal in an NC program being executed, e.g. as a result of an M/H command at the PLC, then a preprocessing strop must be programmed in the following block (STOPRE). Only then can it be ensured that the activation of the traversing range limits are taken into account in the subsequent traversing blocks.

NOTICE

User: Activation and effectiveness

If the traversing range limits of a 360° modulo rotary axis are activated by the NC/PLC interface signal (DB31, ... DBX12.4) in a block in the main run, (e.g. using the M/H function), under certain circumstances, as a result of the preprocessing of already subsequent traversing blocks, the traversing range limits were not taken into account. As a consequence, it is solely the responsibility of the user to apply suitable measures, e.g. by programming STOPRE in the following block after activating the traversing range limits. This means that from the activation time in the NC program, the traversing range limits are also effective in all of the subsequent blocks.



① Direction of rotation +/-

② Software limit switch +/-

Figure 7-4 360° modulo rotary axis with software limit switch

Feedback signal, activation of the traversing range limits

The feedback signal that the traversing range limits of a 360° modulo rotary axis are active in the NC, is realized using the axis-specific NC/PLC interface signal:

DB31, ... DBX74.4 == 1 (modulo rotary axes: traversing range limits active)

Example: Changing the traversing range limits

Pallet 1 with several clamped workpieces is machined on a 360° modulo rotary axis. The pallet 1 is then swapped for a pallet 2 with a built-on axis. The traverse range now has to be monitored.

Parameterization:

- MD30300 $MA_IS_ROT_AX[AX5] = 1
- MD30310 $MA_ROT_IS_MODULO[AX5] = 1
- MD36110 $MA_POS_LIMIT_PLUS[AX5] = 340
- MD36100 $MA_POS_LIMIT_MINUS[AX5] = 350

Extract from part program:

```
Program code   Comment
M123           ; Insert pallet 1 with quadruple clamping
               ; PLC: deactivate traversing range limits of the
               ; B axis => DB35, DBX12.4=0
STOPRE         ; Trigger a preprocessing stop
S1000 M3
G4 F2
G1 X0 Y300 Z500 B0 F5000
CYCLE84(500,400,0,350,0,1,4,10,,0,500,1000) ; drilling cycle
...
G0 Z540 B0
```

```
Program code   Comment
M124           ; Insert pallet 2 with built-on axis
               ; PLC: activate traversing range limits of the
               ; B axis => DB35, DBX12.4=1
STOPRE         ; Trigger a preprocessing stop
B270
```

7.3 Programming of rotary axes

7.3.1 General information

Note

General information about programming, see:
Further information:
Programming Manual Fundamentals

MD30310

Axis-specific machine data
MD30310 ROT_IS_MODULO (modulo conversion for rotary axis)
is used to define whether the rotary axis behaves as a linear axis during programming and positioning or whether rotary-axis special features are taken into account.

These features and any differences (mainly with respect to absolute programming) are explained on the following pages.

Modulo conversion for rotary axes and spindles

MD30310 "Modulo conversion for rotary axes and spindles“

MD30320 "Modulo display for rotary axes and spindles“

If MD30310 and MD30320 are set to "1", the change in Operate to a modulo display takes place as soon as the actual value display changes between 0° and 359.999° - or exceeds 359.999° in the positive direction of rotation, or falls below 0° in the negative direction of rotation.

For internal setpoint and actual values, the behavior of rotary axes and spindles differ:

- For spindles, the modulo conversion takes place for each spindle programming (M3, M4 and M5).
- For rotary axes, the modulo conversion only takes place with a new traversing command.

If the setpoint and actual values are read out, these are the same as the display values in the "Service overview" - "Service axis" under the selected axis. For rotating spindles or rotary axes, these values can also lie below the modulo range of 0° to 359,999°

7.3.2 Rotary axis with modulo conversion (continuously-turning rotary axis).

Absolute programming (AC, ACP, ACN, G90)

Example of the absolute programming for a modulo rotary axis as positioning axis:

```
POS[<axis name>] = ACP(<value>)
```

- <value>: Target position of the modulo rotary axis in the range between 0° and 359.999° Negative values are also possible if a range offset has been realized with the following machine data:
  - MD30340 $MA_ MODULO_RANGE_START
  - MD30330 $MA_MODULO_RANGE
- ACP: Absolute programming of the target position in positive rotation direction (counter clockwise).
- ACN: Absolute programming of the target position in negative rotation direction (clockwise).
- AC / G90: by block / modally valid absolute programming of the target position.
  The direction of rotation depends on the current position of the modulo rotary axis.
  - Target position > Current position: Travel in positive direction.
  - Target position < Current position: Travel in negative direction.

Note

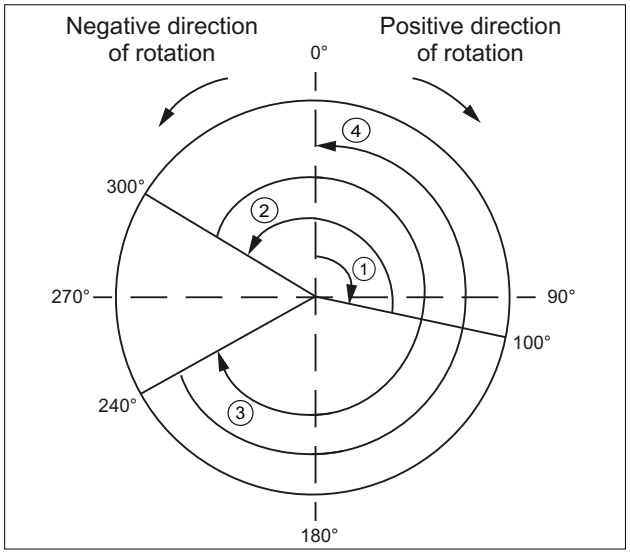
The G90 positioning behavior can be projected via MD30455 $MA_MISC_FUNCTION_MASK, Bit 2. See Section "Modulo 360 degrees (Page 223)".

Application example

ACP / ACN: Defined direction of rotation for asymmetric workpieces in order to exclude collisions during rotation.

Programming examples

Starting position of modulo rotary axis C = 0° (see figure below).



① `POS[C] = ACP(100); Traverse in positive direction of rotation to position 100°`

② `POS[C] = ACN(300); Traverse in negative direction of rotation to position 300°`

③ `POS[C] = ACP(240); Traverse in positive direction of rotation to position 240°`

④ `POS[C] = AC(0); Traverse in negative direction of rotation to position 0°`

Figure 7-5 Absolute programming for modulo rotary axes

Absolute programming along the shortest path (DC)

Example of absolute programming on the shortest path for a modulo rotary axis as positioning axis:

```
POS[<axis name>] = ACP(<value>)
```

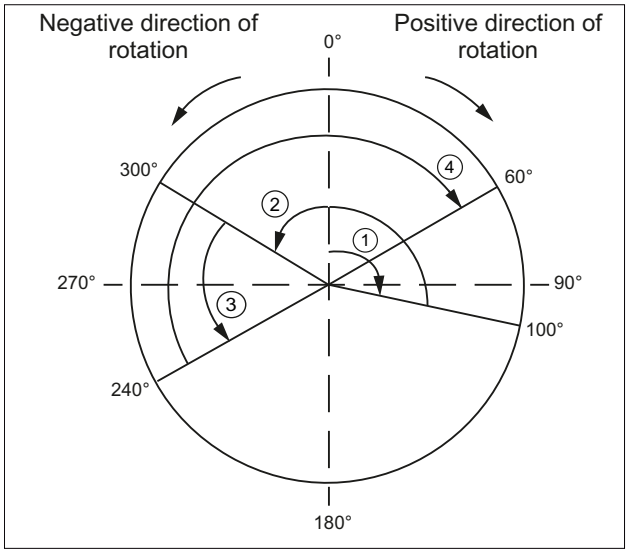
- <value>: Target position of the modulo rotary axis in the range between 0° and 359.999° Negative values are also possible if a range offset has been realized with the following machine data:
  - MD30340 $MA_ MODULO_RANGE_START
  - MD30330 $MA_MODULO_RANGE
- `DC`: Absolute programming of the target position for moving by the shortest direction of rotation.

Note

If the route is the same in both directions (180°), movement in in the positive direction of rotation.

Programming examples

Starting position of modulo rotary axis C = 0° (see figure below).



① `POS[C] = DC(100); Traverse by shortest path to position 100°`
② `POS[C] = DC(300); Traverse by shortest path to position 300°`
③ `POS[C] = DC(240); Traverse by shortest path to position 240°`
④ `POS[C] = DC(60); Traverse by shortest path to position 60°`
`; in positive direction of rotation`

Figure 7-6 Absolute programming with shortest path for modulo rotary axes

Block-search response

After block search with calculation, the collected search run positions can be read by modulo, converted using the system variable $AC_RETPOINT[<axis>].

Supplementary conditions for ASUB after block search with calculation

In this instance, as well as with the cross-channel block-search tool SERUPRO, the modulo conversion simulated in the block search must be performed in the part program.

Incremental programming (IC, G91)

Example for positioning axis: **POS[axis name] = IC(+/-value)**

- The value identifies the rotary-axis traversing distance. The value can be negative and ≥ +/-360°.
- The value's **sign** unequivocally defines the rotary-axis **traversing direction**.
- Application example: milling a spiral groove across several revolutions

Example

Programming	Effect
`POS[C]=IC(720)`	C axis traverses to 720° incrementally in the positive direction (two revolutions)
`POS[C]=IC(-180)`	C axis traverses to 180° incrementally in the negative direction

Program example: Modulo rotary axes as endlessly rotating rotary axis

```
Program code
LOOP:
POS[C] = IC(720) ; Traverse as positioning axis by 270°
GOTOB LOOP       ; Return to Label LOOP
```

7.3.3 Rotary axis without modulo conversion

Deactivating modulo conversion

→ set MD30310 $MA_ROT_IS_MODULO = 0.

Absolute programming (AC, G90)

Example for positioning axis: **POS[axis name] = AC (+/- value)**

- The value and its sign uniquely identify the rotary-axis target position. The value can be ≥ +/-360°. The position value is limited by the software limit switch positions.
- The traversing direction is defined by the control according to the signed rotary axis actual position.
- If `ACP` or `ACN` is programmed, alarm 16810, "ACP traverse instruction cannot be used", or alarm 16820 "ACN traverse instruction cannot be used", is output.
- Application example:
  Linear motion (cam gear) are coupled to the rotary axis, thus certain end positions may not be passed.

Example:

Programming	Effect
`POS[C] = AC (-100)`	Rotary axis C traverses to position -100°; traversing direction depends on the starting position
`POS[C] = AC (1500)`	Rotary axis C traverses to position 1500°

Absolute programming along the shortest path (DC)

POS[axis name] = DC(value)

Even if the rotary axis is not defined as a modulo axis, the axis can still be positioned with `DC` (Direct Control). The behavior is the same as that of a modulo axis.

- The value identifies the rotary axis target position **in a range from 0° to 359.999° (modulo 360°)**. Alarm 16830, "Incorrect modulo position programmed", is output for values with a negative sign or ≥ 360º.
- With DC (Direct Control), the rotary axis approaches the programmed absolute position within one revolution along the **shortest path** (traversing movement max. ±180°).
- The control calculates the direction of rotation and the traverse path according to the current actual position (in relation to modulo 360°). If the path to be traversed is the same in both directions (180°), the positive direction of rotation has priority.
- `DC` application example: the rotary table is required to approach the changeover position in the shortest time (and, therefore, via the shortest path) possible.
- If `DC` is programmed for a linear axis, alarm 16800, "DC traverse instruction cannot be used", is output.

Example:

Programming	Effect
`POS[C] = AC (7200)`	Rotary axis C traverses to position 7200°; traversing direction depends on the starting position
`POS[C] = DC (300)`	Rotary axis C approaches "modulo" position 300° along the shortest path Thus, C traverses through 60° with a negative direction of rotation and stops at absolute position 7140°.
`POS[C] = AC (7000)`	Rotary axis C traverses to position 7000° absolute, C traverses through 140° with a negative direction of rotation

Note

In this example, it would be advisable to activate the modulo 360º display (MD30320 $MA_DISPLAY_IS_MODULO = 1).

Incremental programming (IC, G91)

Example for positioning axis: **POS[axis name] = IC(+/- value)**

When programming with incremental dimensions, the rotary axis traverses along the same path as for the modulo axis. In this case, however, the traversing range is limited by the software limit switches.

- The value identifies the rotary axis traversing distance.
  The value can be negative and ≥ +/-360°.
- The **sign** of the value defines the rotary axis **traversing direction**.

Limited traversing range

The traversing range is limited just the same as for linear axes. The range limits are defined by the "plus" and "minus" software limit switches.

7.3.4 Other programming features relating to rotary axes

Offsets

TRANS (absolute) and ATRANS (additive) can be applied to rotary axes.

Scalings

SCALE or ASCALE are not suitable for rotary axes, since the control always bases its modulo calculation on a 360º full circle.

Preset actual value memory

PRESETON is possible.

Indexing axes

See Section "T1: Indexing axes (Page 239)".

7.4 Activating rotary axes

Procedure

The procedure for activating rotary axes is the same as that for linear axes with a small number of exceptions. It should be noted that, as soon as the axis is defined as a rotary axis (MD30300 $MA_IS_ROT_AX = 1), the axis-specific-machine-/setting-data units are interpreted by the control as follows:

Positions	In "degrees"
Velocities	In "rev/min"
Accelerations	In "rev/sec^2"
Jerk limitation	In "rev/sec^3"

Special machine data

Special rotary-axis machine data may also have to be entered, depending on the application:

MD30310 $MA_ROT_IS_MODULO	Modulo conversion for positioning and programming
MD30320 $MA_DISPLAY_IS_MODULO	Modulo conversion for position display
MD10210 $MN_INT_INCR_PER_DEG	Computational resolution for angular positions

The following overview lists the possible combinations of these machine data for a rotary axis:

Possible combinations of rotary-axis machine data				
MD30300	**MD30310**	**MD30320**	**Application permitted**	**Comment**
0	0	0	Yes	The axis is a linear axis (default).
1	0	0	Yes	The axis is a rotary axis; modulo conversion is not used for positioning, i.e. the software limit switches are active; the position display is absolute.
1	0	1	Yes	The axis is a rotary axis; modulo conversion is not used for positioning, i.e. the software limit switches are active; the position display is modulo; application: for axes with an operating range of +/-1000°, for example
1	1	1	Yes	The axis is a rotary axis; positioning is performed with modulo conversion, i.e. the software limit switches are inactive, the operating range is unlimited; the position display is modulo (setting most frequently used for rotary axes); axis with/without working-area limitation can be used.
1	1	0	Yes	The axis is a rotary axis; positioning is performed with modulo conversion, i.e. the software limit switches are inactive, the operating range is unlimited; the position display is absolute; axis with/without working-area limitation can be used.
0	0 or 1	0 or 1	Not recommended	Axis is not a rotary axis; therefore, the other MD are not evaluated.

JOG velocity for rotary axes

SD41130 $SN_JOG_ROT_AX_SET_VELO (JOG speed for rotary axes)

The above setting data can be used to define a valid JOG velocity for all rotary axes (see also Section "H1: Manual traversing (Page 633)").

If a value of 0 is entered in the setting data, the following axial machine data acts as JOG velocity for the rotary axis:

MD21150 $MC_JOG_VELO (conventional axis velocity)

7.5 Special features of rotary axes

Software limit switch

The software limit switches and working-area limitations are active and are required for swivel axes with a limited operating range. However, in the case of continuously rotating rotary axes (MD30310 $MA_ROT_IS_MODULO = 1), the software limit switches and working area limitations can be deactivated for individual axes.

A modulo rotary axis with/without working-area limitation can be used.

Further information

Function Manual Monitoring and Compensating; Reference point approach (R1)

Function Manual Monitoring and Compensating; Reference point approach (R1)Mirroring of rotary axes

Mirroring can be implemented for rotary axes by programming `MIRROR(C)` or `AMIRROR(C)`.

Reference point approach

Further information
Function Manual Monitoring and Compensating; Reference point approach (R1)

Spindles as rotary axes

For notes concerning the use of spindles as rotary axes (C axis operation), please refer to:

Further information
S1: Spindles (Page 733)

7.6 Examples

Fork head, inclined-axis head

Rotary axes are frequently used on 5-axis milling machines to swivel the tool axis or rotate the workpiece. These machines can position the tip of a tool on any point on the workpiece and take up any position on the tool axis. Various milling heads are required, depending on the application. The figure shows a fork head and an inclined-axis head as example rotary-axis arrangements.

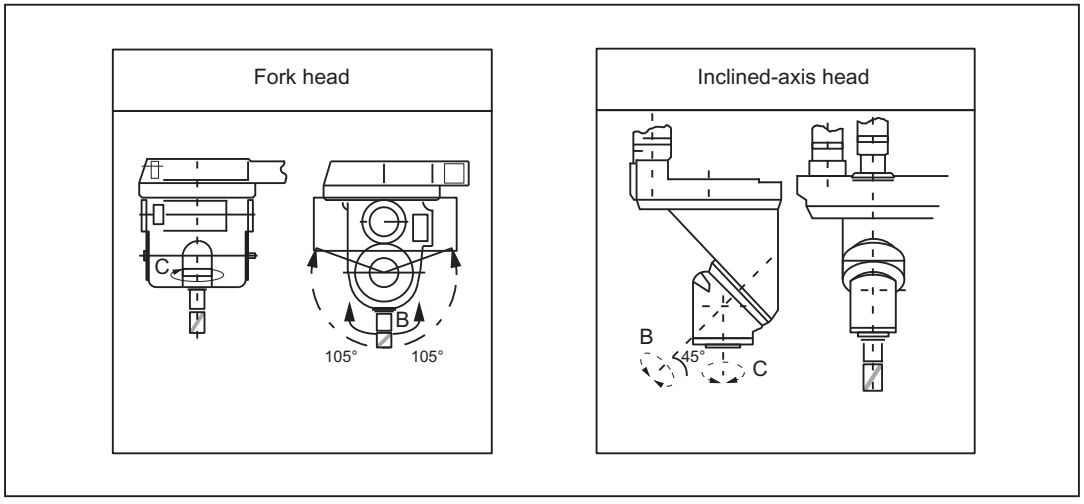


Figure 7-7 Fork head, inclined-axis head

7.7 Data lists

7.7.1 Machine data

7.7.1.1 General machine data

Number	Identifier: $MN_	Description
10210	INT_INCR_PER_DEG	Computational resolution for angular positions

7.7.1.2 Axis/spindlespecific machine data

Number	Identifier: $MA_	Description
30300	IS_ROT_AX	Axis is rotary axis
30310	ROT_IS_MODULO	Modulo conversion for rotary axis

Number	Identifier: $MA_	Description
30320	DISPLAY_IS_MODULO	Actual value display modulo
30330	MODULO_RANGE	Modulo-range size
30340	MODULO_RANGE_START	Modulo-range starting position
30455	MISC_FUNCTION_MASK	Axis functions
36100	POS_LIMIT_MINUS	Minus software limit switch
36110	POS_LIMIT_PLUS	Plus software limit switch

7.7.2 Setting data

7.7.2.1 General setting data

Number	Identifier: $SN_	Description
41130	JOG_ROT_AX_SET_VELO	JOG velocity for rotary axes

7.7.2.2 Axis/spindle-specific setting data

Number	Identifier: $SA_	Description
43420	WORKAREA_LIMIT_PLUS	Plus working-area limitation
43430	WORKAREA_LIMIT_MINUS	Minus working-area limitation

T1: Indexing axes

8

8.1 Brief description

When machine axes only traverse between a certain number of fixed positions, these positions can be parameterized as indexing positions. In NC programs, these machine axes, then known as indexing axes, can be traversed with reference to indexing positions using special commands.

Typical applications for indexing axes include tool magazines, for example (tool revolver, tool chain magazines or tool cartridge magazines). The magazine locations of the tools are parameterized as indexing positions.

It is possible to traverse an indexing axis in the following operating modes:

- AUTOMATIC
- MDI
- JOG
- JOG / INC1, INC10, ... INCvar

8.2 Detailed description

8.2.1 Traversing of indexing axes in the AUTOMATIC mode

Traversal to selected positions

Using the general traversing commands (`G90`, `G91`, `AC`, `IC`, ...) an indexing axis can be traversed to any position within the traversing range of the machine axis.

Traversing to indexing positions

Using the specific traversing commands for indexing axes, it is ensured that only the parameterized indexing positions are approached. See Chapter "Programming (Page 248)".

8.2.2 Traversing of indexing axes in the JOG mode

In the JOG mode, the indexing positions are only taken into account after referencing/ synchronizing the active measuring system of the indexing axis.

Further information

R1: Referencing (Page 291)

Continuous traversing (JOG CONT)

Jog mode

In the jog mode (SD41050 $SN_JOG_CONT_MODE_LEVELTRIGGRD = 1) the indexing axis traverses in the selected direction after the traversing key has been actuated. The indexing axis is stopped at the next possible indexing position after releasing the traversing key. The indexing position where the axis stops is dependent on:

- Distance to the indexing position
- Traversing velocity
- Axis dynamic response

Continuous operation

In continuous operation (SD41050 $SN_JOG_CONT_MODE_LEVELTRIGGRD = 0), after actuating the traversing key (first rising edge), the indexing axis is traversed as usual. The indexing axis is stopped at the next possible indexing position when the traversing key is actuated again (second rising edge). The indexing position where the axis stops is dependent on:

- Distance to the indexing position
- Traversing velocity
- Axis dynamic response

Generally, indexing axes are traversed in the jog mode.

Direction change

If, before reaching the indexing position, the operator changes the direction by actuating the traversing key for the opposite direction, the indexing axis continues to traverse to the next possible indexing position where it can then be stopped. The traversing motion must then be started in the opposite direction by again actuating the traversing key.

Further information

You can find detailed information on the traversing axes in the JOG mode in:

H1: Manual traversing (Page 633)

Incremental traversing (INC)

Independent of the active technology function (INC1, INC10, ... , INCvar), the indexing axis always moves in the selected direction to the next indexing position after the traversing key has been pressed.

Jog mode

In the jog mode (SD41050 $SN_JOG_CONT_MODE_LEVELTRIGGRD = 1), the indexing axis is immediately stopped after releasing the traversing key. Under certain circumstances, the indexing axis is then **not** located at an indexing position. Traversing motion is continued at the indexing position by again actuating the traversing key.

Continuous operation

In continuous operation (SD41050 $SN_JOG_CONT_MODE_LEVELTRIGGRD = 0), after actuating the traversing key (first rising edge), the indexing axis is traversed as usual. The indexing axis is immediately stopped when the traversing key is actuated again (second rising edge). Under certain circumstances, the indexing axis is then **not** located at an indexing position. Traversing motion is continued at the indexing position by again actuating the traversing key.

Revolutional feedrate

The revolution feedrate with which the indexing axis is traversed in the JOG mode is dependent on:

SD41100 $SN_JOG_REV_IS_ACTIVE, bit 0 = <value>

<Value>	Meaning
0	Depending on the traversing mode, the revolutional feed rate is defined by the following setting data: • The indexing axis traverses as **positioning axis**: SD43300 $SA_ASSIGN_FEED_PER_REV_SOURCE (revolutional feedrate for positioning axes) • The indexing axis traverses as **geometry axis**, on which a **frame with rotation** acts: SD42600 $SC_JOG_FEED_PER_REV_SOURCE (revolutional feedrate in JOG)
1	The indexing axis is traversed with a rotational feedrate derived from the master spindle: Corresponding machine data: • MD32050 $MA_JOG_REV_VELO (revolutional feed rate for JOG mode) • MD32040 $MA_JOG_REV_VELO_RAPID (revolutional feed rate for JOG with rapid traverse override)

Further information

- V1: Feedrates (Page 125)
- H1: Manual traversing (Page 633)
- Programming Manual Fundamentals, Feedrate control

8.2.3 Traversing of indexing axes by PLC

Indexing axes can be traversed from the PLC user program.

- Competing positioning axes, see Section "P2: Positioning axes (Page 185)"
- Asynchronous subprograms (ASUBs)
  Further information:
  Function Manual Basic Functions; Mode group, channel, program operation, reset response

See also

R1: Referencing (Page 291)

8.3 Commissioning

8.3.1 Machine data for indexing axes

Indexing axis

An axis is defined as indexing axis by assigning an indexing position table to this axis, using the following axis-specific machine data:

MD30500 $MA_INDEX_AX_ASSIGN_POS_TAB = <value>

<Value>	Meaning
1	The axis is assigned indexing position table 1.
2	The axis is assigned indexing position table 2.
3	Equidistant indexing intervals (Page 245)

First indexing position for indexing axes with equidistant positions

For an indexing axis with **equidistant** positions, the absolute position of the first indexing position is set using the following axes-specific machine data:

MD30503 $MA_INDEX_AX_OFFSET

Indexing position tables

In the indexing position tables, the indexing positions are saved in mm, inches or degrees. Presently, two indexing position tables are available.

MD10910 $MN_INDEX_AX_POS_TAB_1 [<n>] (indexing position table 1)

MD10930 $MN_INDEX_AX_POS_TAB_2 [<n>] (indexing position table 2)

with <n> = 0 ... (maximum number - 1)

Multiple table assignment

Several axes can be assigned to an indexing position table.

Requirement: the axes are of the same type (linear axis, rotary axis, modulo 360° function)

General supplementary conditions

- The indexing positions must be listed in the indexing position table without gaps in an ascending order.
- Indexing positions that follow one another must not be identical.
- The indexing positions must be specified in the basic coordinate system.

Additional secondary conditions for modulo rotary axes

- Permissible range: 0° ≤ indexing position < 360°
- If the indexing axis is at the last indexing position in the indexing position table, when traversing to the next indexing position in the positive direction of rotation, the first indexing position is approached.
- If the indexing axis is at the first indexing position in the indexing position table, when traversing to the next indexing position in the negative direction of rotation, the last indexing position is approached.

No. of indexing positions

A maximum of 60 indexing positions can be entered in an indexing position table. The actual number of indexing positions that is used or the programmable number is defined using machine data:

MD10900 $MN_INDEX_AX_LENGTH_POS_TAB_1 (number of indexing positions of indexing position table 1)

MD10920 $MN_INDEX_AX_LENGTH_POS_TAB_2 (number of indexing positions of indexing position table 2)

Note

Entries in the indexing positions table that exceed the parameterized number of indexing positions are not considered.

System of units for indexing positions

The system of units of indexing positions is defined on an NC-specific basis:

MD10270 $MN_POS_TAB_SCALING_SYSTEM = <value>

<Value>	System of units
0	Metric
1	inch

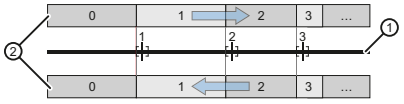
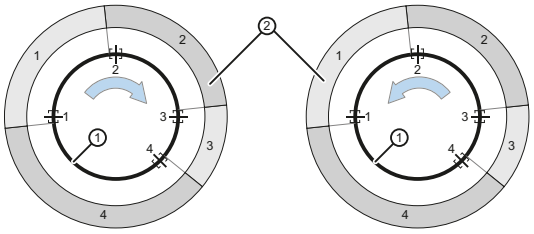
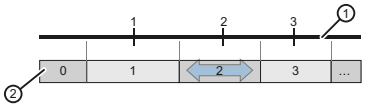
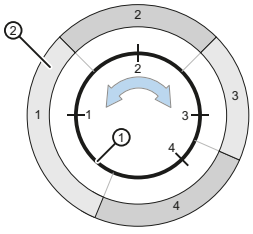
Note

The machine data is only evaluated if the following applies: MD10260 $MN_CONVERT_SCALING_SYSTEM == 1

Displaying the actual indexing position

When traversing indexing axes, the machine data is used to define at which actual position the display of the actual indexing position changes in the system variable $AA_ACT_INDEX_AX_POS_NO:

MD10940 $MN_INDEX_AX_MODE, bit 0 = <value>

Bit	<Value>	Meaning
0	0	When traversing in the direction of ascending indexing positions, the displayed current indexing position changes in the system variable $AA_ACT_INDEX_AX_POS_NO upon reaching/overtraveling the "Exact stop fine" window (MD36010 $MA_STOP_LIMIT_FINE) of the next indexing position. When traversing in the direction of descending indexing positions, the indexing position display changes as soon as you exit the "Exact stop fine" window.  Linear axis  Modulo rotary axis
		① The traversing range of the indexing axis with indexing positions and "Exact stop fine" window
		② Indexing position indicated in the system variable $AA_ACT_INDEX_AX_POS_NO
	1	The actual indexing position indicated in system variable $AA_ACT_INDEX_AX_POS_NO changes when reaching/passing half the traversing distance to the next indexing position in the traversing direction.  Linear axis  Modulo rotary axis
		① Traversing range of the indexing axis with indexing positions
		② Indexing position indicated in the system variable $AA_ACT_INDEX_AX_POS_NO

Note

With the system variables $AA_ACT_INDEX_AX_POS_NO it is not possible to determine whether an indexing position has been reached. To this end, one of the two following NC/PLC interface signals must also be evaluated:

DB31, ... DBX76.6 (indexing position reached)

DB31, ... DBX60.7 (exact stop fine reached)

Example

Assumptions:

- MD10940 $MN_INDEX_AX_MODE, bit 0 == 1
- The axis is located in front of an indexing position outside the tolerance window "exact stop fine" (DB31, ...DBX60.7 == 0)

Although system variable $AA_ACT_INDEX_AX_POS_NO indicates indexing position 2, with the "Traverse to next position" command, indexing position 3 is **not** approached, but first precisely to indexing position 2. Only when the axis is precisely at (DB31, ...DBX60.7 == 1) - or is located after the displayed indexing position, using the "Traverse to next position" command, the next indexing position is approached, in the example, indexing position 3. This means that the indexing position is approached that is the closest **in the traversing direction** ! Under certain circumstances, it is therefore necessary to issue the "traverse to next position" command twice to move from the currently displayed indexing position to the next indexing position.

8.3.2 Machine data for equidistant indexing intervals

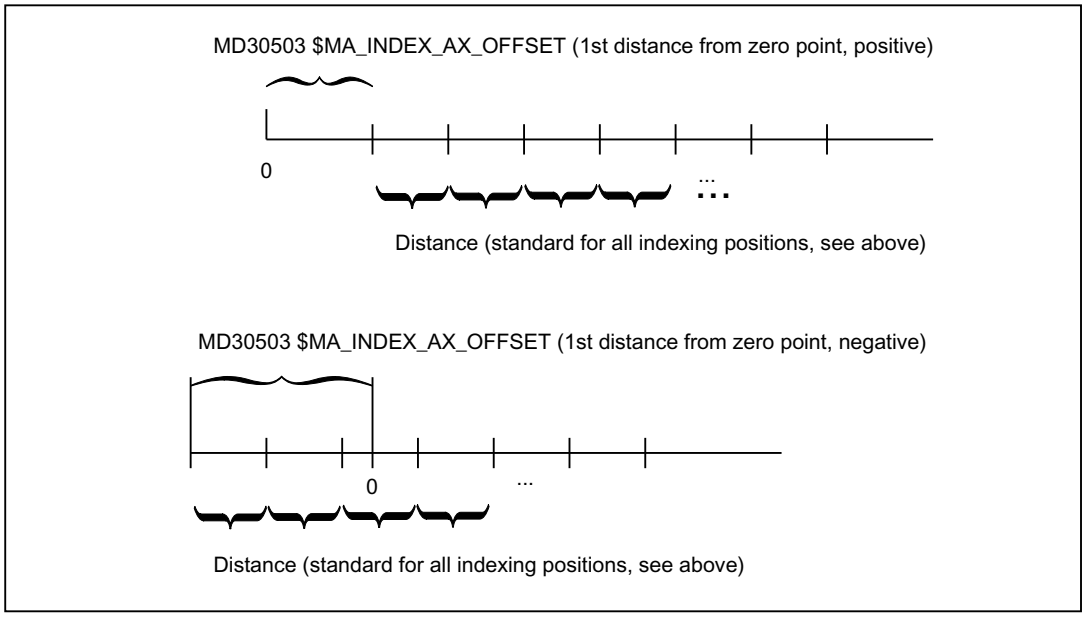
8.3.2.1 Features

Distance between indexes

The index distance is determined for equidistant index intervals according to the following formula:

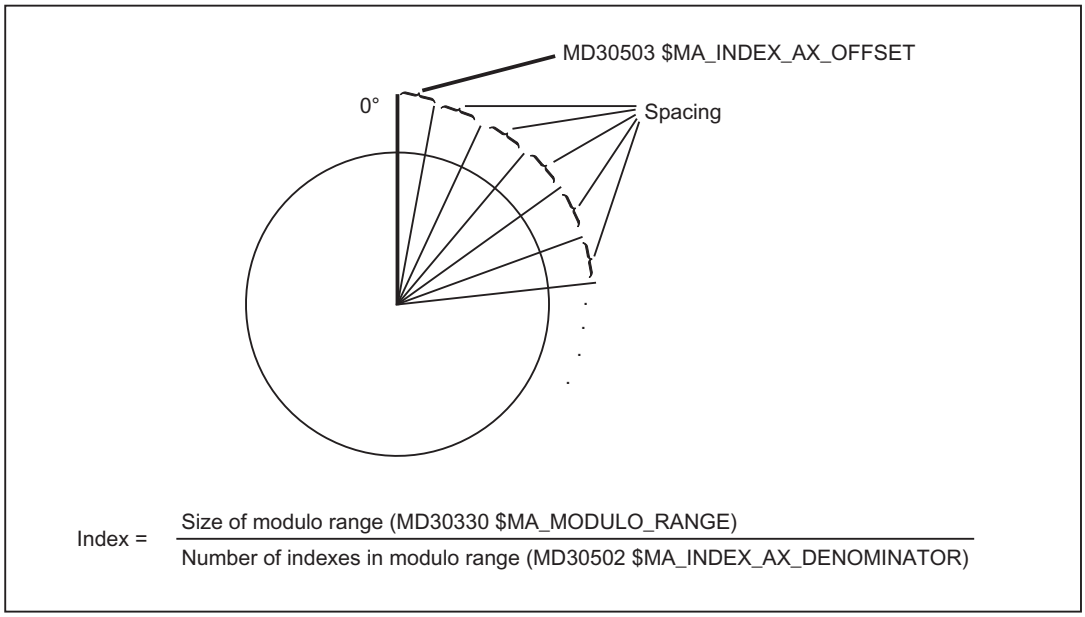
$$\text{Distance} = \frac{\text{Numerator (MD30501 \$MA_INDEX_AX_NUMERATOR)}}{\text{Denominator (MD30502 \$MA_INDEX_AX_DENOMINATOR)}}$$

Linear axes



Modulo rotary axes

$$\text{Index} = \frac{\text{Numerator (MD30330 \$MA_MODULO_RANGE)}}{\text{Denominator (MD30502 \$MA_INDEX_AX_DENOMINATOR)}}$$



8.3.2.2 Hirth axis

Function

For a "Hirth axis", using a special gearing (Hirth gearing) the rotary axis is interlocked when reaching an indexing position. In this case, a locking bolt or a gearwheel is engaged using a linear axis. In order that the mechanical system of the machine is not damaged, the interlock can only be engaged at an indexing position. For a Hirth axis, all of the indexing positions must have the same the distance between one another (equidistant indexing positions).

Requirements

- The "Hirth axis" must be a rotary axis.
- The "Hirth axis" must be an indexing axis. See Section "Commissioning (Page 242)"
- The active measuring system of the indexing axis must be synchronized or referenced: DB31, ... DBX60.4/.5 == 1 (referenced/synchronized, encoder 1/2)
  Further information
  R1: Referencing (Page 291)

Commissioning

Machine data

- MD30300 $MA_IS_ROT_AX = 1 (rotary axis)
- MD30505 $MA_HIRTH_IS_ACTIVE[<axis>] = 1 (indexing axis with Hirth gearing)
- MD30500 $MA_INDEX_AX_ASSIGN_POS_TAB[<axis>] = 3 (indexing axis with equidistant indexing positions)

8.3.3 Axis-specific system variables

$AA_PROG_INDEX_AX_POS_NO

Function

The system variable includes the number of the indexing position programmed for the indexing axis.

Syntax

```
$AA_PROG_INDEX_AX_POS_NO[<axis>]
```

Meaning

$AA_PROG_INDEX_AX_POS_NO	
0	The axis is not an indexing axis, or presently the indexing axis is not traversing to an indexing position
> 0	Number of programmed indexing position

$AA_ACT_INDEX_AX_POS_NO

Function

The system variable includes the number of the indexing position last reached by the indexing axis, or indexing position that was passed by the indexing axis.

Syntax

```
$AA_ACT_INDEX_AX_POS_NO[<axis>]
```

Meaning

$AA_ACT_INDEX_AX_POS_NO	
0	The axis is not an indexing axis.
> 0	Number of the last indexing position reached or passed

8.3.4 NC/PLC interface signals

Axis-specific interface signals

When stopping the indexing axis at an indexing position ± exact stop window fine (MD36010 $MA_STOP_LIMIT_FINE), then the axis-specific interface signal is set:

DB31, ... DBX76.6 = 1 (indexing axis in position)

8.4 Programming

Coded position

To allow indexing axes to be positioned from the NC part program, special instructions are provided with which the indexing numbers (e.g. location numbers) are programmed instead of axis positions in mm or degrees. The availability of a special instruction depends on the axis type (linear or rotary axis):

Statement	Effect	Availability
`CAC(i)`	Traverse **c**oded position in **a**bsolute terms	Linear axis, rotary axis
`CACP(i)`	Traverse **c**oded position **a**in absolute terms in the **p**ositive direction	Rotary axis
`CACN(i)`	Traverse **c**oded position **a**in absolute terms in the **n**egative direction	Rotary axis
`CDC(i)`	Traverse **c**oded position along the **d**irect (shortest) path	Rotary axis
`CIC(i)`	Traverse **c**oded position **i**ncrementally	Linear axis, rotary axis

i:	Coded position (indexing position)
Value range of i:	0 ... 59; whole number (positive and negative values are possible in `CIC`)

Examples

Program code	Comment
POS[B]=CAC(20)	Indexing axis B approaches the coded position (indexing) 20 in **absolute** mode. The direction of traversing depends on the current actual position.
POS[B]=CACP(10)	Indexing axis B approaches the coded position (index position) 10 in **absolute mode with positive direction of rotation** (possible only for rotary axes).
POS[B]=CACN(10)	Indexing axis B approaches the coded position (index position) 10 in **absolute mode with negative direction of rotation** (possible only for rotary axes).
POS[B]=CDC(50)	Indexing axis B approaches indexing position 50 **directly along the shortest path** (possible only for rotary axes).
POS[B]=CIC(-4)	Indexing axis B traverses four indexing positions **incrementally** from its current position. **in a negative direction**.
POS[B]=CIC(35)	Indexing axis B traverses 35 indexing positions **incrementally** from its present position **in a positive direction**.

Special features

- Modulo rotary axis as indexing axis
  On modulo rotary axes, the indexing positions are divided in factors of 360° and approached directly.
- Indexing axis is between two indexing positions
  The specified position instructions have the following effect in the AUTOMATIC mode.

POS[B]=CIC(1)	The **next higher** indexing position is approached.
POS[B]=CIC(-1)	The **next lower** indexing position is approached.
POS[B]=CIC(0)	The indexing axis is **not** traversed.

System variables

Information on the indexing positions can be read via the following system variables:

- $AA_PROG_INDEX_AX_POS_NO
  Number of the last programmed indexing position
- $AA_ACT_INDEX_AX_POS_NO
  Number of the last overtraveled indexing position
  The display depends on the setting in machine data MD10940 $MN_INDEX_AX_MODE.
- $AA_ACT_INDEX_AX_POS_NO
  Number of the last indexing position reached or passed

Indexing positions			
For axis type	**Values**	**Description**	
Modulo rotary axis	1, ... n	n = number of indexing positions (Page 242)	
Linear axis	0, 1, 2, 3, ... 59, 60, 61	0	The actual position lies **below** the lowest indexing position
		61	The actual position lies **above** the highest indexing position

Equidistant indexing positions			
For axis type	**Values**	**Description**	
Modulo rotary axis	1 ... m	m = denominator (counter)	
Linear axis	... -3, -2, -1, 0, 1, 2, 3, ...		

8.5 Supplementary conditions

Indexing axes

Various channel and axis-specific NC/PLC interface signals

If, while an indexing axis is traversed, at least one of the following signals occurs, then the axis stops immediately. The indexing positions are not taken into account.

- Channel-specific signals
  - DB21, ... DBB4 == 0 (feedrate override)
  - DB21, ... DBX6.2 == 1 (delete distance-to-go)
  - DB21, ... DBX7.3 == 1 (NC Stop)
  - DB21, ... DBX7.4 == 1 (NC Stop axes plus spindles)
  - DB21, ... DBX7.7 == 1 (reset)
- Axisspecific signals
  - DB31, ... DBX4.3 == 1 (feedrate stop)

Synchronized actions

- Stop (MOV=0)
  If an indexing axis is stopped with a synchronized action using MOV=0, then the axis stops in the traversing direction at the next possible indexing position
- Delete distance to go (DELDTG)
  If, for an indexing axis, the distance to go is deleted with a synchronized action using DELDTG, then the axis is immediately stopped without stopping at an indexing position

Further information: Function Manual Synchronized Actions

Frames and work offsets

In conjunction with indexing axes, generally frames are not necessary. As a consequence, we recommend that frames and work offsets for indexing axes are suppressed in NC programs.

Software limit switch

When manually traversing an indexing axis in the JOG or JOG/INC operating mode, the indexing axis traverses – as a maximum up – to the last indexing position before the software limit switch.

Manual traversing using a handwheel (DRF)

When traversing an indexing axis using the handwheel, the indexing positions are **not** taken into account. This means that the indexing axis stops immediately, also between two indexing positions, with the end of the handwheel pulse.

"Hirth axes"

In addition to the secondary conditions mentioned above, for "Hirth axes" the following additional secondary conditions must be observed.

Various channel and axis-specific NC/PLC interface signals

If, while a "Hirth axis" is being traversed, at least one of the following signals occurs, then the axis stops immediately at the next possible indexing position in the traversing direction.

- Channel-specific signals
  - DB21, ... DBB4 == 0 (feedrate override)
  - DB21, ... DBX6.2 == 1 (delete distance-to-go)
  - DB21, ... DBX7.3 == 1 (NC Stop)
  - DB21, ... DBX7.4 == 1 (NC Stop axes plus spindles)
  - DB21, ... DBX7.7 == 1 (reset)
- Axisspecific signals
  - DB31, ... DBX4.3 == 1 (feedrate stop)

Emergency Stop

If, while a "Hirth axis" is being traversed, an Emergency Stop is initiated then the axis is immediately stopped:

DB10 DBX56.1 == 1 (Emergency Stop)

Further information: Function Manual Basic Functions, emergency stop

NOTICE
Approaching the indexing position After an Emergency Stop, before the interlocking of the Hirth gearing at the machine is reactivated, either the PLC user program or the operator (manually) must first move the "Hirth axis" to an indexing position in the JOG mode.

Synchronized actions

- Delete distance to go (DELDTG)
  If, for a "Hirth axis", the distance to go is deleted with a synchronized action using DELDTG, then the axis is stopped at the next possible indexing position in the traversing direction
  Further information: Function Manual Synchronized Actions, "Detailed description" > "Actions in synchronized actions" > "Delete distance to go (DELDTG)"

Various axis-specific functions

"Hirth axes" may **not** be used in conjunction with the following functions:

- Kinematic transformation
  Further information: Function Manual Transformations, kinematic transformation
- Work offset by setting the actual value: PRESETON
  Further information: Programming Manual, Job Planning, Section "Coordinate transformations (frames)"
- Revolutional feedrate (G95, FPRAON)
- Manual traversing using a handwheel as well as distance or velocity overlay using the handwheel (DFR)
  Further information: H1: Manual traversing (Page 633)
- Frames
  Further information: Function Manual Basic Functions; Axes, coordinate systems, frames > Frames
- Interpolatory corrections, for example external work offset, DRF offset
- Couplings
  It is not permissible that a "Hirth axis" is a following axis of one of the following couplings:
  - following axis with master value coupling
    Further information: Master value coupling (Page 453)
  - Coupled motion axis
    Further information: Coupled motion (Page 419)
  - gantry following axis
    Further information: G1: Gantry axes (Page 259)

8.6 Examples

8.6.1 Example 1: Rotary axis as indexing axis

Machine axes AX5 is a tool revolver, with 8 revolver locations (endlessly rotating rotary axis). The distances between the revolver locations are constant (equidistant indexing positions).

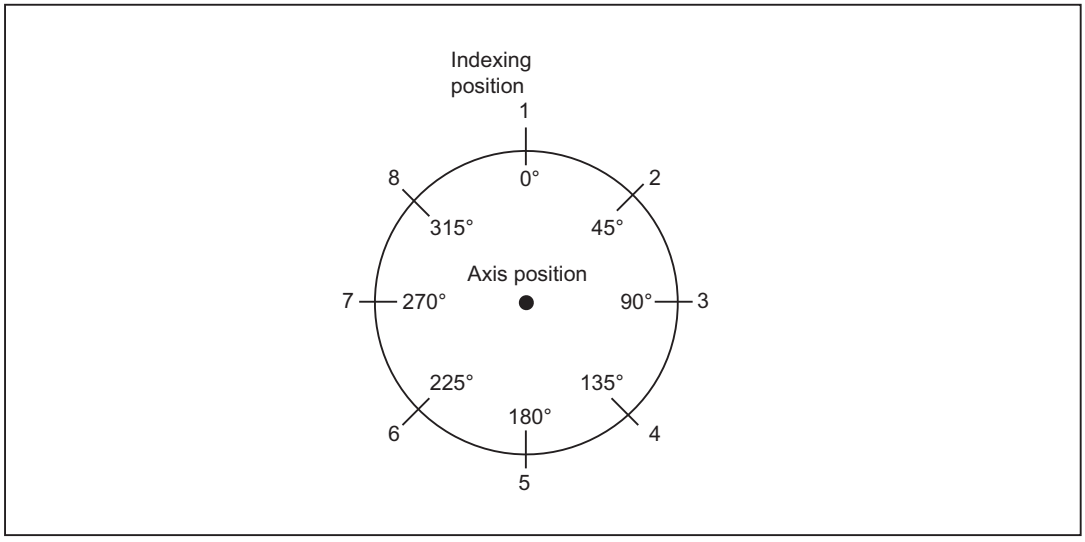


Figure 8-1 Tool turret with 8 locations

Indexing position table 1

MD10910 $MN_INDEX_AX_POS_TAB_1[0] = 0	1st indexing position = 0°
MD10910 $MN_INDEX_AX_POS_TAB_1[1] = 45	2nd indexing position = 45°
MD10910 $MN_INDEX_AX_POS_TAB_1[2] = 90	3rd indexing position = 90°
MD10910 $MN_INDEX_AX_POS_TAB_1[3] = 135	4th indexing position = 135°
MD10910 $MN_INDEX_AX_POS_TAB_1[4] = 180	5th indexing position = 180°
MD10910 $MN_INDEX_AX_POS_TAB_1[5] = 225	6th indexing position = 225°
MD10910 $MN_INDEX_AX_POS_TAB_1[6] = 270	7th indexing position = 270°
MD10910 $MN_INDEX_AX_POS_TAB_1[7] = 315	8th indexing position = 315°

Length of the indexing position table (1)

MD10900 $MN_INDEX_AX_LENGTH_POS_TAB_1= 8	8 indexing positions in table 1

Defining machine axis AX5 as rotary modulo axis and indexing axis

MD30300 $MA_IS_ROT_AX[AX5] = 1	Rotary axis
MD30310 $MA_ ROT_IS_MODULO [AX5] = 1	Modulo rotary axis
MD30500 $MA_INDEX_AX_ASSIGN_POS_TAB [AX5] = 1	Indexing axis, indexing position table 1

8.6.2 Example 2: Indexing axis as linear axis

Machine axis AX6 is a workholder with 10 locations. The distances between the 10 locations are different. The first location is at position -100 mm. Machine axis is AX6.

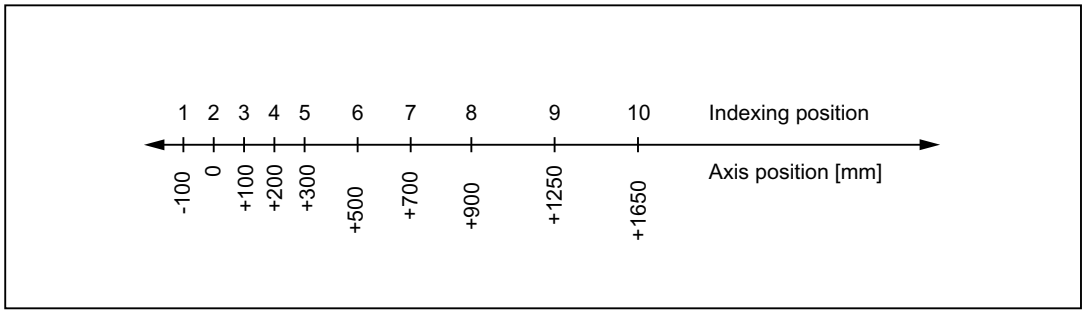


Figure 8-2 Indexing positions for a workholder

Indexing position table 2

MD10930 $MN_INDEX_AX_POS_TAB_2[0] = -100	1st indexing position = -100 mm
MD10930 $MN_INDEX_AX_POS_TAB_2[1] = 0	2nd indexing position = 0 mm
MD10930 $MN_INDEX_AX_POS_TAB_2[2] = 100	3rd indexing position = 100 mm
MD10930 $MN_INDEX_AX_POS_TAB_2[3] = 200	4th indexing position = 200 mm
MD10930 $MN_INDEX_AX_POS_TAB_2[4] = 300	5th indexing position = 300 mm
MD10930 $MN_INDEX_AX_POS_TAB_2[5] = 500	6th indexing position = 500 mm
MD10930 $MN_INDEX_AX_POS_TAB_2[6] = 700	7th indexing position = 700 mm
MD10930 $MN_INDEX_AX_POS_TAB_2[7] = 900	8th indexing position = 900 mm
MD10930 $MN_INDEX_AX_POS_TAB_2[8] = 1250	9th indexing position = 1250 mm
MD10930 $MN_INDEX_AX_POS_TAB_2[9] = 1650	10. Indexing position = 1650 mm

Length of the indexing position table (1)

MD10920 $MN_INDEX_AX_LENGTH_POS_TAB_2 = 10	10 indexing positions in table 2

Defining machine axis AX5 as indexing axis

MD30500 $MA_INDEX_AX_ASSIGN_POS_TAB [AX6] = 2	Indexing axis, indexing position table 2

8.6.3 Example 3: Rotary modulo axis as equidistant indexing axis

Machine axis AX4 is rotary modulo axis with equidistant indexing. Indexing starts at 5°. The difference between the indexing positions is 20°.

The following indexing positions are obtained:

5, 25, 45, 65, 85, 105, 125, 145, 165, 185, 205, 225, 245, 265, 285, 305, 325, 245 [Degrees]

Indexing data

MD30501 $MA_INDEX_AX_NUMERATOR[AX4] ⇒ MD30330 $MA_MODULO_RANGE[AX4] = 360.0	This value is ignored for rotary modulo axes, and MD30330 $MA_MODULO_RANGE (default value: 360°) is used instead.
MD30502 $MA_INDEX_AX_DENOMINATOR[AX4] = 18	Number of indexing positions = 360° / 20° = 18
MD30503 $MA_INDEX_AX_OFFSET[AX4] = 5.0	First indexing position = 5.0°
MD30505 $MA_HIRTH_IS_ACTIVE[AX4] = 0	No Hirth gearing

Defining machine axis AX4 as rotary modulo axis and equidistant indexing axis

MD30300 $MA_IS_ROT_AX[AX4] = 1	Rotary axis
MD30310 $MA_ ROT_IS_MODULO[AX4] = 1	Modulo function active
MD30500 $MA_INDEX_AX_ASSIGN_POS_TAB[AX4] = 3	Equidistant indexing axis

8.6.4 Example 4: Rotary axes as equidistant indexing axis with restricted traversing range

Machine axis AX4 is a rotary axis with equidistant indexing. The equidistant indexing positions are every 20°, starting at 100°. Positions less than 100° and greater than 260° should not be approached.

The following indexing positions are obtained:

100, 120, 140, 160, 180, 200, 220, 240, 260 [degrees]

Indexing data

MD30501 $MA_INDEX_AX_NUMERATOR[AX4] = 360.0	Numerator = 360°
MD30502 $MA_INDEX_AX_DENOMINATOR[AX4] = 18	Number of indexing positions = 360° / 20° = 18
MD30503 $MA_INDEX_AX_OFFSET[AX4] = 100.0	First indexing position = 100.0°
MD30505 $MA_HIRTH_IS_ACTIVE[AX4] = 0	No Hirth gearing

Defining machine axis AX4 as rotary modulo axis and equidistant indexing axis

MD30300 $MA_IS_ROT_AX[AX4] = 1	Rotary axis
MD30310 $MA_ ROT_IS_MODULO[AX4] = 1	Modulo function active
MD30500 $MA_INDEX_AX_ASSIGN_POS_TAB[AX4] = 3	Equidistant indexing axis
MD36100 $MA_POS_LIMIT_MINUS[AX4] = 100.0	1st software limit switch minus = 100.0°
MD36110 $MA_POS_LIMIT_PLUS[AX4] = 260.0	1st software limit switch plus = 260.0°

8.6.5 Example 5: Linear axis as equidistant indexing axis

Machine axis AX4 is a linear axis with equidistant indexing. The equidistant indexing positions are every 10 mm, starting at -200 mm.

The following indexing positions are obtained:

-200, -190, -180, ... 180, 190, 200 [mm]

Indexing data

MD30501 $MA_INDEX_AX_NUMERATOR[AX4] = 10.0	Numerator = 10
MD30502 $MA_INDEX_AX_DENOMINATOR[AX4] = 1	Denominator = 1
MD30503 $MA_INDEX_AX_OFFSET[AX4] = -200.0	First indexing position = -200.0°
MD30505 $MA_HIRTH_IS_ACTIVE[AX4] = 0	No Hirth gearing

Defining machine axis AX4 as linear axis and equidistant indexing axis

MD30300 $MA_IS_ROT_AX[AX4] = 0	Linear axis
MD30310 $MA_ ROT_IS_MODULO[AX4] = 0	Modulo function is not active
MD30500 $MA_INDEX_AX_ASSIGN_POS_TAB[AX4] = 3	Equidistant indexing axis
MD36100 $MA_POS_LIMIT_MINUS[AX4] = -200.0	1st software limit switch minus = -200.0 mm
MD36110 $MA_POS_LIMIT_PLUS[AX4] = 200.0	1st software limit switch plus = 200.0 mm

8.6.6 Example 6: "Hirth axis"

Machine axis AX4 is a "Hirth axis" (rotary modulo axis with equidistant indexing and Hirth gearing). The equidistant indexing positions are every 1°, starting at 0°.

Indexing data

MD30501 $MA_INDEX_AX_NUMERATOR[AX4] ⇒ MD30330 $MA_MODULO_RANGE[AX4] = 360.0	This value is ignored for rotary modulo axes, and MD30330 $MA_MODULO_RANGE (default value: 360°) is used instead.
MD30502 $MA_INDEX_AX_DENOMINATOR[AX4] = 360	Number of indexing positions = 360° / 1° = 360
MD30503 $MA_INDEX_AX_OFFSET[AX4] = 0.0	First indexing position = 0.0°
MD30505 $MA_HIRTH_IS_ACTIVE[AX4] = 1	Hirth gearing

Defining machine axis AX4 as linear axis and equidistant indexing axis

MD30300 $MA_IS_ROT_AX[AX4] = 1	Rotary axis
MD30310 $MA_ ROT_IS_MODULO[AX4] = 1	Modulo function active
MD30500 $MA_INDEX_AX_ASSIGN_POS_TAB[AX4] = 3	Equidistant indexing axis
MD36100 $MA_POS_LIMIT_MINUS[AX4] = -200.0	1st software limit switch minus = -200.0 mm
MD36110 $MA_POS_LIMIT_PLUS[AX4] = 200.0	1st software limit switch plus = 200.0 mm

8.7 Data lists

8.7.1 Machine data

8.7.1.1 General machine data

Number	Identifier: $MN_	Description
10260	CONVERT_SCALING_SYSTEM	Basic system switchover active
10270	POS_TAB_SCALING_SYSTEM	System of measurement of position tables
10900	INDEX_AX_LENGTH_POS_TAB_1	Number of positions for indexing axis table 1
10910	INDEX_AX_POS_TAB_1[n]	Indexing position table 1
10920	INDEX_AX_LENGTH_POS_TAB_2	Number of positions for indexing axis table 2
10930	INDEX_AX_POS_TAB_2[n]	Indexing position table 2
10940	INDEX_AX_MODE	Options for indexing positions

8.7.1.2 Axis/spindlespecific machine data

Number	Identifier: $MA_	Description
30300	IS_ROT_AX	Rotary axis
30310	ROT_IS_MODULO	Modulo conversion for rotary axis
30320	DISPLAY_IS_MODULO	Position display is modulo 360°
30500	INDEX_AX_ASSIGN_POS_TAB	Axis is indexing axis
30501	INDEX_AX_NUMERATOR	Numerator for indexing axes with equidistant positions
30502	INDEX_AX_DENOMINATOR	Denominator for indexing axes with equidistant positions
30503	INDEX_AX_OFFSET	Indexing position for indexing axes with equidistant positions
30505	HIRTH_IS_ACTIVE	Hirth tooth system is active

8.7.2 Setting data

8.7.2.1 General setting data

Number	Identifier: $SN_	Description
41050	JOG_CONT_MODE_LEVELTRIGGRD	JOG continuous in inching mode

8.7.3 System variables

Identifier	Description
$AA_ACT_INDEX_AX_POS_NO[axis]	Number of last indexing position reached or overtraveled
$AA_PROG_INDEX_AX_POS_NO[axis]	Number of programmed indexing position

G1: Gantry axes

9

9.1 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.

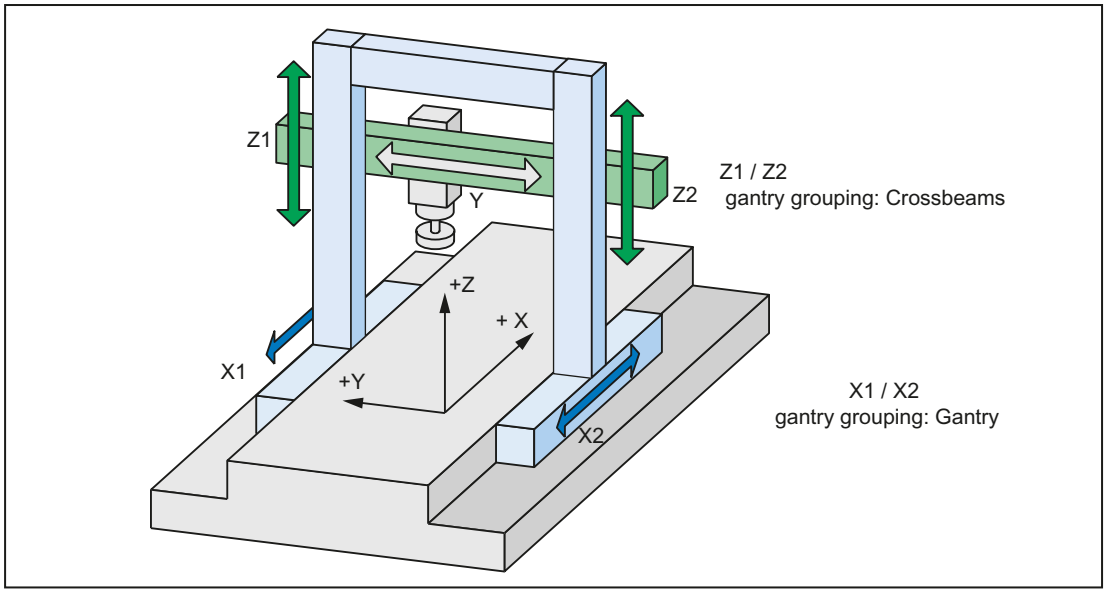


Figure 9-1 Example: Gantry-type milling machine with gantry and transverse beams

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.

9.2 "Gantry axes" function

9.2.1 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)

Note

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.

9.2.2 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.

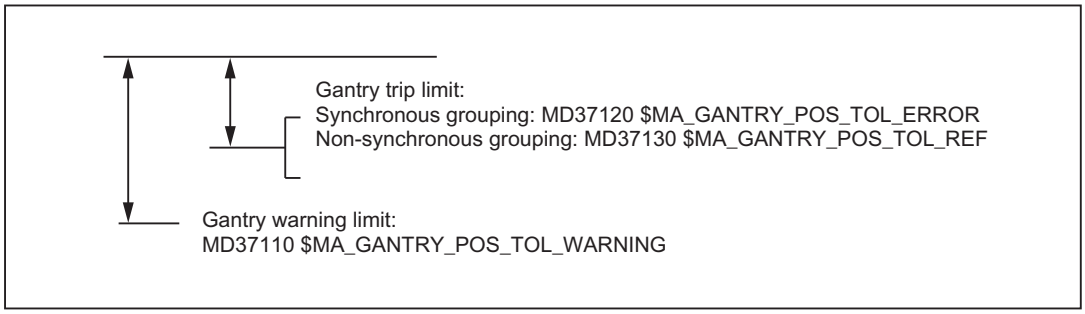
Gantry trip limit

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).

9.2.3 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:

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:
   MD37150 $MA_GANTRY_FUNCTION_MASK, Bit 0 = 0
2. Deleting the synchronism difference displayed in the machine data:
   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:
   MD37150 $MA_GANTRY_FUNCTION_MASK, Bit 0 = 1

9.2.4 Referencing and synchronization of gantry axes

Application

In cases where an incremental measuring system is being used for the guide axis or the synchronous axis, the measuring systems must be referenced during NC run-up, while 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 done, the NC/PLC interface signal is set:

DB31, ... DBX101.5 = 1 (gantry grouping is synchronized)

For the sequence when referencing or synchronizing gantry axes, see Chapter "Referencing and synchronization of gantry axes (Page 264)".

9.2.5 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 274)").

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.

9.2.6 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)

⚠ CAUTION

No synchronous operation

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!

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

9.3 Referencing and synchronization of gantry axes

9.3.1 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 distancecoded 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).

Further information:

R1: Referencing (Page 291)

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).

Further information:
R1: Referencing (Page 291)

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.

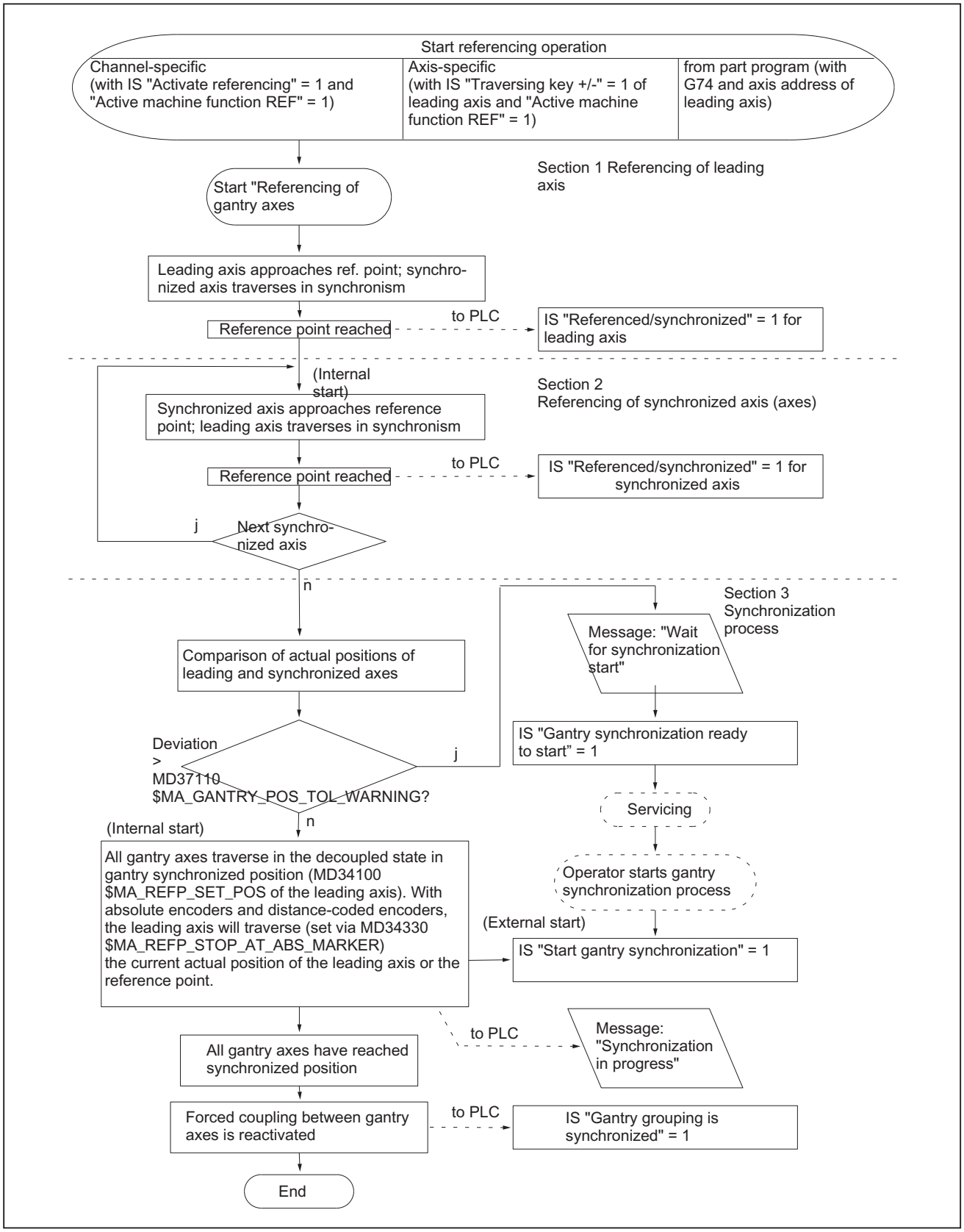

Start referencing operation
Channel-specific (with IS "Activate referencing" = 1 and "Active machine function REF" = 1)
Axis-specific (with IS "Traversing key +/-" = 1 of leading axis and "Active machine function REF" = 1)
from part program (with G74 and axis address of leading axis)
Start "Referencing of gantry axes
Section 1 Referencing of leading axis
Leading axis approaches ref. point; synchronized axis traverses in synchronism
Reference point reached
to PLC
IS "Referenced/synchronized" = 1 for leading axis
(Internal start)
Section 2 Referencing of synchronized axis (axes)
Synchronized axis approaches reference point; leading axis traverses in synchronism
Reference point reached
to PLC
IS "Referenced/synchronized" = 1 for synchronized axis
j
Next synchronized axis
n
Section 3 Synchronization process
Comparison of actual positions of leading and synchronized axes
Message: "Wait for synchronization start"
Deviation > MD37110 $MA_GANTRY_POS_TOL_WARNING?
j
IS "Gantry synchronization ready to start" = 1
n
Servicing
(Internal start)
All gantry axes traverse in the decoupled state in gantry synchronized position (MD34100 $MA_REFP_SET_POS of the leading axis). With absolute encoders and distance-coded encoders, the leading axis will traverse (set via MD34330 $MA_REFP_STOP_AT_ABS_MARKER) the current actual position of the leading axis or the reference point.
Operator starts gantry synchronization process
(External start)
IS "Start gantry synchronization" = 1
to PLC
Message: "Synchronization in progress"
All gantry axes have reached synchronized position
Forced coupling between gantry axes is reactivated
to PLC
IS "Gantry grouping is synchronized" = 1
End

Figure 9-2 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.

Note

For the leading axis, automatic synchronization can be locked using the following NC/PLC interface signal:

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:

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:
  MD37140 $MA_GANTRY_BREAK_UP = 0 (invalidate gantry axis grouping)

The corresponding NC/PLC interface signal is reset:

- DB31, ... DBX60.4 or DBX60.5 == 0 (referenced/synchronized 1 or 2 respectively)
- 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:

DB31, ... DBX29.4 == 1 (start gantry synchronization process)

Requirement is that the following applies to all axes of the gantry grouping:

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:

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 distancecoded 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 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)

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.

9.3.2 Automatic synchronization

Automatic synchronization can take place:

- in the referencing mode (see Section "Introduction (Page 264)")
- 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 the 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)

9.3.3 Points to note

2nd position measuring system 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 reference point approach operations do not have to travel through large distances, it is possible to use just one position measuring system - or a position measuring system as 2nd measuring system with distance-coded reference marks 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.

Further information:

R1: Referencing (Page 291)

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 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)

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 "**Se**arch **R**un by **Pro**gram test") can be used to simulate the traversal of gantry axis groupings.

Note

For further information on the multichannel block search SERUPRO, please see:
Further information
Function Manual Basic Functions; Mode group, channel, program operation, program test

9.4 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 shutdown limit when 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

For symmetrically set up gantry axes, it is recommended that you set the K_v factors and precontrol settings for all of the axes of a group identically.

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)

Further information

Function Manual Monitoring and Compensating; Compensations

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)
- MD32402 $MA_AX_JERK_MODE (filter type for 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)

Further information

Function Manual Basic Functions; Speeds/setpoint/actual value systems, closed-loop control

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 264)"
- Further information: R1: Referencing (Page 291)

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 262)"):

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.

Further information

Function Manual Monitoring and Compensating

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).

Further information

Function Manual Drive Functions SINAMICS S120; 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.

See also

G2: Velocities, setpoint / actual value systems, closed-loop control (Page 31)

9.5 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: **Further information** SINAMICS S120/S150 List Manual

9.6 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 → NC	29.4	X	
No automatic synchronization	PLC → NC	29.5	X	
Gantry axis	NC → PLC	101.7	1	1
Gantry leading axis	NC → PLC	101.6	1	0
Gantry grouping is synchronized	NC → PLC	101.5	X	
Gantry synchronization ready to start	NC → PLC	101.4	X	
Gantry warning limit exceeded	NC → PLC	101.3		X
Gantry trip limit exceeded	NC → PLC	101.2		X

Effect of axial interface signals on gantry axes

a) Axial interface signals from the PLC to the axis (PLC → NC)

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	
		Leading axis	Synchronous axis
Axis/spindle disable	1.3	On all axes in gantry grouping	No effect
Position measuring system 1/2	1.5 and 1.6	Axial [1]	Axial [1]
Controller enable	2.1	On all axes in gantry grouping [2]	
Delete distance to go (axial)	2.2	Axial	No effect
Clamping in progress	2.3	Axial	Axial
Reference point value 1-4	2.4 - 2.7	Axial	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	
2nd software limit switch plus/minus	12.2 and 12.3	Axial	Axial
Ramp-function generator fast stop (RFGFS)	20.1	On all axes in gantry grouping	
Select drive parameter set	21.0 - 21.2	Axial	Axial
Enable Pulses	21.7	Axial	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 the axis to the PLC (NC → 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)

9.7 Miscellaneous points regarding gantry axes

Manual traversing

It is not possible to traverse a synchronous axis directly by hand in JOG mode. Traverse commands entered via the traversing keys of the synchronous axis are ignored internally in the control. Rotation of the handwheel for the synchronous axis has no effect either.

Handwheel override

An overriding motion by means of the handwheel can only be applied to the guide axis in coupled axis mode. In this case, the synchronous axes traverse in synchronism with the guide axis.

DRF offset

A DRF offset can only be applied to the guide axis. In this case, the synchronous axes traverse in synchronism with the guide axis.

Programming in the part program

Only the guide axis of a gantry axis grouping may be programmed in the part program. An alarm is generated while programming a synchronous axis, even when a gantry axis grouping is released (MD37140 $MA_GANTRY_BREAK_UP = 1).

PLC or command axes

Only the guide axis of the gantry grouping can be traversed by the PLC using FC 18 or as a command axis by means of synchronized actions.

Further information

- Function Manual PLC
- Function Manual Synchronized Actions

PRESET

The PRESET function can only be applied to the guide axis. All axes in the gantry grouping are reevaluated internally in the control when PRESET is activated. The gantry axes 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 guide axis is known in several channels, its synchronous 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 interchange

All axes in the gantry grouping are released automatically in response to a RELEASE command (guide axis).

An axis interchange of the guide 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 interchange 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 synchronous axes:

MD30450 $MA_IS_CONCURRENT_POS_AX = 1 (Reset default: neutral axis/channel axis)

The state of the guide 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 guide axis and the synchronous axes. The same applies to the service display values in the "Diagnosis" operating area.

Software limit switch

The software limit switch monitor is processed for the guide axis only. If the guide 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 guide axis and synchronous 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 synchronous axes must be set like the guide axes from the control dynamics perspective. In contrast, the "coupled motion" function permits axes with different dynamic control response characteristics to be coupled.

Further information:
Coupled motion (Page 419)

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.

9.8 Examples

9.8.1 Creating a gantry grouping

Introduction

Setting up gantry grouping, referencing its axes, aligning possible offsets - and finally, synchronizing 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 shutdown limit when 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.

9.8.2 Setting the NC-PLC interface

Introduction

An automatic synchronization process when referencing axes must be disabled initially to prevent damaging/destroying 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 NC 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 NC 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)

9.8.3 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 NC has prepared axis 1 for synchronization and registers this at 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 $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 $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 NC has prepared axis 1 for synchronization and registers this at 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)

- Examine actual positions of machine. Case A or B might apply:

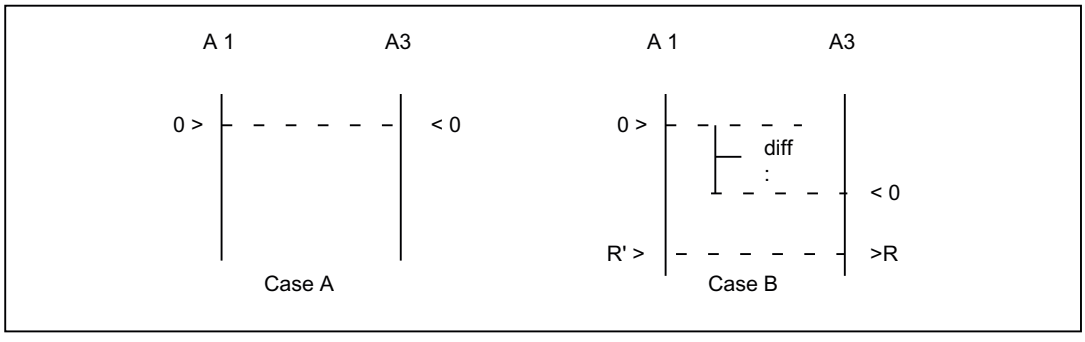


Figure 9-3 Possible results after referencing axis 1 (master axis)

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 1st step (see above).
- Start gantry synchronization. PLC sets:
  DB31, ... DBX29.4 = 1 (start synchronization of gantry)

9.8.4 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, ... DBX101.2 = 0 (gantry trip limit not exceeded)
  DB31, ... DBX101.3 = 0 (gantry warning limit not exceeded)
  DB31, ... DBX101.4 = 0 (gantry synchronization run not ready to start)
  DB31, ... DBX101.5 = 1 (gantry grouping is synchronized)
  DB31, ... DBX101.6 = 1 (gantry guide axis)
  DB31, ... DBX101.7 = 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

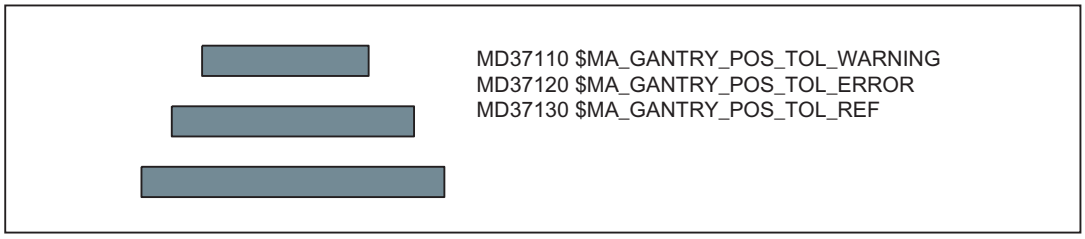
Values are entered in the following machine data:

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 shutdown limit when 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 interface bit "Gantry is synchronous" is not present (e.g. to be resynchronized after canceling alarms without gantry).

9.9 Data lists

9.9.1 Machine data

9.9.1.1 Axis/spindlespecific machine data

Number	Identifier: $MA_	Description
30300	IS_ROT_AX	Rotary axis
32200	POSCTRL_GAIN	K_V factor (servo gain)
32400	AX_JERK_ENABLE	Axial jerk limitation
32402	AX_JERK_MODE	Filter type for axial jerk limitation
32410	AX_JERK_TIME	Time constant for axial jerk filter
32420	JOG_AND_POS_JERK_ENABLE	Initial setting for axial jerk limitation
32430	JOG_AND_POS_MAX_JERK	Axial jerk
32610	VELO_FFW_WEIGHT	Feedforward control factor for speed feedforward control
32620	FFW_MODE	Feedforward control type

Number	Identifier: $MA_	Description
32650	AX_INERTIA	Moment of inertia for torque feedforward control
32800	EQUIV_CURRCTRL_TIME	Equivalent time constant, current control loop for feed-forward control
32810	EQUIV_SPEEDCTRL_TIME	Equivalent time constant, speed control loop for feed forward 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	Shutdown velocity
34070	REFP_VELO_POS	Velocity when approaching the reference point
34080	REFP_MOVE_DIST	Reference point approach distance
34090	REFP_MOVE_DIST_CORR	Reference point offset
34100	REFP_SET_POS	Reference point value
34110	REFP_CYCLE_NR	Axis sequence for channel-specific referencing
34330	REFP_STOP_AT_ABS_MARKER	Distance-coded linear measuring system without destination point
36012	STOP_LIMIT_FACTOR	Factor, exact stop coarse/fine - and zero speed
36030	STANDSTILL_POS_TOL	Standstill 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 shutdown limit
37130	GANTRY_POS_TOL_REF	Gantry shutdown limit when referencing
37140	GANTRY_BREAK_UP	Break up gantry axis grouping

R1: Referencing 10

10.1 Brief Description

Function

When referencing a machine axis, the coordinate system of the machine axis is synchronized with the coordinate system of the machine. The machine axis is traversed to the machine zero and then the actual position of the machine axis is set to zero.

If the machine zero cannot be directly approached as a result of the machine design, then a reference point is defined in the traversing range of the machine axis, which then used to synchronize the machine axis. Its position with reference to the machine zero must be known. When referencing, the actual machine axis position is set to this value

Measuring systems and referencing methods

When referencing, machine axes can be synchronized with the following measuring systems and referencing types:

- Measuring systems
  - Incremental rotary measuring system with at least one zero mark
  - Incremental linear measuring system
  - Rotary measuring system with distancecoded reference marks (supplied by Heidenhain)
  - Linear measuring system with distancecoded reference marks (supplied by Heidenhain)
  - Absolute rotary measuring system
  - Absolute linear measuring system
- Referencing methods
  - Referencing with incremental measuring systems with proximity switch and one-edge and two-edge detection
  - Referencing with incremental measuring systems with replacement of homing cam with proximity switch
  - Referencing with incremental measuring systems with proximity switch with configured approach velocity for spindle applications
  - Referencing with measuring systems with distancecoded reference marks by overtravelling 2 or 4 zero marks
  - Referencing of passive measuring systems using measuring system adjustment
  - Referencing in follow-up mode
  - Referencing with cam switch at the drive

Start

Referencing the machine axis can be manually started, or from the part program:

- Manual: Operation mode JOG and MDI, machine function REF
- Part program: Command `G74`

10.2 Axisspecific referencing

For axis-specific referencing using the reference point approach, the operation must be individually started for each machine axis to be referenced.

Selecting mode and machine function

Before starting reference point approach of the machine axes, you must first place the relevant mode group in JOG or MDI mode:

DB11, ... DBX0.2 (active JOG mode)

DB11, ... DBX0.1 (active MDI mode)

Then machine function REF (reference point approach) must be selected:

DB11, ... DBX1.2 (REF machine function)

Start of reference point approach

In axis-specific reference point approach, each machine axis must be started individually.

Reference point approach is started with the axis-specific traversing keys:

DB31, ... DBX4.6 (Traversing key minus)

DB31, ... DBX4.7 (Traversing key minus)

Direction enable

To avoid operator errors, the direction release must be parameterized. The direction enable specifies which traversing key starts the reference point approach:

MD34010 $MA_REFP_CAM_DIR_IS_MINUS = <value>

<Value>	Meaning
0	Reference point approach in plus direction
1	Reference point approach in minus direction

Jog mode

The following machine data element can be used to specify whether reference point approach is completed when the direction key is pressed once or whether the operator is required to keep the direction key pressed (jogging) for safety reasons:

MD11300 $MN_JOG_INC_MODE_LEVELTRIGGRD (INC and REF in jog mode)

If the machine operator releases the direction key, the machine axis is decelerated to zero speed. Reference point approach is not aborted. Reference point approach is continued the next time the direction key is pressed.

Referencing status

The referencing status of the machine axis is reset with the start of the reference point approach:

DB31, ... DBX60.4 (referenced/synchronized 1)

DB31, ... DBX60.5 (referenced/synchronized 2)

DB21, ... DBX36.2 (all axes with obligatory reference point are referenced)

Distance-coded measuring systems

In distance-coded measuring systems, reference point approach can be started with any traversing key.

Sequence

The machine operator or machine manufacturer (via the PLC user program) is responsible for ensuring that the machine axes are referenced in the proper order.

- Machine operator
  The machine axes must be started by the machine operator in the specified order.
- Machine manufacturer
  The PLC user program of the machine manufacturer allows machine axes to be started only in the proper order.

Simultaneous reference point approach of several machine axes

Several machine axes can be referenced simultaneously, depending on the control:

Completion of reference point approach

Acknowledgment that reference point approach of a machine axis has been successfully completed is given by setting the referencing status:

DB31, ... DBX60.4 (referenced/synchronized 1)

DB31, ... DBX60.5 (referenced/synchronized 2)

Cancellation of reference point approach

In axis-specific reference point approach, the machine axis is traversed in the channel that was assigned as the master channel of the machine axis.

MD30550 $MA_ AXCONF_ASSIGN_MASTER_CHAN

For aborting the reference point approach, either mode group reset or channel reset for the master channel of the machine axis must be activated:

DB11, ... DBX0.7 (mode group reset)

DB21, ... DBX7.7 (channel reset)

All machine axes that have not yet successfully completed reference point approach when the action is cancelled remain in status "Not referenced":

DB31, ... DBX60.4 (referenced/synchronized 1)

DB31, ... DBX60.5 (referenced/synchronized 2)

10.3 Channelspecific referencing

For channel-specific referencing, all machine axes of the channel are referenced in the parameterized sequence when reference point approach is initiated.

Selecting mode and machine function

Before starting the reference point approach of the machine axes, the associated mode group must be switched into the JOG or MDI operating mode:

DB11, ... DBX0.2 (active JOG mode)

DB11, ... DBX0.1 (active MDI mode)

Then machine function REF (reference point approach) must be selected:

DB11, ... DBX1.2 (REF machine function)

Parameterizing the axis sequence

The following machine data element is used to specify the sequence in which the machine axes of the channel are referenced:

MD34110 $MA_REFP_CYCLE_NR = <number>

<number>	Meaning
-1	The machine axis does not have to be referenced for NC START in the channel.
0	The machine axis does not participate in channel-specific reference point approach.
1 - 15	Sequence number in channel-specific reference point approach.

The machine axes are referenced in ascending order of numbers.

Machine axes with the same number will be referenced simultaneously.

Simultaneous reference point approach of several machine axes

Several machine axes can be referenced simultaneously, depending on the control:

Start of reference point approach

Channel-specific reference point approach is started with:

DB21, ... DBX1.0 (activate referencing)

The status of channel-specific reference point approach is indicated by the channel with:

DB21, ... DBX33.0 (referencing active)

Referencing status

The referencing status of the machine axis is reset with the start of the reference point approach:

DB31, ... DBX60.4 (referenced/synchronized 1)

DB31, ... DBX60.5 (referenced/synchronized 2)

Completion of reference point approach

As soon as channel-specific reference approach has been successfully completed for all machine axes involved, this is acknowleged with:

DB21, ... DBX36.2 (all axes with obligatory reference point are referenced)

Cancellation of reference point approach

In channel-specific reference point approach the machine axis is traversed in the channel to which that axis is currently assigned as channel axis.

For aborting the reference point approach either mode group reset or channel reset for the corresponding channel must be activated:

DB11, ... DBX0.7 (mode group reset)

DB21, ... DBX7.7 (channel reset)

All machine axes for which the reference point approach is not yet successfully completed when the action is cancelled remain in status "Not referenced":

DB31, ... DBX60.4 (referenced/synchronized 1)

DB31, ... DBX60.5 (referenced/synchronized 2)

10.4 Reference point appraoch from part program (G74)

Function

With command `G74`, machine axes can be referenced from a part program or synchronized action either for the first time or again.

Referencing must be repeated, for example, after:

- Actual value offset PRESETON
  Further information:
  Programming Guide Advanced; Coordinate transformations (frames) > Preset offset with PRESETON
- Function Manual Monitoring and Compensating, parking a machine axis
- Exceeding the encoder limit frequency of the position measuring system

Programming

Syntax

```
G74 <machine axis> {<machine axis>}
```

Meaning

G74:	Command for referencing machine axes	
	Alone in the block:	Yes
	Effective-ness:	Non-modal
<machine axis>:	The name of a machine axis must be specified (MD10000 $MN_AXCONF_MACHAX_NAME_TAB). The machine axis must be a channel axis of the channel in which the part program or the synchronized action is executed.	

Reset behavior

Mode group reset or channel reset aborts the reference point approach for all programmed machine axes:

- DB11, ... DBX0.7 (mode group reset)
- DB21, ... DBX7.7 (channel reset)

All machine axes for which the reference point approach is not yet successfully completed when the action is canceled remain in status "Not referenced":

- DB31, ... DBX60.4 (referenced/synchronized 1) == 0
- DB31, ... DBX60.5 (referenced/synchronized 2) == 0

10.5 Referencing with incremental measurement systems

10.5.1 Hardware signals

Depending on the machine design and the properties of the incremental measuring system used, different hardware signals must be connected.

Reference cam

- Connection
  The reference cam signal can be connected to a digital input of an external PLC I/O module or to a fast input on the NCU X142 interface.
- NC/PLC interface signal
  The reference cam signal must be transferred from the PLC user program to the axial NC/PLC interface:
  DB31, ... DBX12.7 (deceleration of reference point approach)

Zero mark selection

If during the reference point approach of the axis or spindle several zero marks of the measuring system are detected (e.g. measuring gear between the motor and encoder), then the specific zero mark must be selected with an additional proximity switch signal.

- Connection
  The proximity switch must be connected to a fast digital input on the NCU X122 or X132 interface.
- Activation
  In order that the proximity switch signal is evaluated, the digital input to which the proximity switch is connected must be selected in drive parameter p0493 for the axis/spindle.

Equivalent zero mark

If the used measuring system does not provide a zero mark signal, an equivalent zero mark can be created via a proximity switch signal.

- Connection
  The proximity switch must be connected to a fast digital input on the NCU X122 or X132 interface.
- Activation
  In order that the proximity switch signal is evaluated, the digital input to which the proximity switch is connected must be selected in the p0494 or p0495 drive parameter for the axis/spindle.

Overview

Signal	Connection: Digital input via	Set
Reference cam	Ext. PLC I/O module or NCU: X142	PLC user program: DB31, ... DBX12.7
Zero mark selection	NCU: X122 or X132	Drive parameter: p0493
External zero mark or equivalent zero mark	NCU: X122 or X132	Drive parameter: p0494 or p0495

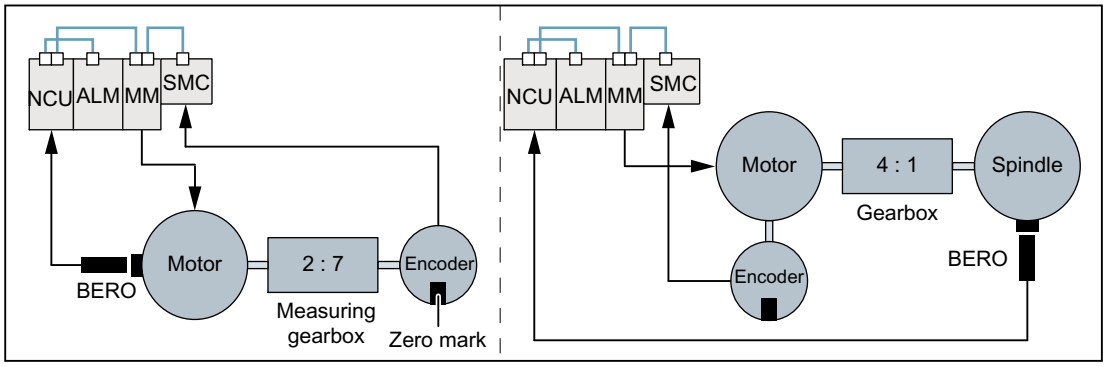
Additional information

- NCU interfaces: Equipment Manual
  Connect > Digital inputs/outputs
- Drive parameter: SINAMICS S120/S150 List Manual

10.5.2 Zero mark selection

Function

Referencing of incremental measuring systems is based on the unique position of the encoder zero mark relative to the overall traversing range of the machine axis. If because of machine-specific conditions, several encoder zero marks are detected in the traversing range of the machine axis (for examples, see figure below), a proximity switch must be mounted on the machine for clear determination of the reference point. The position of the reference point is then derived from the combination of contact-less proximity switch signal and encoder zero mark.



BERO Contact-less proximity switch

Figure 10-1 Measuring gear between the motor and encoder or reduction gear between the motor and spindle

Parameterization

NC: Referencing mode

"Referencing of incremental, rotary or linear measuring systems: Zero pulse on the encoder track" should be parameterized as referencing mode:

MD34200 $MA_ENC_REFP_MODE[<axis>] = 1

Drive: Zero mark selection

The digital input on the NCU interface to which the proximity switch is connected must be set in parameter p0493.

Note

Zero mark selection

The processing of the contact-less proximity switch signal is performed exclusively in the drive. Connection and parameterization, see Section "Hardware signals (Page 296)".

10.5.3 Time sequence

Reference point approach with incremental measuring systems can be divided into three phases:

- Phase 1: "Phase 1: Traversing to the reference cam (Page 300)"
- Phase 2: "Phase 2: Synchronization with the zero mark (Page 302)"
- Phase 3: "Phase 3: Traversing to the reference point (Page 307)"

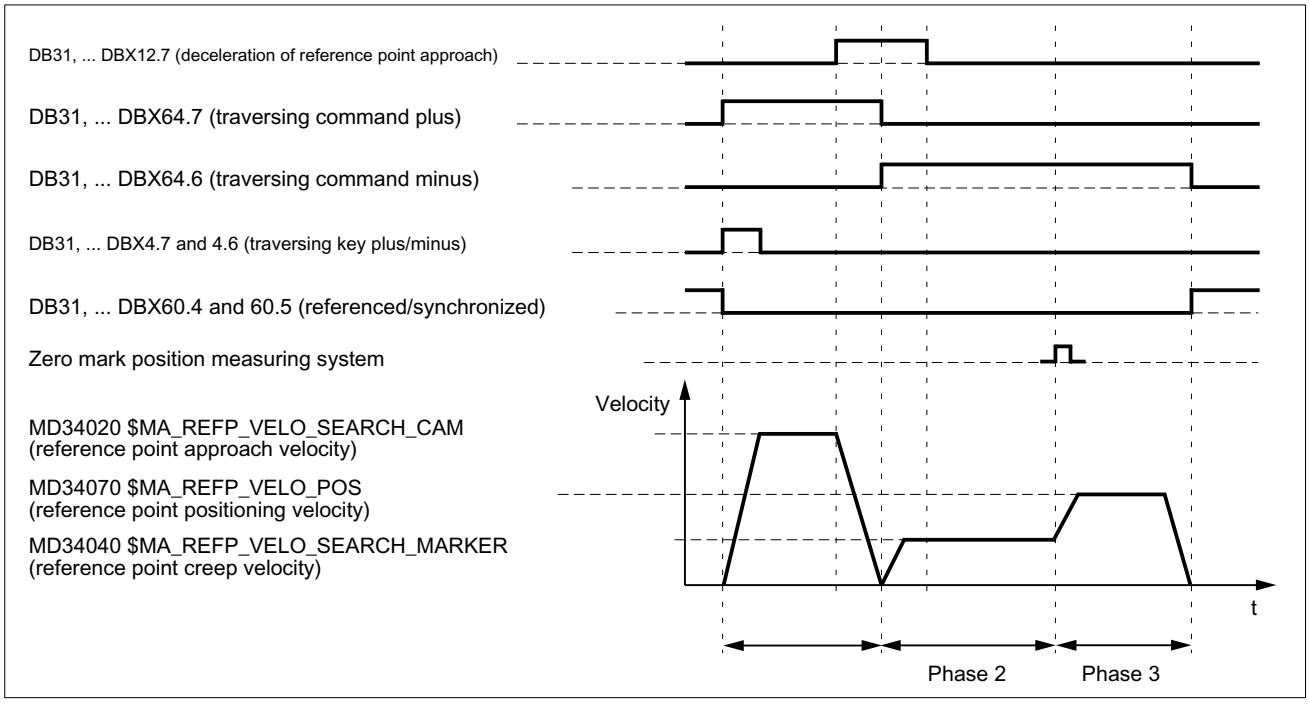


Figure 10-2 Time sequence when referencing with incremental measuring systems (example)

10.5.4 Phase 1: Traversing to the reference cam

Phase 1: Graphic representation

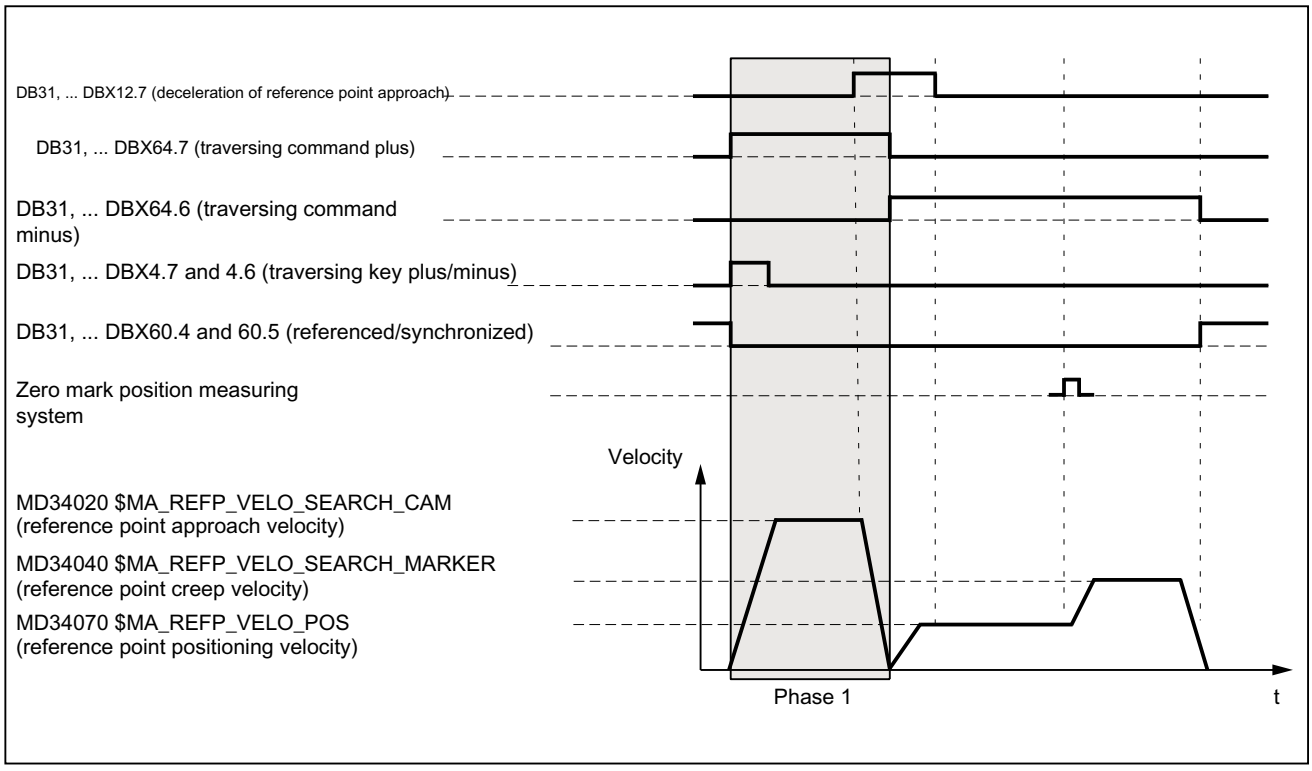


Figure 10-3 Phase 1: Traversing to the reference cam

Phase 1: Start

To start the reference point approach, see Sections "Axisspecific referencing (Page 292)" and "Axisspecific referencing (Page 292)".

Phase 1: Sequence

In Phase 1, depending on the position of the machine axis with reference to the reference cam, we distinguish between three cases:

1. The machine axis is positioned before the reference cam
2. The machine axis is positioned on the reference cam
3. The machine axis has no reference cam

Case 1: The machine axis is positioned before the reference cam

After the start of reference point approach, the machine axis is accelerated in the parameterized direction and to the parameterized reference point approach velocity :

- MD34010 $MA_REFP_CAM_DIR_IS_MINUS (reference point approach in minus direction)
- MD34020 $MA_REFP_VELO_SEARCH_CAM (reference point approach velocity)

The reaching of the reference cam must detected by querying a digital input in the PLC user program and communicated to the NC via the following interface signal:

DB31, ... DBX12.7 = 1 (reference point approach deceleration)

With detection of the NC/PLC interface signal, the machine axis is decelerated to zero speed. Whereby at least the distance s_{min} is traversed. This ensures that the machine axis leaves the reference cam in Phase 2 with the parameterized reference point creep velocity.

$$s_{min} = \frac{(\text{MD34040 \$MA_REFP_VELO_SEARCH_MARKER})2}{2 * \text{MD32300 \$MA_MAX_AX_ACCEL}}$$

Phase 1 is now complete. Reference point approach is continued with Phase 2.

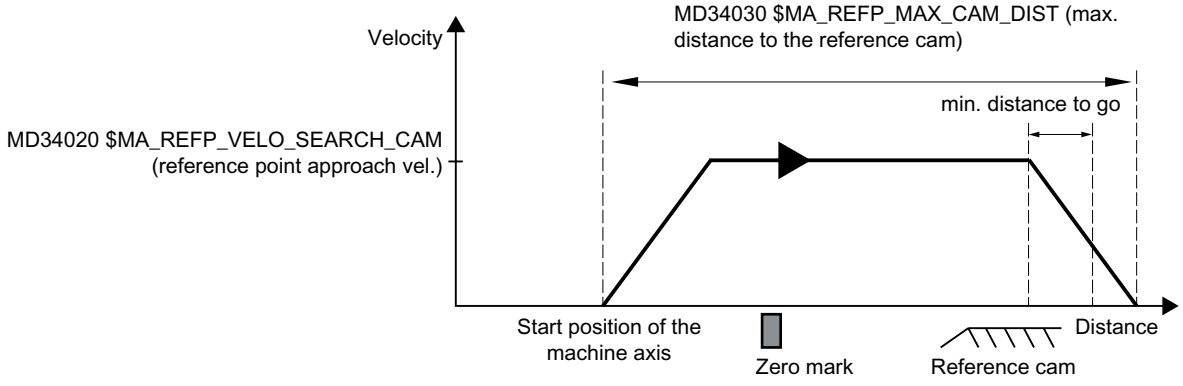


Figure 10-4 Minimum distance for deceleration

Case 2: The machine axis is positioned on the reference cam

The machine axis remains at its starting position.
Phase 1 is now complete. Reference point approach is continued with Phase 2.

Case 3: The machine axis has no reference cam

Machine axes without reference cams remain at their starting position.

These include, for example:

- Machine axes that only have one zero mark along their entire traversing range
- Rotary axes that only have one zero mark per revolution

Zero must be entered in the following machine data for machine axes without a reference cam:

MD34000 $MA_REFP_CAM_IS_ACTIVE = 0 (axis with reference cam)

Phase 1 is now complete. Reference point approach is continued with Phase 2.

Phase 1: Properties

- Feedrate override is active.
- Feed stop (channel-specific and axis-specific) is active.
- NC stop and NC start are active.
- The machine axis is stopped if the reference cam is not reached within the parameterized maximum distance:
  MD34030 $MA_REFP_MAX_CAM_DIST (max. distance to the reference cam)

See also

Channelspecific referencing (Page 294)

10.5.5 Phase 2: Synchronization with the zero mark

Phase 2: Graphic representation

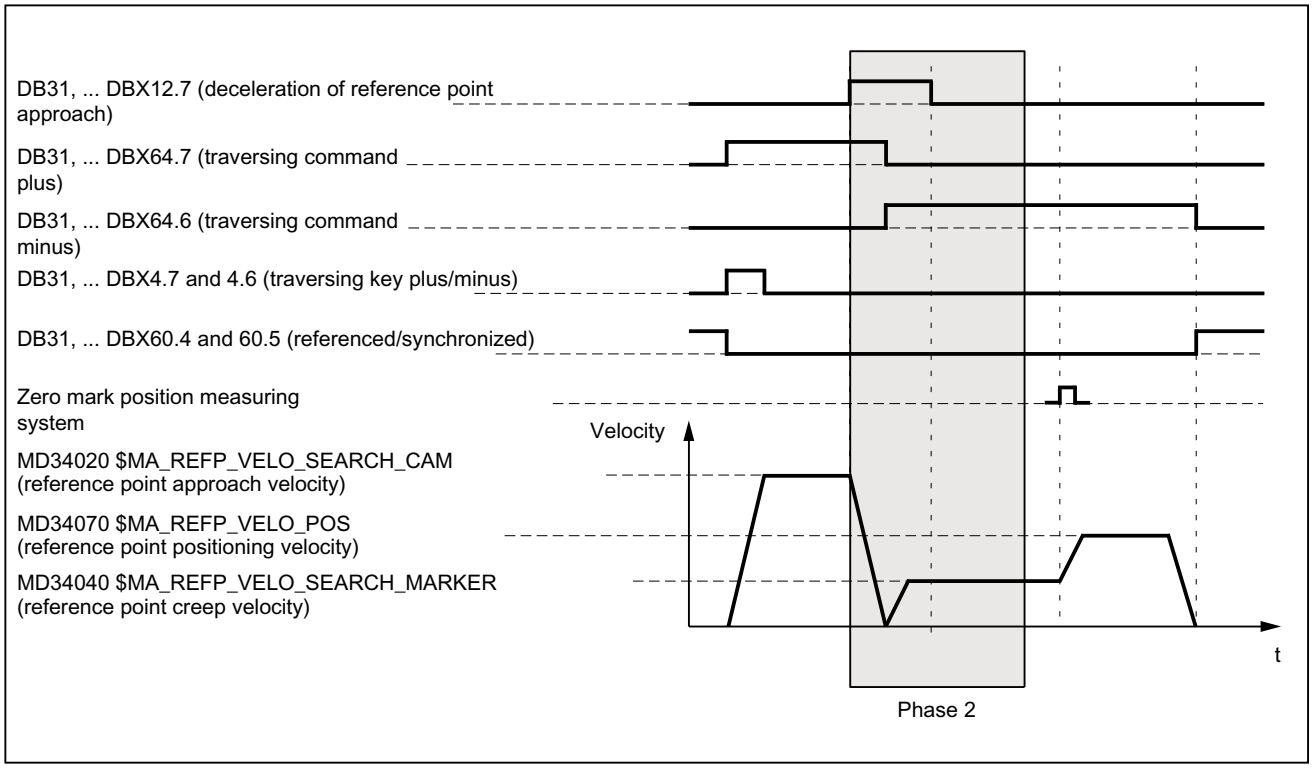


Figure 10-5 Phase 2: Synchronization with the zero mark

Phase 2: Start

Phase 2 is automatically started when Phase 1 has been completed without an alarm.

Initial situation:

The machine axis is positioned on the reference cam.

Zero mark search direction:

The direction of the zero mark search results from the settings in the machine data:

- MD34010 $MA_REFP_CAM_DIR_IS_MINUS (reference point approach in minus direction)
- MD34050 $MA_REFP_SEARCH_MARKER_REVERSE (direction reversal on reference cam)

Phase 2: Sequence

The synchronization in Phase 2 can be performed via the falling or rising edge of the reference cam. The parameterization is performed via:

MD34050 $MA_REFP_SEARCH_MARKER_REVERSE[<axis>] = <value>

Value	Meaning
0	Synchronization with falling reference cam edge
1	Synchronization with rising reference cam edge

Note

If the actual velocity of the machine axis at approach of the reference cam has not yet reached the target velocity of Phase 2 within the parameterized tolerance limits, Phase 1 will be restarted. This will be the case, for example, if the machine axis is positioned on the reference cam when reference point approach is started.

MD35150 $MA_SPIND_DES_VELO_TOL (spindle speed tolerance)

Case 1: Synchronization with falling reference cam edge

During synchronization with falling reference cam edge, the machine axis accelerates to the parameterized reference point creep velocity opposite to the parameterized reference point approach direction (traversing direction of Phase 1)

After leaving the reference cam, the machine axis waits for the next encoder zero mark: DB31, ... DBX12.7 == 0

As soon as the encoder zero mark is detected, Phase 2 comes to an end. The machine axis continues at constant velocity and reference point approach is continued with Phase 3.

- MD34040 $MA_REFP_VELO_SEARCH_MARKER (reference point creep velocity)

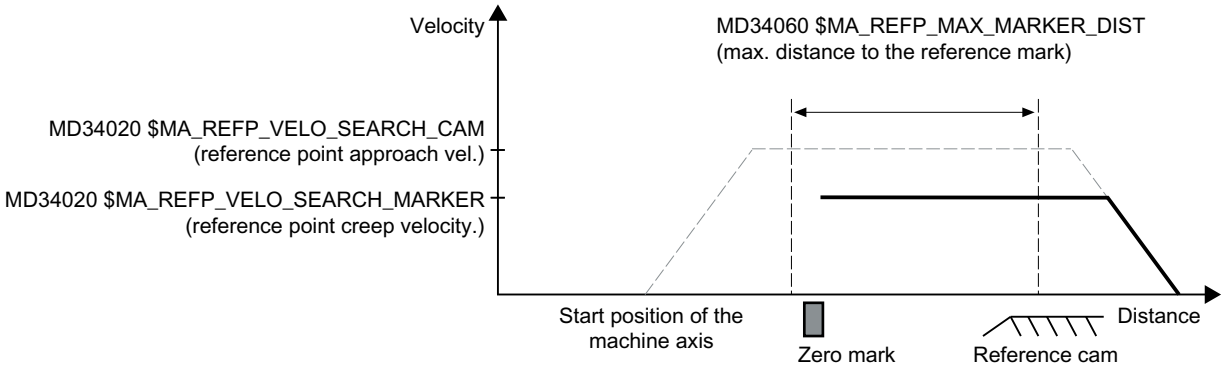


Figure 10-6 Synchronization with falling reference cam edge

Case 2: Synchronization with rising reference cam edge

During synchronization with rising reference cam signal edge, the machine axis accelerates to the parameterized reference point approach velocity against the parameterized reference point approach direction (traversing direction of the Phase 1):

- MD34020 $MA_REFP_VELO_SEARCH_CAM (reference point approach velocity)
- MD34010 $MA_REFP_CAM_DIR_IS_MINUS (reference point approach in minus direction)

After leaving the reference cam, the machine axis decelerated to standstill:
DB31, ... DBX12.7 == 0

The machine axis then travels back to the reference cam at the parameterized reference point creep velocity:

MD34040 $MA_REFP_VELO_SEARCH_MARKER (reference point creep velocity)

After reaching the reference cam (DB31, ... DBX12.7 = 1), the machine axis waits for the next encoder zero mark.

As soon as the encoder zero mark is detected, Phase 2 comes to an end. The machine axis continues at constant velocity and reference point approach is continued with Phase 3.

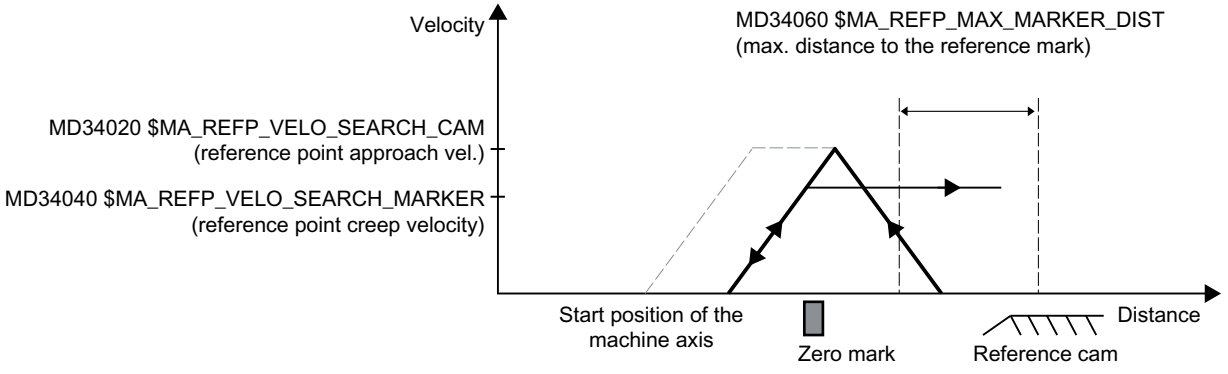


Figure 10-7 Synchronization with rising reference cam edge

Electronic reference cam shift

The electronic reference cam shift is used to compensate for expansions of the reference cam caused by temperature so that synchronization is always to the same encoder zero mark:

MD34092 $MA_ REFP_CAM_SHIFT (electronic reference cam shift for incremental measuring systems with equidistant zero marks)

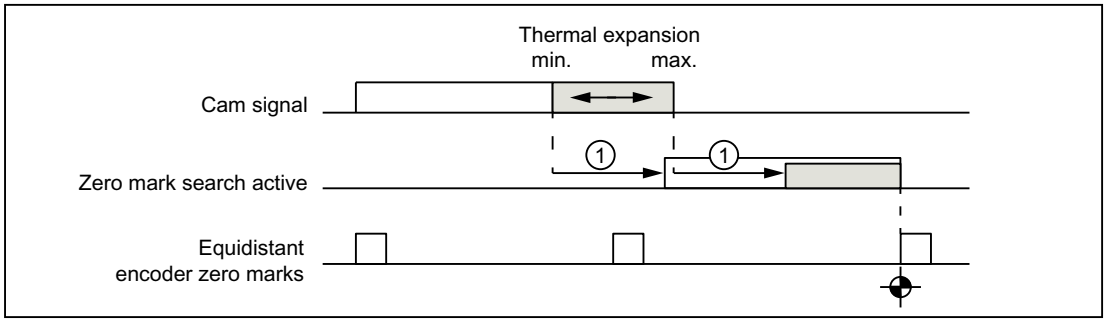
With the electronic reference cam shift, synchronization is not performed immediately to the next encoder zero mark after detection of the reference cam edge, but only after the parameterized offset distance has been traversed.

Due to the determination of the distance traversed in the interpolator clock cycle since the detection of the reference cam edge, the effective shift distance is s_{shift}:

s_{shift_min} = MD34092 $MA_ REFP_CAM_SHIFT

s_{shift_max} = MD34092 $MA_ REFP_CAM_SHIFT +
MD34040 $MA_REFP_VELO_SEARCH_MARKER * interpolator clock cycle

The electronic reference cam shift acts in the direction of zero mark search.



① Reference cam shift

Figure 10-8 Electronic reference cam shift

Requirement

The electronic reference cam shift is only active for machine axes with reference cam:

MD34000 $MA_REFP_CAM_IS_ACTIVE == 1

Reference cam adjustment

Encoder with equidistant zero marks

Always ensure that the reference cam of encoders that supply zero marks at equidistances is accurately adjusted so that the correct zero mark is always detected during reference point approach.

Dynamic response

The following factors influence the dynamic response from the arrival of the reference cam to the machine up to the detection of reference cam signals transferred from the PLC user program to the NC:

- Switching accuracy of the reference cam switch
- Delay of the reference cam switch (NC contact)
- Delay at the PLC input
- PLC cycle time

- Cycle time for updating the NC/PLC interface
- Interpolator clock cycle
- Position control cycle

Notes on setting

- Reference cam
  Aligning the signal edge of the reference cam directly between two zero marks has proven to be the most practical method.
- Electronic reference cam shift

  ⚠ WARNING

  Risk of collision

  If the reference cam adjustment is faulty or inaccurate, an incorrect zero mark can be evaluated. The controller then calculates an incorrect machine zero. As a result, the machine axis will approach the wrong positions. Software limit switches, protected areas and working area limitations act on incorrect positions and are therefore incapable of protecting the machine. The path difference is +/- of the path covered by the machine axis between two zero marks.

  Information needed for parameterizing the electronic reference cam shift is to be found in the read-only machine data:
  MD34093 $MA_REFP_CAM_MARKER_DIST (distance between reference cam/reference mark)
  The indicated value is equivalent to the distance between departure from the reference cam and detection of the reference mark. If the values are too small, there is a risk that the determination of the reference point will be non-deterministic, due to temperature effects or fluctuations in the operating time of the cam signal.

Phase 2: Properties

- Feedrate override is **not** active.
  Traversing is performed internally with feedrate override = 100%.
  If a feedrate override of 0% is specified, an abort occurs.
- Feed stop (channel-specific and axis-specific) is active.
- NC stop and NC start are **not** active.
- If the machine axis does not arrive at Phase 2 within the parameterized distance of the reference mark (encoder zero mark), the machine axis will be stopped:
  MD34060 $MA_REFP_MAX_ MARKER_DIST (max. distance to the reference mark)

10.5.6 Phase 3: Traversing to the reference point

Phase 3: Graphic representation

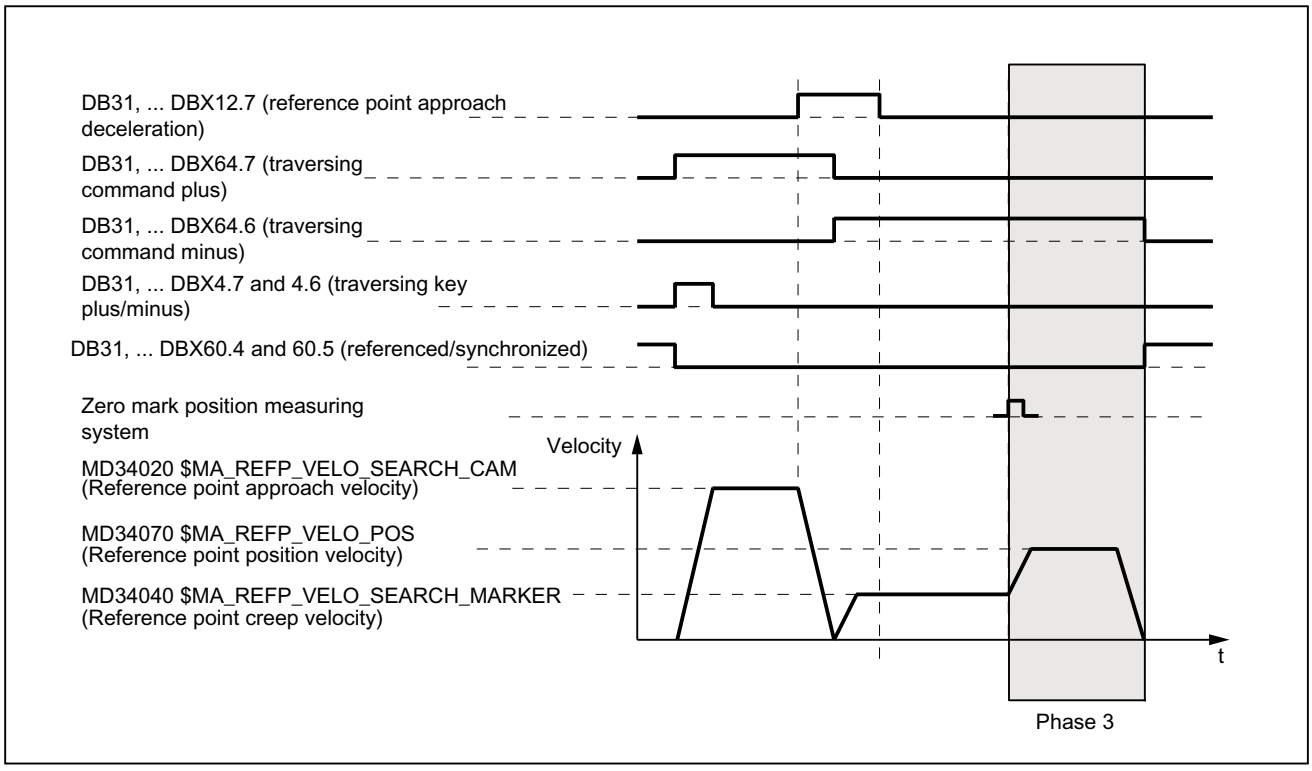


Figure 10-9 Phase 3: Traversing to the reference point

Phase 3: Start

At the end of Phase 2 the machine axis travels at reference point creep velocity. Therefore, as soon as Phase 2 is completed successfully without an alarm, Phase 3 is started without interruption.

Initial situation

The encoder zero mark has been detected.

Phase 3: Sequence

The machine axis moves at the assigned reference point positioning velocity:
MD34070 $MA_REFP_VELO_POS (reference point positioning velocity)
from the encoder zero mark detected in Phase 2 to the reference point.

The path s_{ref} to be covered is calculated from the sum of the reference point distance plus reference point offset:

MD34080 $MA_REFP_MOVE_DIST (reference point distance)

MD34090 $MA_REFP_MOVE_DIST_CORR (reference point offset)

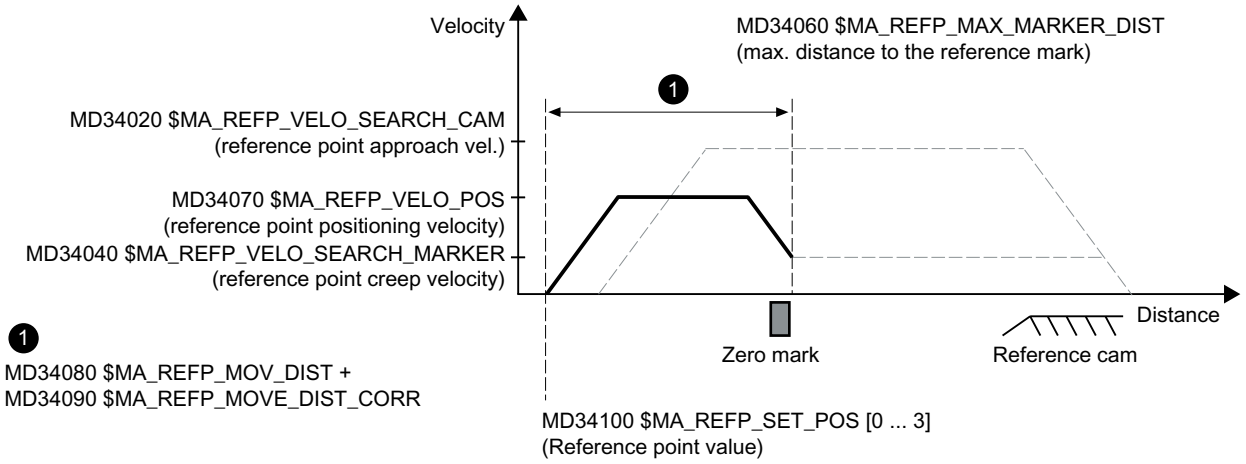


Figure 10-10 Reference point position

When the reference point is reached, the machine axis is stopped and the actual-value system is synchronized with the reference point value n specified by the NC/PLC interface.

MD34100 $MA_ REFP_SET_POS[<n>] (reference point value)

The selection of the reference point value is performed via the NC/PLC interface:

DB31, ... DBX2.4 ... 7 (reference point value 1 ... 4)

The actual-value system is synchronized to the reference point value that was selected at the time the reference cam was reached in Phase 1 (DB31, ... DBX12.7 == 1).

The machine axis is now referenced. The interface signal is set as feedback to the PLC user program, depending on the active measuring system:

DB31, ... DBX60.4/5 (referenced/synchronized 1/2) = 1

Features of Phase 3

- Feedrate override is active.
- Feed stop (channel-specific and axis-specific) is active.
- NC stop and NC start are active.

Special feature of Phase 3

In the following cases, the machine axis stops first after detection of the zero mark and then traverses back to the reference point:

- Because of the reference point positioning velocity, the sum of reference point distance and reference point offset is less than the required braking distance:
  MD34080 + MD34090 < "required braking distance due to MD34070"
- The reference point is located, opposite to the current travel direction, "behind" the reference cam.

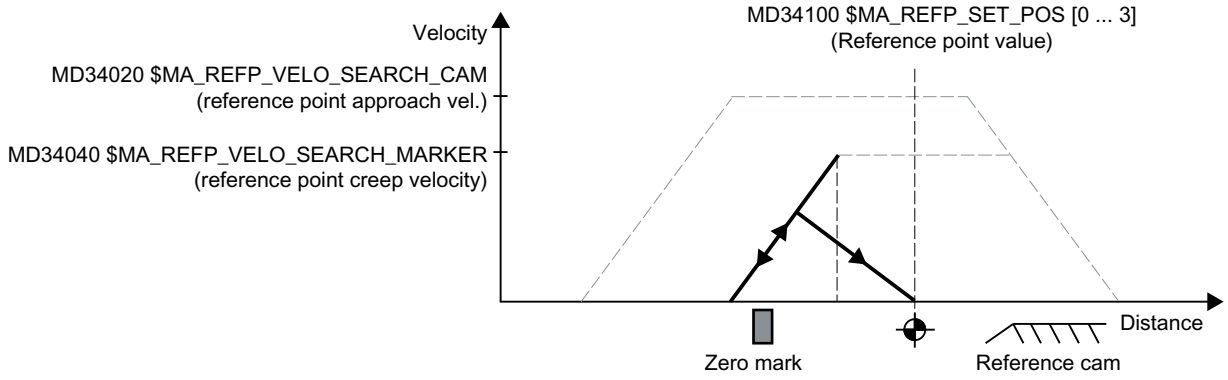


Figure 10-11 Reference point distance plus reference point offset less than braking distance

10.6 Referencing with distance-coded reference marks

10.6.1 General overview

Distancecoded reference marks

Measuring systems with distance-coded reference marks consist of two parallel scale tracks:

- Incremental grating
- Reference mark track

The distance between any two consecutive reference marks is defined. This makes it possible to determine the absolute position of the machine axis when two consecutive reference marks are crossed. For example, if the distance between the reference marks is approx. 10 mm, a traverse path of approx. 20 mm is all that is required to reference the machine axis.

Referencing can be performed from any axis position in the positive or negative direction (exception: end of travel range).

10.6.2 Basic parameter assignment

Linear measuring systems.

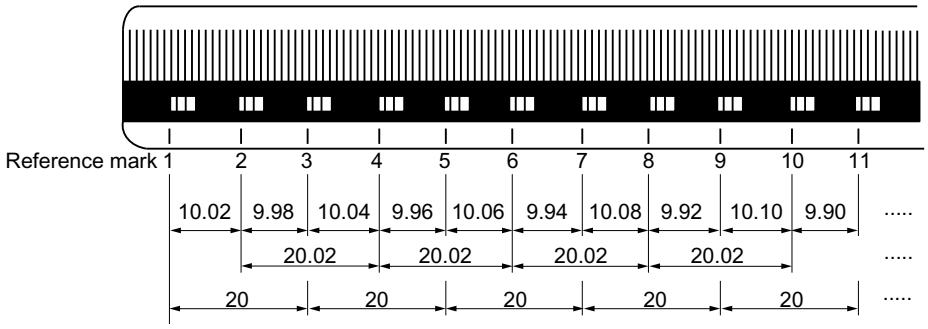


Figure 10-12 Glass measuring scale with distance-coded reference marks, grid spacing: 20 mm

The following data must be set to parameterize linear measuring systems:

- The absolute offset between the machine zero point and the position of the first reference mark of the linear measuring system:
  MD34090 $MA_REFP_MOVE_DIST_CORR (reference point/absolute offset)
  See also below: Determining the absolute offset
- Orientation of the length measuring system (equidirectional or inverse) relative to the machine system coordinate system:
  MD34320 $MA_ENC_INVERS (length measuring system inverse to the machine system)

Rotary measuring system

For rotary measuring systems, the same applies as for linear measuring systems (see above).

Determining the absolute offset

The following procedure is recommended for the determination of the absolute offset between the machine zero point and the position of the first reference mark of a machine axis:

1. Enter the value zero for the absolute offset:
   MD34090 $MA_REFP_MOVE_DIST_CORR = 0
2. Perform reference point approach.
   Note: The reference point should be approached at a point in the machine where the exact position of the machine axis relative to machine zero can be determined easily (using a laser interferometer, for example).
3. Read the displayed actual position of the machine axis in the machine coordinate system MCS from the user interface, e.g. SINUMERIK Integrate.

4. Measure the actual position of the machine axes referred to the machine zero point.
5. Calculate the absolute offset and enter in MD34090.
   The absolute offset is calculated with respect to the machine coordinate system and depending on the orientation of the measuring system as:

Orientation of the measuring system	Absolute offset
Equidirectional	(Measured position) **+** (displayed actual position)
Opposite direction	(Measured position) **-** (displayed actual position)

⚠ WARNING

Reference point deviation

After determining the absolute offset and the entry in MD34090, the reference point traversing for the machine axis must be carried out once more.

Referencing methods

Referencing with distance-coded reference marks can be performed in one of two ways:

- Evaluation of **two** consecutive reference marks:
  MD34200 $MA_ENC_REFP_MODE = 3
  Advantage:
  - Short travel path
- Evaluation of **four** consecutive reference marks:
  MD34200 $MA_ENC_REFP_MODE = 8
  Advantage:
  - Plausibility check by NC is possible
  - Increase in reliability of referencing result

10.6.3 Time sequence

Time sequence

Referencing with distance-coded reference marks can be divided into two phases:

- Phase 1: Travel across the reference marks with synchronization
- Phase 2: Travel to a fixed destination point

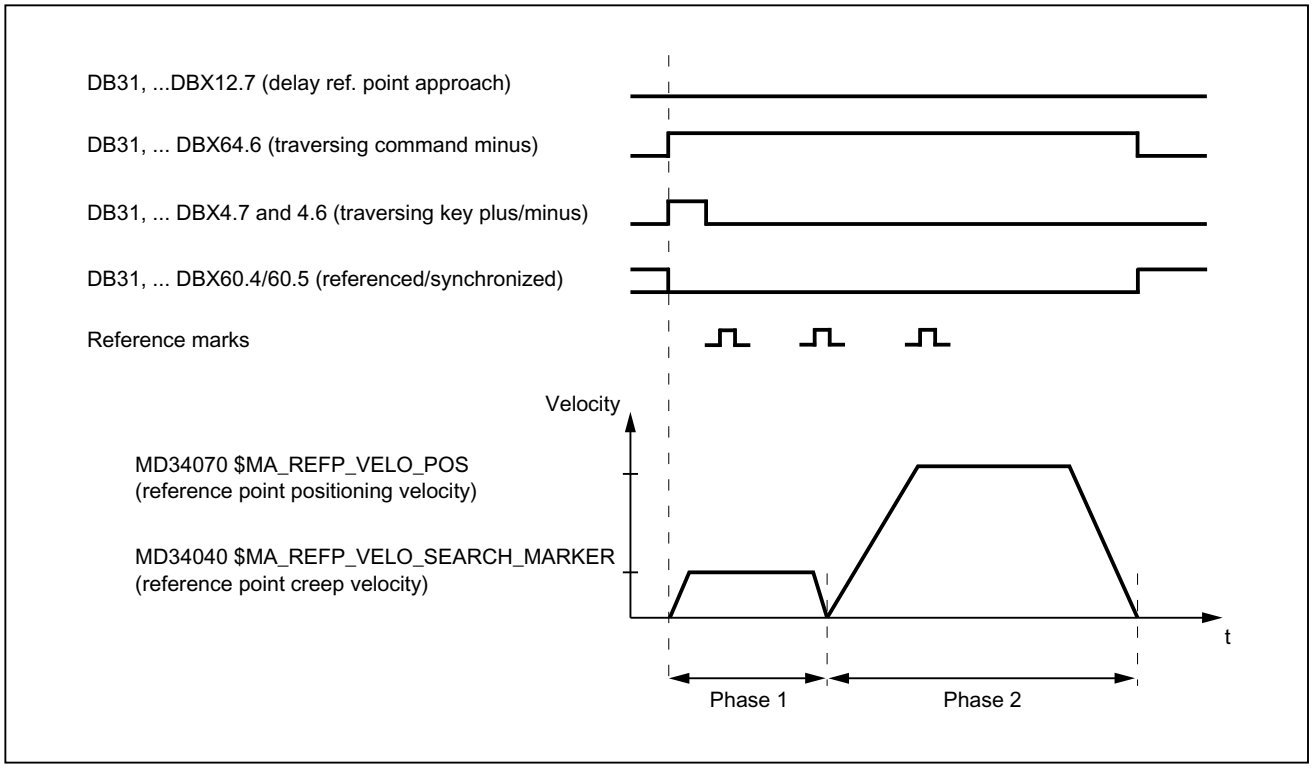


Figure 10-13 Distance-coded reference marks

10.6.4 Phase 1: Travel across the reference marks with synchronization

Phase 1: Start

To start the reference point approach, see Sections "Axisspecific referencing (Page 292)" and "Channelspecific referencing (Page 294)".

Reference cam

In measuring systems with distance-coded reference marks, reference cams are not required for the actual referencing action. For functional reasons, however, a reference cam is required for channel-specific reference point approach and reference point approach from the part program (G74) before the traversing range end of the machine axis.

Phase 1: Sequence

Sequence without contact witha reference cam

Once the reference point approaching process is started, the machine axis accelerates to the reference point shutdown speed set by means of parameter assignment:

MD34040 $MA_REFP_VELO_SEARCH_MARKER (reference point creep velocity)

Once the number of reference marks set by means of parameter assignment has been crossed, the machine axis is stopped again and the actual value system of the machine axis is synchronized to the absolute position calculated by the NC.

Sequence when starting from the reference cam

If the machine axis is at the reference cam at the start of the reference point traversing, it accelerates to the parameterized reference point creep velocity against the parameterized reference point approach direction:

MD34040 $MA_REFP_VELO_SEARCH_MARKER (reference point creep velocity)

MD34010 $MA_CAM_DIR_IS_MINUS (reference point approach in minus direction)

That ensures that the machine axis does not reach the travel range limit before it has crossed the parameterized number of reference marks.

Once the number of reference marks set by means of parameter assignment has been crossed, the machine axis is stopped again and the actual value system of the machine axis is synchronized to the absolute position calculated by the NC.

Sequence when contact is made with reference cam during referencing

Once the reference point approaching process is started, the machine axis accelerates to the reference point shutdown speed set by means of parameter assignment:

MD34040 $MA_REFP_VELO_SEARCH_MARKER (reference point creep velocity)

Before the machine axis travels over the parameterized number of reference marks, it makes contact with the reference cam. It is then reversed and reference mark search is restarted in the opposite direction.

Once the number of reference marks set by means of parameter assignment has been crossed, the machine axis is stopped again and the actual value system of the machine axis is synchronized to the absolute position calculated by the NC.

Plausiblity check of the reference mark distance

An error occurs if, during reference point traversing for two subsequent reference marks, the NC determines a distance greater than twice the parameterized reference mark distance.

MD34300 $MA_ENC_REFP_MARKER_DIST (reference mark distance)

The machine axis will then traverse in opposite direction at half the parameterized reference point creep velocity (MD34040) and the search for reference mark is restarted.

If a faulty reference mark distance is detected again, the machine axis is stopped and the reference point traversing is aborted (alarm 20003 "fault in the measuring system").

Abort criterion

If the parameterized number of reference marks is not detected within the parameterized distance, the machine axis is stopped and reference point traversing is aborted.

MD34060 $MA_REFP_MAX_ MARKER_DIST (max. distance to the reference mark)

Features of Phase 1

After Phase 1 is successfully completed, the actual value system of the machine axis is synchronized.

10.6.5 Phase 2: Traversing to the target point

Phase 2: Start

Phase 2 is automatically started when Phase 1 has been completed without an alarm.

Initial situation:

- The machine axis is positioned directly behind the last of the parameterized number of reference marks.
- The actual value system of the machine axis is synchronized.

Phase 2: Sequence

In Phase 2, the machine axis completes reference point approach by traversing to a defined target position (reference point). This action can be suppressed in order to shorten the reference point approach:

MD34330 $MA_STOP_AT_ABS_MARKER = <value>

Value	Meaning
0	Travel to target position
1	No travel to target position

Travel to target position (normal case)

The machine axis accelerates to the parameterized reference point position velocity and travels to the parameterized target point (reference point):

MD34070 $MA_REFP_VELO_POS (reference point positioning velocity)

MD34100 $MA_REFP_SET_POS (reference point value)

The machine axis is referenced. To identify this, the NC sets an interface signal for the measuring system that is currently active:

DB31, ... DBX60.4/60.5 (referenced/synchronized 1/2) = 1

No travel to target position

The machine axis is now referenced. To identify this, the NC sets an interface signal for the measuring system that is currently active:

DB31, ... DBX60.4/60.5 (referenced/synchronized 1/2) = 1

Features of Phase 2

Phase 2 will display different characteristics, depending on whether a reference cam is parameterized for the machine axis.

Machine axis without reference cam

MD34000 $MA_REFP_CAM_IS_ACTIVE (axis with reference cam) = 0

Properties:

- Feedrate override is active.
- The feed stop (channel-specific and axis-specific) is active.
- NC stop and NC start are active.

Machine axis with reference cam

MD34000 $MA_REFP_CAM_IS_ACTIVE (axis with reference cam) = 1

Properties:

- Feedrate override is **not** active.
  Traversing is performed internally with feedrate override = 100%.
  If a feedrate override of 0% is specified, an abort occurs.
- The feed stop (channel-specific and axis-specific) is active.
- NC stop and NC start are **not** active.
- If the parameterized number of reference marks is not detected within the parameterized distance after the exit of the reference cam, the machine axis will be stopped.
  MD34060 $MA_REFP_MAX_ MARKER_DIST (max. distance to the reference mark)

Special features of rotary measuring systems

On rotary distance-coded measuring systems, the absolute position can only be determined uniquely within one revolution. Depending on the mechanical mounting of the encoder, the overtravel of the absolute position in the hardware does not always coincide with the traversing range of the rotary axis.

Special features of modulo rotary axes

With module rotary axes, the reference point position is mapped on the parameterized modulo range:

MD30330 $MA_MODULO_RANGE (size of the modulo range)

MD30340 $MA_MODULO_RANGE_START (start position of the modulo range)

Note

The reference point position is mapped on the parameterized (fictive) modulo range even with axis function "Determination of reference point position rotary, distance-coded encoder within the configured modulo range":

MD30455 $MA_MISC_FUNCTION_MASK (axis functions), BIT1 = 1

10.7 Referencing by means of actual value adjustment

10.7.1 Actual value adjustment to the referencing measurement system

Function

When actual value adjustment to the referencing measuring system is performed, the resulting absolute actual position after successful referencing of the measuring system of a machine axis is transferred directly to all other measuring systems of the machine axis, and the machine axis is designated as referenced:

DB31, ... DBB60.4 / 60.5 (referenced/synchronized 1/2) = 1

Advantage

When the machine axis switches from an explicitly referenced measuring system to the measuring system referenced by actual value adjustment, continuous servo control is assured (servo enable active) because the matched actual position prevents a sudden change in actual value.

Note

In order to improve positioning precision by determining the measuring-system-specific encoder fine information, we recommend explicitly re-referencing the measuring system previously referenced by actual value adjustment after switching over.

Activation

The activation of the actual value adjustment to the referencing measuring system is machine-specifically carried out via:

MD34102 $MA_REFP_SYNC_ENCS = 1

10.7.2 Actual value adjustment for measuring systems with distance-coded reference marks

Function

In order to improve positioning precision by determining the measuring-system-specific encoder fine information, we recommend explicitly re-referencing the measuring system previously referenced by actual value adjustment after switching over the measuring system.

If an encoder with distance-coded reference marks is used for the passive measuring system, referencing can be avoided under the following conditions:

1. Active measuring system: Indirect measuring system (motor measuring system) with absolute encoder, for example
2. Passive measuring system: Direct measuring system with distance-coded reference marks
3. Traversing motion of the machine axis with the referenced indirect measuring system before measuring system switchover in which the number of reference marks required for referencing are crossed. Whereby, the passive direct measuring system is referenced automatically.

Parameterization

In addition to the specific machine data required to reference the individual measuring systems, the following machine data must be set:

- Enable the actual value adjustment:
  MD34102 $MA_REFP_SYNC_ENCS = 1
- Direct measuring system with distance-coded reference marks:
  - MD34200 $MA_ENC_REFP_MODE[*measuring system*] = 3
    Distance-coded reference marks
  - MD30242 $MA_ENC_IS_INDEPENDENT[*measuring system*] = 2
    In the actual value adjustment, the passive direct measuring system is aligned to the actual position of the active indirect measuring system, but not marked as referenced. After the parameterized number of reference marks is crossed, the passive direct measuring system is referenced automatically. Referencing is performed in every operating mode.

Sequence

1. Initial situation: Both measuring systems are not referenced:
   DB31, ... DBX60.4 = 0 (referenced/synchronized 1)
   DB31, ... DBX60.5 = 0 (referenced/synchronized 2)
2. Referencing of the indirect measuring system according to the measuring system type:
   DB31, ... DBX60.4 = 1 (referenced/synchronized 1)
   DB31, ... DBX60.5 = 0 (referenced/synchronized 2)
3. Traversing of the machine axis across the parameterized number of reference marks.
   This automatically references the direct measuring system:
   DB31, ... DBX60.4 = 1 (referenced/synchronized 1)
   DB31, ... DBX60.5 = 1 (referenced/synchronized 2)

10.8 Referencing in follow-up mode

Function

Incremental measuring systems and measuring systems with distance-coded reference marks can be referenced even when the machine axis is in follow-up mode. Prerequisite for this is the correct parameterization of the reference point approach according to the used measuring system (see Section "Referencing with incremental measurement systems (Page 296)" and "Referencing with distance-coded reference marks (Page 309)").

When referencing in follow-up mode the machine axis is moved not by the NC but by means of an external travel motion over the encoder zero mark and the parameterized number of distance-coded reference marks. The measuring system is referenced when the encoder zero mark or parameterized number of distance-coded reference marks are detected.

Note

Reproducibility of the referencing result

In NC-guided reference point approach, reproducibility of the referencing result is ensured through adherence to the assigned traverse velocities during the referencing operation. During referencing in follow-up mode, responsibility for achieving reproducibility of the referencing results lies with the machine manufacturer / user.

Unique zero mark

Referencing of an incremental measuring system is based on the explicit position of the encoder zero mark relative to the overall traversing range of the machine axis.

Because the reference cam signal is not evaluated by the NC during referencing in follow-up mode, unique identification of the reference point when referencing in follow-up mode will only result with:

- Only one encoder zero mark in the traversing range of the machine axis.
- Linear measuring systems with distance-coded reference marks.
- Modulo rotary axes (absolute position within one revolution).

Zero mark selection when several zero mark signals occur

If several encoder zero marks are detected in the traversing range of the machine axis due to machine-specific factors, e.g. reduction gear between encoder and load, a proximity switch must be mounted on the machine and connected via a digital input of the NCU interface in order to clearly determine the reference point.

Note

Contact-less proximity switch signal: Zero mark selection

The processing of the proximity switch signal is performed exclusively in the drive. Connection and parameterization, see Section "Hardware signals (Page 296)".

Enable

The "Referencing in follow-up mode" function is enabled with:

MD34104 $MA_REFP_PERMITTED_IN_FOLLOWUP = TRUE

Starting the referencing operation

If the machine axis is in follow-up mode (DB31, ... DBX61.3 == TRUE) at the start of reference point approach, the measuring system is referenced in follow-up mode.

If the machine axis is not operating in the follow-up mode at the start of reference point traversing, the "normal" from the NC-controlled reference point travels is carried out.

Referencing in follow-up mode can be started in the following modes:

- JOG-REF: Traversing keys
- AUTOMATIC: Part program command `G74`

Sequence of the referencing operation (JOG-REF mode)

1. Activate follow-up mode of machine axis:
   DB31, ... DBX1.4 (follow-up mode) = 1
   DB31, ... DBX2.1 (controller enable) = 0
2. Wait for activation of follow-up mode:
   DB31, ... DBX61.3 (follow-up active) == 1
3. Switch to JOG mode, REF machine function
4. External motion of machine axis across encoder zero mark or parameterized number of distance-coded reference marks. The referencing operation is started internally in the NC as soon as the machine axis is moved: The following NC/PLC interface signal is reset as feedback:
   DB31, ... DBX61.4 (axis/spindle stationary) == 0
5. The measuring system is referenced after the encoder zero mark or the assigned number of distance-coded reference marks have been successfully detected: The following NC/PLC interface signal is set as feedback:
   DB31, ... DBX60.4/60.5 (referenced/synchronized 1/2) == 1

Aborting the reference operation

An active referencing operation can be aborted by:

- Deselecting follow-up mode
- NC reset

Response when measuring systems are already referenced

A measuring system that has already been referenced can only be re-referenced in AUTOMATIC mode using part program command `G74`.

Sequence of referencing operation (AUTOMATIC mode)

1. Switch to AUTOMATIC mode.
2. Start the part program.
3. Activate follow-up mode of machine axis:
   DB31, ... DBX1.4 (follow-up mode) = 1
   DB31, ... DBX2.1 (controller enable) = 0
4. Wait for activation of follow-up mode:
   DB31, ... DBX61.3 (follow-up active) == 1
5. The referencing operation is started internally in the NC as soon as `G74` command has been processed.
6. External motion of machine axis across encoder zero mark or parameterized number of distance-coded reference marks.
7. The measuring system is referenced after the encoder zero mark or the assigned number of distance-coded reference marks have been successfully detected: The following NC/PLC interface signal is set as feedback:
   DB31, ... DBX60.4 / 60.5 (referenced/synchronized 1/2) == 1
8. The block change occurs after the referencing operation has been successfully completed.

Aborting the reference operation

An active referencing operation can be aborted by:

- Deselecting follow-up mode
- NC reset

Response when measuring systems are already referenced

A measuring system that you have already referenced can be re-referenced.

10.9 Referencing with absolute encoders

10.9.1 Information about the adjustment

Machine axes with absolute encoder

The advantage of machine axes with absolute encoder is that after a one time adjustment procedure, the necessary reference point traversing with incremental measuring systems (e.g. build-up of control, de-selection of "Parking" of machine axes etc.) can be skipped and the actual value system of the machine axis can be immediately synchronized to the determined absolute position.

Adjustment

Adjustment of an absolute encoder involves matching the actual value of the encoder with the machine zero once and then setting it to valid.

The current adjustment status of an absolute encoder is displayed in the following axis-specific machine data of the machine axis, to which it is connected:

MD34210 $MA_ENC_REFP_STATE (status of absolute encoder)

Value	Meaning
0	Encoder not calibrated
1	Encoder adjustment enabled
2	Encoder is calibrated

Adjustment methods

The following adjustment methods are supported:

- Adjustment by entering a reference point offset
- Adjustment by entering a reference point value
- Automatic adjustment with probe
- Adjustment by means of the proximity switch

Readjustment

Readjustment of the absolute encoder is required after:

- Gear change between load and absolute encoder
- Removal/installation of the absolute encoder
- Removal/installation of the motor with the absolute encoder
- Data loss in the static NC memory

- Battery failure
- Set actual value (PRESETON)

⚠ WARNING

Data backup

During the back-up of machine data of a machine A, the encoder status of the machine axis (MD34210) is also backed up.

During loading of this data record into a machine B of the same type, e.g. in the context of a serial start-up or after a case of maintenance, the referenced machine axes will be automatically regarded as adjusted / referenced by the NC. It is the special responsibility of the machine manufacturer / user to undertake a readjustment in such cases.

See also explanations regarding machine data:

MD30250 $MA_ACT_POS_ABS (absolute encoder position at the time of switch-off)

Note

The controller can detect a required readjustment of the absolute encoder only during the following events:

- Gear change with change of gear ratio
- Addressing the zero-mark monitoring
- New encoder serial number after change of the absolute encoder

The controller then sets the status of the absolute encoder to "0":

MD34210 $MA_ENC_REFP_STATE = 0 (encoder not adjusted)

The following alarm is displayed:
Alarm 25022 "Axis <axis name> encoder <number> warning 0"
If the zero-mark monitoring responds, the following alarm is also displayed:
Alarm 25020 "Axis <axis name> zero-mark monitoring of active encoder"

In all other cases (e.g. PRESETON) it is the sole responsibility of the user to display the misalignment of the absolute encoder by **manually** setting the status to "0" and to carry out a readjustment.

10.9.2 Calibration by entering a reference point offset

Function

During adjustment by entering the reference point offset, the difference between the position displayed on the operator interface and the true actual position in the machine is determined and made known to the NC as reference point offset.

Procedure

1. Determining the position of the machine axis with reference to the machine zero point via e.g.:
   - position measurement (e.g. laser interferometer)
   - Moving the machine axis to a known position (e.g., fixed stop)
2. Reading the displayed actual position of the machine axis on the operator interface.
3. Calculating the reference point offset (difference between the actual positions determined under point 1 and 2) and entering in machine data:
   MD34090 $MA_REFP_MOVE_DIST_CORR (reference point offset)
4. Marking the absolute value encoder as adjusted:
   MD34210 $MA_ENC_REFP_STATE = 2

   Note

   The encoder adjustment does not become active until the next time the encoder is activated (e.g., when the controller is powered up).

5. Initiate POWER ON reset.
6. Controlling the position of the machine axis displayed on the operator interface.

   Note

   Backlash compensation

   If backlash compensation is parameterized for a measuring system with absolute value encoder, the following must be observed:

   No backlash is permitted during machine axis travel to the adjusted machine position.

   Activate reference point offset permanently

   The entered reference point offset (MD34090) will be permanently active only after initial POWER ON - Reset. If the machine axis is moved after the absolute encoder adjustment without an interim POWER ON - Reset, the reference point offset entered in the machine data can be overwritten, for example, as part of internal overrun offset.

   Checking the actual position

   Following adjustment of the absolute encoder, we recommend that you verify the actual position of the machine axis the next time you power up the controller (POWER ON).

10.9.3 Adjustment by entering a reference point value

Function

During adjustment by entering the reference point value, the absolute position of the machine axis with reference to the machine zero point is determined by e.g.:

- Position measurement (e.g. laser interferometer)
- Moving the machine axis to a known position (e.g. fixed stop)

This determined position value will be made known to the NC as the reference point value. The NC then calculates the reference point offset from the difference between the encoder absolute value and the reference point value.

Procedure

1. Check whether the referencing mode is set to "Assume the reference point value":
   MD34200 $MA_ENC_REFP_MODE == 0
   If not, enter the value 0 in the machine and trigger a Power On reset.
2. Traverse the machine axis in JOG mode to the position to be measured (e.g. with a laser interferometer) or to a known position (e.g. fixed stop).

   Note

   The machine axis can only be traversed in the direction enabled for referencing with the travel keys:

   MD34010 $MA_REFP_CAM_DIR_IS_MINUS (approach reference point in minus direction)

   To avoid an invalid position because of backlash in the drive train, the known position must be approached at low velocity.

3. Enter the position of the machine axis relative to machine zero as the reference point value in the machine data:
   MD34100 $MA_REFP_SET_POS = *Position*
4. Enable encoder adjustment:
   MD34210 $MA_ENC_REFP_STATE = 1
5. Trigger a Power On reset for acceptance of the entered machine data values.
6. Switch to JOG-REF mode.

7. Operate the travel key used for referencing in step 2.
   The machine axis does not move when the traversing key is actuated!
   The NC calculates the reference point offset from the entered reference point value and that given by the absolute encoder. The result is entered into the machine data:
   MD34090 $MA_REFP_MOVE_DIST_CORR (reference point offset)
   The status of the absolute encoder is set to "Encoder is adjusted":
   MD34210 $MA_ ENC_REFP_STATE = 2
   The actual value system of the machine axis is synchronized.
   The machine axis is now referenced. As identification, the NC sets the appropriate interface signal based on which measuring system is currently active:
   DB31, ... DBB60.4 / 60.5 (referenced/synchronized 1 / 2) = 1
8. Initiate Power On reset.

Note

Activate reference point offset permanently

The entered reference point offset (MD34090) will only be permanently active after Power On reset.

If the machine axis is moved after the absolute encoder adjustment without an interim Power On reset, the reference point offset entered in the machine data can be overwritten, for example, within internal overrun corrections.

Checking the actual position

Following adjustment of the absolute encoder, we recommend that you verify the actual position of the machine axis the next time you power up the controller (Power On).

10.9.4 Automatic calibration with probe

Function

In automatic adjustment with a probe, a known position in the machine is approached with the machine axis from a part program. The position value is stored in the NC as a reference point value. The position is reached when the probe switches, and the NC then calculates the reference point offset from the difference between the encoder value and reference point value.

Note

Part program for automatic adjustment

The part program for automatic adjustment using a probe must be created by the machine manufacturer / user for the specific requirements of the machine.

Freedom from collision

Because actual-value-related monitoring is not active for the machine axes being referenced, the machine operator must take special care to ensure that collisions do not occur in the machine while the machine axes are being moved!

Part program

The part program for automatic adjustment of absolute encoders with probe must perform the points listed below for each axis in the order indicated:

1. Approach the adjustment position of machine axis, which is detected from the probe response.
   The position must be approached several times from the same direction, but at a velocity which is gradually reduced on each approach, to ensure that the measured value obtained is as accurate as possible. The measured value is stored in system variable $AA_IM.
2. Calculating and writing the reference point offset:

   MD34090 $MA_REFP_MOVE_DIST_CORR = MD34100 $MA_REFP_SET_POS – $AA_IM

3. Set the absolute encoder status to "Encoder is adjusted":
   MD34210 $MA_ ENC_REFP_STATE = 2

Sequence

Proceed as follows for automatic adjustment with probe:

1. Enable part program start even for non-referenced machine axes:
   MD20700 $MC_REFP_NC_START_LOCK = 0
2. Enter the machine axis position relative to machine zero when probe is switched as the reference point value for all relevant machine axes:
   MD34100 $MA_REFP_SET_POS = reference point value
3. Activate NCK-Reset for the acceptance of the entered machine data values.
4. Start part program.
5. After completion of the part program, re-secure the partial program start for machine axes which are not referenced:
   MD20700 $MC_REFP_NC_START_LOCK = 1
6. Initiate POWER ON - Reset so that the reference point offset written by the part program is permanently active:
   MD34090 $MA_REFP_MOVE_DIST_CORR (reference point offset)

Note

Activate reference point offset permanently

The entered reference point offset (MD34090) will only be permanently active after POWER ON - Reset.

If the machine axis is moved after the absolute encoder adjustment without an interim POWER ON - Reset, the reference point offset entered in the machine data can be overwritten, for example, as part of internal overrun offset.

Checking the actual position

Following adjustment of the absolute encoder, we recommend that you verify the actual position of the machine axis the next time you power up the controller (POWER ON).

10.9.5 Adjustment with BERO

Function

For adjustment using proximity switch, a reference point approach to a defined machine position is performed the same as for incremental measuring systems. In this case, the proximity switch replaces the encoder zero mark that the absolute encoder does not have. After successful completion of reference point approach, the NC automatically calculates the reference point offset from the difference between the encoder absolute value and the parameterized reference point value.

Parameterization

NC: Referencing mode

The referencing mode should be set to "Referencing of incremental, rotary or linear measuring systems: Zero pulse on the encoder track":

MD34200 $MA_ENC_REFP_MODE[<axis>] = 1

NC: Reference point value

The reference point value is parameterized via:

MD34100 $MA_REFP_SET_POS[<axis>] = <reference point value>

Drive: Equivalent zero mark

The digital input on the NCU interface to which the proximity switch is connected must be set in parameter p0494 or p0495.

Execution

Reference point approach can be started manually in JOG-REF mode or in AUTOMATIC or MDI mode via a part program (`G74`).

After successful completion of the reference point approach, the absolute encoder is adjusted and the actual-value system of the machine axis is synchronized.

As feedback for the PLC user program, the NC sets the NC/PLC interface signal for the machine axis, depending on the active measuring system:

DB31, ... DBB60.4/60.5 (referenced/synchronized 1/2) = 1

Note

If the proximity switch is removed after adjustment of the absolute encoder, the referencing mode must be re-parameterized to "Referencing with absolute encoder".

MD34200 $MA_ENC_REFP_MODE[<axis>] = 0

10.9.6 Reference point approach with absolute encoders

Parameter assignment

Traversing movement release

If for a machine axis with adjusted absolute value encoder as active measuring system, reference point traversing is activated (manually in the JOG-REF mode or automatically via the part program instruction `G74`), the machine axis travels depending on the parameterized traversing movement release.

MD34330 $MA_REFP_STOP_AT_ABS_MARKER = <value>

Value	Meaning
0	Traversing is **enabled**. When reference point approach is initiated, the machine axis moves to the reference point position. When reaching the reference point position, the reference point approach is completed.
1	Traversing is **not** enabled. After the activation of the reference point travel, the machine axis does not travel and the reference point travel is immediately completed.

10.9.7 Reference point approach for rotary absolute encoders with equivalent zero mark

Function

To ensure that the reference point approach via zero mark (see Section "Referencing with incremental measurement systems (Page 296)") can also be used with absolute encoders, the zero mark, which is not provided in the hardware of absolute encoders, is simulated. For this, the controller generates the signal for the equivalent zero mark once per encoder revolution, always at the same position within the revolution.

Difference compared to referencing with incremental encoders

An absolute encoder with replacement zero mark should not be considerd as a complete equivalent of an incremental encoder. All the properties of the absolute encoder are retained. The following table lists the different properties of incremental and absolute encoders:

Table 10-1 Properties of incremental and absolute encoders

Property	Incremental encoder	Absolute encoder
Encoder type	MD30240 $MA_ENC_TYPE =	
	1	4
Internal encoder position	MD30250 $MA_ACT_POS_ABS =	
	Value is updated only in MD34210 ≥ 1	Value is updated only in MD30270 = 0
Traversing range extension	MD30270 $MA_ENC_ABS_BUFFERING =	
	No effect	= 0 (default): Active

Property	Incremental encoder	Absolute encoder
Reference point offset	MD34090 $MA_REFP_MOVE_DIST_CORR =	
	Value input allowed	Value is updated exclusively via control
Supported referencing types	MD34200 $MA_ENC_REFP_MODE =	
	1, 3, 4, 8	0, 1
Adjustment status	MD34210 $MA_ENC_REFP_STATE = 0, 1, 2	
	Automatic encoder misalignment during shut down while in motion.	Automatic encoder misalignment during parameter set change with position jump or during serial number change.
Absolute position modulo range	MD34220 $MA_ENC_ABS_TURNS_MODULO =	
	0	1 - 4096
Encoder serial no.	MD34230 $MA_ENC_SERIAL_NUMBER =	
	0	The value must be updated from the PLC during each encoder change, otherwise loss of adjustment plus alarm.
Transfer of series startup files	Without any restrictions.	Due to encoder characteristics (MD30250, MD30270, MD34090, MD34210, MD34220, MD34230) only possible with certain restrictions.
Activation time	0 seconds	several seconds
Zero mark	1 per encoder revolution	None
Zero mark monitoring	Hardware	Software
Position after POWER ON without actual value buffering	0.0	Last position within MD34220.
	MD34210 = 0	MD30270 = 1
Position after POWER ON with actual value buffering	Last standstill position before deactivation.	Last position including small movements during POWER OFF.
	MD34210 = 1	MD30270 = 0
Referenced after POWER ON	depends on adjustment status	

Requirement

The function can be used only with rotary absolute encoders:

- MD31000 $MA_ENC_IS_LINEAR == 0
- MD30240 $MA_ENC_TYPE == 4

Parameterization

Reference point approach with equivalent zero mark:

MD34200 $MA_ENC_REFP_MODE = 1

Supplementary conditions

- A reference point offset (MD34090 $MA_REFP_MOVE_DIST_CORR) may not be parameterized.
  This MD describes, in connection with absolute encoders, the offset between machine and absolute encoder zero, and it therefore has a different meaning.
- The load-side zero mark search rate (MD34040 $MA_REFP_VELO_SEARCH_MARKER) should not exceed the limiting frequency of the absolute trace of the encoder (MD36302 $MA_ENC_FREQ_LIMIT_LOW)
  . If the speed is too high, absolute information cannot be read any more, and thus, no equivalent zero marks are generated.
- If no zero mark is found within the parameterized path (MD34060 $MA_REFP_MAX_MARKER_DIST), an alarm is output:
- The following MD must be set if the absolute encoder retains even the referenced status through POWER OFF, besides the last position:
  MD34210 $MA_ENC_REFP_STATE = 2

Data backup and standard commissioning

Some properties of an absolute encoder restrict the transfer of a commissioning archive to other machines. The following machine data must be checked and possibly corrected after loading a commissioning archive to the controller:

- MD30250 $MA_ACT_POS_ABS (internal encoder position)
- MD30270 $MA_ENC_ABS_BUFFERING (traversing range extension)
- MD34090 $MA_REFP_MOVE_DIST_CORR (absolute offset)
- MD34210 $MA_ENC_REFP_STATE (adjustment status)
- MD34220 $MA_ENC_ABS_TURNS_MODULO (modulo range)
- MD34230 $MA_ENC_SERIAL_NUMBER (encoder serial number)

10.9.8 Enabling the measurement system

The measuring system of a machine axis is activated in the following cases:

- Power up of the control (POWER ON)
- Activation of the measuring system via interface signal (deselection of "parking"):
  DB31, ... DBB1.5 / 1.6 (position measuring system 1/2)
  DB31, ... DBB2.1 (servo enable)
- Violation of the assigned encoder limit frequency (spindles):
  MD36300 $MA_ENC_FREQ_LIMIT

When the measuring system is activated, the NC synchronizes the actual value system of the machine axis with the current absolute value. Traversing is disabled during synchronization for axes but not for spindles.

Parameterizing the encoder limit frequency (spindles)

The EQN 1325 absolute encoder made by Heidenhain has an incremental track and an absolute track.

If a spindle is driven at a speed above the encoder limit frequency of the incremental track, the substantially lower limit frequency of absolute track must be parameterized as the encoder limit frequency.

MD36300 $MA_ENC_FREQ_LIMIT

Otherwise an incorrect absolute position would be read because the parameterized encoder limit frequency is not reached when the measuring system is activated. This would cause a position offset in the actual value system of the machine axis.

Determining the encoder limit frequency

The encoder limit frequency to be parameterized is derived from the smaller of the two following limit speeds:

- Encoder
  The limit speed or encoder limit frequency is listed in the data sheet of the encoder (e.g., limit speed = 2000 [rpm])
- NC
  Due to the NC-internal evaluation process, the maximum limit speed for which error-free calculation of the absolute value by the NC is possible is 4 encoder revolutions per interpolator clock cycle.
  For an interpolator clock cycle of, for example, 12 ms: Limit speed = 4 / 12 ms = 20,000 rpm
  The limiting frequency corresponding to the limiting speed is calculated to be:

$$MD36300 = \frac{4 * MD31020}{MD10050 * MD10070}$$

MD31020 $MA_ENC_RESOL (encoder pulses per revolution)
MD10050 $MN_SYSCLOCK_CYCLE_TIME (basic system clock cycle)
MD10070 $MN_IPO_SYSCLOCK_TIME_RATIO (factor for the interpolation cycle)

Note

The position control switching speed relevant for spindles is set according to the encoder limiting frequency of the absolute value encoder of the spindle:

MD35300 $MA_SPIND_POSCTRL_VELO (position control switching speed)

MD36300 $MA_ENC_FREQ_LIMIT (Encoder limit frequency)

10.9.9 Referencing variants not supported

The following referencing variants are not supported when used with absolute encoders:

- Referencing/calibrating with encoder zero mark
- Distance-coded reference marks
- Proximity switch with two-edge evaluation

10.10 Automatic restoration of the machine reference

10.10.1 Overview

Without a defined machine reference when traversing machine axes, no position-dependent functions such as transformations or tool frames can be executed. In different machine situations, these functions must immediately be available for traversing axes when the encoder is activated, e.g. after the controller is switched on or after the parking of a machine axis has ended (see Function Manual "Monitoring and Compensating"). However, the machine axes should not or cannot be traversed again for referencing.

Absolute encoders

For measuring systems with adjusted absolute encoders, the machine reference is restored immediately without any additional measures when the encoder value is read.

Incremental encoders

With incremental measuring systems, the machine reference can be restored without traversing the axes through "Automatic referencing" or "Restoration of the actual position".

Boundary conditions

⚠ WARNING

Incorrect synchronization of the position measuring system caused by offset of the actual machine axis position

During the time in which the measuring system of the machine axis is switched off, it is **not** permissible that the axis is mechanically moved. Otherwise this results in an offset between the last buffered actual position and the real actual position of the machine axis. This would lead to an incorrect synchronization of the measuring system resulting in danger to personnel and machine.

The machine manufacturer must provide such measures as holding brakes, etc. on the machine so that the actual position is not changed, and this must be ensured by the user. The responsibility for this rests exclusively with the machine manufacturer / user.

If axis motion cannot be prevented mechanically in the shutdown state, either an absolute encoder must be used or the axis must be referenced again with reference point approach after switching on.

Note

SMExx Sensor Modules

Automatic referencing or restoration of the actual position to the last buffered position after restarting the control is only possible in conjunction with SM**E**xx (externally mounted) Sensor Modules. When using SM**C**xx (cabinet) or SM**I**xx (integrated) Sensor Modules, the actual position **cannot** be restored after restarting the control (power on). The measuring system of the machine axis must be referenced again.

10.10.2 Automatic referencing

Function

During automatic referencing, the actual position of the active measuring system of the machine axis is set to the last buffered position and "referenced" set as encoder state after switching on the control. This makes it possible to start programs in the AUTOMATIC and MDI modes directly after run-up of the control.

Requirements

- The active measuring systems when the control is switched on must already have been referenced once before switching off.
- At the time the control is switched off, the machine axis must be at standstill with "Exact stop fine" (DB31, ... DBX60.7 == 1) are located.

  Note

  If the machine axis is **not** at standstill with "Exact stop fine" when switching off, the actual position will be initialized with " 0" when switching on. "Not referenced" is displayed as the encoder state.

Parameterization

The automatic referencing is enabled by setting the encoder state to "Automatic referencing is enabled, but the encoder has not been referenced":

MD34210 $MA_ENC_REFP_STATE[<encoder>] = **1**

After the measuring system has been referenced, the encoder state displays that automatic referencing will be executed the next time the encoder is activated:

MD34210 $MA_ENC_REFP_STATE[<encoder>] == **2**

NC/PLC interface signals

After automatic referencing, the encoder state "Referenced" is displayed for the active measuring system:

DB31, ... DBX60.4/.5 == **1** (referenced/synchronized 1/2)

Boundary conditions

Encoder activation with MD34210 $MA_ENC_REFP_STATE[<encoder>] == 1

An encoder state equal to "1" at the time of the encoder activation means that "Automatic referencing" has been enabled. However, the measuring system has either not been referenced yet or the machine axis was not switched off at standstill in the "Exact stop fine" state. The following is set for the machine axis or the active measuring system:

- Actual position = 0
- Active measuring system, encoder state = "Not referenced":
  DB31, ... DBX60.4 / .5 = **0** (referenced/synchronized 1/2)

Further information

Referencing with incremental measurement systems (Page 296)

10.10.3 Restoration of the actual position

Function

When restoring the actual position to the last buffered position, the encoder state of the active measuring system is set to "**Restored**". The axis can only be traversed manually.

AUTOMATIC mode

To enable NC start for the automatic execution of programs in the AUTOMATIC mode, the measuring system of the machine axis must be re-referenced.

MDI mode and overstore

In the MDI mode and for the overstore function, machining can also be performed, without referencing the axes, with restored positions. To do this, NC start with restored positions must be enabled explicitly for a specific channel:

MD20700 $MC_REFP_NC_START_LOCK = 2

Requirement

The measuring system that is active when the control is switched on must already have been referenced once before switching off.

Parameterization

Release: Restoration of the actual position

The enable to restore the actual position is performed by setting the encoder state to "The last buffered axis position before switching off will be restored, no automatic referencing":

MD34210 $MA_ENC_REFP_STATE[<encoder>] = **3**

Release: NC START for "MDI" and "Overstore" modes

The enable of NC START for execution of part programs or part program blocks in the "MDI" and "Overstore" modes with the state "Position restored" is performed via:

MD34110 $MA_REFP_CYCLE_NR **≠ -1** (axis sequence for channel-specific referencing)

MD20700 $MC_REFP_NC_START_LOCK = **2** (NC START lock without reference point)

NC/PLC interface signals

The restored actual position is not considered to be equivalent to an actual position after reference point approach. Therefore, the state "Position restored" and not "Referenced/ synchronized" is displayed for the measuring system of the machine axis.

Actual position restored:

- DB31, ... DBX60.4/.5 = **0** (referenced/synchronized 1/2)
- DB31, ... DBX71.4/.5 = **1** (**position restored**, encoder 1/2)

Measuring system referenced:

- DB31, ... DBX60.4/.5 = 0 → **1** (**referenced/synchronized** 1/2)
- DB31, ... DBX71.4/.5 = 1 → **0** (position restored, encoder 1/2)

Note

The monitoring of the traversing range limits (software limit switches, working area limitation, etc.) is already active in the "Position restored" state.

Boundary conditions

Spindles

If the encoder limit frequency is exceeded, a spindle is reset to the "Not referenced/ synchronized" state:

- DB31, ... DBX60.4/.5 = 1 → **0** (referenced/synchronized 1/2)
- DB31, ... DBX71.4/.5 = 1 → **0** (position restored, encoder 1/2)

Further information

Referencing with incremental measurement systems (Page 296)

10.11 Supplementary conditions

10.11.1 Large traverse range

Linear axes with a traversing range > 4096 encoder revolutions, rotatory absolute encoder EQN 1325 and a parameterized absolute encoder range of MD34220 $MA_ENC_ABS_TURNS_MODULO = 4096:

The maximum possible travel range corresponds to that of incremental encoders.

Endlessly turning rotary axes with absolute encoders:

- Any number of integer transmission ratios are permitted.
- We recommend that you parameterize endlessly turning rotary axes with absolute encoders as modulo rotary axes (traversing range 0...360 degrees):
  MD30310 $MA_ROT_IS_MODULO = 1
  Otherwise, the rotary axis may require a very large traversing path to reach absolute zero when the measuring system is activated.

Machine axes with absolute encoders:

In order that the controller correctly determines the current actual position after the restart of the measuring system, the machine axis may only be moved less than half the absolute encoder range when the measuring system is switched off:

MD34220 $MA_ENC_ABS_TURNS_MODULO

Notes on uniqueness of encoder positions

Note

Linear absolute encoders

The absolute value of linear position encoders, e.g. Heidenhain LC181, is always unique for the scale lengths available.

Rotary absolute encoders

The absolute value of rotary absolute encoders is only unique within the range of the specific maximum encoder revolutions.

For example, the EQN 1325 rotary absolute encoder by Heidenhain supplies a unique absolute value in the range of 0 to 4,096 encoder revolutions.

Depending on how the encoder is connected that will result in:

- Rotary axis with encoder on load: 4096 load revolutions
- Rotary axis with encoder on motor: 4096 motor revolutions
- Linear axis with encoder on motor: 4096 motor revolutions

Example:
An EQN 1325 rotary absolute encoder is mounted on the motor of a linear axis. For an effective leadscrew pitch of 10 mm this will result in a unique absolute value within the travel range -20.48 to +20.48 m.

10.12 Data lists

10.12.1 Machine data

10.12.1.1 NC-specific machine data

Number	Identifier: $MN_	Description
11300	JOG_INC_MODE_LEVELTRIGGRD	INC/REF in jog/continuous mode

10.12.1.2 Channelspecific machine data

Number	Identifier: $MC_	Description
20700	REFP_NC_START_LOCK	NC start disable without reference point

10.12.1.3 Axis/spindlespecific machine data

Number	Identifier: $MA_	Description
30200	NUM_ENCS	Number of encoders
30240	ENC_TYP	Actual value encoder type
30242	ENC_IS_INDEPENDENT	Encoder is independent
30250	ACT_POS_ABS	Absolute encoder position at time of deactivation
30270	ENC_ABS_BUFFERING	Absolute encoder: Traversing range extension
30300	IS_ROT_AX	Rotary axis / spindle
30310	ROT_IS_MODULO	Modulo conversion for rotary axis / spindle
30330	MODULO_RANGE	Size of the modulo range
30340	MODULO_RANGE_START	Starting position of modulo range
30355	MISC_FUNCTION_MASK	Axis functions
31122	BERO_DELAY_TIME_PLUS	BERO delay time in plus direction
31123	BERO_DELAY_TIME_MINUS	BERO delay time in minus direction
34000	REFP_CAM_IS_ACTIVE	Axis with reference cam
34010	REFP_CAM_DIR_IS_MINUS	Reference point approach in minus direction
34020	REFP_VELO_SEARCH_CAM	Reference point approach velocity
34030	REFP_MAX_CAM_DIST	Maximum distance to reference cam
34040	REFP_VELO_SEARCH_MARKER	Reference point creep velocity
34050	REFP_SEARCH_MARKER_REVERSE	Direction reversal on reference cam
34060	REFP_MAX_MARKER_DIST	Maximum distance to reference mark; maximum distance to two reference marks with distance-coded scales
34070	REFP_VELO_POS	Reference point positioning velocity

Number	Identifier: $MA_	Description
34080	REFP_MOVE_DIST	Reference point distance / destination point for distance-coded system
34090	REFP_MOVE_DIST_CORR	Reference point offset / absolute offset, distance-coded
34092	REFP_CAM_SHIFT	Electronic reference cam shift for incremental measuring systems with equidistant zero marks
34093	REFP_CAM_MARKER_DIST	Reference cam / reference mark distance
34100	REFP_SET_POS	Reference point value
34102	REFP_SYNC_ENCS	Actual value adjustment to the referencing measurement system
34104	REFP_PERMITTED_IN_FOLLOWUP	Enable referencing in follow-up mode
34110	REFP_CYCLE_NR	Axis sequence for channel-specific referencing
34120	REFP_BERO_LOW_ACTIVE	Polarity change of the BERO cam
34200	ENC_REFP_MODE	Referencing mode
34210	ENC_REFP_STATE	Status of absolute encoder
34220	ENC_ABS_TURNS_MODULO	Absolute encoder range for rotary encoders
34230	ENC_SERIAL_NUMBER	Encoder serial number
34232	EVERY_ENC_SERIAL_NUMBER	Range of the encoder serial number
34300	ENC_REFP_MARKER_DIST	Basic reference mark distance for distance-coded encoders
34310	ENC_MARKER_INC	Interval between two reference marks with distance-coded scales
34320	ENC_INVERS	Linear measuring system inverse to machine system
34330	REFP_STOP_AT_ABS_MARKER	Distance-coded linear measuring system without destination point
35150	SPIND_DES_VELO_TOL	Spindle speed tolerance
36302	ENC_FREQ_LIMIT_LOW	Encoder limit frequency resynchronization
36310	ENC_ZERO_MONITORING	Zero mark monitoring

S9: Setpoint switchover

11

11.1 Brief description

Function

The "Setpoint switchover" function is required if only one motor is to be used to drive several axes/spindles. For example, for milling heads, where the spindle motor is to be used to drive the tool as well as to orientate the milling head.

Requirements

The following conditions must be fulfilled so that the setpoint can be switched over:

- All of the axes involved in the setpoint switchover are at a standstill
- No sign-of-life error and no errors are active on PROFIBUS
- None of the following functions in the axis with drive control are active:
  - Reference point approach
  - Measuring
  - Travel to fixed stop
  - Function generator
  - Star/delta switchover
  - Parameter set switchover

Position control loop

The drive train, and therefore also the position control loop, are disconnected while the setpoint is being switched over. In order to avoid instability, the setpoint is only switched over at standstill - and once all controller enable signals have been cleared (deleted).

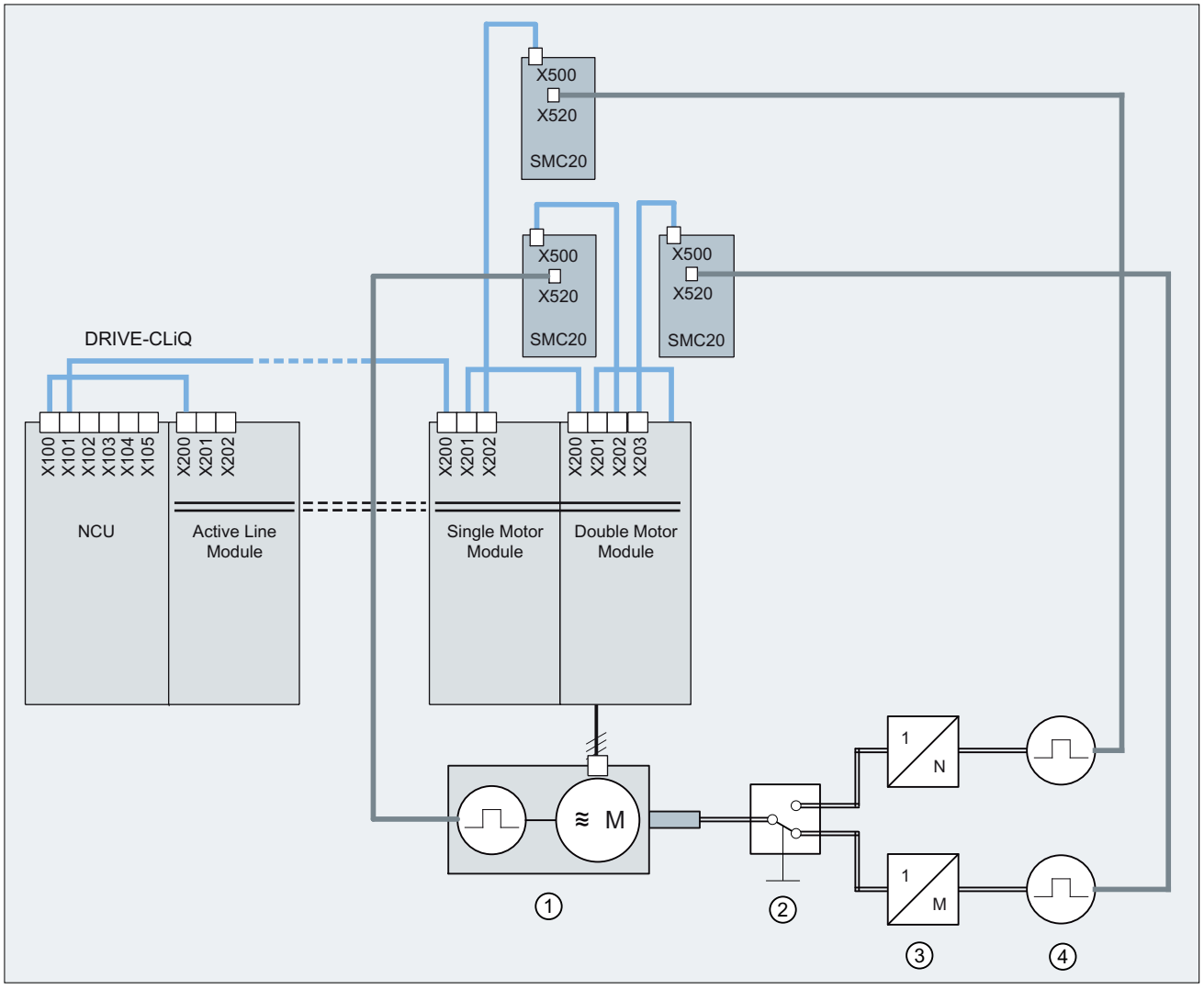
When using a single drive, only one control loop can be closed at the same instant in time. Axes without drive control are operated with open position controller and tracked positions.

Note

Vertical axes

The controller enable signals for all of the axes involved are automatically withdrawn by the control for the duration of the switchover. Axes without drive control are not in closed-loop control. Therefore, a brake control must be implemented for vertical axes.

Example



① Motor with encoder
② Mechanical switchover device
③ Gearbox 1
④ Encoder 1 (e.g. for the milling head)
⑤ Gearbox 2
⑥ Encoder 2 (e.g. for the spindle)

Figure 11-1 Setpoint switchover with 2 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.

11.2 Startup

Machine data

Setpoint assignment

The same setpoint channel of a drive is assigned a number of times to define the axes participating in the setpoint switchover. For this, the following machine data must be pre-assigned with the same logical drive number for each axis:

MD30110 $MA_CTRLOUT_MODULE_NR (setpoint assignment: module number)

Note

Alarm 26018 is output if the option is missing.

Encoder assignment

The encoder is assigned in the machine data on an axis-for-axis basis:

MD30230 $MA_ENC_INPUT_NR(actual value assignment: Input on drive module/measuring circuit module)

NC/PLC interface signals

Note

Access rights

Despite assignment of a drive to several machine axes, the use of NC/PLC interface signals remains unchanged. In the PLC user program this requires an explicit coordination of the access operations to the NC/PLC interface signal of the individual machine axes involved in the setpoint switchover.

Request for drive control

The request to assume drive control is realized via:

DB31, ... DBX24.5 (setpoint switchover: request drive control)

Status of the drive control

The current status of drive control is displayed via:

DB31, ... DBX96.5 (setpoint switchover: drive control active)

Status signals

After drive control is assumed for the first time for one of the machine axes involved in the setpoint switchover (DB31, ... DBX24.5 = 1 (setpoint switchover: switchover)) the following status signals are always displayed the same for all of the machine axes involved:

DB31, ... DBB92 - DBB95

Control signals

The control signals are only active in the machine axis that currently has the drive control (DB31, ... DBX96.5 == 1 (setpoint switchover active)):

DB31, ... DBB20 to DBB21

Controller enable

The controller enable (DB31, ... DBX2.1) is only active in the machine axis that currently has drive control (DB31, ... DBX96.5 == 1 (status setpoint switchover))

Follow-up mode

Machine axes **without** drive control, involved in the setpoint switchover, are always switched into the follow-up mode in the control itself (DB31, ... DBX61.3 == 0).

Note

Only one axis with drive control

At any one point in time, only one axis must have drive control DB31, ... DBX24.5 == 1 and DB31, ... DBX96.5 == 1.

Special cases

Several simultaneous requests to take over

No switchover takes place for several simultaneous takeover requests. The last axis used keeps drive control. The same effect applies if there are no takeover requests.

No takeover request when powering up

If there is no takeover request when the control powers up, then the **control** assigns the drive control to the first machine axis with the same logical drive number.

```
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 ; setpoint switchover: 1st
axis
MD30110 $MA_CTRLOUT_MODULE_NR[0,AX5] = 4 ; setpoint switchover: 2nd
axis
```

In order to be able to traverse the axis using the drive, the NC/PLC interface signal to request drive control (DB31, ... DBX24.5) must be set.

11.3 Flow diagram

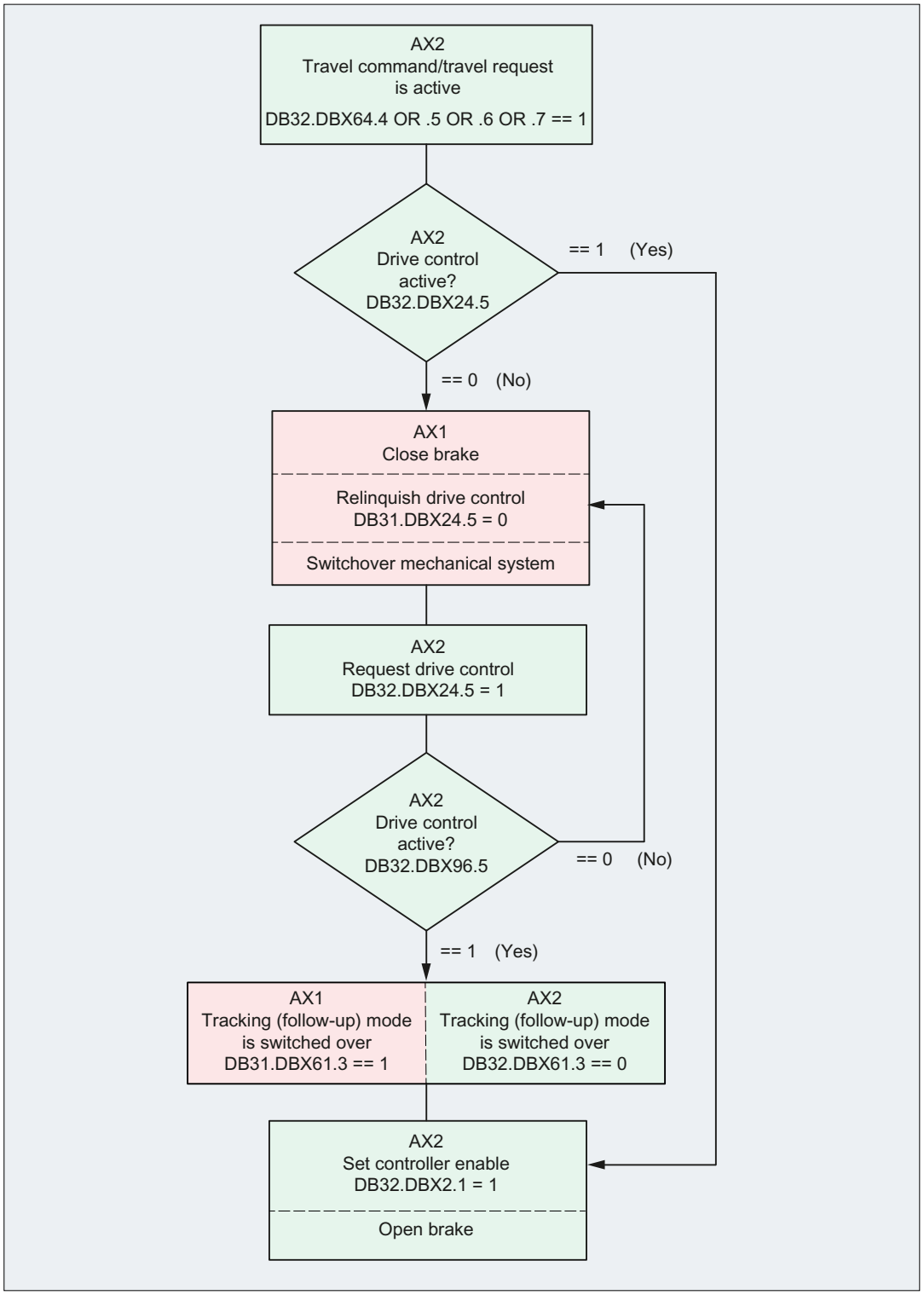


Figure 11-2 Sequence when switching over the setpoint from machine axes AX1 to AX2

11.4 Boundary conditions

Alarms

Only axes with active drive control display drive alarms.

Position control loop

The drive train, and therefore also the position control loop, are disconnected while the setpoint is being switched over. In order to avoid instability, the setpoint is only switched over at standstill - and once all controller enable signals have been cleared (deleted).

When using a single drive, only one control loop can be closed at the same instant in time. Axes without drive control are operated with open position controller and tracked positions.

Reference points

The use of load-side encoders does not affect the axis-specific reference points of a setpoint switchover.

However, the mechanical reference to the load can be lost following setpoint switchover for a load-side position derived solely from the motor encoder. These types of axis must be referenced again after every setpoint switchover.

Parameter set switchover

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

If, due to secondary conditions, a setpoint switchover is not performed, then a message is **not** displayed.

"Parking" state

If an axis is in the "Parking" state, this state can only be canceled if this axis assumes drive control.

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 only be used to commission the setpoint switchover via the expert list.

Safety Integrated

A detailed description of the supplementary conditions for setpoint switchover in conjunction with Safety Integrated is available in:

Further information:
SINUMERIK Safety Integrated manual

11.5 Data lists

11.5.1 Machine data

11.5.1.1 Axis/spindlespecific 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

B2: Acceleration and jerk

12

12.1 Brief description

12.1.1 General information

Scope of functions

The Description of Functions covers the following sub-functions:

- Acceleration
- Jerk
- Kneeshaped acceleration characteristic

Acceleration and jerk

The effective acceleration and jerk can be optimally matched to the machine and machining situation concerned using axis- and channel-specific programmable maximum values, programmable acceleration profiles in part programs and synchronized actions, and dynamic adaptations and limitations.

Kneeshaped acceleration characteristic

The knee-shaped acceleration characteristic means that, in the case of machine axes featuring a motor (in particular stepper motors) with a torque characteristic that is highly dependent upon speed, acceleration can be set at the level required to ensure optimum utilization of the motor whilst at the same time protecting it against overload.

12.1.2 Features

Acceleration

Axis-specific functions:

- Programmable maximum acceleration value
- Acceleration profile that can be selected via part-program instruction:
  Acceleration without jerk limitation (`BRISKA`)
- Setting of maximum value using part-program instruction (`ACC`)
- Specific maximum value for programmed rapid traverse (`G00`).
- Specific maximum value for traverse with active jerk limitation
- Excessive acceleration for non-tangential block transitions

Channel-specific functions:

- Acceleration profile that can be selected via part-program instruction:
  Acceleration without jerk limitation (BRISK)
- Programmable constant travel time for the purpose of avoiding extreme sudden acceleration
- Programmable acceleration margin for overlaid traversing
- Adjustable acceleration limitation
- Adjustable acceleration for specific real-time events
- Programmable acceleration margin for radial acceleration

Jerk

Axis-specific functions:

- Acceleration profile that can be selected via part-program instruction:
  Acceleration with jerk limitation (SOFTA)
- Programmable maximum jerk value for single-axis interpolation
- Programmable maximum jerk value for path interpolation

Channel-specific functions:

- Acceleration profile that can be selected via part-program instruction:
  Acceleration with jerk limitation (SOFT)
- Adjustable jerk limitation
- Adjustable path jerk for specific real-time events
- Specific maximum value for programmed rapid traverse (G00)
- Excessive jerk for block transitions without constant curvature

Kneeshaped acceleration characteristic

A knee-shaped acceleration characteristic is parameterized using the following characteristic data:

- Maximum velocity v_{max}
- Maximum acceleration a_{max}
- Creep velocity v_{red}
- Creep acceleration a_{red}
- Nature of the acceleration reduction (constant, hyperbolic, linear)

12.2 Functions

12.2.1 Acceleration without jerk limitation (BRISK/BRISKA) (channel/axis-specific)

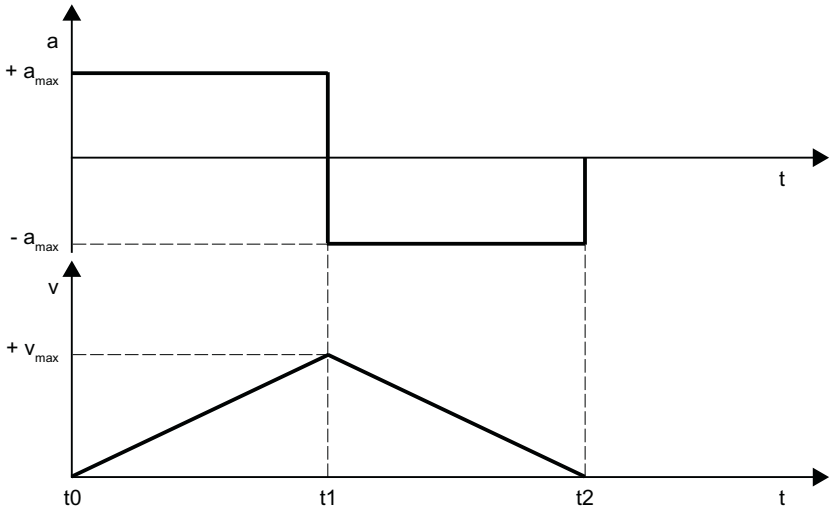
12.2.1.1 General Information

General Information

In the case of acceleration without jerk limitation (jerk = infinite) the maximum value is applied for acceleration immediately. As regard to acceleration with jerk limitation, it differs in the following respects:

- **Advantages**
  Shorter processing times with the same maximum values for velocity and acceleration.
- **Disadvantages**
  Increased load on the machine's mechanical components and risk of inducing high-frequency and difficult-to-control mechanical vibrations.

Acceleration profile



a_{max}: Maximum acceleration value

v_{max}: Maximum velocity value

t: Time

Figure 12-1 Velocity and acceleration schematic for stepped acceleration profile

The following features of the acceleration profile can be identified from the figure above:

- Time: t_0
  Sudden acceleration from 0 to $+a_{max}$
- Interval: t_0 - t_1
  Constant acceleration with $+a_{max}$; linear increase in velocity

- Time: t_1
  Sudden acceleration from 2 * a_{max} with immediate switchover from acceleration to braking

  Note

  The sudden acceleration can normally be avoided by specifying a constant velocity time (see Section "Constant traversing time (channel-specific) (Page 352)").

- Interval: t_1 - t_2
  Constant acceleration with -a_{max}; linear decrease in velocity

12.2.1.2 Parameterization

Maximum axial acceleration for path motions

The maximum axial acceleration for path motions can be set for the specific technology for each machine axis via the following machine data:

MD32300 $MA_MAX_AX_ACCEL[<parameter set index>]

With <parameter set index> = 0, 1, 2 ... (max. parameter set number - 1)

For the technology-specific parameter sets, see the Function Manual "Basic Functions", Section "Dynamic mode for path interpolation".

The path parameters are calculated by the path planning of the preprocessing so that the parameterized maximum values of the machine axes involved in the path are not exceeded.

Note

It is possible for the maximum value to be exceeded in connection with specific machining situations (see Sections "Acceleration adaptation (ACC) (axis-specific) (Page 354)" and "Path acceleration for real-time events (channel-specific) (Page 357)").

Maximum axial acceleration for positioning axis motions

With positioning axis motions, one of the two following maximum values is effective depending on the set positioning axis dynamic response mode:

- MD32300 $MA_MAX_AX_ACCEL [0] (maximum axial acceleration for path motions in the dynamic response mode DYNNORM)
- MD32300 $MA_MAX_AX_ACCEL [1] (maximum axial acceleration for path motions in the dynamic response mode DYNPOS)

The positioning axis dynamic response mode is set in the NC-specific machine data:

MD18960 $MN_POS_DYN_MODE = <mode>

<mode>	Meaning
0	Effective maximum axial acceleration: MD32300 $MA_MAX_AX_ACCEL[0]
1	Effective maximum axial acceleration: MD32300 $MA_MAX_AX_ACCEL[1]

Maximum axial acceleration for JOG motions

See Section "Acceleration and jerk for JOG motions (Page 388)".

12.2.1.3 Programming

Path acceleration without jerk limitation (BRISK)

Syntax

```
BRISK
```

Functionality

The `BRISK` part-program instruction is used to select the "without jerk limitation" acceleration profile for the purpose of path acceleration.

G group: 21

Effective: Modal

Reset response

The channel-specific initial setting is activated via a reset:

MD20150 $MC_GCODE_RESET_VALUES[20]

Supplementary conditions

If the acceleration profile is changed in a part program during machining (`BRISK`/`SOFT`) an exact stop is performed at the end of the block.

Single-axis acceleration without jerk limitation (BRISKA)

Syntax

`BRISKA (`*axis{,axis}*`)`

Function

The `BRISKA` part-program command is used to select the "without jerk limitation" acceleration profile for single-axis movements (JOG, JOG/INC, positioning axis, reciprocating axis, etc.).

G group: -

Effectiveness: Modal

Axis:

- Value range: Axis name of the channel axes

Axis-specific initial setting

Acceleration without jerk limitation can be set as the axis-specific initial setting for single-axis movements:

MD32420 $MA_JOG_AND_POS_JERK_ENABLE = FALSE

Reset behavior

The axis-specific initial setting is activated via a reset:

MD32420 $MA_JOG_AND_POS_ENABLE

12.2.2 Constant traversing time (channel-specific)

12.2.2.1 General Information

Overview

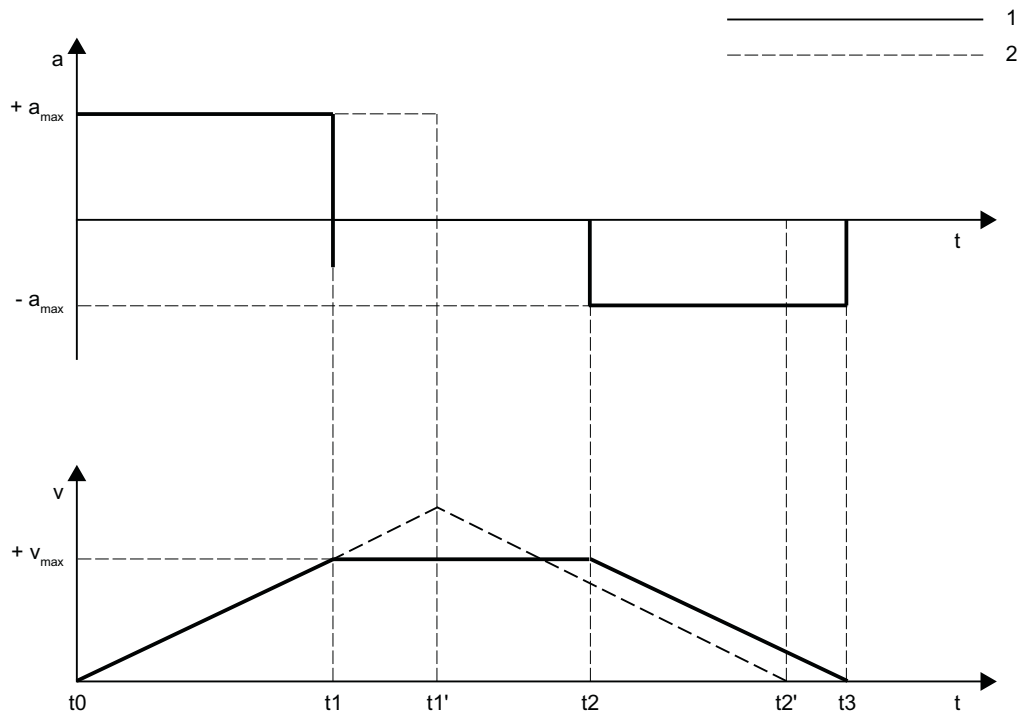
For acceleration without jerk limitation, an acceleration jump of 2 * a_{max} occurs when changing from acceleration and braking. To prevent this acceleration jump, a channel-specific constant traversing time can be parameterized. The constant traversing time specifies the time for constant velocity traversing between the acceleration and the braking phase:

MD20500 $MC_CONST_VELO_MIN_TIME (minimum time with constant velocity)

Note

Constant traversing time does not act for:

- Active function: LookAhead
- In traversal blocks with a traversal time less than or equal to the interpolation cycle.



1: Curve with constant traversing time

2: Curve without constant traversing time

a_{max}: Maximum acceleration value

v_{max}: Maximum velocity value

t: Time

Figure 12-2 Principle characteristic for an abrupt acceleration

The above figure shows the effect of the constant traversing time:

- Instant: t_1
  End of the acceleration phase with acceleration jump 1 * a_{max}
- Interval: t_1 - t_2
  Acceleration 0; constant velocity over the parameterized constant traversing time
- Instant: t_2
  Start of the braking phase with acceleration jump 1 * a_{max}

The instants t_0, t_1' and t_2' indicate the associated curve that would have resulted without constant traversing time.

12.2.2.2 Parameterization

The constant travel time is parameterized for specific channels using machine data:

MD20500 $MC_CONST_VELO_MIN_TIME
(minimum time with constant velocity)

12.2.3 Acceleration adaptation (ACC) (axis-specific)

12.2.3.1 General Information

Function

Using the ACC command, the currently effective maximum axis acceleration parameterized in the acceleration-specific machine data can be reduced for a specific axis. The reduction is in the form of a percentage factor, which is specified when programming the command.

For instance, reducing the maximum possible axis acceleration in a machining segment can be used to prevent mechanical vibration as a result of high machining forces.

Effectiveness

Reducing the acceleration with ACC is **effective** for all interpolation types in the AUTOMATIC and MDI operating modes – and for the dry run feedrate.

Acceleration reduction is **not effective**:

- in the JOG mode
- During the machine function REF (reference point approach).
- If, as a result of a fault that has been detected, the axis is stopped with a fast stop (setpoint = 0).

12.2.3.2 Programming

Syntax

```
ACC[<axis>]=<reduction factor>
ACC[SPI(<spindle number>)]=<reduction factor>
ACC(S<spindle number>)=<reduction factor>
```

Meaning

<ACC>:	Command for the axis-specific reduction of the currently maximum possible acceleration, derived from the machine data	
	Effectiveness:	Modal
<axis>:	Channel axis name of path axis	
	Data type:	AXIS
	Value range:	Channel axis names
SPI(<spindle number>):	The SPI(...) function converts the spindle number into the corresponding channel axis name.	
S<spindle number>:	Spindle name in the channel	

Further information

System variable

The acceleration reduction (set using ACC), currently active in the channel, can be read on an axis-for-axis basis using:

```
$AA_ACC[<axis>]
```

Reset response

The acceleration reduction set using ACC can be kept after a channel reset or after the end of the program. The parameterization is performed via:

MD32320 $MA_DYN_LIMIT_RESET_MASK

12.2.4 Acceleration reserve (channel-specific)

12.2.4.1 General Information

General information

Under normal circumstances, preprocessing makes maximum use of the parameterized maximum values of the machine axes for the purpose of path acceleration. In order that an acceleration margin may be set aside for overlaid movements, e.g., within the context of the "Rapid lift away from the contour" function, path acceleration can be reduced by a programmable factor. When, for example, a factor of 0.2 is applied, preprocessing will only use 80% of the maximum possible path acceleration. 20% is set aside as an acceleration margin for overlaid movements.

12.2.4.2 Parameterization

Parameters for the acceleration margin are assigned for each channel by means of machine datum:
MD20610 $MC_ADD_MOVE_ACCEL_RESERVE
(acceleration margin for overlaid motions)

12.2.5 Limiting the path acceleration (channel-specific)

12.2.5.1 General Information

General Information

To enable a flexible response to the machining situations concerned, setting data can be used to limit the path acceleration calculated during preprocessing for specific channels:

SD42500 $SC_SD_MAX_PATH_ACCEL (maximum path acceleration)

The value specified in the setting data is only taken into account if it is smaller than the path acceleration calculated during preprocessing.

The limitation must be activated for specific channels using setting data:

SD42502 $SC_IS_SD_MAX_PATH_ACCEL = TRUE

12.2.5.2 Parameterization

Parameterization is carried out for specific channels using setting data:

SD42500 $SC_SD_MAX_PATH_ACCEL (maximum path acceleration)

SD42502 $SC_IS_SD_MAX_PATH_ACCEL (activation of path-acceleration limitation)

12.2.5.3 Programming

Limit value

Syntax

```
$SC_SD_MAX_PATH_ACCEL = limit value
```

Functionality

The path-acceleration limitation can be adjusted for the situation by programming the setting data.

Limit value:

- Value range: ≥ 0
- Unit: m/s^2

Application:

- Part program
- Static synchronized action

Switch ON/OFF

Syntax

```
$SC_IS_SD_MAX_PATH_ACCEL = value
```

Functionality

The path-acceleration limitation can be activated/deactivated by programming the setting data.

Parameter: *Value*

- Value range: TRUE, FALSE

Application:

- Part program
- Static synchronized action

12.2.6 Path acceleration for real-time events (channel-specific)

12.2.6.1 General Information

General Information

So that no compromise has to be made between machining-optimized acceleration on the one hand and time-optimized acceleration in connection with the following real-time events on the other:

- NC Stop / NC Start
- Changing the feedrate override
- Changing the velocity default for "safely reduced velocity" within the context of the "Safety Integrated" function

For the real-time events mentioned above, the path acceleration can be specified using a channel-specific system variable:

`$AC_PATHACC =` *path acceleration*

Real-time event acceleration will only be active for the duration of the change in velocity in respect of one of the real-time events specified above.

Limitation

If the specified path acceleration exceeds the capabilities of the machine axes that are of relevance for the path, a limit will be imposed on the path acceleration within the controller so that the resulting axial acceleration (a_{res}) is restricted to less than 2x the parameterized maximum axial value (a_{max}).

$a_{res} = 2 * a_{max,}$ with a_{max} = MD32300 $MA_MAX_AX_ACCEL

Note

Path acceleration for real-time events is enabled, irrespective of the radial acceleration.

Effectiveness

Effective	Real-time event acceleration is only enabled in AUTOMATIC and MDA operating modes in conjunction with the following real-time events: • NC Stop / NC Start • Override changes • Changing the velocity default for "safely reduced velocity" within the context of the "Safety Integrated" function
Not effective	Path acceleration for real-time events is ineffective for changes in path velocity that are attributable to path planning during preprocessing for the channel, such as contour curvatures, corners, kinematic transformation limitations, etc. Real-time-event path acceleration is ineffective if the programmed value is smaller than the path acceleration calculated during preprocessing for the path section concerned.

Programming

For information about programming system variables in the part program or synchronized actions, see Section "Programming (Page 358)".

12.2.6.2 Programming

Syntax

`$AC_PATHACC =` *path acceleration*

Functionality

Real-time-event path acceleration is set via the channel-specific system variables.

Parameter: *Path acceleration*

- Value range: Path acceleration ≥ 0
- Unit: m/s^2

Deactivation: `$AC_PATHACC = 0`

Application:

- Part program
- Static synchronized action

Reset response

Real-time-event path acceleration is deactivated on reset.

Supplementary conditions

Programming `$AC_PATHACC` in the part program automatically triggers a preprocessing stop with REORG (`STOPRE`).

12.2.7 Acceleration with programmed rapid traverse (G00) (axis-specific)

12.2.7.1 General Information

Frequently, the acceleration for the machine axes involved in the machining process must be set lower than the machine's performance capability officially allows because of the supplementary conditions associated with the specific process concerned.

For time-optimized traversing of the machine axes with programmed rapid traverse (part-program instruction G00), a specific maximum value can be programmed for the axis-specific acceleration.

JOG setup mode

This function does not affect acceleration in respect of a rapid traverse override in JOG setup mode.

12.2.7.2 Parameterization

The maximum value for axis-specific acceleration with programmed rapid traverse is parameterized (G00) using the axis-specific machine data:

MD32434 $MA_G00_ACCEL_FACTOR
(scaling of the acceleration limitation with G00)

This is used to generate the maximum value for axis-specific acceleration with programmed rapid traverse (G00) that is taken into account by the path planning component during preprocessing:

Acceleration[axis] =
MD32300 $MA_MAX_AX_ACCEL * MD32434 $MA_G00_ACCEL_FACTOR

12.2.8 Acceleration with active jerk limitation (SOFT/SOFTA) (axis-specific)

12.2.8.1 General Information

Function

Compared with acceleration without jerk limitation, acceleration with jerk limitation results in a certain degree of time loss, even when the same maximum acceleration value is used. To compensate for this time loss, a specific maximum value can be programmed for the axis-specific acceleration as far as traversing of the machine axes with active jerk limitation (SOFT/SOFTA) is concerned.

The maximum value for acceleration with active jerk limitation is parameterized using a factor calculated in relation to the axis-specific maximum value. This is used to generate the maximum value for axis-specific acceleration with active jerk limitation that is taken into account by the path planning component during preprocessing:

Acceleration[axis] =
MD32300 $MA_MAX_AX_ACCEL * MD32433 $MA_SOFT_ACCEL_FACTOR

12.2.8.2 Parameterization

The maximum value for acceleration with active jerk limitation (SOFT/SOFTA) is parameterized using the axis-specific machine data:

MD32434 $MA_SOFT_ACCEL_FACTOR
(scaling of the acceleration limitation with SOFT)

12.2.9 Exceeding the acceleration with non-tangential block transitions (axis-specific)

12.2.9.1 General Information

Function

In the case of non-tangential block transitions (corners), the programmable controller may have to decelerate the geometry axes significantly in order to ensure compliance with the parameterized axis dynamics. For the purpose of reducing/avoiding deceleration in connection with non-tangential block transitions, a higher level of axis-specific acceleration can be enabled.

Excessive acceleration is parameterized using a factor calculated in relation to the axis-specific maximum value. This is used to generate the maximum value for axis-specific acceleration with non-tangential block transitions that is taken into account by the path planning component during preprocessing:

Acceleration[axis] =
MD32300 $MA_MAX_AX_ACCEL * MD32310 $MA_MAX_ACCEL_OVL_FACTOR

12.2.9.2 Parameterization

Excessive acceleration for non-tangential block transitions is parameterized using the axis-specific machine data:
MD32310 $MA_MAX_ACCEL_OVL_FACTOR
(overload factor for velocity jumps)

12.2.10 Acceleration reserve for the radial acceleration (channel-specific)

12.2.10.1 General Information

Overview

In addition to the path acceleration (tangential acceleration), radial acceleration also has an effect on curved contours. If this is not taken into account during parameterization of the path parameters, the effective axial acceleration during acceleration and deceleration on the curved contour can, for a short time, reach 2x the maximum value.

Effective axial acceleration =
Path acceleration + radial acceleration =
2 * (MD32300 $MA_MAX_AX_ACCEL)

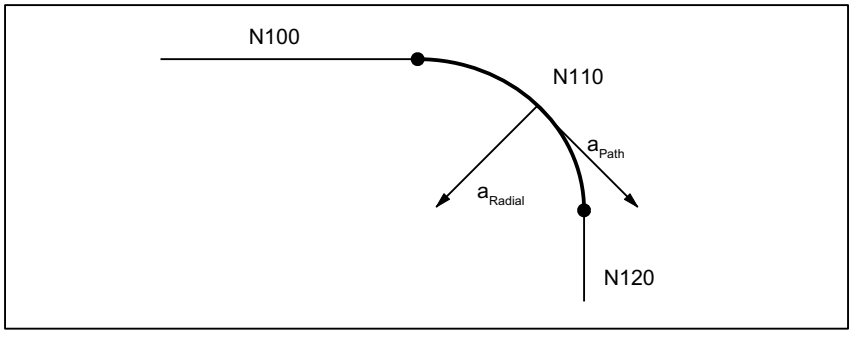


Figure 12-3 Radial and path acceleration on curved contours

The channel-specific machine data:
MD20602 $MC_CURV_EFFECT_ON_PATH_ACCEL
(influence of path curvature on dynamic path response)
can be used to set the proportion of the axis-specific acceleration that is to be taken into account for radial acceleration.

When, for example, a value of 0.75 is applied, 75% of the axis-specific acceleration will be made available for radial acceleration and 25% for path acceleration.

The corresponding maximum values are generally calculated as follows:

Radial acceleration =
MD20602 $MC_CURV_EFFECT_ON_PATH_ACCEL * MD32300 $MA_MAX_AX_ACCEL

Path acceleration =
(1 - MD20602 $MC_CURV_EFFECT_ON_PATH_ACCEL) * MD32300 $MA_MAX_AX_ACCEL

Example

The following machine parameters apply:

- MD32300 $MA_MAX_AX_ACCEL for all geometry axes: 3 m/s
- Maximum path velocity with a path radius of 10 mm due to mechanical constraints of the machine: 5 m/min.

The radial acceleration is calculated as follows:

$$a_{Radial} = \frac{v^2_{Path}\,[m/min]}{r\,[mm] * 3.6\,[m/s^2]} = \frac{5^2}{10 * 3.6} = 0.694\ m/s^2$$

The acceleration margin is set as follows:

$$\text{MD20602 \$MC_CURV_EFFECT_ON_PATH_ACCEL} = \frac{a_{Radial}\,[m/s^2]}{\text{MD32300 \$MA_MAX_AX_ACCEL}\,[m/s^2]} = \frac{0.694}{3} \approx 0.23$$

Linear motions

The acceleration margin referred to above is ineffective in the case of linear motions (linear interpolation) without active kinematic transformation.

12.2.10.2 Parameterization

The proportion of maximum available axis acceleration to be taken into account as an acceleration margin for radial acceleration on curved contours is parameterized using the channel-specific machine data:

MD20602 $MC_CURV_EFFECT_ON_PATH_ACCEL
(influence of path curvature on dynamic path response)

12.2.11 Jerk limitation with path interpolation (SOFT) (channel-specific)

12.2.11.1 General Information

Overview

As far as the functionality described in the rest of this document is concerned, constant acceleration, i.e., acceleration with jerk limitation (jerk = infinite value), is the assumed acceleration profile. In the case of acceleration with jerk limitation, linear interpolation is applied in respect of acceleration from 0 to the maximum value.

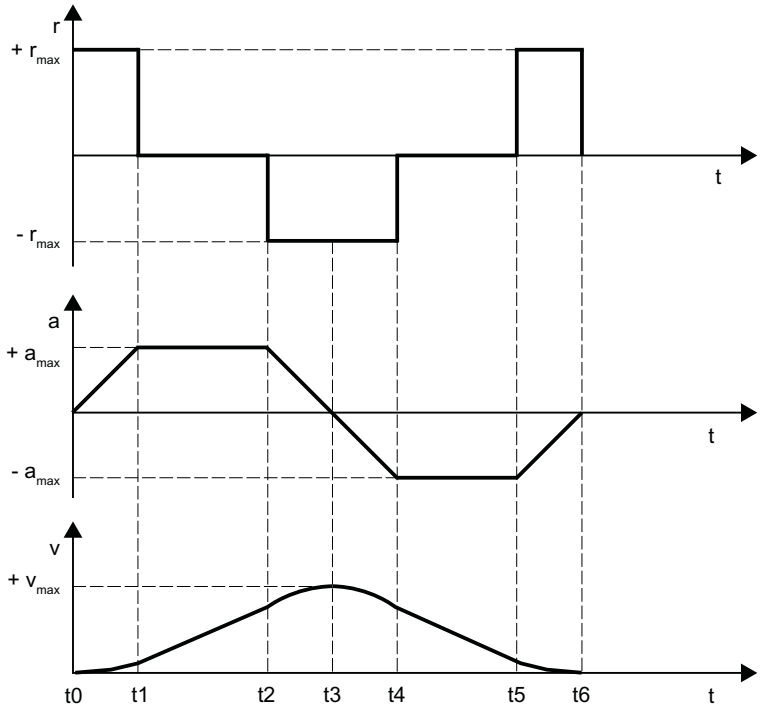
Advantages

Minimal load on the machine's mechanical components and low risk of high-frequency and difficult-to-control mechanical vibrations thanks to constant excessive acceleration.

Disadvantages

Longer machining times compared with stepped acceleration profile when the same maximum velocity and acceleration values are used.

Acceleration profile



r_{max}: Maximum jerk value

a_{max}: Maximum acceleration value

v_{max}: Maximum velocity value

t: Time

Figure 12-4 Jerk, acceleration and velocity schematic with jerk limitation acceleration profile

The following features of the acceleration profile can be identified from the figure above:

- Interval: t_0 - t_1
  Constant jerk with $+r_{max}$; linear increase in acceleration; quadratic increase in velocity
- Interval: t_1 - t_2
  Constant acceleration with $+a_{max}$; linear increase in velocity
- Interval: t_2 - t_3
  Constant jerk with $-r_{max}$; linear decrease in acceleration; quadratic decrease in excessive velocity until maximum value $+v_{max}$ is reached
- Interval: t_3 - t_4
  Constant jerk with $+r_{max}$; linear increase in braking acceleration; quadratic decrease in velocity
- Interval: t_4 - t_5
  Constant braking acceleration with $-a_{max}$; linear decrease in velocity
- Interval: t_5 - t_6
  Constant jerk with $-r_{max}$; linear decrease in braking acceleration; quadratic decrease in velocity reduction until zero velocity is reached v = 0

12.2.11.2 Parameterization

Maximum jerk value for path motions (axis-specific)

The maximum axial jerk for path motions can be set for the specific technology for each machine axis via the following machine data:

MD32431 $MA_MAX_AX_JERK[<parameter set index>]

With <parameter set index> = 0, 1, 2 ... (max. parameter set number - 1)

For the technology-specific parameter sets, see the Function Manual "Basic Functions", Section "Dynamic mode for path interpolation".

The path parameters are calculated by the path planning of the preprocessing so that the parameterized maximum values of the machine axes involved in the path are not exceeded.

Note

It is possible for the maximum value to be exceeded in connection with specific processing situations (see Section "Path jerk for real-time events (channel-specific) (Page 369)").

Maximum jerk value for path motions (channel-specific)

In addition to the axis-specific setting, the maximum jerk value can also be specified as channel-specific path parameter via the following machine data:

MD20600 $MC_MAX_PATH_JERK (path-related maximum jerk)

In order to exclude the mutual influencing of axis and channel-specific maximum jerk values, the channel-specific maximum value must be set to a value greater than the axial maximum values.

12.2.11.3 Programming

Syntax

```
SOFT
```

Functionality

The `SOFT` part-program instruction is used to select the acceleration profile with jerk limitation for the traversing operations of geometry axes in the channel.

G group: 21

Effective: Modal

Reset response

The channel-specific initial setting is activated via a reset:

MD20150 $MC_GCODE_RESET_VALUES[20]

Boundary conditions

If the acceleration mode is changed in a part program during machining (BRISK ↔ SOFT), a block change is performed at the point of transition with an exact stop at the end of the block, even in continuous-path mode.

12.2.12 Jerk limitation with single-axis interpolation (SOFTA) (axis-specific)

12.2.12.1 Parameterization

Basic setting of axis-specific jerk limitation

Acceleration with jerk limitation can be set as the axis-specific initial setting:

MD32420 $MA_JOG_AND_POS_JERK_ENABLE== TRUE

Maximum axis-specific jerk for positioning axis movements

When traversing positioning axes with active jerk limitation, the value from one of the following machine data takes effect as the maximum axis-specific jerk:

- MD32430 $MA_JOG_AND_POS_MAX_JERK (maximum axis-specific jerk)
- MD32431 $MA_MAX_AX_JERK [0] (maximum axis-specific jerk for path motions in the dynamic response mode DYNNORM)
- MD32431 $MA_MAX_AX_JERK [1] (maximum axis-specific jerk for path motions in dynamic response mode DYNPOS)

The machine data to be used is determined by the set positioning axis dynamic response mode:

MD18960 $MN_POS_DYN_MODE = <mode>

<mode>	Meaning
	The following applies as the maximum axis-specific jerk for positioning axis motions:
0	MD32430 $MA_JOG_AND_POS_MAX_JERK With active G75 (fixed-point approach): MD32431 $MA_MAX_AX_JERK[**0**]
1	MD32431 $MA_MAX_AX_JERK[**1**]

Maximum axial jerk for JOG motions

See Chapter "Acceleration and jerk for JOG motions (Page 388)".

12.2.12.2 Programming

Syntax

SOFTA (*Axis* {*Axis*})

Functionality

The SOFTA part-program command is used to select acceleration with jerk limitation for single-axis movements (positioning axis, reciprocating axis, etc.)

G group: -

Effectiveness: Modal

Axis:

- Value range: Axis name of the channel axes

Axis-specific initial setting

Acceleration with jerk limitation can be set as the axis-specific initial setting for single-axis movements:

MD32420 $MA_JOG_AND_POS_JERK_ENABLE = TRUE

Reset behavior

The axis-specific initial setting is activated via a reset:

MD32420 $MA_JOG_AND_POS_ENABLE

12.2.13 Limiting the path jerking (channel-specific)

12.2.13.1 General Information

Overview

To enable a flexible response to the machining situations concerned, setting data can be used to limit the path jerk calculated during preprocessing for specific channels:

SD42510 $SC_SD_MAX_PATH_JERK (maximum path jerk)

The value specified in the setting data is only taken into account in the channel if it is smaller than the path jerk calculated during preprocessing.

The limitation must be activated for specific channels using setting data:

SD42512 $SC_IS_SD_MAX_PATH_JERK = TRUE

12.2.13.2 Parameterization

Parameterization is carried out for specific channels using setting data:

SD42510 $SC_SD_MAX_PATH_JERK (maximum path jerk)

SD42512 $SC_IS_SD_MAX_PATH_JERK
(activation of path-jerk limitation)

12.2.13.3 Programming

Maximum path jerk

Syntax

$SC_SD_MAX_PATH_JERK = *jerk value*

Functionality

The path-jerk limitation can be adjusted for the situation by programming the setting data.

Jerk value:

- Value range: ≥ 0
- Unit: m/s^3

Application:

- Part program
- Static synchronized action

Switch ON/OFF

Syntax

$SC_IS_SD_MAX_PATH_JERK = *value*

Functionality

The path-jerk limitation can be activated/deactivated by programming the setting data.

Parameter: *Value*

- Value range: TRUE, FALSE

Application:

- Part program
- Static synchronized action

12.2.14 Path jerk for real-time events (channel-specific)

12.2.14.1 General Information

Overview

So that no compromise has to be made between machining-optimized jerk on the one hand and time-optimized jerk in connection with the following real-time events on the other:

- NC Stop / NC Start
- Changing the feedrate override
- Changing the velocity default for "safely reduced velocity" within the context of the "Safety Integrated" function

for the real-time events mentioned, the path jerk can be specified using a channel-specific system variable:

$AC_PATHJERK = *path jerk*

Path jerk for real-time events will only be active for the duration of the change in velocity in respect of one of the real-time events specified above.

Limitation

As the jerk is not a physical variable of any relevance to the drive, **no** limit is imposed on the jerk set.

Effectiveness

Effective	Path jerk for real-time events is only enabled in AUTOMATIC and MDA operating modes in conjunction with the following real-time events: • NC Stop / NC Start • Override changes • Changing the velocity default for "safely reduced velocity" within the context of the "Safety Integrated" function
Not effective	Path jerk for real-time events is ineffective for changes in the path velocity that are attributable to path planning during preprocessing for the channel, such as contour curvatures, corners, kinematic transformation limitations, etc. Path jerk for real-time events is ineffective if the programmed value is smaller than the path jerk calculated during preprocessing for the path section concerned.

Programming

For the purpose of setting the jerk for real-time events in accordance with the acceleration, the system variables can be set as follows:

$AC_PATHJERK = $AC_PATHACC/smoothing time

- $AC_PATHACC: Path acceleration [m/s^2]
  Smoothing time: Freely selectable, e.g. 0.02 s

For information about programming system variables in the part program or synchronized actions, see Section "Programming (Page 370)".

12.2.14.2 Programming

Syntax

$AC_PATHJERK = *path jerk*

Functionality

The path jerk for real-time events is set via the channel-specific system variables.

Jerk value:

- Value range: Path jerk ≥ 0
- Unit: m/s^3

Application:

- Part program
- Static synchronized action

Reset behavior

The function is deactivated on reset.

Boundary conditions

Programming $AC_PATHJERK in the part program automatically triggers a preprocessing stop with REORG (`STOPRE`).

12.2.15 Jerk during programmed rapid traverse (G00) (axis-specific)

12.2.15.1 General Information

Overview

Frequently, the maximum jerk for the machine axes involved in the machining process must be set lower than the machine's performance capability officially allows because of the supplementary conditions associated with the specific process concerned.

For time-optimized traversing of the machine axes with programmed rapid traverse (part-program instruction G00), a specific maximum value can be programmed for the axis-specific jerk.

JOG setup mode

This function does not affect jerk in respect of a rapid traverse override in JOG setup mode.

12.2.15.2 Parameterization

The maximum value for axis-specific jerk with programmed rapid traverse is parameterized (G00) using the axis-specific machine data:

MD32434 $MA_G00_ACCEL_FACTOR
(scaling of the acceleration limitation with G00)

This is used to generate the maximum value for axis-specific jerk with programmed rapid traverse (G00) that is taken into account by the path planning component during preprocessing:

Jerk[axis] =
MD32431 $MA_MAX_AX_JERK * MD32435 $MA_G00_JERK_FACTOR

12.2.16 Excessive jerk for block transitions without constant curvature (axis-specific)

12.2.16.1 General Information

Overview

In the case of block transitions without constant curvature (e.g. straight line > circle), the programmable controller has to decelerate movement of the geometry axes significantly in order to ensure compliance with the parameterized axis dynamics. For the purpose of reducing/avoiding deceleration in connection with block transitions without constant curvature, a higher level of axis-specific jerk can be enabled.

The excessive jerk is parameterized using a dedicated axis-specific maximum value.

12.2.16.2 Parameterization

The excessive jerk for block transitions without constant curvature is parameterized using the axis-specific machine data:

MD32432 $MA_PATH_TRANS_JERK_LIM
(excessive jerk for block transitions without constant curvature)

12.2.17 Velocity-dependent jerk adaptation (axis-specific)

Function

The dynamic path response results from the parameterized, constant axial maximum values for velocity, acceleration and jerk of the axes involved in the path:

- MD32000 $MA_MAX_AX_VELO (maximum axis velocity)
- MD32300 $MA_MAX_AX_ACCEL (maximum axis acceleration)
- MD32431 $MA_MAX_AX_JERK (max. axial jerk for path motion)

For **contours with non-constant curvature** (torsion), as for example in connection with free-form surfaces, fluctuations in the path velocity, particularly in the upper velocity range, can result mainly due to the axial jerk. The fluctuations of the path velocity lead to adverse affects in the surface quality.

The influence of the axial jerk on the path velocity is decreased for contours with non-constant curvature through a velocity-dependent increase of the permissible axial jerk. Fluctuations in the path velocity can be avoided with the appropriate parameterization.

The velocity-dependent increase of the permissible axial jerk has no effect on the maximum possible path acceleration and path jerk. These result from the constant axial maximum values parameterized in the in the machine data even when jerk adaptation is active.

As both curvature and torsion are zero in the case of linear motion, the velocity-dependent jerk adaptation has no effect with linear motions.

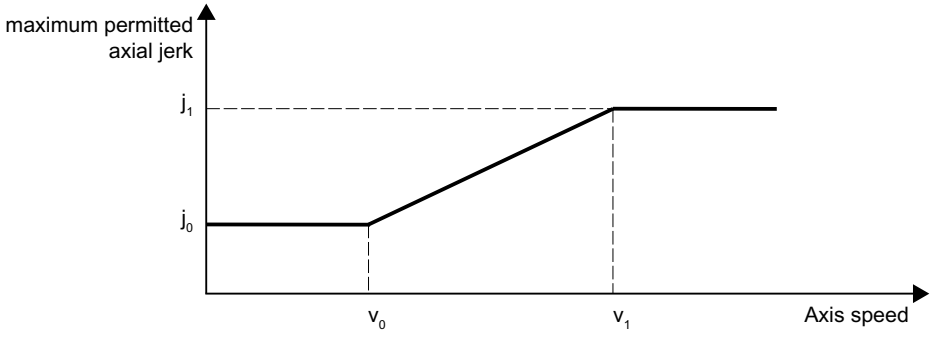
Availability

The "velocity-dependent jerk adaptation" function is available independent of the function "Freeform surface mode: Basic functions". See Function Manual "Basic Functions", Section "Freeform Surface mode: Basic functions".

Parameterization

The "Velocity-dependent jerk adaptation" function is parameterized with the following machine data:

- MD32437 $MA_AX_JERK_VEL0[<n>] = <threshold value$_{lower}$>
  Lower velocity threshold of the jerk adaptation. Velocity-dependent jerk adaptation takes effective as of this velocity.
  The lower speed threshold can be set separately via the Index n, for each dynamic mode (see Function Manual "Basic Functions", Section "Dynamic mode for path interpolation"):
- MD32438 $MA_AX_JERK_VEL1[<n>] = <threshold value$_{upper}$>
  Upper velocity threshold of the jerk adaptation. The velocity-dependent jerk reaches its maximum value j_{max} parameterized with MD32439 $MA_MAX_AX_JERK_FACTOR at this velocity.
  The upper speed threshold can be set separately via the Index n, for each dynamic mode (see Function Manual "Basic Functions", Section "Dynamic mode for path interpolation"):
- MD32439 $MA_MAX_AX_JERK_FACTOR = <factor>
  Factor for the parameterization of the maximum velocity-dependent jerk j_{max} on reaching the upper velocity threshold MD32438 $MA_AX_JERK_VEL1[<n>]:
  j_{max} = (MD32431 $MA_MAX_AX_JERK) * (MD32439 $MA_MAX_AX_JERK_FACTOR)
  The velocity-dependent jerk adaptation is active at a value > 1.0.
  The velocity-dependent jerk adaptation is inactive at a value = 1.0.



v_0: MD32437 $MA_AX_JERK_VEL0

v_1: MD32438 $MA_AX_JERK_VEL1

j_0: MD32431 $MA_MAX_AX_JERK

j_1: MD32439 $MA_MAX_AX_JERK_FACTOR * MD32431 $MA_MAX_AX_JERK

Figure 12-5 Axial jerk as a function of the axis velocity

Note

The velocity-dependent jerk adaptation is only active, if:

MD32439 $MA_MAX_AX_JERK_FACTOR > 1.0

Example

Example of parameter assignment:

- MD32437 $MA_AX_JERK_VEL0 = 3000 mm/min
- MD32438 $MA_AX_JERK_VEL1 = 6000 mm/min
- MD32439 $MA_MAX_AX_JERK_FACTOR[AX1] = 2.0
- MD32439 $MA_MAX_AX_JERK_FACTOR[AX2] = 3.0
- MD32439 $MA_MAX_AX_JERK_FACTOR[AX3] = 1.0

Effect

- The speed-dependent jerk adaptation is active for the 1st and 2nd axis, whereas the function for the 3rd axis is not active.
- The parameterized jerk is effective at axis velocities in the range of 0 to 3000 mm/min.
- The maximum jerk is linearly increased for axis velocities in the range 3000 mm/min to 6000 mm/min.
- The maximum permitted jerk of the 1st axis is, for axis velocities greater than 6000 mm/min, increased by a factor of 2 - for the 2nd axis, by factor of 3.
- The parameterized values apply in each dynamic response mode.

12.2.18 Jerk filter (position setpoint filter, axis-specific)

12.2.18.1 General Information

Overview

In certain application scenarios, e.g. when milling free-form surfaces, it may be beneficial to smooth the position setpoint characteristics of the machine axes. This enables the surface quality to be improved by reducing the mechanical vibration generated in the machine.

To smooth the position setpoint characteristic of a machine axis, a jerk filter can be activated at the position controller level, independent of channel- and axis-specific jerk limitations that are taken into account at interpolator level. Jerk limitation can be set using the following machine data, for example:

- MD20600 $MA_MAX_PATH_JERK (pathrelated maximum jerk)
- MD32431 $MA_MAX_AX_JERK (maximum axial jerk for path motion)

All position setpoint filter types have a lowpass characteristic, and always round a contour at the center of the curvature (towards the inside). Larger time constants or lower filter frequencies round the contour more significantly; however, that also means that machine resonance points (machine oscillation) are excited less. A compromise must always be found between an adequate damping effect and contour accuracy.

The damping effect can be monitored using the positioning response, the contour accuracy, using a circularity test, for example.

Various filter types are available to enable the jerk filter to be optimally matched to the specific machine characteristics:

- 2nd order filter (PT2)
- Moving average calculation
- Band-stop
- Double moving average calculation

Filter type: 2nd order filter

Because it is a simple low-pass filter, a "2nd order filter" mode can only meet the requirements specified above where relatively small filter time constants (around 10 ms) are involved. When used in conjunction with larger time constants, inadmissible contour deviations soon become apparent. The filter effect is relatively low.

For historical reasons, this filter mode is set as the default.

Filter type: Moving average calculation

Where low contour deviations are required, filter time constants within the range of 20 - 40 ms can be set using the "moving average calculation" filter mode. The smoothing effect is largely symmetrical.

The display of the calculated servo gain factor (K_V) in the user interface, indicates smaller values than would normally be appropriate for the filter. The contour accuracy is higher than the displayed servo gain factor (K_V) appears to suggest.

When changing from "2nd-order filter" to "moving average calculation" filter mode, the displayed servo gain factor (K_V) may therefore decrease (with identical filter time constant), even though there is an improvement in contour accuracy.

Filter type: Band-stop

The use is not recommended and is (still) only available for compatibility reasons.

The band-stop filter is a 2nd order filter in terms of numerator and denominator:

$$H(s) = \frac{\frac{s^2}{(2 * \pi * f_Z)^2} + \frac{2 * s * D_Z}{(2 * \pi * f_Z)}}{\frac{s^2}{(2 * \pi * f_N)^2} + \frac{2 * s * D_N}{2 * \pi * f_N}}$$

With:

f_N: Numerator natural frequency

f_D: Denominator natural frequency

D_N: Numerator damping

D_D: Denominator damping

Since a vibration-capable filter setting is not expected to yield useful results in any case, as with the "2nd-order filter" (PT2) low-pass filter (PT2) mode of the jerk filter there is no setting option for the denominator damping D_D. The denominator damping D_D is permanently set to 1.

The band-stop filter can be parameterized in two different ways:

- Real band-stop filter
- Band-stop filter with additional amplitude response increase/decrease at high frequencies

Real band-stop filter

A real band-stop filter is obtained when identical numerator and denominator natural frequencies are selected:

$f_N = f_D = f_{block}$ (blocking frequency)

If the numerator damping setting = 0 is selected, the blocking frequency is equivalent to complete attenuation. In this case the 3 dB bandwidth is calculated as follows:

$f_{3\ dB\ bandwidth} = 2 * f_{block}$

If instead of complete attenuation, a reduction by a factor of k is all that is required, then numerator damping should be selected in accordance with k. In this case the above formula for calculating the 3 dB bandwidth no longer applies.

Band-stop filter with additional amplitude response increase/decrease at high frequencies

In this case, the numerator and denominator natural frequencies are set to different values. The numerator natural frequency determines the blocking frequency.

By selecting a lower/higher denominator natural frequency than the numerator natural frequency, you can increase/decrease the amplitude response at high frequencies. An amplitude response increase at high frequencies can be justified in most cases, as the controlled system generally has a lowpass characteristic itself, i.e. the amplitude response decreases at high frequencies anyway.

Boundary conditions

If the numerator natural frequency is selected to be too high, then the filter is deactivated. In this case, the frequency limit f_{Zmax} depends on the position controller clock cycle:

$$f_{Zmax} = \frac{1}{2 * \pi * T_{Zmin}} = \frac{1}{2 * \pi * T_{Position\ control\ cycle\ clock}} \text{ (Shannon-Theorem)}$$

Filter type: Double moving average calculation

Double moving average calculation has a stronger smoothing effect than basic moving average calculation. By additionally parameterizing a second moving average calculation, differences in the dynamic response between axes that are interpolating with one another can be compensated.

12.2.18.2 Parameterization

Activation

The jerk filter is activated using machine data:

MD32400 $MA_AX_JERK_ENABLE (axial jerk limitation)

The jerk filter is active in all operating modes and with all types of interpolation.

Filter type

The filter type is selected via the machine data:

MD32402 $MA_AX_JERK_MODE (filter type for axial jerk limitation)

Value	Filter type
1	2nd order filter
2	Moving average calculation
3	Band-stop
4	Double moving average calculation
5	FIR lowpass

Axial jerk filter (1, 2, 3)

The axial jerk filters are set using the following machine data:

- For filter types 1 and 2 with MD32410 $MA_AX_JERK_TIME.
- For filter type 3 using machine data MD32412 AX_JERK_FREQ and MD 32414 AX_JERK_DAMP.

The jerk filter is only effective when the time constant is greater than a position control clock cycle.

Double moving average calculation (4)

A double moving average calculation requires that the following machine data are set:

- MD32410 $MA_AX_JERK_TIME
- MD32411 $MA_AX_JERK_TIME_ADD

If both machine data are parameterized the same, a delta filter is implemented, which produces extremely strong damping, but also significant contour distortion.

FIR filter (5)

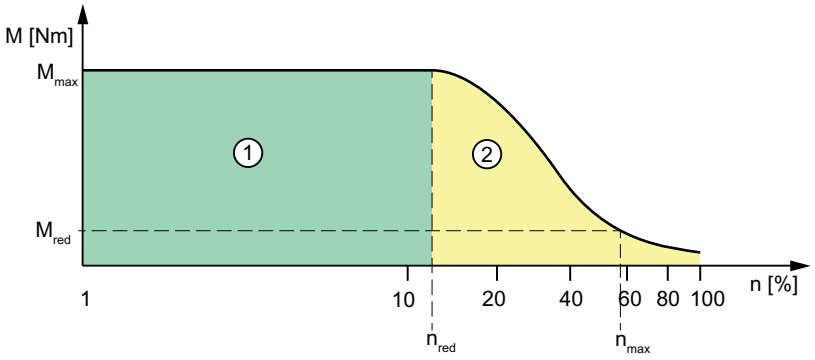
A FIR lowpass filter requires that the following machine data are set:

- MD32407 $MA_AX_JERK_FIR_FREQ = (corner frequency (-6dB) of the axial FIR jerk filter smoothing)
- MD32408 $MA_AX_JERK_FIR_ORDER (order of the filter for the axial FIR jerk filter smoothing)
- MD32409 $MA_AX_JERK_FIR_ORDER (window type of the axial FIR jerk filter smoothing):
  - Window type 0 does not use a window
  - Window type 1 uses a Hamming window.
  - Window type 2 uses a Blackman-Harris window.
  - Window type 3 uses a Gauss window.
- MD38020 $MA_MM_CPREC_FIR_POINTS (number of points for characteristic approximation of FIR filters for CPRECON)
  The function "programmable contour accuracy" uses a table for approximation of the inverse characteristic of FIR filters. The memory-configuring machine data MD38020 specifies the number of table values. The larger the value, the more precise the approximation. The recommended value is the value of MD32407 $MA_AX_JERK_FIR_FREQ. If no value is entered, it is not possible to execute the function and alarm 10990 is output.
  Further information:
  For information about "Programmable contour accuracy", see:
  Function Manual Monitoring and Compensating; Contour tunnel monitoring > Programmable contour accuracy

12.2.19 Knee-shaped acceleration characteristic

12.2.19.1 Function: Adaptation to the motor characteristic curve

Various motor types, particularly stepper motors, have a torque characteristic that is highly dependent on the speed with a steep decline in the torque in the upper speed range. For optimal utilization of the motor characteristic, the acceleration of the associated NC axis must be reduced as of a specific speed.



①	Normal range
②	Reducing range
n_{red}	Speed as of which traversing is done with reduced torque
n_{max}	Maximum speed
M_{max}	Maximum torque
M_{red}	Torque at n_{max} (corresponds to acceleration reduction)

Characteristic curve types

As a controller-internal emulation of the torque characteristic curve of the motor, the following characteristic curve types can be selected for the reducing range (see chapter "Parameterization (Page 383)"):

- Constant torque characteristic
- Hyperbolic torque characteristic
- Linear torque characteristic

12.2.19.2 Function: Effects on the path acceleration

The path acceleration characteristic curve results from the types of characteristic curve for the axes that are of relevance for the path. If axes with different types of characteristic curve are interpolated together, the acceleration profile for the path acceleration will be determined on the basis of the reduction type that is most restrictive.

The following order of priorities applies, whereby 1 = top priority:

1. Acceleration reduction: 0 = constant characteristic
2. Acceleration reduction: 1 = hyperbolic characteristic

3. Acceleration reduction: 2 = linear characteristic
4. No acceleration reduction effective
   A situation where no acceleration reduction is active arises for example when:
   MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT = 1
   and / or
   MD35230 $MA_ACCEL_REDUCTION_FACTOR = 0

Note

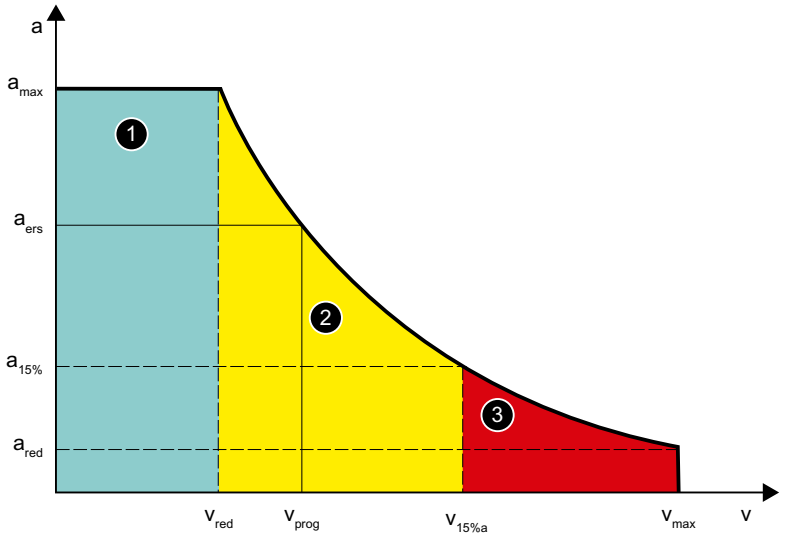
Machine axes featuring stepper motor and DC drive can be interpolated together.

12.2.19.3 Function: Substitute curve

If the programmed path cannot be traversed with the acceleration curve parameterized in the machine data of the participating axes, e.g. if kinematic transformation and acceleration types `BRISK` or `SOFT` are active, a substitute curve is generated by reducing the dynamic limit values. The reduced dynamic limit values are calculated to ensure that the substitute characteristic curve provides the best possible compromise between maximum velocity and constant acceleration.

Substitute characteristic curve with linear path sections

Limitation to this value is applied if the programmed path velocity is greater than that at which 15 % of the maximum acceleration capacity is still available ($v_{15\%a}$). Consequently, 15 % of the maximum acceleration capacity/motor torque always remains available, whatever the machining situation.

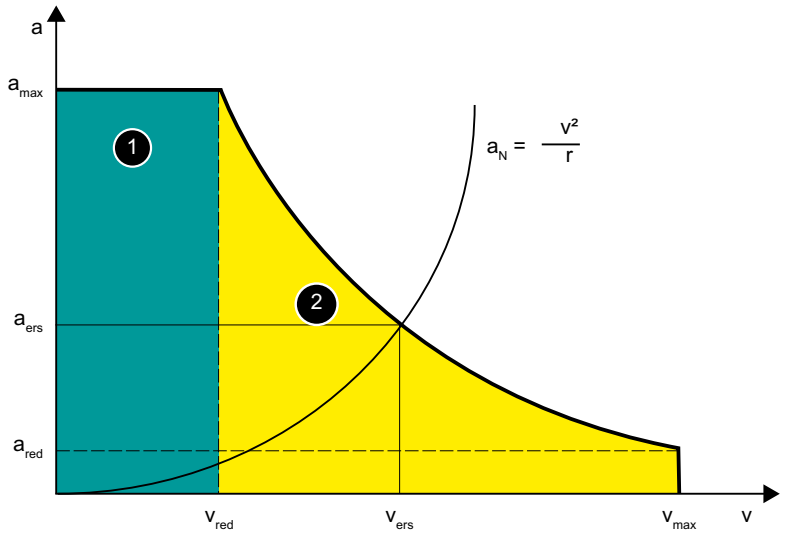


① Normal range
② Reducing range
③ Locked area
a_{ers}: Substitute characteristic curve constant acceleration
$a_{15\%}$ Minimal constant acceleration
$a_{15\%} = 0.15 * (a_{max} - a_{red}) + a_{red}$
v_{ers} Substitute characteristic curve velocity
v_{prog} Programmed velocity
$v_{15\%a}$ Velocity at $a_{15\%}$

Figure 12-6 Substitute path characteristic curve: Linear path

Substitute characteristic curve with curved path sections

In the case of curved path sections, normal and tangential acceleration are considered together. The path velocity is reduced so that only up to 25 % of the speed-dependent acceleration capacity of the axes is required for normal acceleration. The remaining 75 % of the acceleration capacity is set aside for the tangential acceleration, i.e., deceleration/acceleration on the path.



① Normal range
② Reducing range
a_N Normal acceleration
a_{ers}: Substitute characteristic curve constant acceleration
v_{ers} Substitute characteristic curve velocity
r Path radius

Figure 12-7 Substitute path characteristic curve: Curved path

Block transitions with continuous-path mode

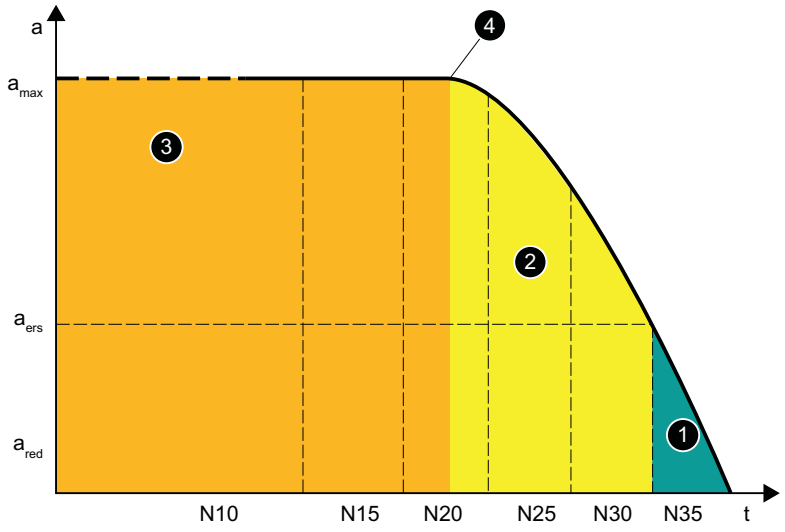
If continuous-path mode is active, non-tangential block transitions result in axial velocity jumps when the programmed path velocity is used for traversing.

As a result, the path velocity is controlled in such a way that prevents any axial velocity proportion from exceeding the creep velocity v_{red} at the time of the block transition.

Deceleration ramp with continuous-path mode and LookAhead

In the case of consecutive traversing blocks with short paths, an acceleration or deceleration operation may be spread over several part program blocks.

In such a situation the "LookAhead" function also takes into account the parameterized speed-dependent acceleration characteristic.



① Normal range ⇒ a = a_{max}
② Reducing range ⇒ a < a_{max}
③ Constant travel range ⇒ a = 0 m/s²
④ Brake application point
v_{red}: Creep velocity
v_{max} Maximum velocity
Nx Traversing block with block number Nx

Figure 12-8 Deceleration with LookAhead

12.2.19.4 Parameterization

Path axis

Parameter assignment

The parameterization of the axis-specific velocity limit, above which the reduced acceleration acts as an axis for traversing movements, is done via the machine data:

MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT = <speed limit>

Single axis

Activation

The activation of the axis-specific velocity limit, above which the reduced acceleration acts as a single axis for traversing movements, is done via machine data:

MD35240 $MA_ACCEL_TYPE_DRIVE = TRUE

Parameter assignment

The following machine data is relevant for parameterizing the axis-specific acceleration characteristic curve above the configured velocity limit:

- MD32000 $MA_MAX_AX_VELO (maximum axis velocity)
- MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT = <speed limit>
  From the parameterized velocity, the reduced acceleration is in effect>
- MD35230 $MA_ACCEL_REDUCTION_FACTOR = <reduction factor>
  Beyond the reduction speed, interpolation between the maximum and minimum acceleration occurs up to the maximum speed. The minimum acceleration is calculated as: minimum acceleration = maximum acceleration * (1 - reduction factor)
- MD35242 $MA_ACCEL_REDUCTION_TYPE (see paragraph below: "Selection of the torque characteristic curve")
- MD32300 $MA_MAX_AX_ACCEL (Maximum axis acceleration)

Selection of the torque characteristic curve

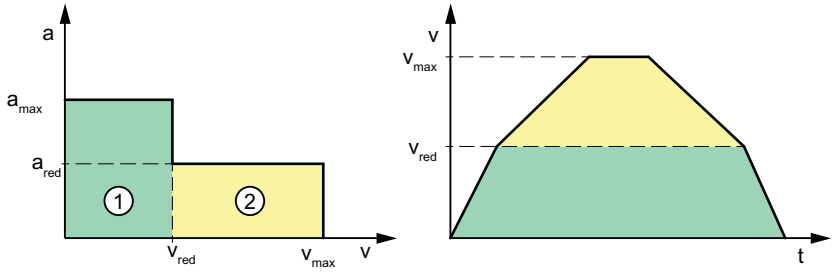
The following characteristic curve types can be selected for the reducing range via the machine data as a controller-internal emulation of the torque characteristic curve of the motor:

MD35242 $MA_ACCEL_REDUCTION_TYPE = <value>

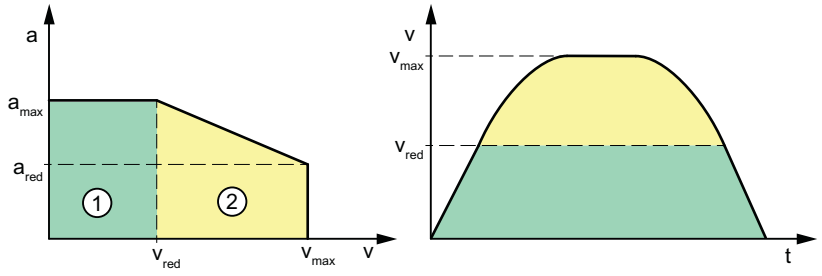
Value	Meaning
0	Constant torque characteristic
1	Hyperbolic torque characteristic
2	Linear torque characteristic

The following figures show the principal velocity and acceleration characteristic curves for the respective types of characteristic curve:

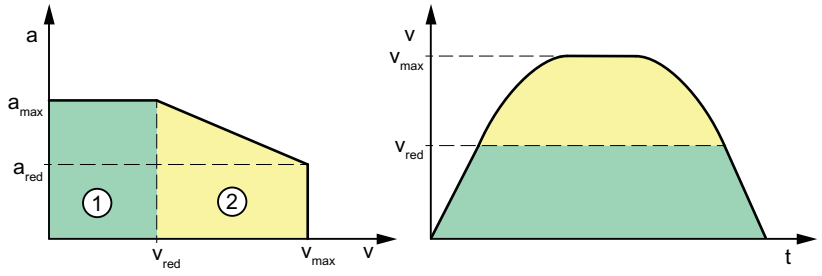
- Constant torque characteristic (value = 0)



- Hyperbolic torque characteristic (value = 1)



- Linear torque characteristic (value = 2)



Characteristic parameters

The characteristic curve parameters result from the following machine data:

① Normal range
② Reducing range
v_{max} $MA_MAX_AX_VELO
v_{red} $MA_ACCEL_REDUCTION_SPEED_POINT * $MA_MAX_AX_VELO
a_{max} $MA_MAX_AX_ACCEL
a_{red} (1 - $MA_ACCEL_REDUCTION_FACTOR) * $MA_MAX_AX_ACCEL

12.2.19.5 Programming: Channel-specific activation (DRIVE)

Functionality

The axis-specific velocity limit, above which the reduced acceleration acts as an axis for traversing movements, is activated with the command DRIVE.

Syntax

```
DRIVE
```

Meaning

DRIVE:	Channel-specific switch-on of the reduced acceleration above the parameterized velocity limit	
	G group:	21
	Basic position:	MD20150 $MC_GCODE_RESET_VALUES[20]
	Effectiveness:	Modal

Supplementary conditions

Characteristic changeover

If the acceleration reduction is active due to the programming of DRIVE and the acceleration profile is changed over by SOFT or BRISK, traversing is then done with a substitute characteristic with reduced dynamic limit values.

With the re-programming of DRIVE, the reduction in acceleration can be re-activated beyond the parameterized velocity limit.

See also

Parameterization (Page 383)

12.2.19.6 Programming: Axis-specific activation (DRIVEA)

Functionality

The axis-specific velocity limit, above which the reduced acceleration acts as a single axis for traversing movements, is activated for the programmed axes with the pre-defined procedure DRIVEA().

Syntax

```
DRIVEA(<Axis_1>, <Axis_2>, ...)
```

Meaning

DRIVEA:	Axis-specific activation of the parameterized knee-shaped acceleration characteristic curve.	
	Effectiveness:	Modal
<axis_x>:	Axis for which the parameterized knee-shaped acceleration characteristic curve is to be activated.	
	Data type:	AXIS
	Value range:	Channel axis names

Supplementary conditions

If the knee-shaped acceleration characteristic curve is parameterized for an axis, then this becomes the default acceleration profile for traversing operations.

If the effective acceleration profile is changed for a specific axis using the SOFTA or BRISKA part-program commands, then an appropriate substitute characteristic curve is used in place of the knee-shaped acceleration characteristic curve.

It is possible to switch back to the knee-shaped acceleration characteristic curve for a specific axis by programming DRIVEA.

12.2.19.7 Boundary conditions

Single axis interpolation

After activating the knee-shaped acceleration characteristic curve in case of single-axis interpolations (positioning axis, oscillating axis, manual traversing, etc.), the machine axis is traversed exclusively in the mode DRIVEA .

It is not possible to switch over the acceleration profile via the following part program instructions:

- Abrupt acceleration changes (BRISKA)
- Acceleration with jerk limitation (SOFTA)

Path interpolation

If for a machine axis involved in a programmed path, the knee-shaped acceleration characteristic curve parameterized without the part program instruction DRIVE is active, then a substitute characteristic curve with reduced dynamic limit values is determined for the path.

Kinematic transformation

The knee-shaped acceleration characteristic curve is not considered in an active kinematic transformation. With internal control, a switchover is done to acceleration without jerk limitation (BRISK) and a substitute characteristic curve becomes active for the path acceleration.

12.2.20 Acceleration and jerk for JOG motions

The axis-specific acceleration and jerk limitation values also take effect in JOG mode.

It is also possible to limit acceleration and jerk channel-specifically for manual traversing of geometry and orientation axes. This enables better handling of the kinematics that generate Cartesian motions entirely via rotary axes (robots).

12.2.20.1 Parameterization

Axis-specific limitation of acceleration and jerk

Maximum axis-specific acceleration

The maximum acceleration for manual traversing of an axis is defined in machine data:

MD32300 $MA_MAX_AX_ACCEL [0] (maximum axis-specific acceleration for path motions in dynamic response mode DYNNORM)

Note

Only the dynamic response mode DYNNORM is always effective for JOG mode.

See also Chapter "Acceleration without jerk limitation (BRISK/BRISKA) (channel/axis-specific) (Page 349)".

Basic setting of axis-specific jerk limitation

Acceleration with jerk limitation can be specified as the axis-specific basic setting for JOG mode:

MD32420 $MA_JOG_AND_POS_JERK_ENABLE== TRUE

Maximum axis-specific jerk

For JOG motions with active jerk limitation, the value from the following machine data takes effect as the maximum axis-specific jerk:

MD32430 $MA_JOG_AND_POS_MAX_JERK (maximum axis-specific jerk)

See also Chapter "Jerk limitation with single-axis interpolation (SOFTA) (axis-specific) (Page 366)".

Channel-specific limitation of acceleration and jerk

Maximum acceleration when manually traversing geometry axes

The maximum acceleration when manually traversing geometry axes can be specified for each channel via the machine data:

MD21166 $MC_JOG_ACCEL_GEO [<geometry axis>]

With <geometry axis> = 0, 1, 2

Note

With MD21166 $MC_JOG_ACCEL_GEO [<geometry axis>], there is no direct limitation to MD32300 $MA_MAX_AX_ACCEL.

Note

When a transformation is active, MD32300 $MA_MAX_AX_ACCEL determines the maximum possible axis-specific acceleration.

Maximum jerk when manually traversing geometry axes

The maximum jerk when manually traversing geometry axes in the SOFT acceleration mode (acceleration with jerk limitation) can be specified for each channel via the machine data:

MD21168 $MC_JOG_JERK_GEO [<geometry axis>]

With <geometry axis> = 0, 1, 2

Note

MD21168 $MC_JOG_JERK_GEO acts only when the axis-specific jerk limitation in JOG mode has been enabled for the base machine axes:

MD32420 $MA_JOG_AND_POS_JERK_ENABLE [<axis>] == TRUE

With MD21168 = 0, instead of the channel-specific jerk limitation, the axis-specific limit value from MD32430 $MA_JOG_AND_POS_MAX_JERK is effective.

Maximum jerk when manually traversing orientation axes

The maximum jerk when manually traversing orientation axes can be specified for each channel via the machine data:

MD21158 $MC_JOG_JERK_ORI [<orientation axis>]

For MD21158 to take effect, the channel-specific jerk limitation for the manual traversing of orientation axes must be enabled via the following machine data:

MD21159 $MC_JOG_JERK_ORI_ENABLE == TRUE

12.2.20.2 Supplementary conditions

Behavior for the manual traversing of geometry axes with active rotation

When manually traversing geometry axes in the SOFT acceleration mode (acceleration with jerk limitation), the value from MD32430 $MA_JOG_AND_POS_MAX_JERK is also used with active rotation or active orientable tool carrier.

Part program instruction SOFTA / BRISKA / DRIVEA

The part program instruction `SOFTA(<axis1>,<axis2>, ...)` is also effective in JOG mode, i.e. the maximum axis-specific jerk from MD32430 $MA_JOG_AND_POS_MAX_JERK is effective for the specified axes when traversing in JOG mode (exactly as when setting MD32420 $MA_JOG_AND_POS_JERK_ENABLE [<axis>] == TRUE).

Note

On the other hand, the SOFT part program instruction has no effect on JOG mode.

As with SOFTA, the part program instructions BRISKA and DRIVEA are also effective in JOG mode, i.e. the acceleration is without jerk limitation, even when MD32420 $MA_JOG_AND_POS_JERK_ENABLE is set to "TRUE" for the relevant machine axes.

Note

Manual traversing of orientation axes is not affected by BRISKA/SOFTA/DRIVEA.

See also

Acceleration without jerk limitation (BRISK/BRISKA) (channel/axis-specific) (Page 349)

Jerk limitation with single-axis interpolation (SOFTA) (axis-specific) (Page 366)

12.3 Examples

12.3.1 Acceleration

12.3.1.1 Path velocity characteristic

The following example shows the path velocity characteristic, based on programmed traversing motion and the actions initiated in a part program segment.

Part program extract

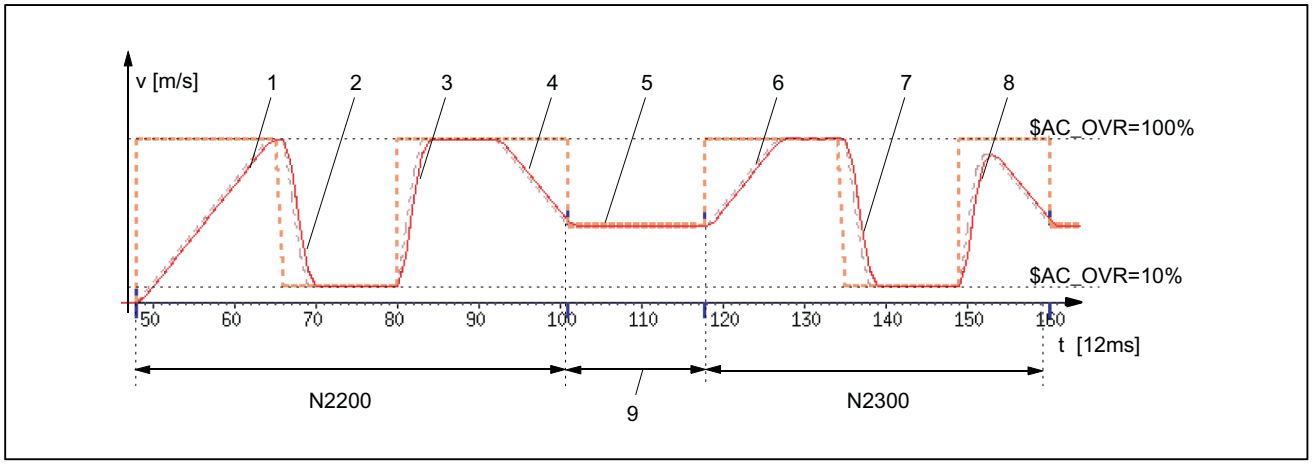
Program code

```
; Synchronized action: Acceleration switchover depending on fast input 1
($A_IN[1]):
N53 ID=1 WHEN $A_IN[1] == 1 DO $AC_PATHACC = 2.*$MA_MAX_AX_ACCEL[X]
; Synchronized action: Test override profile (simulation of external
interventions):
N54 ID=2 WHENEVER ($AC_TIMEC > 16) DO $AC_OVR=10
N55 ID=3 WHENEVER ($AC_TIMEC > 30) DO $AC_OVR=100
;Approach
N1000 G0 X0 Y0 BRISK
```

Program code

```
N1100 TRANS Y=-50
N1200 AROT Z=30 G642
; Contour
N2100 X0 Y0
N2200 X = 70 G1 F10000 RNDM=10 ACC[X]=30 ACC[Y]=30
N2300 Y = 70
N2400 X0
N2500 Y0
```

Path velocity characteristic



Acceleration profile: BRISK

1: Accelerate to 100% of path velocity (F10000) in accordance with acceleration default: ACC (N2200...)

2: Brake to 10% of path velocity as a result of override modification ($AC_OVR) in accordance with real-time acceleration $AC_PATHACC (N53/N54...)

3: Accelerate to 100% of path velocity as a result of override modification ($AC_OVR) in accordance with real-time acceleration $AC_PATHACC (N53/N55...)

4: Brake to block end velocity for intermediate smoothing block in accordance with acceleration default: ACC (N2200...)

5: Speed limitation as a result of smoothing (see 9)

6: Accelerate to 100% of path velocity ($AC_OVR) in accordance with acceleration default: ACC (N2300...)

7: Decelerate as a result of override modification at a rate of acceleration that is in accordance with real-time acceleration $AC_PATHACC (N53/N54...)

8: Accelerate to 100% of path velocity as a result of override modification ($AC_OVR) in accordance with real-time acceleration $AC_PATHACC (N53/N55...)

9: Intermediate block inserted within the control as a result of the programmed smoothing (RNDM) (N2200...)

Figure 12-9 Switching between path acceleration specified during preprocessing and real-time acceleration

12.3.2 Jerk

12.3.2.1 Path velocity characteristic

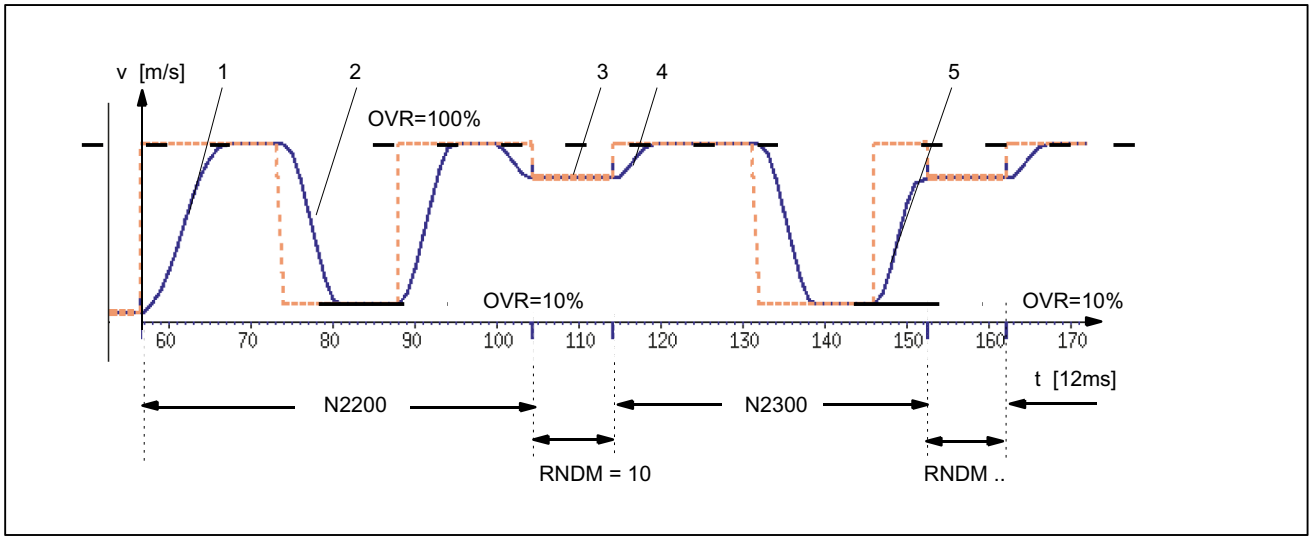
The following example shows the characteristic of path velocity and jerk based on programmed traversing motion and the actions initiated in a part program segment.

Part program extract

Program code

```
; Setting of path acceleration and path jerk in the event of external
intervention:
N0100 $AC_PATHACC = 0.0
N0200 $AC_PATHJERK = 4.0 * ($MA_MAX_AX_JERK[X] + $MA_MAX_AX_JERK[Y]) / 2.0
; Synchronized action:  varying the override (simulation of external
interventions)
N0530 ID=1 WHENEVER ($AC_TIMEC > 16) DO $AC_OVR=10
N0540 ID=2 WHENEVER ($AC_TIMEC > 30) DO $AC_OVR=100
;Approach
N1000 G0 X0 Y0 SOFT
N1100 TRANS Y=-50
N1200 AROT Z=30 G642
; Contour
N2100 X0 Y0
N2200 X = 70 G1 F10000 RNDM=10
N2300 Y = 70
N2400 X0
N2500 Y0
```

Characteristic of path velocity and jerk



Acceleration profile: SOFT

1: Jerk according to $MA_MAX_AX_JERK[..]

2: Jerk according to $AC_PATHJERK

3: Jerk according to $MA_MAX_AX_JERK[..] (approach block end velocity)

4: Velocity limiting using arcs

5: Jerk according to $AC_PATHJERK

Figure 12-10 Switching between path jerk specified during preprocessing and $AC_PATHJERK

12.3.3 Acceleration and jerk

The following example shows the characteristic of velocity and acceleration of the X axis based on the programmed traversing motion of the part program extract. Further, which of the velocity and acceleration-relevant machine data are decisive for which section of the contour.

Part program extract

Program code	Comment
N90 F5000 SOFT G64	; Continuous-path mode, jerk-limited acceleration
N100 G0 X0 Y0 Z0	; Rapid traverse
N110 G1 X10	; Straight line
N120 G3 CR=5 X15 Y5	; Circular arc, radius 5 mm, block transition: Tangential
N130 G3 CR=10 X5 Y15	; Circular arc, radius 10 mm, block transition: Tangential
N140 G1 X-5 Y17.679	; Straight line, 15° kink

Contour

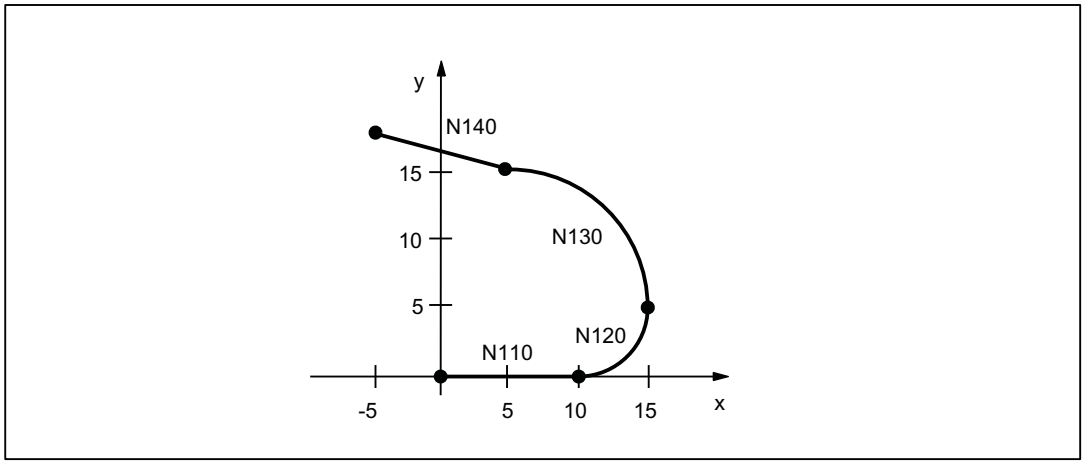


Figure 12-11 Contour of the part program extract

Velocity and acceleration characteristic

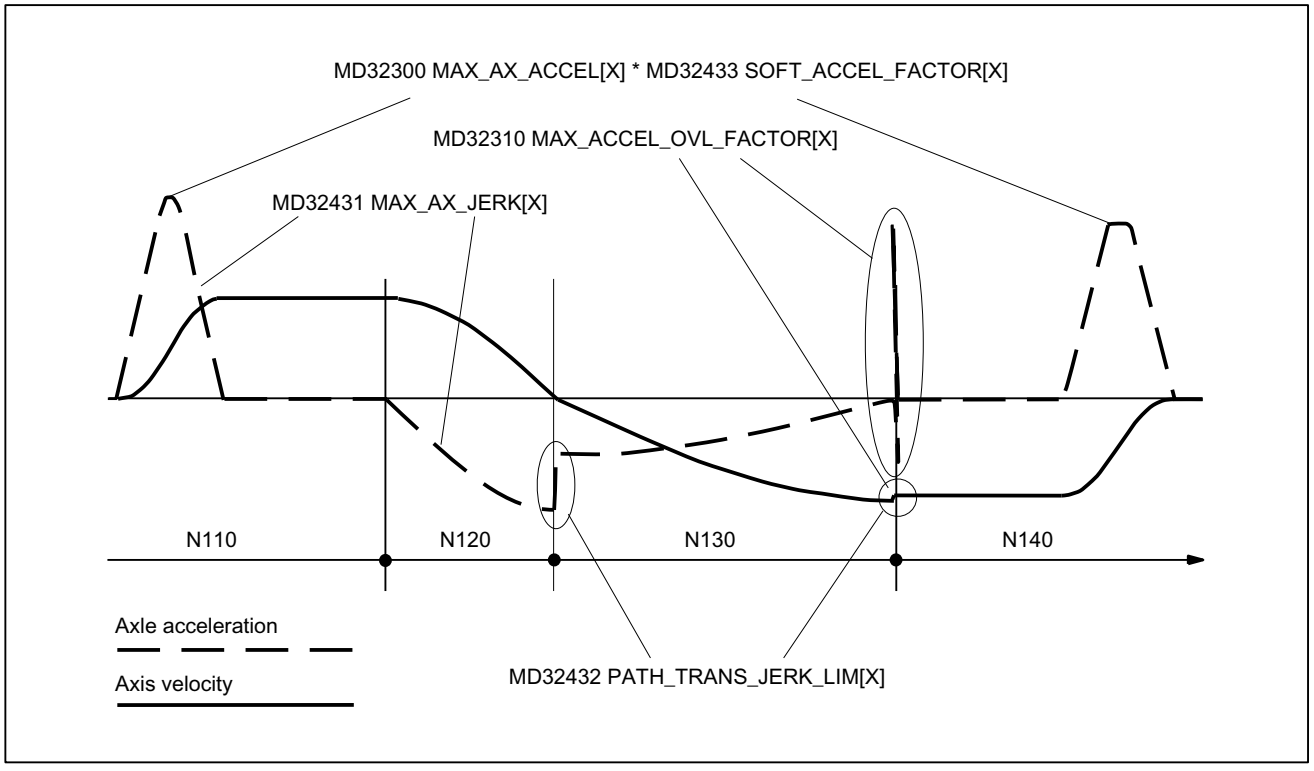


Figure 12-12 Velocity and acceleration characteristic curves X axis

12.3.4 Knee-shaped acceleration characteristic

12.3.4.1 Activation

The example illustrates how the knee-shaped acceleration characteristic curve is activated on the basis of the machine data and a part program extract.

Machine data

Machine data	Value
MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT[**X**]	= 0.4
MD35230 $MA_ACCEL_REDUCTION_FACTOR[**X**]	= 0.85
MD35242 $MA_ACCEL_REDUCTION_TYPE[**X**]	= 2
MD35240 $MA_ACCEL_TYPE_DRIVE[**X**]	= TRUE
MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT[**Y**]	= 0.0
MD35230 $MA_ACCEL_REDUCTION_FACTOR[**Y**]	= 0.6
MD35242 $MA_ACCEL_REDUCTION_TYPE[**Y**]	= 1
MD35240 $MA_ACCEL_TYPE_DRIVE[**Y**]	= TRUE
MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT[**Z**]	= 0.6
MD35230 $MA_ACCEL_REDUCTION_FACTOR[**Z**]	= 0.4
MD35242 $MA_ACCEL_REDUCTION_TYPE[**Z**]	= 0
MD35240 $MA_ACCEL_TYPE_DRIVE[**Z**]	= FALSE

Activation by entering as channel-specific initial setting:

MC_GCODE_RESET_VALUE[20] = 3 (DRIVE)

Part program extract

Program code	Comment
N10 G1 X100 Y50 Z50 F700	; Path motion (X,Y, Z) with DRIVE
N15 Z20	; Path motion (Z) with DRIVE
N20 BRISK	; Switchover to BRISK
N25 G1 X120 Y70	; Path motion (Y, Z) with substitute characteristic curve
N30 Z100	; Path motion (Z) with BRISK
N35 POS[X] = 200 FA[X] = 500	; Positioning motion (X) with DRIVEA
N40 BRISKA(Z)	; Activate BRISKA for Z
N40 POS[Z] = 50 FA[Z] = 200	; Positioning motion (Z) with BRISKA
N45 DRIVEA(Z)	; Activate DRIVEA for Z
N50 POS[Z] = 100	; Positioning motion (Z) with DRIVE
N55 BRISKA(X)	; results in error message

12.4 Data lists

12.4.1 Machine data

12.4.1.1 NC-specific machine data

Number	Identifier: $MN_	Description
18960	POS_DYN_MODE	Type of positioning axis dynamic response

12.4.1.2 Channel-specific machine data

Number	Identifier: $MC_	Description
20150	GCODE_RESET_VALUES	Initial setting of the G groups
20500	CONST_VELO_MIN_TIME	Minimum time with constant velocity
20600	MAX_PATH_JERK	Path-related maximum jerk
20602	CURV_EFFECT_ON_PATH_ACCEL	Influence of path curvature on path dynamic response
20610	ADD_MOVE_ACCEL_RESERVE	Acceleration reserve for overlaid motions
21158	JOG_JERK_ORI	Maximum jerk of the orientation axes when traversing in JOG
21159	JOG_JERK_ORI_ENABLE	Initial setting of the channel-specific jerk limitation of orientation axes for traversing in JOG mode
21166	JOG_ACCEL_GEO	Maximum acceleration rate of the geometry axes when traversing in JOG
21168	JOG_JERK_GEO	Maximum jerk of the geometry axes when traversing in JOG

12.4.1.3 Axis/spindlespecific machine data

Number	Identifier: $MA_	Description
32000	MAX_AX_VELO	Maximum axis velocity
32300	MAX_AX_ACCEL	Maximum axis acceleration
32310	MAX_ACCEL_OVL_FACTOR	Overload factor for velocity jump
32320	DYN_LIMIT_RESET_MASK	Reset behavior of dynamic limits
32400	AX_JERK_ENABLE	Axial jerk limitation
32402	AX_JERK_MODE	Filter type for axial jerk limitation
32407	AX_JERK_FIR_FREQ	Corner frequency axial jerk filter, type 5
32408	AX_JERK_FIR_ORDER	Filter order, axial jerk filter, type 5
32409	AX_JERK_FIR_WINDOW	Window type, axial jerk filter, type 5
32410	AX_JERK_TIME	Time constant for axial jerk filter
32411	AX_JERK_TIME_ADD	Time constant for the second axial jerk filter, type 4

Number	Identifier: $MA_	Description
32412	AX_JERK_FREQ	Blocking frequency of the axial jerk filter
32414	AX_JERK_DAMP	Damping, axial jerk filter
32420	JOG_AND_POS_JERK_ENABLE	Basic setting for axial jerk limitation
32430	JOG_AND_POS_MAX_JERK	Axial jerk for single axis motion
32431	MAX_AX_JERK	Maximum axial jerk at the block change in continuous-path mode
32432	PATH_TRANS_JERK_LIM	Max. axial jerk of a geometry axis at block boundary
32433	SOFT_ACCEL_FACTOR	Scaling of acceleration limitation for SOFT
32434	G00_ACCEL_FACTOR	Scaling of acceleration limitation for G0
32435	G00_JERK_FACTOR	Scaling of axial jerk limitation for G0
32437	AX_JERK_VEL0	First velocity threshold for velocity-dependent jerk adaptation
32438	AX_JERK_VEL1	Second velocity threshold for the velocity-dependent jerk adaptation
32439	MAX_AX_JERK_FACTOR	Factor to set the maximum jerk for higher velocities (velocity-dependent jerk adaptation)
35220	ACCEL_REDUCTION_SPEED_POINT	Speed for reduced acceleration
35230	ACCEL_REDUCTION_FACTOR	Reduced acceleration
35240	ACCEL_TYPE_DRIVE	DRIVE acceleration characteristic for axes on/off
35242	ACCEL_REDUCTION_TYPE	Type of acceleration reduction

12.4.2 Setting data

12.4.2.1 Channelspecific setting data

Number	Identifier: $SC_	Description
42500	SD_MAX_PATH_ACCEL	Max. path acceleration
42502	IS_SD_MAX_PATH_ACCEL	Analysis of SD 42500: ON/OFF
42510	SD_MAX_PATH_JERK	Max. path-related jerk
42512	IS_SD_MAX_PATH_JERK	Analysis of SD 42510: ON/OFF

12.4.3 System variables

Identifier	Description
$AC_PATHACC	Path acceleration for real-time events
$AC_PATHJERK	Path jerk for real-time events

N3: Software cams, position switching signals 13

13.1 Brief description

Function

The "Software cams" function generates position-dependent switching signals for axes that supply an actual position value (machine axes) and for simulated axes.
These cam signals can be output to the PLC and also to the NC I/O.

The cam positions at which signal outputs are set can be defined and altered via setting data. The setting data can be read and written via HMI, PLC and part program.

Activation

The "Software cams" function can be activated and used in all operating modes. The function remains active in the event of reset or Emergency Stop.

Field of application

Output cam signals can be used, for example:

- To activate protection zones
- To initiate additional movements as a function of position
- As reversing signals for hydraulically controlled oscillation axes

Axis types

Software cams can be used on linear and modulo rotary axes that are defined as machine axes.

Cam range/cam pair

Cams are always assigned in pairs to axes. A pair consists of a plus and a minus cam. 32 cam pairs are available.

The plus and minus cams each simulate a mechanical cam which is actuated at a defined point (cam position) in a specific approach direction when the axis reaches the cam position.

Cam ranges are assigned to the plus and minus cams as follows:

- Cam range plus: All positions ≥ plus cam
- Cam range minus: All positions ≤ minus cam

13.2 Cam signals and cam positions

13.2.1 Generation of cam signals for separate output

Separate output of the plus and minus cam signals makes it easy to detect whether the axis is within or outside the plus or minus cam range.

Linear axes

The switching edges of the cam signals are generated as a function of the axis traversing direction:

- The minus cam signal switches from 1 to 0 when the axis traverses the minus cam in the positive axis direction.
- The plus cam signal switches from 0 to 1 when the axis traverses the plus cam in the positive direction.

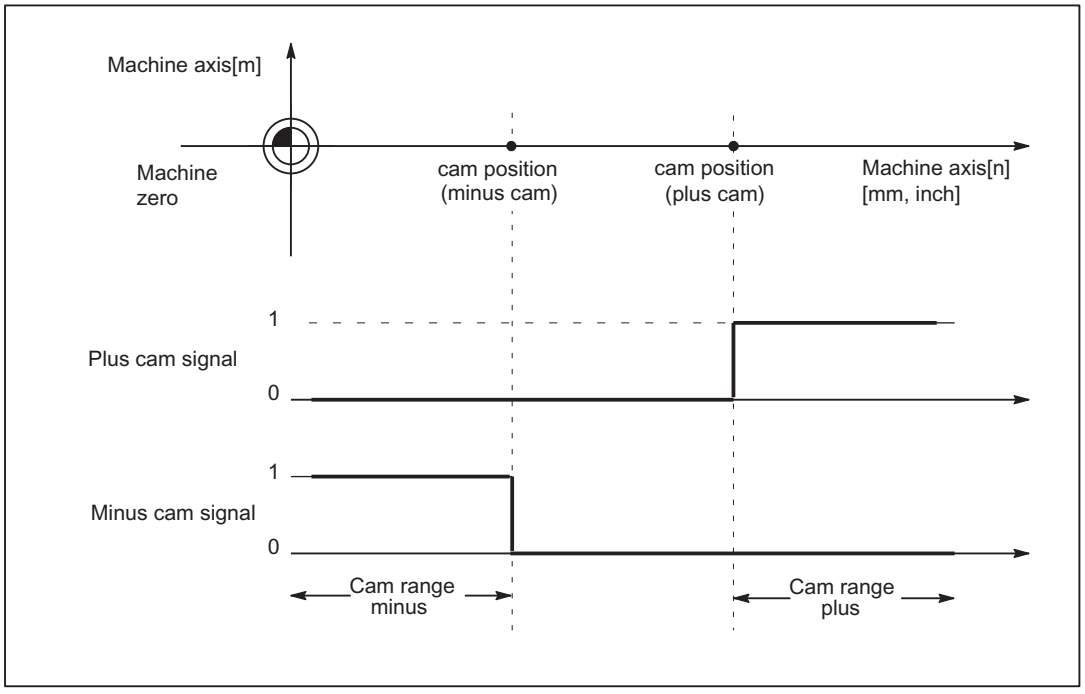


Figure 13-1 Software cams for linear axis (minus cam < plus cam)

Note

If the axis is positioned exactly at the output cam position (plus or minus), the defined output flickers. If the axis moves one increment further, the output becomes a definite zero or one.

Flickering of the actual position causes the signals to flicker in this manner.
The actual position is evaluated.

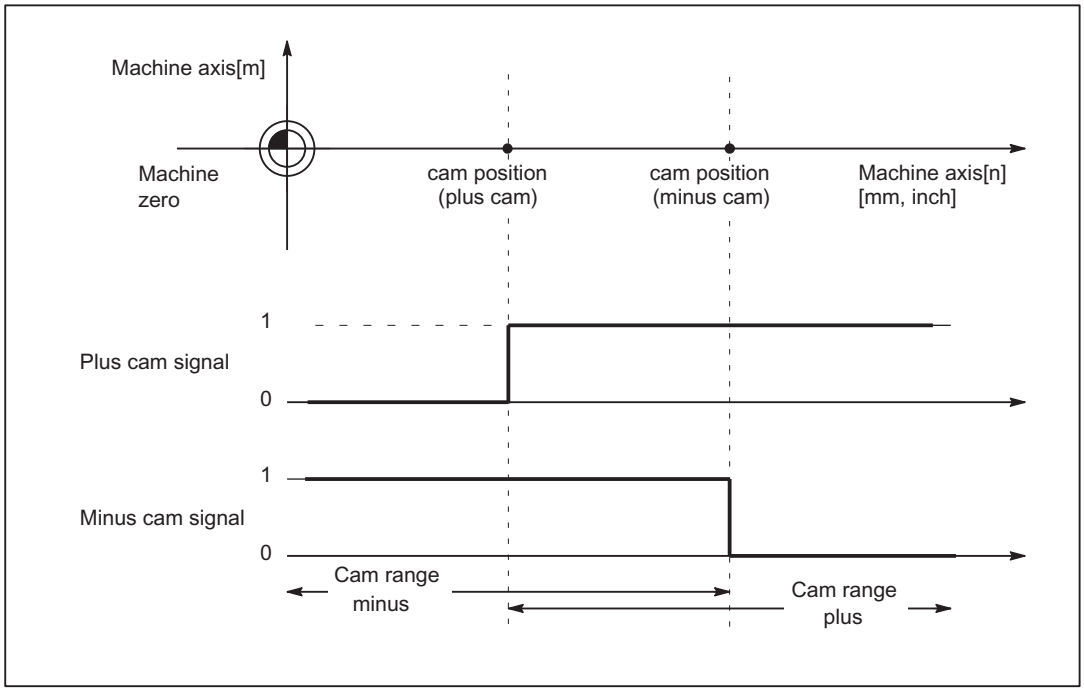


Figure 13-2 Software cams for linear axis (plus cam < minus cam)

Modulo rotary axes

The switching edges of the cam signals are generated as a function of the rotary axis traversing direction:

- The plus cam signal switches from 0 to 1 when the axis traverses the minus cam in a positive axis direction and from 1 back to 0 when it traverses the plus cam.
- The minus cam signal changes level in response to every positive edge of the plus cam signal.

Note

The described response of the plus cam applies on **condition**that:

plus cam - minus cam < 180 degrees

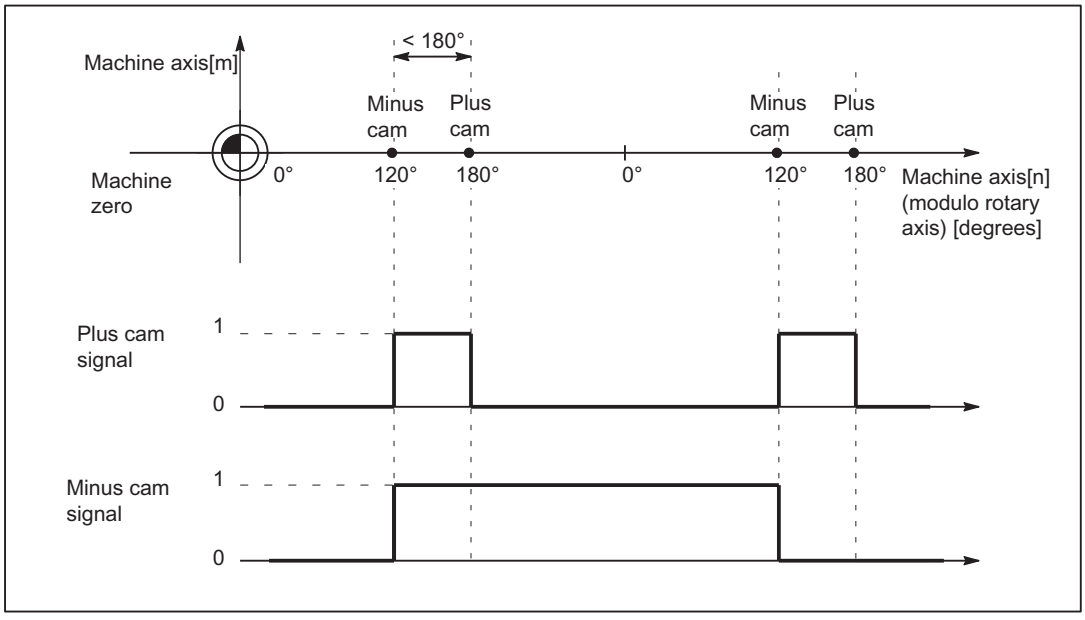


Figure 13-3 Software cams for modulo rotary axis (plus cam - minus cam < 180 degrees)

The signal change of the minus cam makes it possible to detect traversal of the cam even if the cam range is set so small that the PLC cannot detect it reliably.

Both cam signals can be output to the PLC and to the NC I/O. Separate output of the plus and minus cam signals makes it easy to detect whether the axis is within or outside the plus or minus cam range.

If this condition (plus cam - minus cam < 180 degrees) is not fulfilled or if the minus cam is set to a greater value than the plus cam, then the response of the plus cam signal is inverted. The response of the minus cam signal remains unchanged.

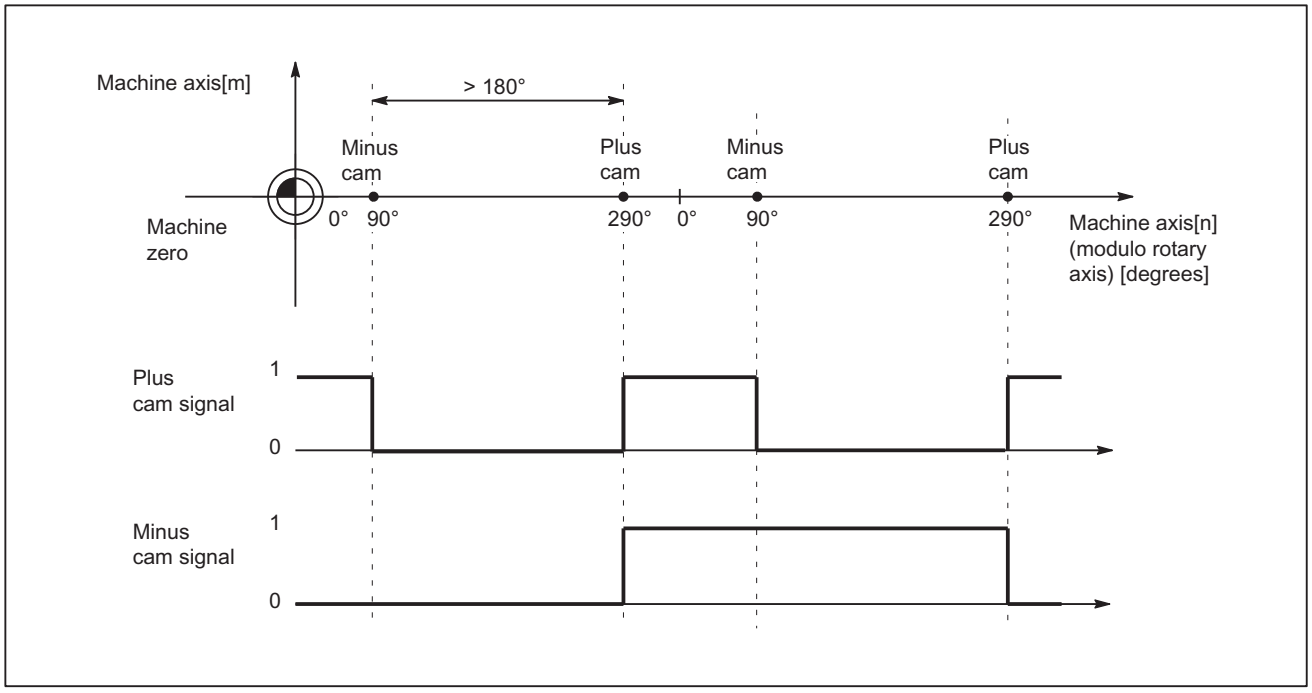


Figure 13-4 Software cams for modulo rotary axis (plus cam - minus cam > 180 degrees)

13.2.2 Generation of cam signals with gated output

The plus and minus cam output signals are gated in the case of:

- timer-controlled cam signal output to the four onboard outputs on the NCU
- Output to the NCK I/O, if the 2nd byte in the following machine data was not specified (= "0"):
  MD10470 SW_CAM_ASSIGN_FASTOUT_2
  ...
  MD10473 SW_CAM_ASSIGN_FASTOUT_4

Linear axes

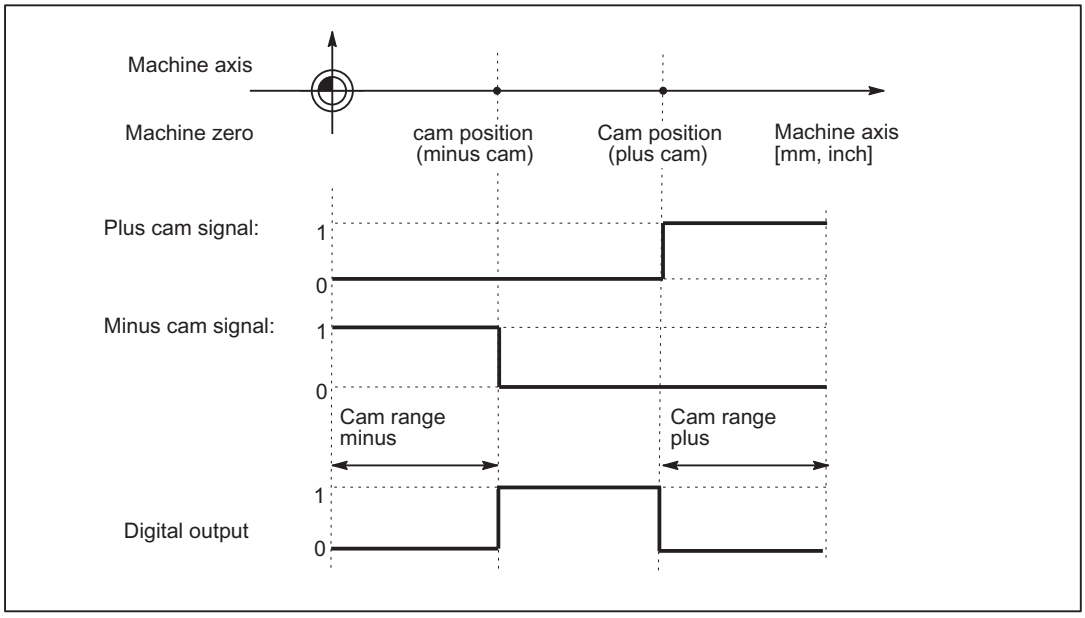


Figure 13-5 Position switching signals for linear axis (minus cam < plus cam)

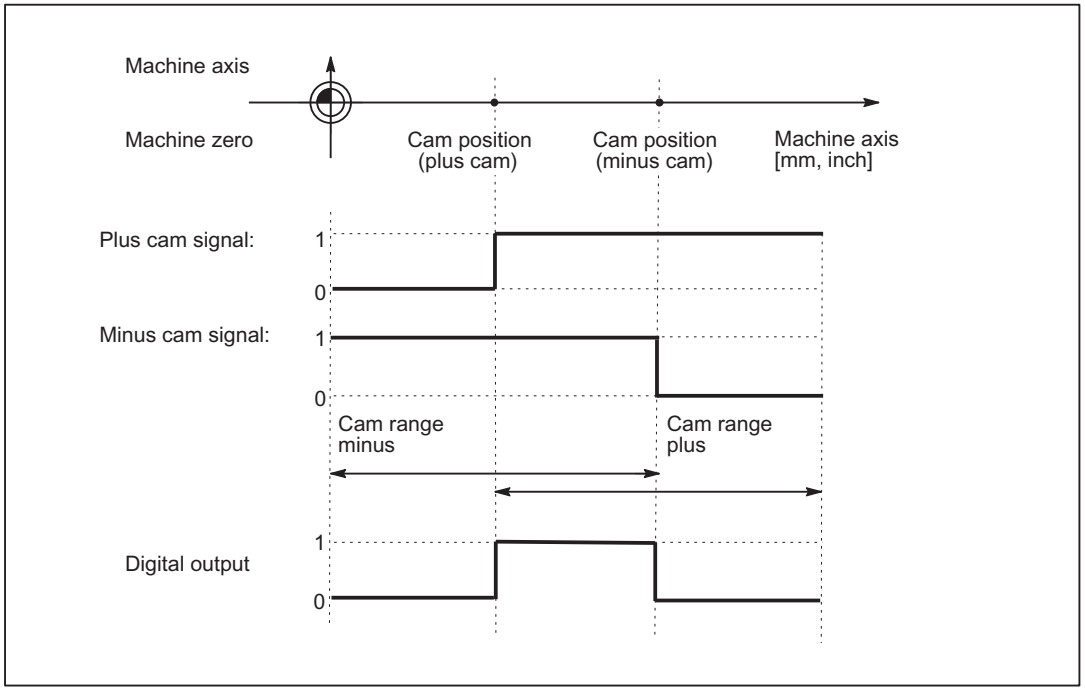


Figure 13-6 Position switching signals for linear axis (plus cam < minus cam)

Modulo rotary axis

The default signal response for modulo rotary axes is dependent on the cam width:

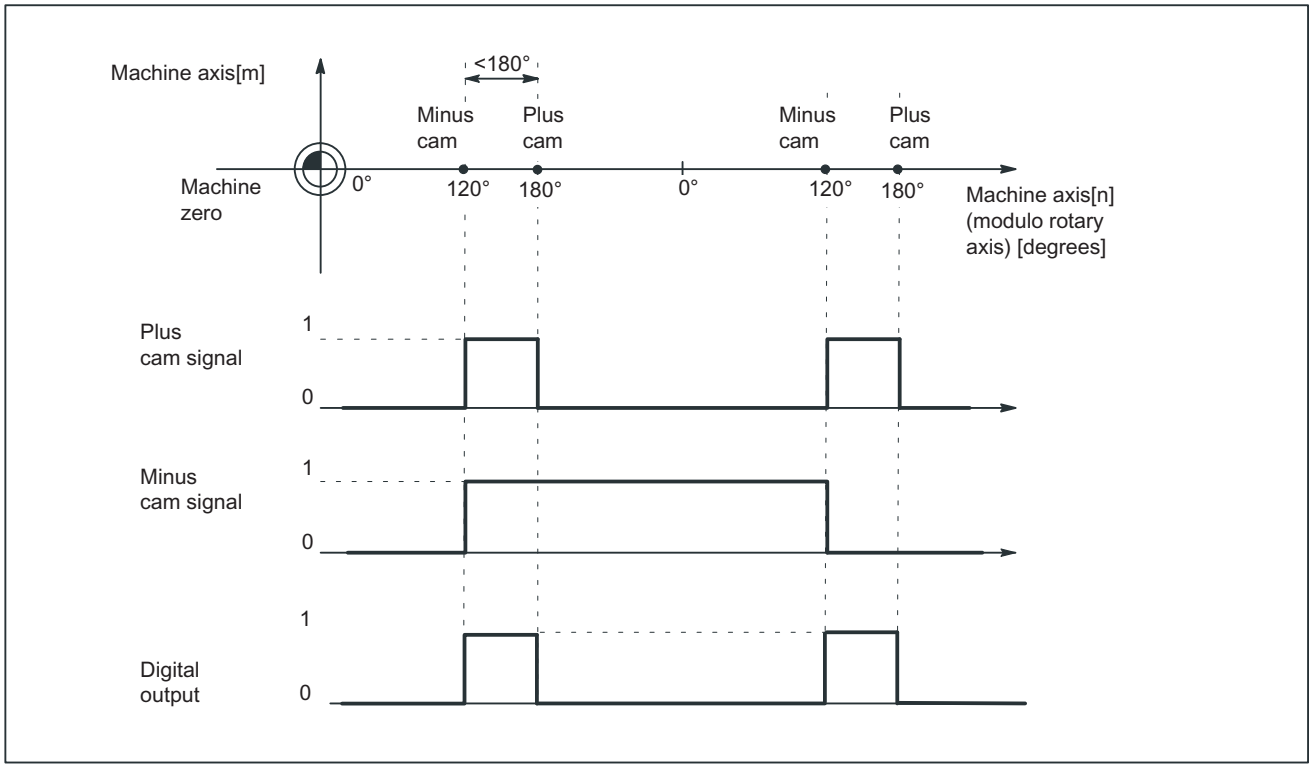


Figure 13-7 Software cams for modulo rotary axis (plus cam - minus cam < 180 degrees)

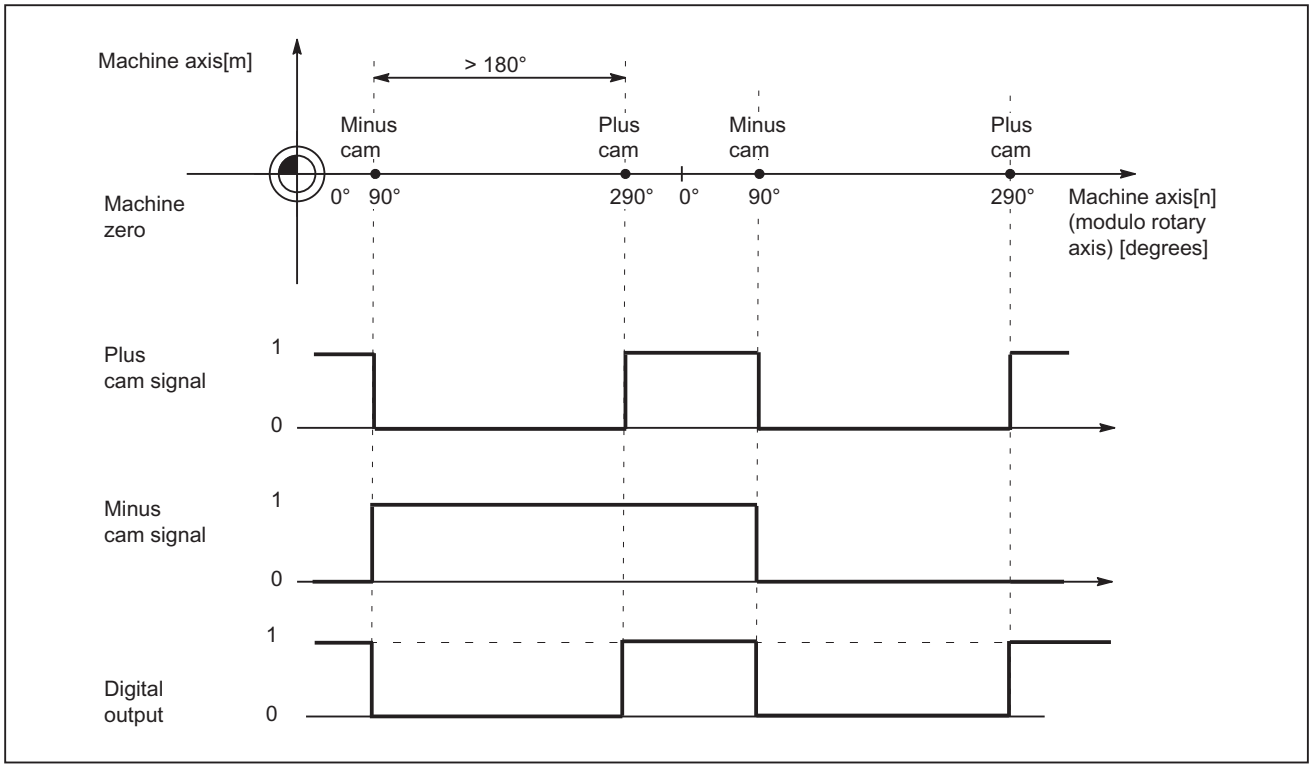


Figure 13-8 Software cams for modulo rotary axis (plus cam - minus cam > 180 degrees)

Suppression of signal inversion

With the following setting, selection of signal inversion for "plus cam - minus cam > 180 degrees" can be suppressed.

MD10485 SW_CAM_MODE bit 1=1

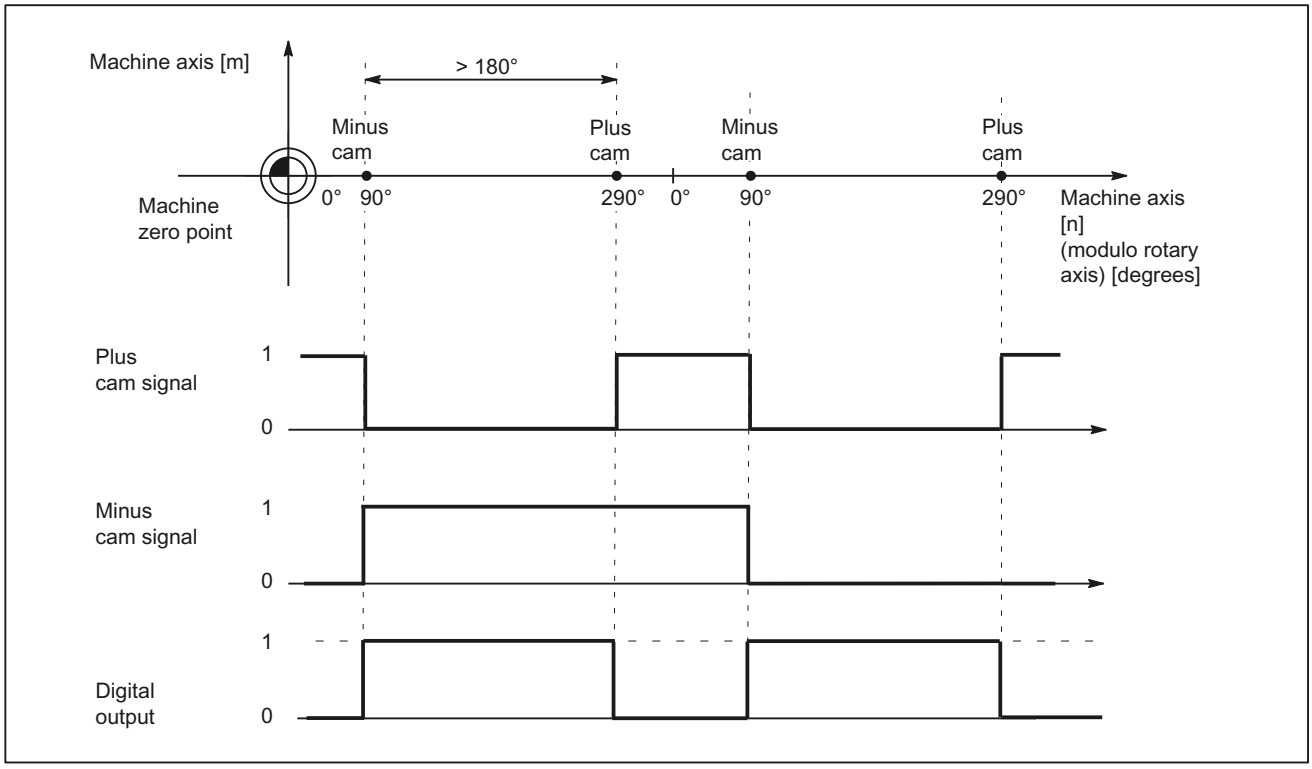


Figure 13-9 Software cams for modulo rotary axis (plus cam - minus cam > 180 degrees) and suppression of signal inversion

13.2.3 Cam positions

Setting cam positions

The positions of the plus and minus cams are defined using general setting data:

- SD41500 SW_CAM_MINUS_POS_TAB_1[n] Position of minus cams 1 - 8
- SD41501 SW_CAM_PLUS_POS_TAB_1[n] Position of plus cams 1 – 8
- SD41502 SW_CAM_MINUS_POS_TAB_2[n] Position of minus cams 9 - 16
- SD41503 SW_CAM_PLUS_POS_TAB_2[n] Position of plus cams 9 - 16
- SD41504 SW_CAM_MINUS_POS_TAB_3[n] Position of minus cams 17 - 24
- SD41505 SW_CAM_PLUS_POS_TAB_3[n] Position of plus cams 17 - 24
- SD41506 SW_CAM_MINUS_POS_TAB_4[n] Position of minus cams 25 - 32
- SD41507 SW_CAM_PLUS_POS_TAB_4[n] Position of plus cams 25 - 32

Note

Owing to the grouping of cam pairs (eight in each group), it is possible to assign different access authorization levels (e.g. for machine-related and workpiece-related cam positions). The positions are entered in the machine coordinate system. No check is made with respect to the maximum traversing range.

Dimension system metric/inch

With the setting:
MD10260 CONVERT_SCALING_SYSTEM = 1
, the cam positions no longer refer to the configured basic dimension system, but to the dimension system set in the following machine data:

MD10270 POS_TAB_SCALING_SYSTEM (measuring system of position tables)

Value	Meaning
0	Metric
1	inch

MD10270 therefore defines the dimension system for position data from setting data SD41500 … SD41507.

A switchover with G70/G71 or G700/G710 has no effect.

Sensing of cam positions

To set the cam signals, the actual position of the axes is compared to the cam position.

Writing/reading of cam positions

The setting data can be read and written via HMI, PLC and part program.

Accesses from the part program are not synchronous to machining.
Synchronization can only be achieved by means of a programmed block preprocessing stop (STOPRE command).

It is possible to read and write the cam positions with FB2 and FB3 in the PLC user program.

Axis/cam assignment

A cam pair is assigned to a machine axis using the general machine data:

MD10450 SW_CAM_ASSIGN_TAB[n] (assignment of software cams to machine axes)

Note

Changes to an axis assignment take effect after the next NC power-up.

Cam pairs to which no axis is assigned are not active.

A cam pair can only be assigned to one machine axis at a time.

Several cam pairs can be defined for one machine axis.

13.2.4 Lead/delay times (dynamic cam)

Function

To compensate for any delays, it is possible to assign two lead or delay times with additive action to each minus and plus cam for the cam signal output.

The two lead or delay times are entered in a machine data and a setting data.

Note

The input of negative time values causes a delay in the output of cam signals.

Lead or delay time in machine data

The **first** lead or delay time is entered in the following general machine data:

- MD10460 SW_CAM_MINUS_LEAD_TIME[n] (lead or delay time at the minus cams)
- MD10461 SW_CAM_PLUS_LEAD_TIME[n] (lead or delay time at the plus cams)

For example, the following can be entered into the machine data:

- Constant internal delay times between actual value sensing and cam signal output (e.g. as determined by an oscilloscope)
- Constant external delay times

Lead or delay time in setting data

The **second** lead or delay time is entered into the following general setting data:

- SD41520 SW_CAM_MINUS_TIME_TAB_1[n] (lead or delay time at the minus cams 1 – 8)
- SD41521 SW_CAM_PLUS_TIME_TAB_1[n] (lead or delay time at the plus cams 1 – 8)
- SD41522 SW_CAM_MINUS_TIME_TAB_2[n] (lead or delay time at the minus cams 9 – 16)
- SD41523 SW_CAM_PLUS_TIME_TAB_2[n] (lead or delay time at the plus cams 9 – 16)
- SD41524 SW_CAM_MINUS_TIME_TAB_3[n] (lead or delay time at the minus cams 17 – 24)
- SD41525 SW_CAM_PLUS_TIME_TAB_3[n] Lead or delay time on plus cams 17 - 24

- SD41526 SW_CAM_MINUS_TIME_TAB_4[n] (lead or delay time at the minus cams 25 – 32)
- SD41527 SW_CAM_PLUS_TIME_TAB_4[n] (lead or delay time at the plus cams 25 – 32)

Delay times which may change during machining must, for example, be entered in the setting data.

13.3 Output of cam signals

13.3.1 Activating

The status of the cam (cam signals) can be output to the PLC as well as to the NC I/O.

Activation of cam signal output

The output of cam signals of an axis is activated via the NC/PLC interface signal:

DB31, ... DBX2.0 (cam activation)

Check-back signal to PLC

The successful activation of all cams of an axis is signaled back to the PLC using the following NC/PLC interface signal:

DB31, ... DBX62.0 (cams active)

Note

The activation can be linked with other conditions by the PLC user (e.g. axis referenced, reset active).

13.3.2 Output of cam signals to PLC

Output to PLC

The status of the cam signals for all machine axes with activated software cams is output to the PLC.

The status is output in the IPO cycle and is transferred to the PLC asynchronously.

Minus cam signals

The status of the minus cam signals is entered into the following NC/PLC interface signals:

DB10 DBX110.0 to 113.7 (minus cam signal 1 to 32)

Plus cam signals

The status of the plus cam signals is entered into the following NC/PLC interface signals:

DB10 DBX114.0 to 117.7 (plus cam signals 1 to 32)

Note

If no measuring system has been selected or NC/PLC interface signal DB31, ... DBX2.0 (cam activation) is set to 0, then the following NC/PLC interface signals are also set to 0:

- DB10 DBX110.0-113.7 (minus cam signals 1-32)
- DB10 DBX114.0-117.7 (plus cam signals 1-32)
- DB31, ... DBX62.0 (cams active)

13.3.3 Output of cam signals to NCK I/Os in position control cycle

Signal output in position control cycle

For cams assigned to an HW byte via machine data MD10470 to MD10473 (see Section "Hardware assignment"), the signal is output in the position control cycle.

The 4 onboard outputs on the NCU and a total of 32 optional external NC outputs are available as the digital outputs of the NC I/Os.

Further information

See Function Manual Basic Functions; Digital and analog NC I/O.

Hardware assignment

The assignment to the hardware bytes used is made for each eight cam pairs in the following general machine data:

- MD10470 SW_CAM_ASSIGN_FASTOUT_1 — Hardware assignment for the output of cams1 - 8 to NC I/O
- MD10471 SW_CAM_ASSIGN_FASTOUT_2 — Hardware assignment for the output of cams 9 - 16 to NC I/O
- MD10472 SW_CAM_ASSIGN_FASTOUT_3 — Hardware assignment for the output of cams 17 - 24 to NC I/O
- MD10473 SW_CAM_ASSIGN_FASTOUT_4 — Hardware assignment for the output of cams 25 - 32 to NC I/O

Note

It is possible to define one HW byte for the output of eight minus cam signals and one HW byte for the output of eight plus cam signals in each machine data.

In addition, the output of the cam signals can be inverted with the two machine data.

If the 2nd byte is not specified (= "0"), then the 8 cams are output as a logic operation of the minus and plus cam signals via the 1st HW byte using the 1st inversion screen form.

Status query in the part program

The status of the HW outputs can be read in the part program with main run variable **$A_OUT[n]** (n = no. of output bit).

Switching accuracy

Signals are output to the NC I/O or onboard outputs in the position control cycle.
As a result of the time scale of the position control cycle, the switching accuracy of the cam signals is limited as a function of the velocity.

The following applies:

Delta pos = V_{act} * position control cycle

With: Delta pos = switching accuracy (depending on position control cycle)
V_{act} = Current axis velocity

Example:

V_{act} = 20 m/min, PC cycle = 4 ms	Delta pos = 1.33 mm
V_{act} = 2000 rpm, PC cycle = 2 ms	Delta pos = 24 degrees

13.3.4 Timer-controlled cam signal output

Timer-controlled output

A significantly higher degree of accuracy can be achieved by outputting the cam signals independently of the position control cycle using a timer interrupt.

The following machine data can be used to select the timer-controlled output to the 4 NCU onboard outputs for 4 cam pairs:

MD10480 SW_CAM_TIMER_FASTOUT_MASK (screen form for the output of cam signals via timer interrupts on the NCU)

In this case, the minus and plus signals of a cam pair are logically combined for output as one signal.

Signal generation

Previously, it had to be specified in which way the signals to be logically combined should be generated. This is realized using bit 1 in machine data:

MD10485 SW_CAM_MODE (behavior of the software cams)

Bit	Value	Signal generation
1	0	Inversion of the signal behavior of the plus cam when: plus cam - minus cam ≥ 180 degrees
	1	**No** inversion of the signal behavior of the plus cam when: plus cam - minus cam ≥ 180 degrees

Note

This function operates independently of the HW assignment selected in MD10470 ... MD10473.

The onboard byte may not be used a multiple number of times.

Restrictions

The following applies to the mutual position of the cam positions:

Only **one** timer-controlled output takes place per interpolation cycle.

If signal changes for more than one cam pair are active in the same interpolation cycle, the output is prioritized:

The cam pair with the lowest number (1...32) determines the output time for **all** active signals, i.e. the signal change of the other cam pairs takes place at the same time.

PLC interface

The NC image of the onboard outputs and the status of the plus and minus cams is displayed at the PLC interface.

However, these signals are irrelevant or correspondingly inaccurate for the **timer-controlled** cam output version, as described in the following paragraphs. The signals for the plus and minus cams are generated (once) in synchronism with the interpolator clock cycle and transmitted together to the PLC.

Pulses shorter than an interpolator clock cycle are therefore not visible in the PLC.
The onboard outputs are set and reset asynchronously to the interpolator clock cycle for each interrupt. The status of the onboard outputs is detected and transmitted to the PLC in synchronism with the update time of the PLC interface.

Depending on the current status at the moment the PLC interface is updated, pulses shorter than one interpolator clock cycle are not visible or are displayed stretched by one or several IPO cycles.

Further settings

The following bit must be set to "0" if the behavior described here is to be activated:

MD10485 SW_CAM_MODE bit 0 = 0

13.3.5 Independent, timer-controlled output of cam signals

Independent, timer-controlled cam output

Each switching edge is output separately per interrupt due to the timer-controlled, independent (of interpolator clock cycle) cam output.

The mutual influence of the cam signals is no longer applicable as a result of:

- single output per interpolator clock cycle
- output time determined by highest priority cam pair (lowest cam pair number)

A total of 8 timer-controlled cam outputs per interpolator clock cycle can be configured for setting/resetting the 4 onboard outputs. The signal states of the plus and minus cams are also made available as standard on the PLC interface for the timer-controlled variant. However, these signals are not relevant or are correspondingly inaccurate with a timer-controlled output.

Signal generation

Previously, it had to be specified in which way the signals to be logically combined should be generated. This is realized using bit 1 in machine data:

MD10485 SW_CAM_MODE (behavior of the software cams)

Bit	Value	Signal generation
1	0	Inversion of the signal behavior of the plus cam when: plus cam - minus cam ≥ 180 degrees
	1	**No** inversion of the signal behavior of the plus cam when: plus cam - minus cam ≥ 180 degrees

Settings

Cam pairs are assigned to onboard outputs using machine data:

MD10480 SW_CAM_TIMER_FASTOUT_MASK (screen form for the output of cam signals via timer interrupts on the NCU)

In addition, this type of processing must be explicitly selected:

MD10485 SW_CAM_MODE bit 0 = 1

Note

This function operates independently of the HW assignment selected in MD10470 … MD10473.

The onboard byte may not be used a multiple number of times.

13.4 Position-time cams

Position-time cams

The term "position-time cam" refers to a pair of software cams that can supply a pulse of a certain duration at a defined axis position.

Solution

The position is defined by a pair of software cams.
The pulse duration is defined by the lead/delay time of the plus cam.

Machine data can be used to specify that cam pairs with "minus cam position = plus cam position" should be processed as position-time cams.

Properties of position-time cams

- The pulse duration is independent of the axis velocity and travel direction reversal.
- The pulse duration is independent of changes in the axis position (Preset).
- Activation (rising signal edge) takes place only when the cam position is crossed. Moving the axis position (e.g. preset) does not trigger activation.
- A lead/delay time is operative for the minus cam and causes a time displacement of the pulse.
- Activation (ON edge) and pulse duration are independent of the travel direction.
- The cam is not deactivated if the cam position is crossed again when the cam is active (direction reversal).
- The cam time (pulse width) is not interrupted and the cam time not restarted when the cam position is crossed again.
  This behavior is particularly relevant with respect to modulo axes, i.e. if the cam time is greater than the modulo range crossing time, the cam is not switched in every revolution.

Settings

The following settings must be made to program a position-time cam:

- **Position**
  The position must be defined by a cam pair with which the minus cam position is equal to the plus cam position.
  This is defined using setting data:
  SD41500 SW_CAM_MINUS_POS_TAB_1
  ...
  SD41507 SW_CAM_PLUS_POS_TAB_4.
- **Pulse duration**
  The pulse duration is calculated by adding together the associated entries for the cam pair in:
  MD10461 SW_CAM_PLUS_LEAD_TIME[n]
  SD41521 SW_CAM_PLUS_TIME_TAB_1[n]...
  SD41527 SW_CAM_PLUS_TIME_TAB_4[n]

- **Offset**
  The time displacement of the position-time cam is calculated by adding together the associated entries for the cam pair in:
  MD10460 SW_CAM_MINUS_LEAD_TIME[n]
  SD41520 SW_CAM_MINUS_TIME_TAB_1[n]...
  SD41526 SW_CAM_MINUS_TIME_TAB_4[n]
- **Mode**
  MD10485 SW_CAM_MODE
  Bit 2 = 1 must be set in the machine data to ensure that all cam pairs with the same values for minus cam and plus cam positions are treated as position-time cams.

13.5 Boundary conditions

Availability of function "Software cams, position switching signals"

The function is an option ("position-switching signals/cam controller"), which must be assigned to the hardware through the license management.

13.6 Data lists

13.6.1 Machine data

13.6.1.1 General machine data

Number	Identifier: $MN_	Description
10260	CONVERT_SCALING_SYSTEM	Basic system switchover active
10270	POS_TAB_SCALING_SYSTEM	System of units for positioning tables
10450	SW_CAM_ASSIGN_TAB[n]	Assignment of software cams to machine axes
10460	SW_CAM_MINUS_LEAD_TIME[n]	Lead or delay time at the minus cams 1-16
10461	SW_CAM_PLUS_LEAD_TIME[n]	Lead or delay time at the plus cams 1-16
10470	SW_CAM_ASSIGN_FASTOUT_1	Hardware assignment for the output of cams 1-8 to NC I/O
10471	SW_CAM_ASSIGN_FASTOUT_2	Hardware assignment for the output of cams 9-16 to NC I/O
10472	SW_CAM_ASSIGN_FASTOUT_3	Hardware assignment for the output of cams 17-24 to NC I/O
10473	SW_CAM_ASSIGN_FASTOUT_4	Hardware assignment for the output of cams 25-32 to NC I/O

Number	Identifier: $MN_	Description
10480	SW_CAM_TIMER_FASTOUT_MASK	Screen form for the output of the cam signals via timer interrupts on the NCU
10485	SW_CAM_MODE	Behavior of the software cams

13.6.2 Setting data

13.6.2.1 General setting data

Number	Identifier: $SN_	Description
41500	SW_CAM_MINUS_POS_TAB_1[n]	Position of minus cams 1 -8
41501	SW_CAM_PLUS_POS_TAB_1[n]	Position of plus cams 1 -8
41502	SW_CAM_MINUS_POS_TAB_2[n]	Position of minus cams 9 -16
41503	SW_CAM_PLUS_POS_TAB_2[n]	Position of plus cams 9 -16
41504	SW_CAM_MINUS_POS_TAB_3[n]	Position of minus cams 17 -24
41505	SW_CAM_PLUS_POS_TAB_3[n]	Position of plus cams 17 -24
41506	SW_CAM_MINUS_POS_TAB_4[n]	Position of minus cams 25 -32
41507	SW_CAM_PLUS_POS_TAB_4[n]	Position of plus cams 25 -32
41520	SW_CAM_MINUS_TIME_TAB_1[n]	Lead or delay time on minus cams 1 -8
41521	SW_CAM_PLUS_TIME_TAB_1[n]	Lead or delay time on plus cams 1 -8
41522	SW_CAM_MINUS_TIME_TAB_2[n]	Lead or delay time on minus cams 9 -16
41523	SW_CAM_PLUS_TIME_TAB_2[n]	Lead or delay time on plus cams 9 -16
41524	SW_CAM_MINUS_TIME_TAB_3[n]	Lead or delay time on minus cams 17 -24
41525	SW_CAM_PLUS_TIME_TAB_3[n]	Lead or delay time on plus cams 17 -24
41526	SW_CAM_MINUS_TIME_TAB_4[n]	Lead or delay time on minus cams 25 -32
41527	SW_CAM_PLUS_TIME_TAB_4[n]	Lead or delay time on plus cams 25 -32

M3: Coupled axes 14

14.1 Coupled motion

14.1.1 Brief description

14.1.1.1 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.

14.1.1.2 Preconditions

"Coupled motion" function

The "Coupled motion" function is a fixed component of the NC 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 NC software is supplied with the relevant options of generic coupling (refer to " Preconditions " in the "Brief description" of generic coupling).

14.1.2 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 = motion of the coupled motion axis / motion of the leading axis

Negative coupling factors (motion of the coupled motion axis in the opposite direction) are also permitted.

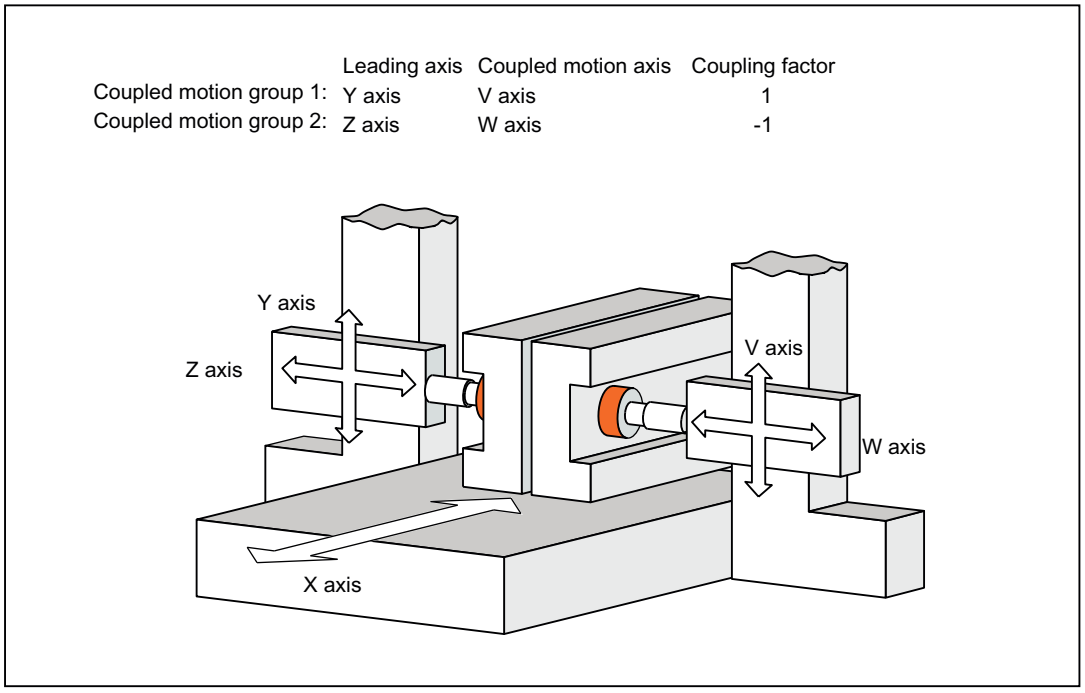


Figure 14-1 Application example: Two-sided machining

Multiple couplings

Up to 2 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).

Activating/deactivating

Coupled motion can be activated/deactivated via the part programs and synchronous actions. In this context please ensure activating/deactivating is undertaken with the same programming:

- Activate: Part program → Deactivate: Part program
- Activate: Synchronous action → Deactivate: Synchronized 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

  No coupling

  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 for NC-START)

Bit	Value	Meaning
8	0	Coupled-axis groupings are maintained in NC Start.
	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	Meaning
8	0	Coupled-axis groupings are invalidated on RESET / part program end.
	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.

14.1.3 Programming

14.1.3.1 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: `TRAILON(<coupled motion axis>, <leading axis>, [<coupling factor>])`

Effectiveness: modal

Parameters:

- Coupled motion axis: Type AXIS:
  - Range of values: All defined axis and spindle names in the channel
- Leading axis: Type AXIS:
  - Range of values: All defined axis and spindle names 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 = Path of the coupled-motion axis/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 * 10-308 … 1,8 * 10+308)
  - Default value: +1.0

Example:

Program code	**Comment**
`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.`

14.1.3.2 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>)
	TRAILOF(<coupled-motion axis>)
Effectiveness:	modal
Parameters:	
Coupled motion axis:	Type: AXIS
	Range of values: All defined axis and spindle names in the channel
Leading axis:	Type: AXIS
	Range of values: All defined axis and spindle names in the channel

Example:

```
Program code          Comment
TRAILOF(V,Y)          ;   Deactivation of the coupling of the coupled-motion
                      ;   axis
                          V to leading axis Y.
```

14.1.4 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 (feedrate override)
- DB31, ... DBX1.3 (axis blocking)
- DB31, ... DBX2.1 (control system enable)
- DB31, ... DBX4.0 - 4.2 (activate handwheel)
- DB31, ... DBX4.3 (feed stop)
- …

This allows the speed to be changed for the independent motion of a coupled motion axis using a feedrate 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 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/spindel active)

DB31, ... DBX99.1 (following axis/spindel active)

$AA_COUP_ACT[axis name] (see: Status of coupling)

14.1.5 Status of coupling

The coupling status of an axis can be determined using the following system variables:

$AA_COUP_ACT [axis name]

Value	Meaning
0	No coupling active
1, 2, 3	Tangential tracking
4	Synchronous spindle coupling
8	Coupled motion active

Value	Meaning
16	Master value coupling
32	Following axis of electronic gearbox

Note

Only one coupling mode may be active at any given time.

14.1.6 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 ($AA_TYP ≠ 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.

⚠ CAUTION

Axis overload

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.

14.1.7 Supplementary conditions

Control system dynamics

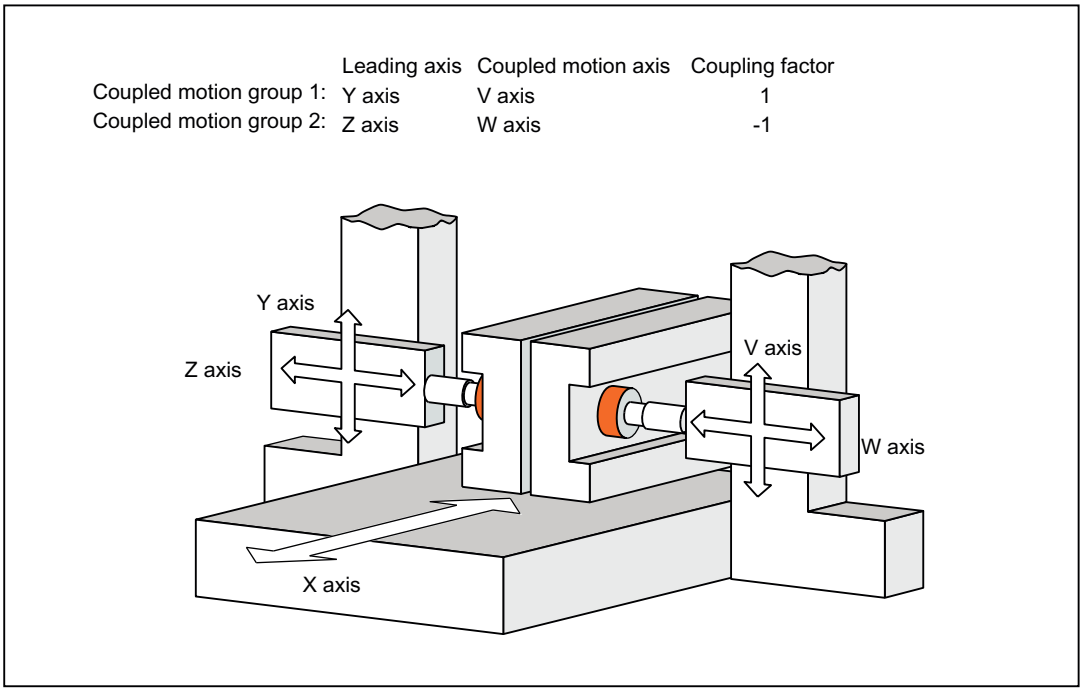
It is recommended to align the position control parameters of the leading axis and the coupled motion axis within a coupled axis group.

Note

Alignment of the position control parameters of the leading axis and the coupled motion axis can be performed via a parameter set changeover.

14.1.8 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

Program code	Comment
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	;
N02 TRAILON(U,X,1)	; Activation of coupled axis group
N03 G1 F2000 X200	; Dependent movement of U, Upos=200, UTrail=100
N04 U201	; Independent movement, Upos=U201+UTrail=301
N05 X250	; Dependent movement of U, UTrail=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)	

14.2 Curve tables

14.2.1 Product brief

14.2.1.1 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.

14.2.1.2 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)

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)

14.2.2 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.

Further information
Programming Manual Advanced

Curve segments are used if:

- Polynomials 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.

14.2.3 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 433)").

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 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.

14.2.4 Commissioning

14.2.4.1 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	
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 polynomes** that can be stored in the static NC memory.

Dynamic NC memory	
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_SEGNENTS_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 polynomes** 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.

14.2.4.2 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.

14.2.4.3 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 tabels are normally created in the static NC memory.
1	Curve tabels are normally created in the dynamic NC memory.

14.2.5 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
  CTABSSV(leading value, n, degrees, [following axis, leading axis])
- Read end value (following axis value) of a table segment
  CTABSEV(master value, n, degrees, [following axis, master axis])

Note

If curve table functions such as CTAB(), CTABINV(), CTABSSV() etc., in **synchronous actions** are used, only **main traverse variable**, e.g. $AC_PARAM[...] or $R[...] is permissible for the return value and the argument "degrees" of the function.

Example:

ID=1 WHEN TRUE DO $R1 = CTABSSV(10, 1, $R2)

or

ID=1 WHEN TRUE DO $AC_PARAM[1] = CTABSSV(10, 1, $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: CTABLOCK(n, m, memType)
- **Cancel** deletion and overwrite **block**.
  CTABUNLOCK releases the tables locked with CTABLOCK. Tables involved in an active coupling remain locked, i.e. they cannot be deleted. However, the CTABLOCK command is cancelled, i.e. the table can be deleted as soon as the coupling is deactivated. It is not necessary to call CTABUNLOCK again.
  General form: CTABUNLOCK(n, m, memType)

Applications of the forms:

Curve table with number n

CTABLOCK(n)

Curve tables in the number range n to m.

CTABLOCK(n, m)

All curve tables, irrespective of memory type

CTABLOCK()

All curve tables in the specified memory type

CTABLOCK(, , memType)

Curve table with number n

CTABUNLOCK(n)

Curve tables in the number range n to m.

CTABUNLOCK(n, m)

All curve tables, irrespective of memory type

CTABUNLOCK()

All curve tables in the specified memory type

CTABUNLOCK(, , 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
  CTABNO()
- Number of **defined** tables in SRAM or DRAM of NC memory
  CTABNOMEM(memType)
- Number of **possible** curve tables in memory memType.
  CTABFNO(memType)
- Table number of **nth** curve table.
  General form: CTABID(n, memType)
  Generates the table number of the nth curve table with memory type memType. CTABID(1, memType) is used to read out the highest curve number (105) of the memory type specified.
  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)
  CTABID(p)
  Generates the ID (table number) of the curve table entered in the memory as the pth curve table.

  Note

  If for example, the sequence is changed between consecutive calls of CTABID(), then 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.
  CTABISLOCK(n)
- **Checks** curve table number n.
  CTABEXISTS(n)
- Returns the **memory** in which curve table number n is stored.
  CTABMEMTYP(n)
- Returns the **table periodicity**.
  CTABPERIOD(n)
- Number of **curve segments** already used in memory memType.
  CTABSEG(memType, segType)
- Number of curve segments used in **curve table** number n
  CTABSEGID(n, segType)
- Number of **still** possible **curve segments** in memory memType.
  CTABFSEG(memType, segType)

- **Maximum** number of possible **curve segments** in memory memType.
  CTABMSEG(memType, segType)
- Number of **polynomials already used** in memory memType.
  CTABPOL(memType)
- Number of **curve polynomials** used by curve table number n.
  CTABPOLID(n)
- Number of **still** possible **polynomials** in memory memType.
  CTABFPOL(n)
- **Maximum** number of possible **polynomials** in memory memType.
  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.
  CTABTSV(n, degrees, F axis), following value at the beginning of the curve table
- The value at the **end** of the curve table is read by a following axis.
  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.
  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.
  CTABTEP(n, degrees, FAxis), master value at the end of the curve table
- Determine the value range of the following value.
  CTABTMIN(n, FAxis), minimum following value of curve table
  CTABTMAX(n, FAxis), maximum following value of the curve table

Parameter

- Following axis:
  Name of axis via which the following axis is programmed in the definition.
- Leading axis:
  Name 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.
- memType
  Optional parameter for specifying memory type to be used n 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

Further information
Programming Manual Advanced; Axis couplings, curve tables (CTAB)

Restrictions

The following restrictions apply when programming:

- It is not permissible that the NC block generates 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 hidden.

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	; 1st Motion instruction specifies the start value
	; leading value: 0, Following value: 0
N110 AX1=20 AX2=0	; 1st Curve segment: Master value: 0...20,
	; following value: Starting value …0
N120 AX1=100 AX2=6	; 2nd Curve segment: Master value: 20...100,
	; following value: 0...6
N130 AX1=150 AX2=6	; 3rd Curve segment: Master value: 100...150,
	; following value 6
N130 AX1=180 AX2=0	; 4th Curve segment: Master value: 150...180,
	; following 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 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 CTABEND.

```
Program code                              Comment
N10 CTABDEF(Y, X, 1, 0)                   ; start of the definition for the non-
                                          ; periodic curve table no. 1
N20 X0 Y0
N30 G41 X10 Y0                            ; tool radius compensation on
N40 X20 Y20
N50 X40 Y0
N60 X60 Y20
N70 X80 Y0
N80 G40 X90 Y0                            ; tool radius 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 CTABDEF and CTABEND must be specified for precisely the axis names that have been programmed in CTABDEF as the leading axis name and following axis name. In the case of programming errors, alarms or incorrect contours may be generated.

14.2.6 Access to table positions and table segments

Reading table positions

With the program commands CTAB and CTABINV the following value for a master value (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 CTABSSV and 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 interpolator clock cycle.

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 CTABSSV and CTABSEV:

Program code	Comment
N10 DEF REAL STARTPOS	; Beginning 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)	; Start 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=	; End position Y in segment 2 = 40
CTABSEV(30.0,1,GRADIENT)	; Segment 2 belongs to LW X = 30.0

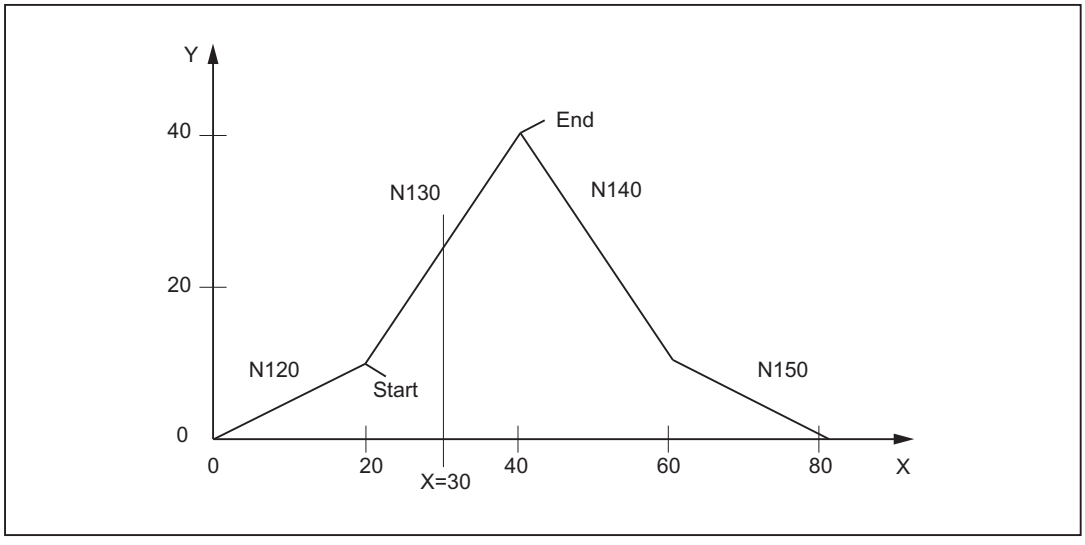


Figure 14-2 Determining the curve segment associated with master value X = 30

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 the definition for the start and initial values of the curve table
N20 DEF REAL ENDVAL	
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)	; Start 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 curve segment ...
N150 X80 Y20	; End position of the 4th curve 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 ; Read master value at the end of the curve table from the value range of the following axis
N240 U_MINVAL=CTABTMIN(1)	; Minimum value when Y = 5 and
N250 U_MAXVAL=CTABTMAX(1)	; Maximum value when Y = 40

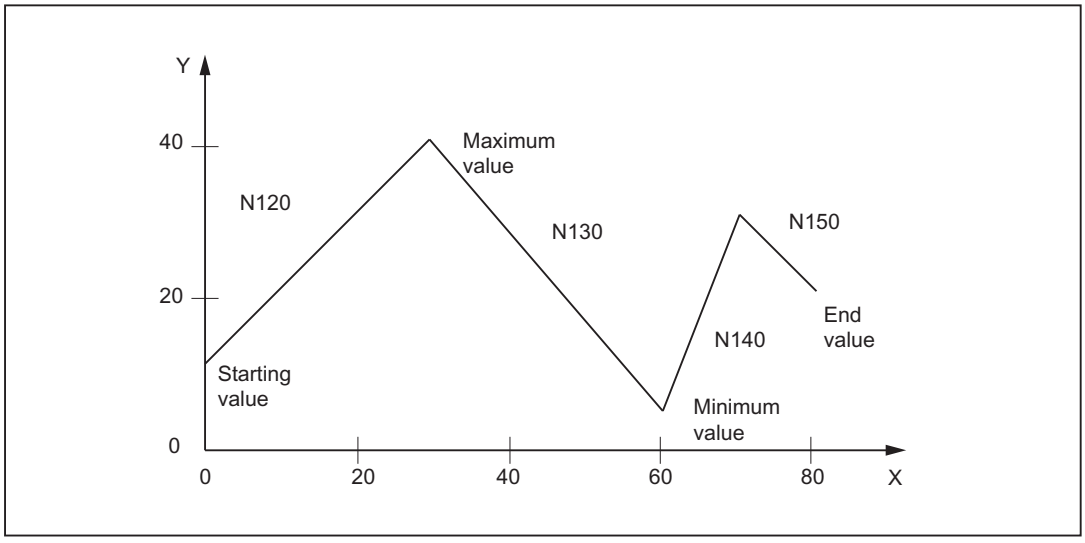


Figure 14-3 Determining the minimum and maximum values of the table

14.2.7 Activation/deactivation

Activation

The coupling of real axes to a curve table is activated through this command:

`LEADON` (<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:

```
...
N1000 LEADON(A,X,3)          ; Axis A follows the master value X according to the rule
                               of motions defined in Curve Table No. 3
...
```

Deactivation

The switch off of the coupling to a curve table takes place through the following command:

`LEADON` (<Following axis>, <Leading axis>)

Deactivation is possible:

- In the part program
- in synchronized actions

Note

While programming LEADOF, the abbreviated form is also possible without specification of the leading axis.

Example:

```
...
N1010 LEADOF(A,X)          ; The coupling of Axis A with its leading axis is canceled
...
```

Multiple use

A curve table can be used several times in a single part program to couple different channel axes.

14.2.8 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 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 round (360°) = 570°

14.2.9 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.

14.2.10 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.

14.2.11 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
  `CTABID(p, memType)`
  If memType is not specified, the memory type specified in the following machine data:
  MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE
  Result:
  Table number or
  Alarm for invalid p or memType

When using the `CTABID(p, memType)` function, no assumptions should be made regarding the sequence of the curve tables in the memory. The `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 `CTABID()` CTABID(), e.g. due to the deletion of curve tables with `CTABDEL()`, the `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 `CTABLOCK(...)` language command. In this case, it should be noted that the curve tables concerned are then unlocked with `CTABUNLOCK()`.

- Determine **block condition**
  Table n
  `CTABISLOCK(n)`
  Result:
  > 0: Table is blocked
  Reason for block:
  1: by `CTABLOCK`()
  2: by an active coupling
  3: by `CTABLOCK()` and by an active coupling
  = 0: Table is not blocked
  - 1: Table does not exist
- Check whether the curve table **exists**
  `CTABEXISTS(n)`
  Result:
  1: Table exists
  0: Table does not exist
- 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 polynomisl 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
  `CTABSEGID(n, segType)`
  Result:
  >= 0: Number of curve segments
  -1: Curve table with number n does not exist
  -2: segType not equal "L" or "P"
- Determine number of **free** curve segments of the type memType in the memory range
  `CTABFSEG(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 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
  `CTABMSEG(memType, 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
  `CTABPOL(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
  `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
  `CTABFPOL(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 free curve polynomials
  -2: Invalid memory type
- Determine the **maximum** number of polynomials of the memory type
  `CTABMPOL(memType)`
  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 polynomials
  -2: Invalid memory type

14.2.12 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.

14.2.13 Examples

Definition of a curve table with linear sets

```
%_N_TAB_1_NOTPERI_MPF
;$PATH=/_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

```
%_N_TAB_1_NOTPERI_MPF
;$PATH=/_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)
```

```
%_N_TAB_1_NOTPERI_MPF
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)
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 LEADOF(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 LEADOF(Y,X)
N230 M30
```

14.3 Master value coupling

14.3.1 Product brief

14.3.1.1 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.

14.3.1.2 Preconditions

The option "Master Value Coupling and Curve Tables Interpolation" or the relevant option of generic coupling (refer to " Preconditions " in the "Brief description" for generic coupling) is a prerequisite for utilization of the function.

14.3.2 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 clock cycle, the interpolator for the axis is initialized by the NC 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.

SD43104 $SA_LEAD_SCALE_IN_POS[FA]
The input value is multiplied.

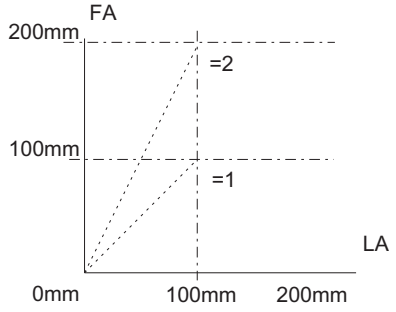
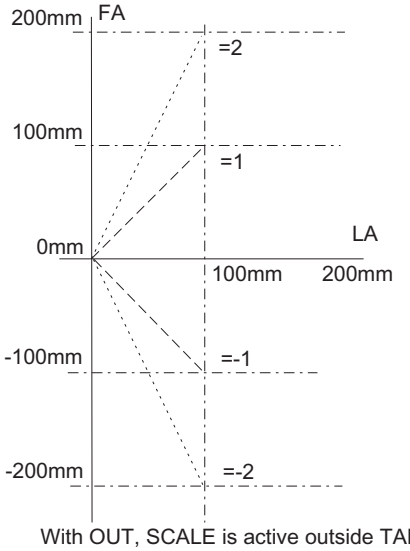


Table definition for example:
CTABDEF(FA,LA,CTABNR,1)
LA=0 FA=0
LA=100 FA=100
CTABEND

With IN, SCALE is not active outside TAB!

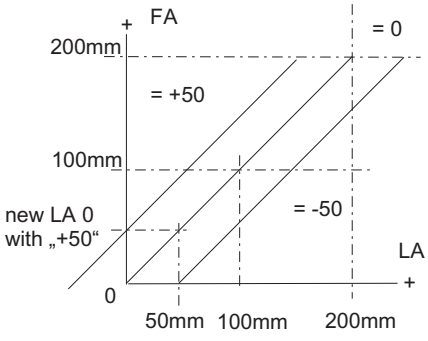
SD43108 $SA_LEAD_SCALE_OUT_POS[FA]
The output value is multiplied. Setting data is effective outside of the table definition, also in the negative direction.



With OUT, SCALE is active outside TAB!

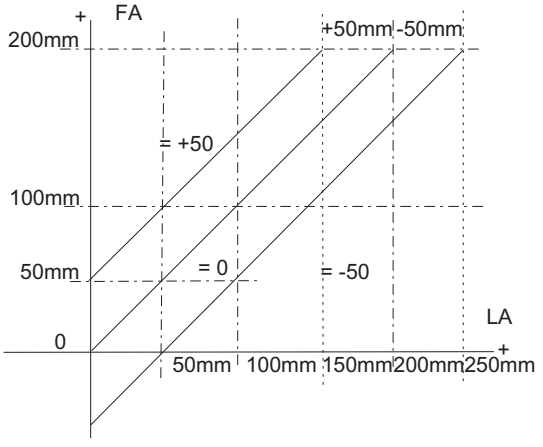
Figure 14-4 Master value coupling offset and scaling (multiplied)

SD43102_$SA_LEAD_OFFSET_IN_POS[FA]
The calculated position of the following axis is offset with the specified increments.



With IN, OFFSET is not active outside TAB!

SD43106 $SA_LEAD_OFFSET_OUT_POS[FA]
The calculated position of the following axis is offset with the specified increments. If the TAB is moved outside of the FA LEADON position, KOP cannot be switched on. Setting data is effective outside of the table definition, also in the negative direction.



With OUT, OFFSET active outside TAB!

Figure 14-5 Master value coupling offset and scaling (with increment offset)

Reaction to Stop

All leading 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 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.

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)	; Enable 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.

14.3.3 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 Value range: 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)

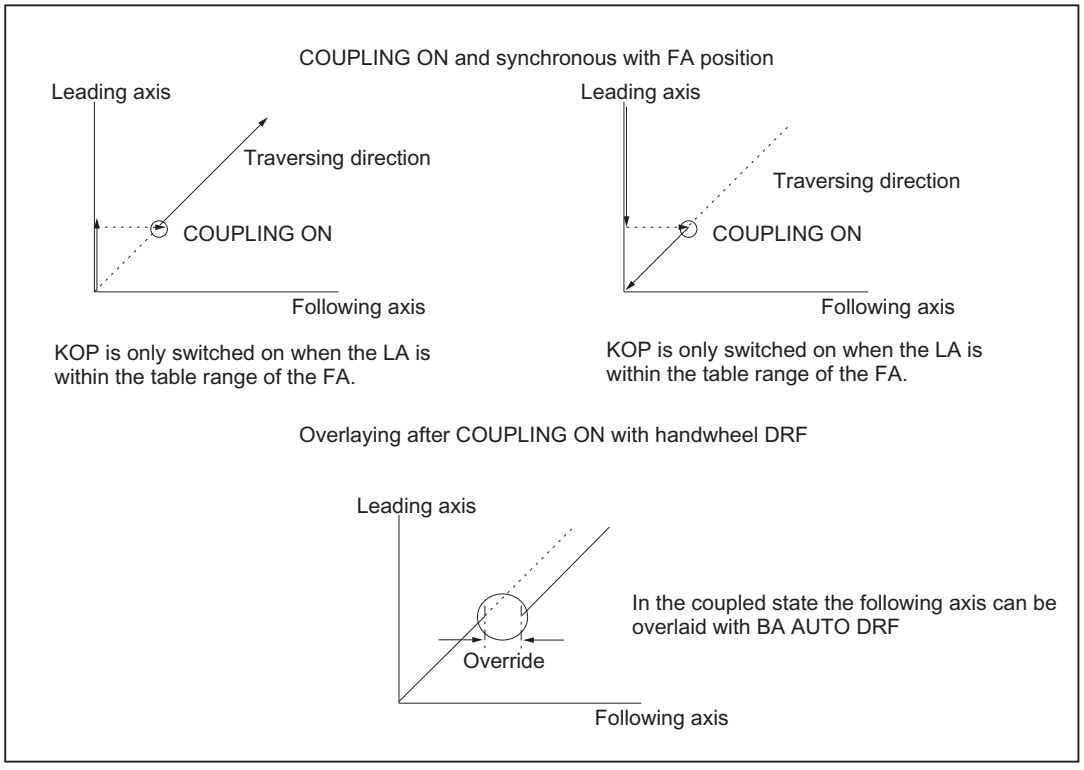


Figure 14-6 Activating master value coupling

Switch off

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 overwritten by another command with a synchronous action.

Syntax:

```
LEADON(<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.

Further information
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 is: $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_IN_SYNC[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.

14.3.4 Behavior in AUTOMATIC, MDA and JOG modes

Effectiveness

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 should 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)

- MD20110 $MC_RESET_MODE_MASK=2001H
  &&
  MD20112 $MC_START_MODE_MASK=0H
  → Master value coupling remains valid after RESET and START
- MD20110 $MC_RESET_MODE_MASK=2001H
  &&
  MD20112 $MC_START_MODE_MASK=2000H
  → Master value coupling remains valid after RESET and is canceled with START. However, master value coupling activated via IDS=... remains valid.
- MD20110 $MC_RESET_MODE_MASK=1H
  → Master value coupling is canceled with 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 RESET and is canceled with START, irrespective of machine data:
  MD20112 $MC_START_MODE_MASK
  However, master value coupling activated via IDS=... remains valid.

Further information
Function Manual Basic Functions; Coordinate systems, axis types, axis configurations, ...

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 (definition of initial control settings after RESET / TP End)
  and
  MD20112 $MC_START_MODE_MASK (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

14.3.5 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.

14.3.6 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 ($K_V$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 Chapter "Status of coupling (Page 426)"

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.

14.3.7 Supplementary conditions

External master value axes

When using the `REPOS` or `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 `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.

14.4 Electronic gear (EG)

14.4.1 Product brief

14.4.1.1 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
- 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

14.4.1.2 Preconditions

The "Electronic Gearbox" option or the relevant option of generic coupling (refer to " Preconditions " in the "Brief description" of Generic Coupling) is required for the usage of function.

14.4.2 Electronic gear (EG)

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.

$FA_{set} = SynPosFA + (LA_{1}\text{-}SynPosLA_{1})*CF_{1} + ... + (LA_{5}\text{-}SynPosLA_{5})*CF_{5}$

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
Application faults Knowledge of the control technology and measurements with servo trace are an absolute prerequisite for using this function.

Further information

- CNC Commissioning Manual: NC, PLC, drive
- G2: Velocities, setpoint / actual value systems, closed-loop control (Page 31)

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:

EGDEF

(see Section "Definition of an EG axis group (Page 473)")

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:

EGON

EGONSYN

EGONSYNE

(see Section "Definition of an EG axis group (Page 473)")

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".

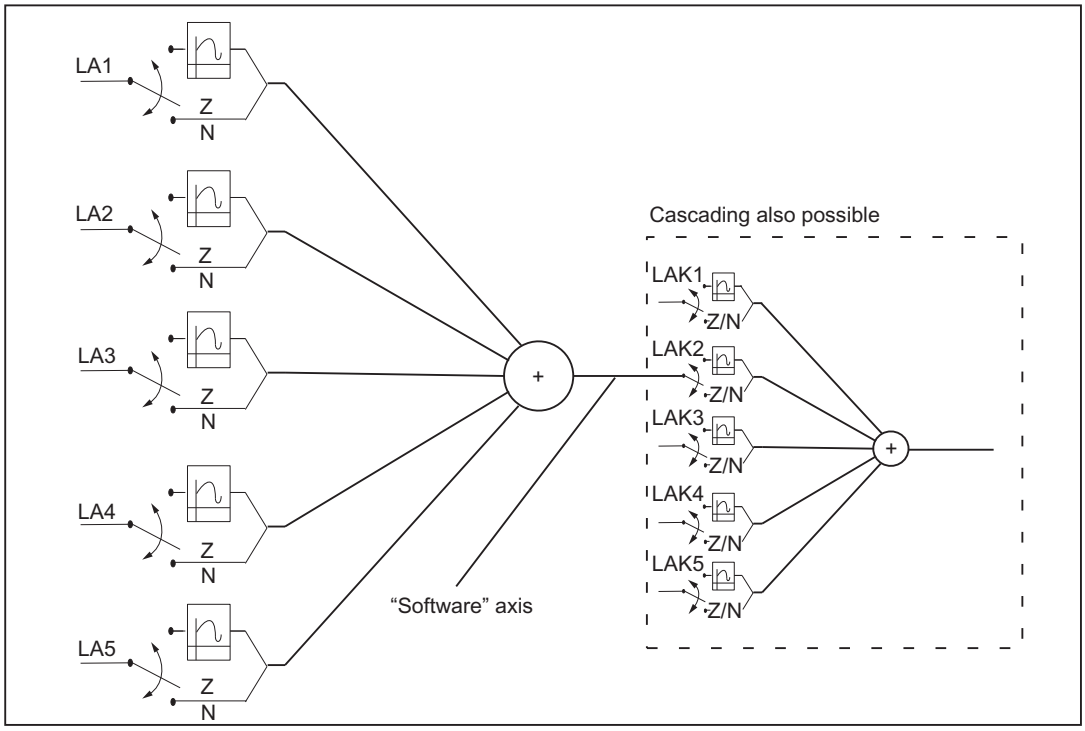


Figure 14-7 Block diagram of an electronic gearbox

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 474)")
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 474)")
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 474)")

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 clock 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 **synchronism difference** 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 synchronism difference can be queried from the part program (see below).

Changes in the synchronism difference

The mass inertia of the axis systems during acceleration can cause dynamic fluctuations in the synchronism difference. The synchronism difference is checked continuously and the tolerance values in the machine data used to produce interface signals.

The synchronism difference is compared with the following machine data:

MD37200 $MA_COUPLE_POS_TOL_COARSE

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

NST "synchronous traverse coarse" DB31, ... DBX98.1

Difference > .. TOL_COARSE

As long as the synchronism 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 synchronism 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 the synchronism 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

Synchronism difference for EG cascades

Synchronism difference for EG cascades is the deviation of the actual position of the following axis from setpoint position that results for the rule of motion for the real axes involved.

Example:

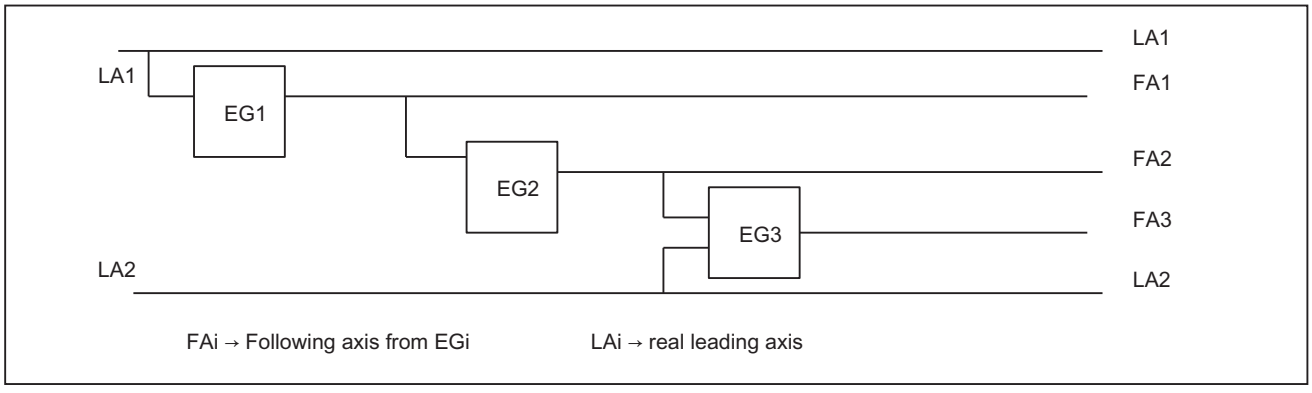


Figure 14-8 Three-level EG cascade

According to the definition given, the synchronism difference of following axis FA3 in the example below is determined by the value of following axis $FA3_{Act}$ and the value of leading axis $FA2_{Act}$ and $LA2_{Act}$, but not by $LA1_{Act}$ and $FA1_{Act}$.

If FA2 is not a real axis, the actual value $FA2_{Act}$ is not available. In this case, the **setpoint** of the axis derived solely from the leading axis value $FA1_{Act}$ 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

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 synchronism 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 synchronism 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.

14.4.3 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 type 2,...)

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".

14.4.4 Activating an EG axis group

Without synchronization

The EG axis group is activated **without synchronization selection** with:

EGON(FA, block change mode, LA1, Z1, N1, LA2 , Z2, N2,..LA5, Z5, N5.)

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_i: Axis name of the leading axis i

Z_i: Counter for coupling factor of leading axis i

N_i: Denominator for coupling factor of leading axis 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

EGONSYN(FA, block change mode, SynPosFA, LA_i, $SynPosLA_i$, Z_LA_i, 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

LA_i: Axis name 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, This way the fastest possible synchronized position is approached reliably, e. g., the next tooth from the tooth spacing (360 degree* Zn/Ni) and the associated synchronized positions.
(So-called *relative synchronization*.)

thereby ensuring that they approach the fastest possible synchronized position. (So-called *relative synchronization*, e.g. the next tooth gap after "centering".)

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. EGONSYNE

EGONSYNE(FA, block change mode, SynPosFA, Approach mode, LA_i,$SynPosLA_i$, Z_LA_i, 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

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": AbsoluteCoordinateNegative, Absolute measurement specification, rotary axis traverses in negative rotation direction

"ACP": AbsoluteCoordinatePositive, Absolute measurement specification, rotary axis traverses in positive rotation direction

"DCT": DirectCoordinateTime-optimized, Absolute measurement specification, rotary axis traverses time-optimized to programmed synchronized position

"DCP": DirectCoordinatePath-optimized, Absolute measurement specification, rotary axis traverses path-optimized to programmed synchronized position

LA_i: Axis name 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).

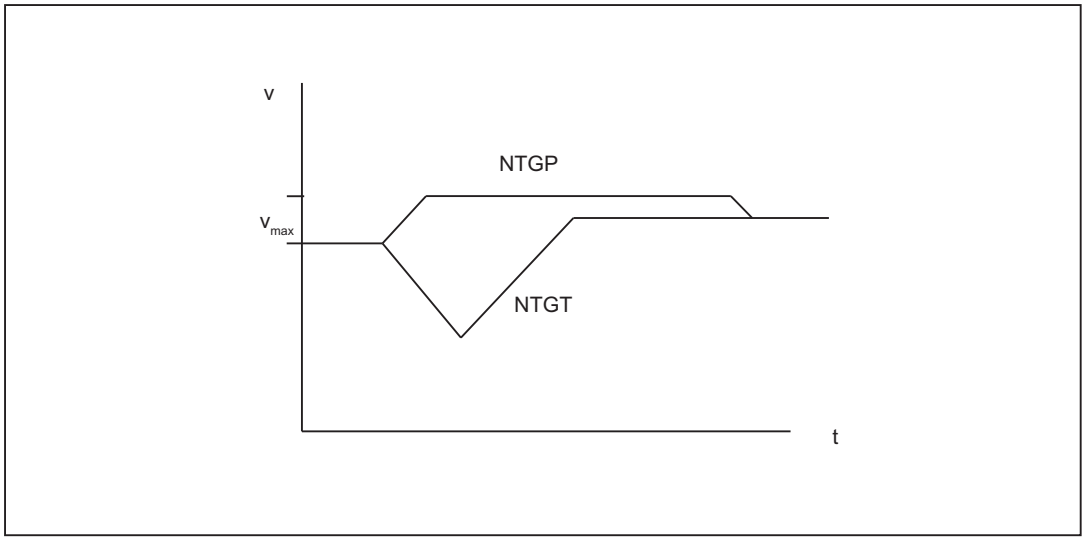


Figure 14-9 Reaching the next tooth gap, FA path-optimized (top) vs. time-optimized (bottom)

Sample notations

EGONSYNE(A, "FINE", 110, "NTGT", B, 0, 2, 10)

couple A to B, synchronized position A = 110, B = 0, coupling factor 2/10, approach mode = NTGT

EGONSYNE(A, "FINE", 110, "DCT", B, 0, 2, 10)

couple A to B, synchronized position A = 110, B = 0, coupling factor 2/10, approach mode = DCT

EGONSYNE(A, "FINE", 110, "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 (N_i) 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 (Z_i) 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 (LA_i) corresponds to the one specified for coupling via coupling factor (linear coupling).

14.4.5 Deactivating an EG axis group

Variant 1

There are different ways to deactivate an active EG axis grouping.

EGOFS(following axis)

The electronic gear 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 name must be programmed.

14.4.6 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.

14.4.7 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. Please note the following with respect to this command:

- 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 motion of the following axis are not taken into account.

Further information

V1: Feedrates (Page 125)

14.4.8 Response to POWER ON, RESET, operating mode change, block search

Function	Behavior regarding electronic gearbox	
	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 while a coupling is active (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()

Further information

Function Manual Basic Functions; Mode group, channel, program operation, reset response, block search type 5 SERUPRO > Lock program section for continue machining at the contour

14.4.9 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 14-1 System variables, R means: Read access possible

Name	Type	Access		Preprocessing stop		Meaning, value	Cond. Index
		part program	Sync act.	part program	Sync act.		
$AA_EG_TYPE[a,b]	INT	R		R		Type of coupling: 0: actual value 1: Setpoint value coupling	Axis name a: Following axis b: Leading axis
$AA_EG_NUMERA[a,b]	REAL	R		R		Numerator of coupl. factor KF KF = numerator/denominator preset: 0 Number of curve table when $AA_EG_DENOM[a,b] is 0.	Axis name a: Following axis b: Leading axis
$AA_EG_DENOM[a,b]	REAL	R		R		Denominator of coupl. fact. KF KF = 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.	Axis name a: Following axis b: Leading axis
$AA_EG_SYN[a,b]	REAL	R		R		Synchronized position for specified leading axis Default: 0	Axis name a: Following axis b: Leading axis
$AA_EG_SYNFA[a]	REAL	R		R		Synchronized position for specified following axis Default: 0	Axis name a: Following axis

Name	Type	Access		Preprocessing stop		Meaning, value	Cond. Index
		part program	Sync act.	part program	Sync act.		
$P_EG_BC[a]	STRING	R		R		Block change criterion for EG activation calls: EGON, EGONSYN: "NOC": immediately "FINE": Synchronoous traverse fine "COARSE": Synchronous traverse coarse "IPOSTOP": Setpoint-based synchronism	Axis name 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 name a: Following axis
$AA_EG_AX[n,a]	AXIS	R		R		Axis name of leading axis whose index n has been specified.	Axis name 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 name a: Following axis b: Leading axis
$VA_EG_SYNCDIFF[a]	REAL	R	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 name a: Following axis

14.4.10 Examples

14.4.10.1 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.

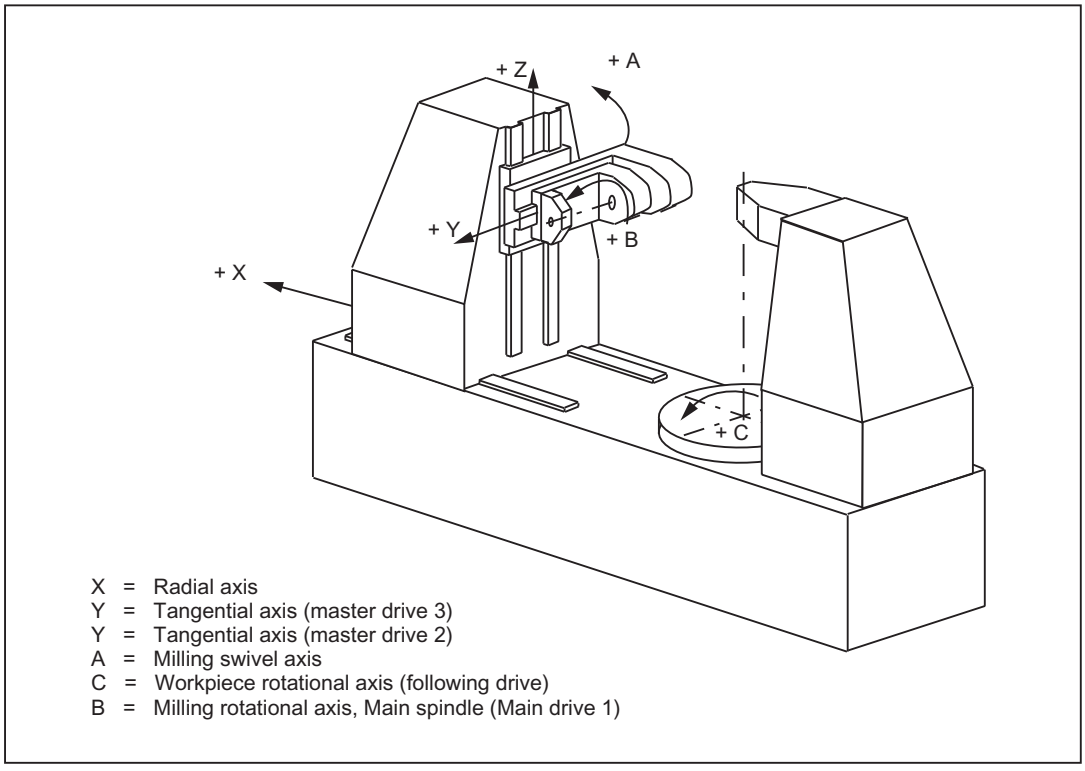
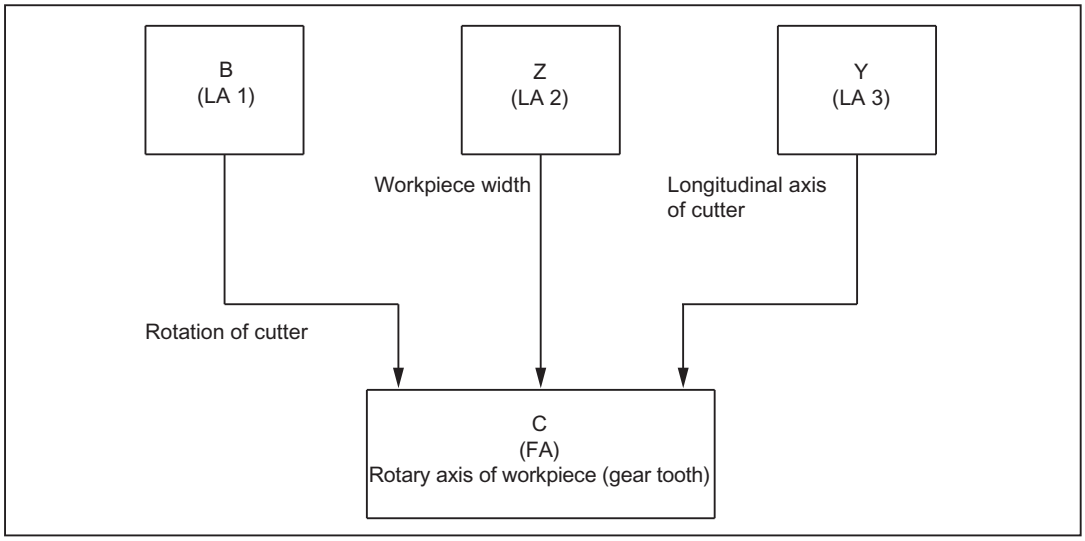


Figure 14-10 Definition of axes on a gear hobbing machine (example)

The functional interrelationships on the gear hobbing machine are as follows:



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:

$n_c = n_b * (z_0 / z_2) + v_z * (u_{dz} / z_2) + v_y * (u_{dy} / z_2)$

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.

$u_{dz} = (\sin \beta° / (m_n * \pi)) * 360$ [degrees/mm]
$u_{dy} = (\cos \gamma° / (m_n * \pi\square)) * 360$ [degrees/mm]

with:
- m_n = Normal module (in mm)
- $\beta°$ = Incline angle of gear wheel
- $\gamma°$ = 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,udz,z2,Y,udy,z2)`	`; Activate coupling.`
…	

14.4.10.2 Extended example with non-linear components

Introduction

The following example extends the example (see "Figure 14-10 Definition of axes on a gear hobbing machine (example) (Page 483)") 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 center of the tooth is stressed more than the edges during operation.

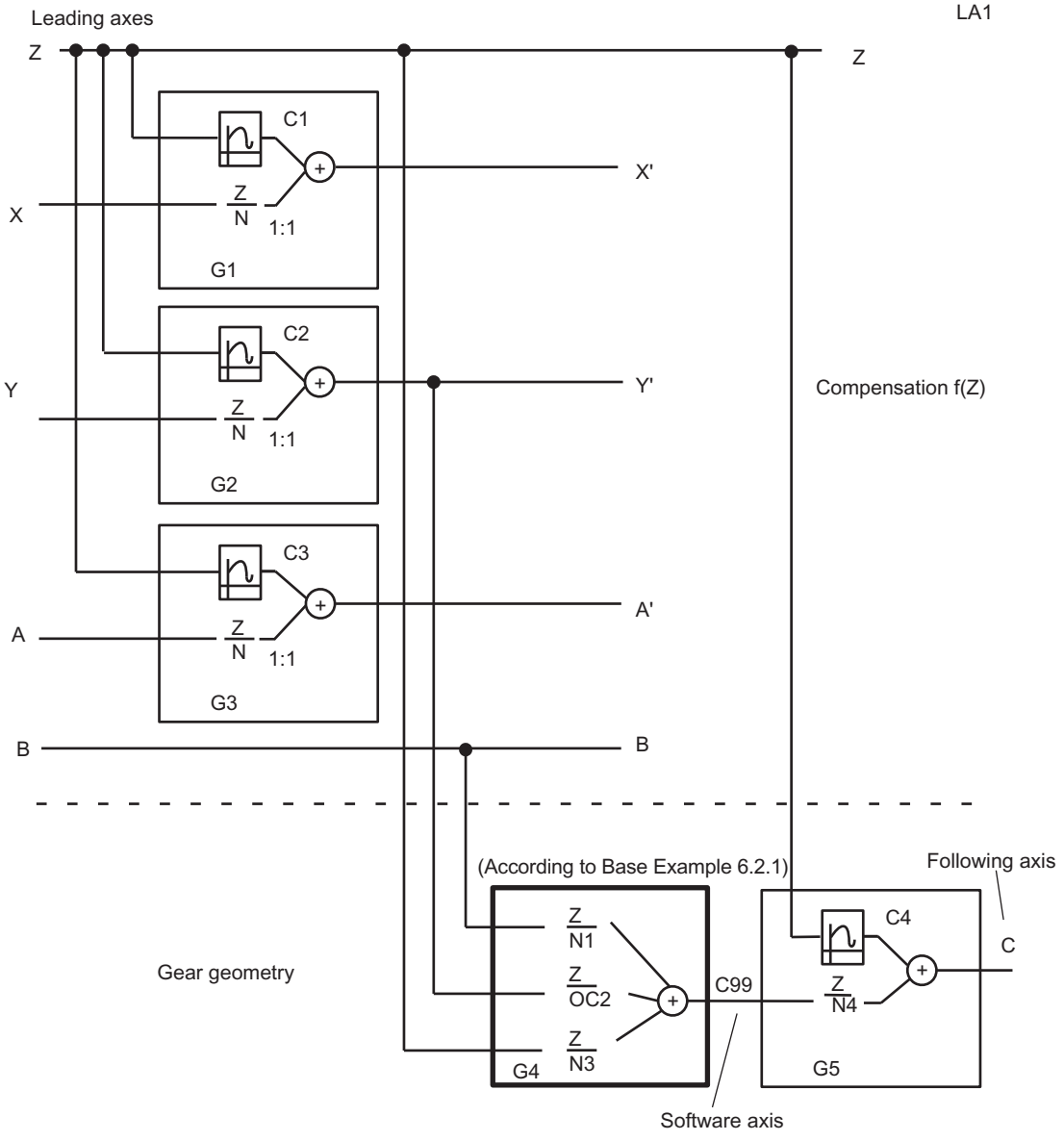


Figure 14-11 Extended example with non-linear machine fault compensation and non-linear components on the tooth geometry

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.

```
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,     ; declaration and specification of non-periodic curve table C2
0)
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 TABEND
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",    ; Path switch-on via C1
     <SynPosX>, Z,
     <SynPosX_Z>, 1, 0)
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,           ; Path switch-on via C2
     "COARSE",
      <SynPosY>, Z,
      <SynPosY_Z>, 2, 0)
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",   ; Path switch-on via C3
     <SynPosA>, Z,
     <SynPosA_Z>, 3, 0)
; 1st Gear stage, C99 is the software axis between the two electronic gears
N800 EGDEF(C99, Y, 1, Z,
     1, B, 1 )
N810 EGONSYN(C99, "NOC",  ; Switch-on of leading axis B
     <SynPosC99>, B,
     <SynPosC99_B>,
18.2, &
     Y, <Syn-             ; Switch-on of leading axis Y
PosC99_Y>,R1 * π, 1, &
```

```
     Z, <Syn-                  ; Switch-on of leading axis Z
PosC99_Z>,10, 1)
                               ; "&" character means: command continued in next line, no LF nor comment per-
                               missible in program
; 2nd Gear stage
N900 EGDEF(C, C99, 1,          ; declaration of following C99 of step 1 as leading axis of step 2,
Z, 1)                          ; Setpoint value coupling
N910                           ; Declaration of path via C4, setpoint coupling
N900                           ; Switch-on of software axis C99
N910                           ; and of leading axis Z via C4
N920 EGONSYN(C, "NOC",
     <SynPosC>, C99,
     <SynPosC_C99>, 1,
1, &
     Z, <SynPosC_Z>, 4,
0)
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	;Max. Number of curve segments
$MN_MM_NUM_CURVE_POLYNOMS = 100	;Max. Number of curve polynomials

Setting data

If the scaling described in Section "Electronic gear (EG) (Page 464)" 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.

Further information
List 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                       ; name of leading axis
$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                       ; name of leading axis
$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                       ; name of leading axis
$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
$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                     ; name of leading axis Y
$AA_EG_AX[1, C99] = Z                     ; name of leading axis Z
$AA_EG_AX[2, C99] = B                     ; name of leading axis B
$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                      ; name of leading axis Z
$AA_EG_AX[1, C] = C99                    ; name of leading axis C99
$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_ASSIGN_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_ASSIGN_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_ASSIGN_TO_MACHAX[AX10] = 0
$MA_IS_ROT_AX[AX10] = TRUE
$MA_ROT_IS_MODULO[AX10] = TRUE
```

14.5 Generic coupling

14.5.1 Brief description

14.5.1.1 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.
  Further information
  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 542)").

14.5.1.2 Requirements

CP version

The generic coupling is available in two versions:

- CP-BASIC
- CP-COMFORT

The number of required couplings (following axes/spindles) and their properties are decisive in the selection of versions.

Example:

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 variant 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 variant would be required.

The versions are independent of one another. They can be combined and can be activated simultaneously.

Table 14-2 Quantity structure, which is dependent on the execution of coupling modules that can be simultaneously activated

Type	CP versions allow one or more different CPSETTYPE coupling objects simultaneously:	CP-BASIC	CP-COMFORT
A	Coupled motion	4	4
B	Synchronous spindle with |1|:1 coupling	-	-
C	o./u. Synchronous spindles, multi-edge turning o./u. Master value coupling / curve tables interpolation o./u. MCS coupling	1	4
D	o./u. Electronic gearbox "plain" o./u. Free generic coupling "basic"	-	1
E	o./u. Electronic gearbox o./u. Free generic coupling	-	-

o./u. stands for over/under

Table 14-3 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					
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
-		-	+	+	Coupling between a spindle and an axis
+		+	+	+	Coupling between a rotary axis and a linear axis
-		+	+	+	From part program and synchronous actions
+		+	+	+	Superimposition / speed difference permitted
-		-	-	+	Cascading permitted
BCS		BCS / MCS	BCS / MCS	BCS / MCS	Coordinate reference (default: CPFRS="BCS")
Synchronous spindle with |1|:1 coupling					
-	1	-	-	-	Maximum number of synchronous spindles / multi-edge turning with the following properties: → refer to CPSETTYPE="COUP"[1)]

Type A	Type B	Type C	Type D	Type E	
	1				Maximum number of master values
	-				From part program and synchronous actions
	-				Superimposition / speed difference permitted
	-				Cascading permitted
	MCS				Coordinate reference fix (CPFRS="MCS")
COUP - synchronous spindle/multi-edge turning					
-	-	13	9	5	Maximum number of synchronous spindles / multi-edge turning with the following properties: → refer to CPSETTYPE="COUP"[1)]
		1	1	1	Maximum number of master values
		-	-	-	From part program and synchronous actions
		+	+	+	Superimposition / speed difference permitted
		-	-	-	Cascading permitted
		MCS	MCS	MCS	Coordinate reference fix (CPFRS="MCS")
LEAD - master value coupling / curve tables interpolation					
-	-	13	9	5	Maximum number of master value couplings / curve tables interpolation with the following properties: → refer to CPSETTYPE="LEAD"[1)]
		1	1	1	Maximum number of master values
		+	+	+	From part program and synchronous actions
		+	+	+	Superimposition / speed difference permitted
		-	-	+	Cascading permitted
		BCS / MCS	BCS / MCS	BCS / MCS	Coordinate 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
			-	-	From part program and synchronous actions
			+	+	Superimposition / speed difference permitted
			-	+	Cascading permitted
			BCS / MCS	BCS / MCS	Coordinate reference (default: CPFRS="BCS")
			-	+	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

Type A	Type B	Type C	Type D	Type E	
			BCS / MCS	BCS / MCS	Coordinate reference (default: `CPFRS="BCS"`)
			-	+	Non-linear coupling law (`CPLCTID`) permitted
[1] Refer to " Coupling types (CPSETTYPE) ".					

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.

14.5.2 Fundamentals

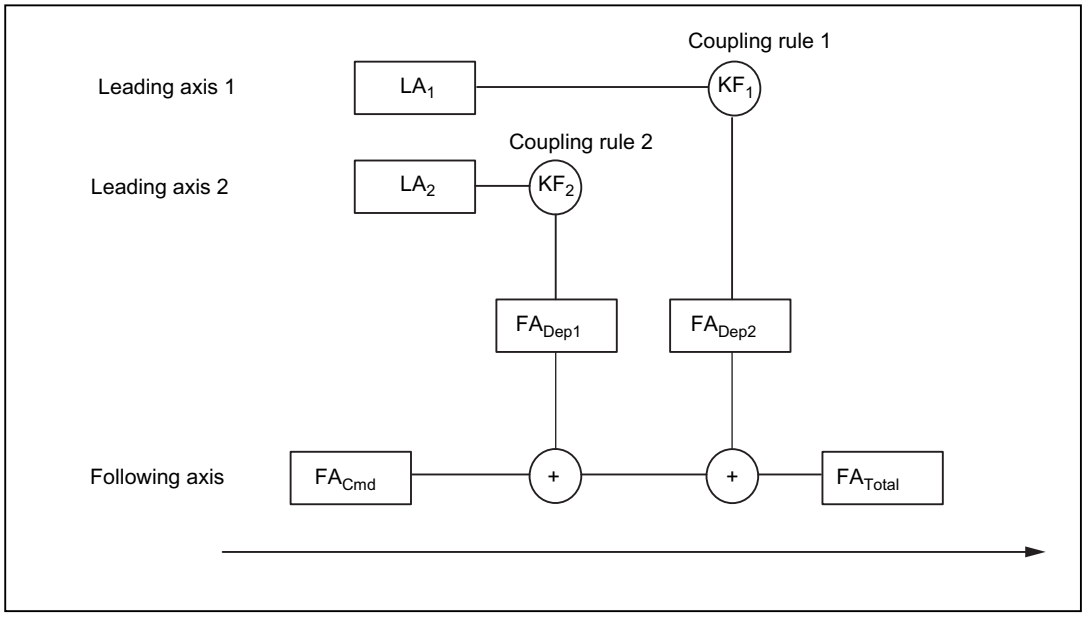
14.5.2.1 Coupling module

With the help of a coupling module, the motion of one axis (→ following axis) can be interpolated depending on other axes (→ 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}	Total setpoint value of the following axis
FA_{Cmd}	Setpoint value set in part program = independent motion component of the following axis
FA_{DEP1}	Dependent motion component of leading axis 1
FA_{DEP2}	Dependent motion component of leading axis 2
LA_1	Setpoint or actual value of the 1st leading axis
LA_2	Setpoint or actual value of the 2nd leading axis
$SynPosLA_1$	Synchronized position of the 1st leading axis
$SynPosLA_2$	Synchronized position of the 2nd leading axis
KF_1	Coupling factor of the 1st leading axis
KF_2	Coupling factor of the 2nd leading axis

The following axis position results from the overlay (summation) of the dependent motion components (FA_{DEP1} and FA_{DEP2}), which result from the individual coupling relationships to the leading axes, and of the independent motion component (FA_{Cmd}) of the following axis:

$FA_{Total} = FA_{Cmd} + FA_{DEP1} + FA_{DEP2}$

The motion components of the following axis are calculated as follows:

Dependent motion component of leading axis 1:	$FA_{DEP1} = (LA_1 - SynPosLA_1) * KF_1$
Dependent motion component of leading axis 2:	$FA_{DEP2} = (LA_2 - SynPosLA_2) * KF_2$
Independent motion component of the following axis:	FA_{Cmd}

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.

14.5.2.2 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:

```
CPON=(X1) CPLA[X1]=(X2) CPLNUM[X1,X2]=2
```

`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 **C**ou**p**ling). 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)]
CP**F***	Describes the characteristics of the following axis (**F**ollowing axis).	CPFPOS[1)]
CP**L***	Describes a property referring to the leading axis (**L**eading axis) or the coupling rule.	CPLON[1)], CPLNUM[1)]
CP**M***	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 (CPSET-TYPE="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
CPFMSON	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
CPMPRT	Coupling response when the part program starts under block 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
CPLOUTSC	Scaling factor for the output value of a coupling	1.0

Keyword	Coupling characteristics / meaning	Default setting (CPSETTYPE="CP")
CPSYNCOP	Threshold value of position synchronism "Coarse"	MD37200
CPSYNFIP	Threshold value of position synchronism "Fine"	MD37210
CPSYNCOP2	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)
CPMALARM	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 543)").

14.5.2.3 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
$**P**A_CP	Reading of channel referenced axis specific coupling characteristics in block preparation (**P**reparation)	Use in synchronous actions is not possible. Does not generate an implicit preprocessing stop.
$**A**A_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 $PA_CP.. CP system variable only differs from the values of the corresponding $AA_CP.. 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 575)").

The exhaustive description of the system variables can be found in:
Further information
List Manual System Variables

14.5.3 Creating/deleting coupling modules

14.5.3.1 Creating a coupling module (CPDEF)

An axial coupling module is created through the definition of the following axis.

Programming

Syntax: `CPDEF=` (<following axis/spindle>)

Designation: **C**oup**l**ing **Def**inition

Functionality: Definition of a coupling module The coupling is not activated.

Following axis/ spindle: Type: AXIS

Range of values: All defined axis and spindle names in the channel

Example:

Programming	Comment
CPDEF=(X2)	; A coupling module is created with axis X2 as following axis.

Boundary conditions

- The maximum number of coupling modules is limited (see Section "Requirements").
- The application of CPDEF to an already created coupling module is possible and will not result in an alarm being generated.

14.5.3.2 Delete coupling module (CPDEL)

A coupling module created with CPDEF can be deleted with CPDEL.

Programming

Syntax: CPDEL= (<following axis/spindle>)

Designation: 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 names in the channel

Example:

Programming	Comment
CPDEL=(X2)	; Deletion of the coupling module with following axis X2.

Boundary conditions

- The switch command CPDEL results in a preprocessing stop with active coupling. Exception: CPSETTYPE="COUP" does not result in a preprocessing stop.
- 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.

14.5.3.3 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: CPLDEF[FAx]= (<leading axis/spindle>)

Designation: **C**oup**l**ing **L**ead **A**xis **Def**inition

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 names in the channel

Example:

Programming	Comment
CPLDEF[X2]=(X1)	; Definition of leading axis X1 for following axis X2.

Programming with CPLA and CPDEF

Syntax: CPLA[FAx]= (<leading axis/spindle>)

Designation: **C**oup**l**ing **L**ead **A**xis

Functionality: Definition of leading axis/spindle for following axis/spindle FAx.

Leading axis/spindle: Type: AXIS

Range of values: All defined axis and spindle names in the channel

Example:

Programming	Comment
CPDEF=(X2) CPLA[X2]=(X1)	; Definition of leading axis X1 for following axis X2.

Boundary conditions

- 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.)
- The maximum number of leading axis modules per coupling module is limited (see Section "Requirements").
- 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.

14.5.3.4 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: CPLDEL[FAx]= (<leading axis/spindle>)

Designation: **C**oup**l**ing **L**ead **A**xis **Del**ete

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 names 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>)

Designation: 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 names in the channel

Example:

Programming	Comment
CPDEL=(X2) CPLA[X2]=(X1)	; Deletion of leading axis X1 of the coupling to following axis X2.

Boundary conditions

- 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.

14.5.4 Switching coupling on/off

14.5.4.1 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 508)").

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 names in the channel

Example:

Programming	Comment
CPON=(X2)	; Activation of coupling of the following axis X2.

Boundary 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.

14.5.4.2 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: CPOF= (<Following axis / spindle>)

Designation: 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 names in the channel

Example:

Programming	Comment
CPOF=(X2)	; Deactivation of coupling of following axis X2.

Boundary conditions

- 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.

14.5.4.3 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>)

Designation: **C**oup**l**ing **L**ead 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 names in the channel

Example:

Programming	Comment
CPLON[X2]=(X1)	; The coupling of leading axis X1 to following axis X2 is activated.

Boundary conditions

CPON can be programmed in synchronous actions.

14.5.4.4 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[FAx]= (<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.

14.5.4.5 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.

14.5.5 Programming coupling characteristics

14.5.5.1 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: CPLNUM[FAx,LAx]= <value>

Designation: **C**oup**l**ing **L**ead **Num**erator

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: CPLDEN[FAx,LAx]= <value>

Designation: **C**oup**l**ing **L**ead **Den**ominator

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: CPLCTID[FAx,LAx]= <value>

Designation: **C**oup**l**ing **L**ead **C**urve **T**able **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} 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").

14.5.5.2 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: CPLSETVAL[FAx,LAx]= <value>

Identifiers: **C**oup**l**ing **L**ead **Set Val**ue

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"	**Com**manded **Pos**ition	Setpoint value coupling
"CMDVEL"	**Com**manded **Vel**ocity	Speed coupling
"ACTPOS"	**Act**ual 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.

14.5.5.3 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: **C**ou**p**ling **F**ollowing **R**elation **S**ystem

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”	**B**asis **C**o-ordinate **S**ystem	Basic Coordinate System

”MCS”	**M**achine **C**oordinate **S**ystem	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.

14.5.5.4 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: `CPBC[FAx]=` "<block change criterion>"

Identifiers: **C**oup**B**ling **B**lock **C**hange Criterium

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: `WAITC(FAx1,BC)`

Identifiers: **Wait** for **C**oupling 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
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.

14.5.5.5 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 (CPFMSON), taking into account the dynamic response limits.

Programming

Syntax: CPON=FAx CPFPOS[FAx]= <value>

Designation: **C**oup**l**ing **F**ollowing **Pos**ition

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 519)").
- 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).

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.

14.5.5.6 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) CPFPOS[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.

14.5.5.7 Synchronization mode (CPFMSON)

Synchronization mode determines synchronization behavior during switch-on of the coupling.

Programming

Syntax: CPFMSON[FAx]= "<synchronization mode>"

Identifiers: **C**oup**l**ing **F**ollowing **M**ode **S**trategy **O**n

Functionality: Determines the synchronization mode during coupling.

Synchronization mode: Type: STRING

Range of values:

"CFAST"	**C**losed Coupling **F**ast	The coupling is closed time-optimized.

"CCOARSE"	**C**losed If Gab **C**oarse	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"	**N**ext **T**ooth **G**ap **T**ime Optimized	The next tooth gap is approached time-optimized.
"NTGP"	**N**ext **T**ooth **G**ap **P**ath Optimized	The next tooth gap is approached path-optimized.
"NRGT"	**N**ext **R**atio **G**ap **T**ime 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"	**N**ext **R**atio **G**ap **P**ath 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"	**A**bsolute **C**oordinate **N**egative	For rotary axes only! The rotary axis traverses towards the synchronized position in the **negative** axis direction. Synchronization is effected immediately.
"ACP"	**A**bsolute **C**oordinate **P**ositive	For rotary axes only! The rotary axis traverses to the synchronized position in the **positive** axis direction. Synchronization is effected immediately.
"DCT"	**D**irect **C**oordinate **T**ime Optimized	For rotary axes only! The rotary axis traverses to the programmed synchronized position in **time-optimized** fashion. Synchronization is effected immediately.
"DCP"	**D**irect **C**oordinate **P**ath 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
CPFMSON[X2]="CFAST"	; CFAST is taken as synchronization mode of the coupling to following axis X2.

14.5.5.8 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: **C**oup**l**ing **F**ollowing **M**ode **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"	**Cont**inue	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"	**Add**itional	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.

14.5.5.9 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: CPFMOF[FAx]= "<switch-off behavior>"

Identifiers: **C**oup**l**ing **F**ollowing **M**ode **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"	**Cont**inue	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.

14.5.5.10 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: `CPOF=(FAx) CPFPOS[FAx]=` <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
`CPOF=(X2) 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 514)").
- 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`.

14.5.5.11 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: **C**oup**l**ing **M**ode **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 543) ".

14.5.5.12 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 543) ".

14.5.5.13 Status during part program start in search run via program test (CPMPRT)

When starting the part program under block 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>

Designation: **C**oup**l**ing **M**ode **Pr**ogram **T**est

Functionality: Defines the behavior of a coupling when the part program starts under block search run via program test.

Value: Type: STRING

Value range:

"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.
"OFF"	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"`	`; When the part program starts under block search run via program test, the coupling to following axis X2 is switched on.`

Supplementary conditions

- The coupling characteristics set with CPMPRT is retained until the coupling module is deleted with (CPDEL).
- If CPMPRT="NONE" is set, then the response when the part program starts under block 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 when the part program starts during block 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 initial control settings for NC-START)
  → See Section "Defaults" in " Coupling types (CPSETTYPE) (Page 543) ".

14.5.5.14 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:

$$FA_{Total} = FA_{cmd} + \{ transOut_1 + scaleOut_1 * [(LA_1 - SynPosLA_1) * scaleIn_1 + transIn_1] * KF_1 \} + \{ transOut_2 + scaleOut_2 * [(LA2 - SynPosLA_2) * scaleIn_2 + transIn_2] * KF_2 \}$$

FA_{Total}	Total setpoint value of the following axis
FA_{Cmd}	Setpoint set in the part program
$LA_{1 / 2}$	Setpoint or actual value of the 1st or 2nd leading axis/value
$SynPosLA_{1 / 2}$	Synchronized position of the 1st or 2nd leading axis/value
$scaleIn_{1 / 2}$	Scaling factor of the 1st or 2nd lead value
$transIn_{1 / 2}$	Offset of the 1st or 2nd lead value
$KF_{1 / 2}$	Coupling factor of the 1st or 2nd leading axis/value
$scaleOut_{1 / 2}$	Scaling factor of the 1st or 2nd output value
$transOut_{1 / 2}$	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: CPLINTR[FAx,LAx]= <value>

Designation: **C**oup**l**ing **L**ead **In** **Tr**anslation 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: CPLINSC[FAx,LAx]= <value>

Designation: **C**oup**l**ing **L**ead **In Sc**ale 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: CPLOUTTR[FAx,LAx]= <value>

Designation: **C**oup**l**ing **L**ead **Out Tr**anslation 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: CPLOUTSC[FAx,LAx]= <value>

Designation: **C**oup**l**ing **L**ead **Out Sc**ale 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.

Note

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.

14.5.5.15 Synchronism monitoring stage 1 (CPSYNCOP, CPSYNFIP, CPSYNCOV, CPSYNFIV)

Synchronism monitoring stage 1

In each interpolator clock 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 510)"):
  - "Coarse" position synchronous operation threshold value
  - "Fine" position synchronous operation threshold value
- For velocity coupling (see "Coupling relationship (CPLSETVAL) (Page 510)"):
  - 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	
$AA_SYNC [FAx]	State of the coupling	
	Value	State
	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 615)").

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: `CPSYNCOP[FAx]=` <value>

Designation: **C**oup**l**ing **Syn**chronous Difference **Co**arse **P**osition

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: `CPSYNFIP[FAx]=` <value>

Designation: **C**oup**l**ing **Syn**chronous Difference **Fi**ne **P**osition

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: `CPSYNCOV[FAx]=` <value>

Designation: **C**oup**l**ing **Syn**chronous Difference **Co**arse **V**elocity

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[FAx]=` <value>

Designation: **C**oup**l**ing **Syn**chronous Difference **Fi**ne **V**elocity

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 [FAx]

Example

Program code	Comment
`CPDEF=(S2) CPLA[S2]=(S1)`	`; Definition of a spindle coupling: Leading spindle S1 to following spindle S2`
`CPON=(S2) CPSYNCOP[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.

14.5.5.16 Synchronous operation monitoring stage 2 (CPSYNCOP2, CPSYNFIP2)

Synchronism monitoring stage 2

For active CP position coupling (setpoint or actual value coupling, see "Coupling relationship (CPLSETVAL) (Page 510)"), after reaching the "COARSE"/"FINE" block change criterion (see "Block change behavior (CPBC) (Page 512)"), the second stage of the synchronous operation monitoring can be used to monitor that one of the threshold values of the first stage is maintained on the actual value side, independent of the synchronism 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.

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: **C**ou**p**ling **Syn**chronous Difference **Co**arse **P**osition **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: **C**ou**p**ling **Syn**chronous Difference **Fi**ne **P**osition **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:
  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 (CPSYNCOP, CPSYNFIP, CPSYNCOV, CPSYNFIV) (Page 527)") is reached:
  DB31, ... DBX98.1 (coarse synchronous operation) = 1 / DB31, ... DBX98.0 (fine synchronous operation) = 1

Monitor

As long as the synchronism 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:

DB31, ... DBX103.4 (synchronous operation 2 fine)

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, CPLOF, CPLDEL)
- New synchronization request by:
  - CPON / CPLON / CPDO / CPLOF
  - 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.

Boundary 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 558)").

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 542) ").

Example

```
Program code                                    Comment
G0 Z-300 X50
CPDEF=(S2) CPLA=(S1) CPLNUM=2 CPLDEN=1          ; Gear ratio in accordance with tool.
CPBC="FINE"
M3 S2000
CPON=(S2) CPLA=(S1) CPSYNCOP2[X]=1.6            ; Block change is performed on reaching "fine synchronous
CPSYNFIP2[X]=0.8                                operation".
                                                ; Synchronous operation monitoring stage 2 is also activa-
                                                ted
                                                ; (tolerances: coarse 1.6, fine 0.8).
X20
G1 Z-200                                        ; Machining block, material removal.
                                                ; Synchronous operation determines the quality.
CPOF=(S2)                                       ; Deselect coupling, deactivate the monitoring function.
```

14.5.5.17 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[FAx]= <value>

Designation: **C**oup**l**ing **M**ode **Brake**

Functionality: CPMBRAKE is a bit-coded CP keyword that defines the braking behavior of the following axis FAx for the following events:

Bit	Event
0	• NST DB31, ... DBX4.3 (feed stop / spindle stop) present or • CP SW limit stop is set (see "CP SW limit monitoring (Page 552)")

Note:
Bit 0 is significant only for a "Freely programmable" coupling type (CPSETTYPE=CP).

Value	Meaning
0	The brake is not transferred to the leading axes.
1	The brake is transferred to the leading axes depending on the context.

1 - 31 Reserved

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 for pre-processing couplings with CPSETTYPE=CP (otherwise bit 0 = 0)

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" or "CP SW limit 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" or "CP SW limit stop" should brake the coupling group.`
`...`	

14.5.5.18 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: **C**oup**l**ing **M**ode **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 FA	A/S disable LA_1	A/S disable LA_2	CPMVDI Bit 3/5	CPMVDI Bit 4/6	FA_{Total}	Meaning for FA
0	0	0	0	0	FA_{Cmd} + FA_{DEP1} + FA_{DEP2}	Real movement.
0	0	0	1	0	FA_{Cmd} + FA_{DEP1} + FA_{DEP2}	Real movement.
0	1	1	0	0	FA_{Cmd} + FA_{DEP1} + FA_{DEP2}	Simulated movement

A/S disable FA	A/S disable LA_1	A/S disable LA_2	CPMVDI Bit 3/5	CPMVDI Bit 4/6	FA_{Total}	Meaning for FA
0	0	1	0	0	-	Alarm 16773, different leading axis states with ref. to the A/S disable
1	0	0	0	0	FA_{Cmd} + FA_{DEP1} + FA_{DEP2}	Real movement, FA spindle disable has no effect.
1	0	1	1	0	FA_{Cmd} + FA_{DEP1} + FA_{DEP2}	Simulated movement, FA spindle disable has an effect.
1	0	1	1	1	FA_{Cmd} + FA_{DEP2}	Simulated movement, as bit 4 is set, FA_{DEP1} is suppressed.
0	1	1	0	1		Alarm 16694 bit 4/6 = 1 is only supported if bit 3/5 = 1.
1	0	0	0	1		Alarm 16694 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

LA_1: Leading axis 1

LA_2: Leading axis 2

FA_{Total}: Total setpoint value of the following axis

FA_{Cmd}: Independent motion component of the following axis

FA_{DEP1}: Dependent motion component of leading axis 1

FA_{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.

14.5.5.19 Alarm suppression (CPMALARM)

The CP keyword `CPMALARM` can be used to suppress coupling-related alarms.

Programming

Syntax: `CPMALARM[FAx]=` <value>

Designation: **C**oup**l**ing **M**ode **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		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 cou-
                                  pling of the X2 following axis.
```

14.5.6 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[X2]=(X1) CPDEF=(A2) CPLA[A2]=(X2) CPLA[A2]=(A1)`

Supplementary conditions

- The availability of cascading is option-based (see Section "Requirements").
- 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).

14.5.7 Compatibility

14.5.7.1 Adaptive cycles

Adaptive cycles

The provision of adaptive cycles as fixed component of the NC 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

Coupling commands	Adaptive cycle
LEADOF	cycle703
COUPDEF	cycle704
COUPON	cycle705
COUPONC	cycle706
COUPOF	cycle707
COUPOFS	cycle708
COUPDEL	cycle709
COUPRES	cycle710
EGDEF	cycle711
EGON	cycle712
EGONSYN	cycle713
EGONSYNE	cycle714
EGOFC	cycle715
EGOFS	cycle716
EGDEL	cycle717

Storage 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 NC software, a log file is saved in the "CST" directory (Changelog), indicating necessary changes of the adaptive cycles.

14.5.7.2 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[FAx]=` <value>

Designation: **C**oup**l**ing **S**et **T**ype

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:

Programming	Comment
`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		-	-		
CPDEL		-	-		
CPLDEF		-	-		
CPLDEL		-	-		
CPON					
CPOF					

Keyword	Coupling type				
	Default (CP)	Coupled motion (TRAIL)	Master value coupling (LEAD)	Electronic gear (EG)	Synchronous spindle (COUP)
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
CPFMSON	CFAST	CFAST	CCOARSE	NRGT	CFAST
CPFMON	STOP	VL[1]: STOP HL[2]: CONT	VL[1]: CONT HL[2]: CONT	STOP	CONT
CPFMOF	STOP	VL[1]: STOP HL[2]: STOP	VL[1]: STOP HL[2]: STOP	STOP	CONT
CPLPOS + CPON	Not set	-	-	Not set	-
CPMRESET	NONE	MD20110	MD20110	MD20110	MD20110
CPMSTART	NONE	MD20112	MD20112	MD20112	MD20112
CPMPRT	NONE	MD20112 / MD22620[3]	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
CPLOUTTR	0.0	0.0	0.0	0.0	0.0
CPLOUTSC	1.0	1.0	1.0	1.0	1.0
CPSYNCOP	MD37200	MD37200	MD37200	MD37200	MD37200
CPSYNFIP	MD37210	MD37210	MD37210	MD37210	MD37210
CPSYNCOP2	MD37202	MD37202	MD37202	MD37202	MD37202
CPSYNFIP2	MD37212	MD37212	MD37212	MD37212	MD37212
CPSYNCOV	MD37220	MD37220	MD37220	MD37220	MD37220
CPSYNFIV	MD37230	MD37230	MD37230	MD37230	MD37230
CPMBRAKE					
Bit 0	1	-	-	-	-

Keyword		Coupling type				
		Default (CP)	Coupled motion (TRAIL)	Master value coupling (LEAD)	Electronic gear (EG)	Synchronous spindle (COUP)
CPMVDI						
	Bit 3	0	0	0	0	0
	Bit 4	1	1	1	1	1
	Bit 5	0	0	0	0	0
	Bit 6	1	1	1	1	1
CPMALARM		MD11410 MD11415	MD11410 MD11415	MD11410 MD11415	MD11410 MD11415	MD11410 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 (`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/CPLOF	CPLON/CPLOF	CPLON/CPLOF	CPLDEF/CPLDEL	CPLDEF/CPLDEL
Cascading	+	+	+	+	-
Dynamic observation of the leading spindle	-	-	-	-	+

	Default (CP)	Coupled motion (TRAIL)	Master value coupling (LEAD)	Electronic gear (EG)	Synchronous spindle (COUP)
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").

Example:

The coupled motion coupling type (`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

- `CPSETTYPE` can be programmed in synchronous actions.
- If the coupling type (`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	Alarm 16686	
CPLNUM	Alarm 16686	Alarm 16686		
CPLDEN	Alarm 16686	Alarm 16686		
CPLCTID	Alarm 16686			Alarm 16686
CPLSETVAL	Alarm 16686 with CMDVEL	Alarm 16686 with CMDVEL	Alarm 16686 with CMDVEL	

CPSETTYPE=	TRAIL	LEAD	EG	COUP
CPFRS				Alarm 16686 with BCS
CPBC	Alarm 16686	Alarm 16686		
CPFPOS + CPON	Alarm 16686	Alarm 16686		
CPFPOS + CPOF	Alarm 16686	Alarm 16686	Alarm 16686	
CPFMSON	Alarm 16686	Alarm 16686		Alarm 16686
CPFMON	Alarm 16686	Alarm 16686 with ADD/STOP	Alarm 16686	Alarm 16686 with STOP
CPFMOF	Alarm 16686	Alarm 16686 with ADD/STOP	Alarm 16686 with ADD	Alarm 16686 with ADD
CPLPOS + CPON	Alarm 16686	Alarm 16686		Alarm 16686
CPMRESET	Alarm 16686	Alarm 16686	Alarm 16686	Alarm 16686
CPMSTART	Alarm 16686	Alarm 16686	Alarm 16686	Alarm 16686
CPMPRT	Alarm 16686	Alarm 16686	Alarm 16686	Alarm 16686
CPMBRAKE	Alarm 16686	Alarm 16686	Alarm 16686	Alarm 16686
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				Alarm 14092 with axis

14.5.7.3 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.

Further information

S3: Synchronous spindle (Page 847)

Programming

Syntax: `CPRES=` (<following spindle>)

Designation: **C**oup**l**ing **Res**tore

Functionality: Activates projected data of the synchronous spindle coupling to following spindle FAx.

Following spindle: Type: AXIS

Value range: All defined spindle names in the channel

Example:

Programming	Comment
CPLON[S2]=(S1) CPSETTYPE[S2]="COUP"	; 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.

Boundary conditions

- 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.

14.5.8 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.

Further information
Function Manual Basic Functions; Mode groups, channels, axis replacement

Leading axes

Axis change of leading axes can be performed independently of the state of the coupling.

14.5.9 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 = Modulo reduced rotary axis, X = Linear axis

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 through additional 50mm to 250.

Programming	Comment
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.

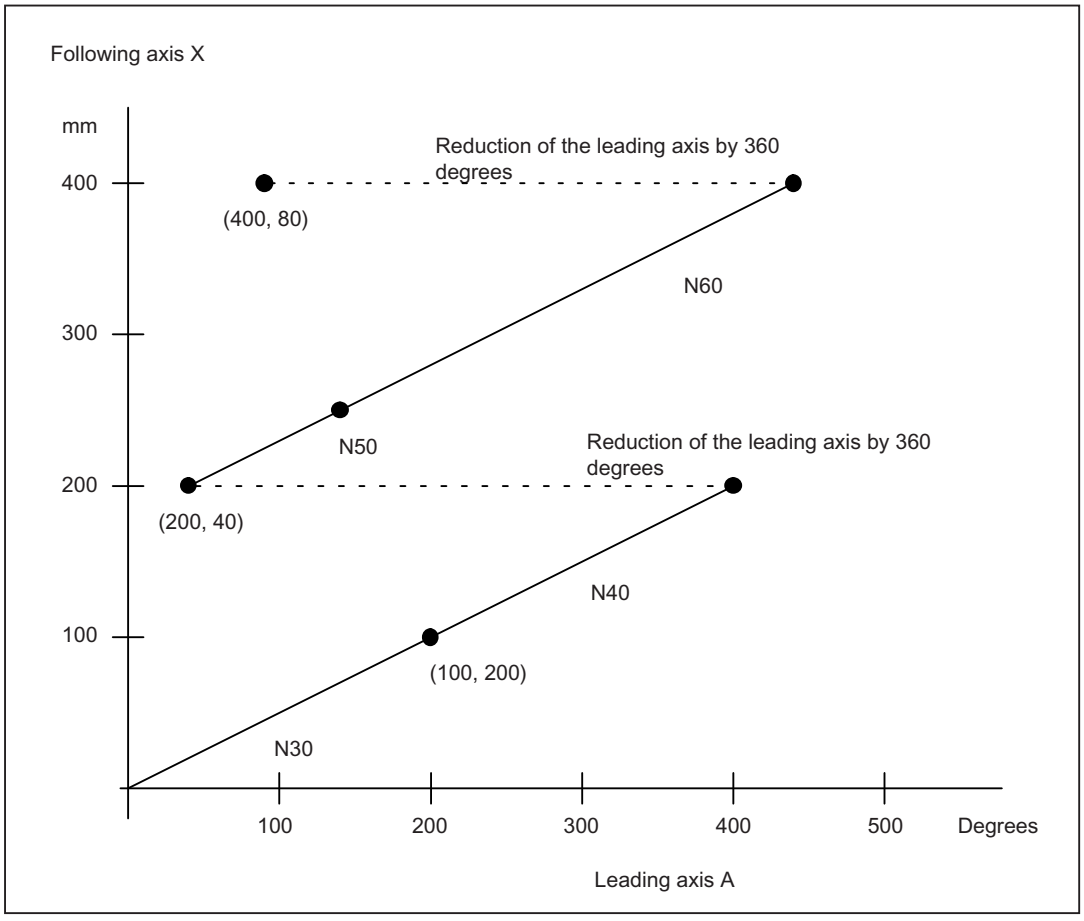


Figure 14-12 Example: Modulo reduced rotary axis to linear axis

(...) Position indication for X, A

14.5.10 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 `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.

14.5.11 CP SW limit monitoring

14.5.11.1 Function

What is the purpose of the function?

The "CP-SW limit monitoring" function improves the braking response of a following axis or a following spindle in axis operation when approaching a software limit switch.

Advantages:

- This avoids that the software limit switch is passed.
- If possible, synchronous operation of the coupling is kept.

Availability

The "CP SW limit monitoring" function can only be activated for following axes from:

- Generic couplings, type "Freely programmable (CPSETTYPE[FAx] = "CP")
- Couplings (generic coupling with CPSETTYPE[FAx] not equal to "CP", coupled motion, electronic gearbox, master value coupling or synchronous spindle) with a maximum of one active leading axis/spindle

In all other cases (and when "CP SW limit monitoring" is not activated) the previous function of the software limit switch monitoring is active (see Chapter "Limit switch monitoring" in the Function Manual, Basic Functions, A3: Axis Monitoring, Protection Areas).

Monitoring and setting the brake

The "CP-SW limit monitoring" function checks, in every IPO cycle, as to whether the movement of the following axis/spindle can be enabled for the following IPO cycle, so that the axis can always stop in plenty of time before the software limit switch. If this is not the case, for the following axis, a "CP-SW limit stop" is set; this signifies unconditional direction-specific braking along the contour.

Transferring the brake to the axes

Setting the brake means that the coupling-independent motion component (CMD and CORR component) of the following axis is stopped. On the other hand, the coupling-dependent motion component (DEP component) of the following axis can only be braked by braking the leading axes.

For couplings (generic coupling with CPSETTYPE[FAx] not equal to "CP", coupled motion, electronic gearbox, master value coupling or synchronous spindle) with a maximum of one active leading axis/spindle, the brake is also set for the (only) active leading axis in the collision direction (depending on the coupling factor).

For generic couplings, type "freely programmable (CPSETTYPE[FAx] = "CP"), the brake is only transferred to all of the active leading axes, if the coupling property CPMBRAKE[FAx] Bit 0 = 1 has been programmed (see "Reaction to stop signals and commands (CPMBRAKE) (Page 535)").

The collision direction depends on the coupling factor for the specific leading axis: With a negative coupling factor, the collision direction reverses, there is no transfer for a coupling factor of "zero". For non-linear couplings (e.g. curve table), the coupling factor is derived from the gradient, determined as linear approximation.

Note

A brake set for a following axis can only influence its leading axes as long as the coupling is active.

Braking behavior

The normal acceleration ramp is used when braking the axes towards the software limit switch along the contour, maintaining the coupling definition.

When braking, the stop state 75 "brake request" and situation-dependent, the higher priority stop states 22 "wait: spindle enable missing", 12 "wait for axis/spindle release" or 71 "wait for enable, transformation axis" are displayed.

If the following axis is stationary as result of the "CP-SW limit stop", and cannot approach the software limit switch any closer, then the following alarm is displayed:

10625 "%?C{channel %1: %}block %3 following axis/spindle %2 with CP-SW limit stop %4"

This means that up to the "final" stopping position, the situation can be somewhat relieved by moving the following axis in the opposite direction.

Note

For the "CP-SW limit monitoring", coupling involves maintaining synchronous operation. As a consequence, it cannot be guaranteed that the axis stops at precisely the correct position, if the coupling rule is to be maintained. If the brake is to be transferred to the leading axes, and the interpolators brake as required, then the leading axis comes to a standstill in front of the software limit switch. The maximum distance to the software limit switch is then obtained from the actual maximum acceleration rate of the following axes multiplied by the square of the interpolation cycle time.

Retraction

The user is responsible for retraction.

For diagnostic purposes, there are:

- the stop state 75 "brake request"
  (if an axis has a travel command, but stops or may not traverse as a result of a braking request)
- and the system variables:
  $AA_BRAKE_STATE (actual brake status)
  $AA_BRAKE_CONDB (context-sensitive conditions for the interpolator stop in the BCS)
  $AA_BRAKE_CONDM (context-sensitive conditions for the interpolator stop in the MCS)
  including the OPI variables aaBrakeState, aaBrakeCondB and aaBrakeCondM derived from these.

Example for retraction:

1. acknowledge alarm 10625.
2. Switch to the JOG mode.
3. Traverse the leading axis using the traversing keys so that the following axis moves away from the software limit switch.

14.5.11.2 Parameterization

Activation

The "CP SW limit monitoring" function is activated on an axis-for-axis basis using machine data:

MD30455 $MA_MISC_FUNCTION_MASK (axis functions)

Bit	Value	Meaning
11	0	CP SW limit monitoring is not active.
	1	CP SW limit monitoring is active.

14.5.11.3 Programming

Transferring the brake to the leading axes

For generic couplings, type "freely programmable" (CPSETTYPE[FAx] = "CP"), by programming the coupling property CPMBRAKE (see "Reaction to stop signals and commands (CPMBRAKE) (Page 535)") it can be set as to whether the brake of the following axis, initiated using the "CP-SW limit monitoring" function, should also be transferred to the leading axes.

14.5.11.4 Boundary conditions

Possible fault sources

If the axes do not brake as required, this could be due to the following reason:

At least one leading axis does not react to the brake, because it is also a leading axis in its own right (cascade), and the brake is not transferred to its leading axes. A brake is only transferred to the leading axes (recursive), if the following applies for the following axes involved: CPSETTYPE[FAx] = "CP" and CPMBRAKE[Fax] bit 0 = 1

Even if all of the axes brake as required, the contour can be violated as a result of different acceleration rates and/or coupling factors if these values have not been appropriately configured and/or programmed.

Even if all of the previous preconditions are fulfilled, there are still some reasons why it cannot be avoided that a following axis passes a software limit switch:

- The actual maximum acceleration of a leading axis changes.
- The actual maximum acceleration of the following axis becomes smaller.
- The velocity of the following axis when switching in the coupling or when the CP-SW limit monitoring responds is already so high that with the actual maximum acceleration it is no longer possible to stop the axis in the appropriate time.

If the axis cannot be stopped in the appropriate time, then as before, the previous software limit switch monitoring responds, and stops the following axis at the software limit switch.

Braking response for transformations

If the axis to be braked is the output of a transformation, and it has a (MCS) travel command in the collision direction, then the brake is transferred to all input axes of the transformation in both directions, and a path stop is executed for this transformation. However, the following exceptions apply: It does not apply to the independent axes of the transformation. For TRANSMIT and TRAANG, these are the axes specified in machine data $MC_TRAFO_AXES_IN_*[2]. For the transformations, braking is only transferred to their dependent input axes.

DRF offset

Setting a CP-SW limit stop can cause a DRF movement to be canceled, as shown in the following example:

The basic motion is path motion with extremely low velocity in the positive direction. A larger DRF correction motion applies to this in the opposite direction. This results in a travel command in the negative direction. If these movements are now stopped using a CP-SW limit stop, then for standard machine data, this means that DRF motion is canceled, and only path motion is continued after the stop is withdrawn.

14.5.11.5 Examples

Example 1: Generic coupling type "freely programmable" (CPSETTYPE[FAx] = "CP")

Configuration:

MD26110 $MA_POS_LIMIT_PLUS[AX2]=15 ;	Position of the 1st software limit switch of the following axis for the traversing range limit in the positive direction
MD30455 $MA_MISC_FUNCTION_MASK[AX2] = 'H800' ;	Bit 11 = 1 (CP-SW limit monitoring is activated)

Programming:

```
; start position of X and Y
N100 G0 X0 Y0
; coupling activation FA=Y LA=X
; set with CPMBRAKE bit 0  (transfer of the brake to the leading axis)
N120 CPON=Y CPSETTYPE="CP" CPLA=X CPLDEN=2 CPMBRAKE B_OR='H1'
N140 G4 F2

; traverse leading axis X
N200 G1 X=1000 F1000
```

Example 2: Couplings with a maximum of one active leading axis/spindle

With the following calls, the CP-SW limit monitoring is executed – including the transfer of the brake to the leading axis – if MD30455 MA_MISC_FUNCTION_MASK[AX2] bit 11 is set:

```
TRAILON(Y,X,0.5) ;    Definition and switching in the coupling of the coupled motion
                      axis Y two leading axis X.
```

```
EGDEF(Y,X,1) ;                Definition of an EG axis group with setpoint coupling
                              from X to Y (following axis).
EGON( Y,"FINE",X,1,2) ;       Activating coupling.
```

14.5.12 Disturbance characteristic

14.5.12.1 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.

14.5.13 Tracking the deviation from synchronism

14.5.13.1 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 NC (see "Measuring the deviation from synchronism (Page 559)").
2. The deviation value is already known and entered by the user directly (see "Entering the deviation from synchronism directly (Page 562)").

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 492)").

14.5.13.2 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).

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. K_V 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].

14.5.13.3 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.

14.5.13.4 Synchronism correction

If correction value $AA_COUP_CORR[S<n>] is a value other than zero and a generic machine coupling has been active for following spindle S<n> (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<n>] 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<n>] 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 `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.

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.

14.5.13.5 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.

14.5.13.6 Resetting synchronism correction

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.

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 559)").

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.

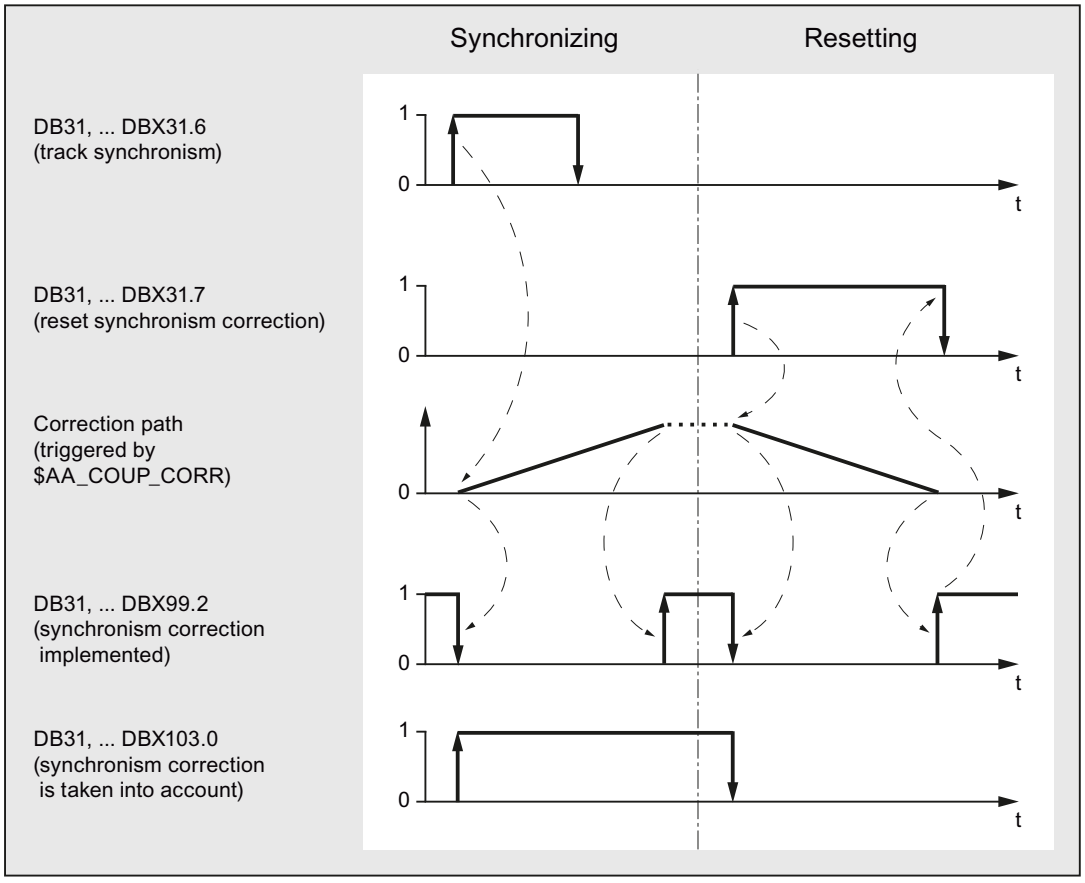


Figure 14-13 Time diagram for synchronizing and resetting synchronism correction

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.

14.5.13.7 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.

14.5.14 Examples

14.5.14.1 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.

```
CPON=(X2) CPLA[X2]=(X1) CPLNUM[X2,X1]=2
...
CPOF=(X2)                          ; 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.

```
CPON=(X2) CPLA=(X1) CPLNUM[X2,X1]=2 CPLA=(Z) CPLNUM[X2,Z]=3
...
CPOF=(X2)                          ; 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.

```
CPON=(X2) CPLA[X2]=(X1) CPLNUM[X2,X1]=2 CPLA[X2]=(Z) CPLNUM[X2,Z]=1.2
...
CPOF=(X2) CPLA[X1]=(Z)             ; The coupling with leading axis Z is deactiva-
                                     ted 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.

```
N10 CPDEF=(X2) CPLA[X2]=(X1) CPLA[X2]=(Z) CPLA[X2]=(A)
N20 CPON=(X2)                          ; All leading axes become active, i.e. all
                                         contribute a position component accord-
                                         ing to the coupling rule (coupling com-
                                         ponent) to axis X2.
N30 CPOF=(X2)                          ; All leading axes are deactivated.
```

```
N40 CPLON[X2]=(X1)                          ; Leading axis X1 is activated, only this
                                              axis supplies a coupling component.
                                              Leading axes Z and A remain deactivated.
N50 CPLON[X2]=(A)                           ; Leading axis X1 remains active, leading
                                              axis A is deactivated, X1 and A contrib-
                                              ute coupling components (→ selective
                                              switch-on is additive, the condition of
                                              the other leading axes is retained).
N60 CPLON[X2]=(Z) CPLOF[X2]=(A)             ; Leading axis Z is activated, leading ax-
                                              is A deactivated. Leading axes X 1and Z
                                              are now active.
N70 CPLOF[X2]=(X1)                          ; 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!

```
CPDEF=(X2) CPLA[X2]=(X1) CPLNUM[X2,X1]=2 CPLA[X2]=(Z) 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)
```

14.5.14.2 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
;$PATH=/_N_CST_DIR/_N_CYCLE700_SPF
;USER V1.1 Mar 22, 2006
;classic TRAILON(FA,LA,Factor)
PROC CYCLE700(AXIS _CPF=NO_AXIS, AXIS _CPL=NO_AXIS,REAL _CPLF=1) IPTRLOCK SBLOF DISPLOF ICYCOF
;*IF _CPLF==0 GOTOF _CPOF
CPLON[_CPF]=(_CPL) CPSETTYPE[_CPF]="TRAIL" CPLNUM[_CPF,_CPL]=(_CPLF) CPFRS[_CPF]="MCS"
RET
_CPOF:
IF ($P_TECCYCLE==TRUE)
  IF ($AA_CPSETTYPE[_CPF]<>"TRAIL") GOTOF _EXIT
ELSE
  IF ($PA_CPSETTYPE[_CPF]<>"TRAIL") GOTOF _EXIT
ENDIF
IF _CPL==NO_AXIS
  CPSETTYPE[_CPF]="TRAIL" CPOF=(_CPF)
ELSE
  CPSETTYPE[_CPF]="TRAIL" CPLOF[_CPF]=(_CPL)
ENDIF
_EXIT: RET

;* SysError 500918 therefore IF command commented
```

Figure 14-14 Cycle 700 after adaption. Changes are indicated by a colored bar.

14.6 Dynamic response of following axis

14.6.1 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)

14.6.2 Programmed dynamic limits

14.6.2.1 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	Part 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	Effect
Dynamic reduction	Prolongs the synchronism difference. The monitoring from leading to following value may exceed the allowed range for extended periods.
Dynamic increase	Shortens the synchronism 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.

⚠ CAUTION

Application faults

This option can only be used in conjunction with servo trace and with the appropriate technical knowledge of the control.

Further information

- CNC Commissioning Manual: NC, PLC, drive
- G2: Velocities, setpoint / actual value systems, closed-loop control (Page 31)

POWER ON

During POWER ON the values of `VELOLIMA` and `ACCLIMA` are initialized to 100%.

Mode change

The dynamic offsets remain valid only on transition from AUTO → JOG mode.

RESET

The validities of the (`VELOLIMA` and `ACCLIMA`) dynamic offsets after RESET depend on the setting in the channel-specific machine data:

MD22410 $MC_F_VALUES_ACTIVE_AFTER_RESET (F Function is active even after RESET)

Value	Meaning
0	The values of VELOLIMA[FA] and ACCLIMA[FA] are set to 100% after RESET.
1	The last programmed values of VELOLIMA[FA] and 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 MD22410 = 0, then the IDS synchronized action should trigger a repeat or continuous writing of the dynamic offset.

Further information
Function Manual Synchronized Actions

14.6.2.2 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
N130 VELOLIMA[AX4]=50                    ; reduced speed
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)
...
```

14.6.2.3 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
Preprocessing			
$PA_ACCLIMA[n]	REAL	Acceleration offset set with ACCLIMA[Ax]	%
$PA_VELOLIMA[n]	REAL	Velocity offset set with VELOLIMA[Ax]	%
Main run			
$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.

14.7 General supplementary conditions

Note

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.

14.8 Data lists

14.8.1 Machine data

14.8.1.1 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

14.8.1.2 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

14.8.1.3 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"

14.8.2 Setting data

14.8.2.1 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

14.8.3 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	Name for nth leading axis

Identifier	Meaning
$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	Synchronism 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_CPFACT	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
$AA_CPFMOF	Behavior of the following axis at switch-off
$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_CPLCTID	Table number of the active curve table
$AA_CPLDEN	Denominator of the coupling factor
$AA_CPLNUM	Numerator of the coupling factor

Identifier	Meaning
$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_CPSYNCOV	Threshold value of velocity synchronism "Coarse" (main run)
$AA_CPSYNFIP	Threshold value of position synchronism "Fine" (main run)
$AA_CPSYNFIV	Threshold value of velocity synchronism "Fine" (main run)
$AA_CPLINSC	Scaling factor of the input value of a leading axis (main run)
$AA_CPLINTR	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_EG_ACTIVE	Coupling for leading axis b is active
$AA_EG_AX	Name 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_NUM_LA	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)
$AA_LEAD_V	Current leading velocity value
$AA_SYNC	Coupling status of following axis
$AA_SYNCDIFF[FA]	Synchronism 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_CPFPOSSTAT	Validity of synchronous and stop position

Identifier	Meaning
$PA_CPSYNCOP	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	Synchronism difference
$VA_EG_SYNCDIFF_S	Synchronism difference with sign
$VA_SYNCDIFF[FA]	Synchronism difference of the actual value
$VA_SYNCDIFF_STAT[FA]	Status of the synchronism difference
$P_COUP_OFFS[S2]	Programmed position offset of the synchronous spindle
$P_EG_BC	Block change criterion

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

P5: Oscillation 15

15.1 Brief description

Definition

When the "Oscillation" function is selected, an oscillation axis oscillates backwards and forwards at the programmed feedrate or a derived feedrate (revolutional feedrate) between two reversal points. Several oscillation axes can be active at the same time.

Oscillation variants

Oscillation functions can be classified according to the axis response at reversal points and with respect to infeed:

- Asynchronous oscillation across block boundaries
  During reciprocating movement, other axes can interpolate at will. The oscillation axis can act as the input axis for dynamic transformation or as the master axis for gantry or coupled-motion axes. Oscillation is not automatically linked to the AUTOMATIC mode.
- Oscillation with continuous infeed.
  Simultaneous infeed in multiple axes is possible. However, there is no interpolative connection between the infeed and oscillation movements.
- Oscillation with infeed in both reversal points or only in the left-hand or right-hand reversal point. The infeed can be initiated at a programmable distance from the reversal point.
- Sparking-out strokes after oscillation.
- Beginning and end of oscillation at defined positions.

Response at reversal points

The change in direction is initiated:

- without the exact stop limit being reached (exact stop fine or coarse)
- After reaching the programmed position or
- after the programmed position is reached and expiry of a dwell.
- by an external signal (from the PLC).

Control methods

Oscillation movements can be controlled by various methods:

- The oscillation movement and/or infeed can be interrupted by delete distance-to-go.
- The reversal points can be altered via NC program, PLC, HMI, handwheel or directional keys.

- The feedrate velocity of the oscillation axis can be altered through a value input in the NC program, PLC, HMI or via an override. The feedrate can be programmed to be dependent on a master spindle, rotary axis or spindle (revolutional feedrate).
  Further information
  V1: Feedrates (Page 125)
- The oscillation movement can be controlled entirely by the PLC.

Methods of oscillation control

There are two modes of oscillation:

1. Asynchronous oscillation:
   Is active across block boundaries and can also be started from the PLC/HMI.
2. Oscillation by synchronized movement actions:
   In this case the asynchronous oscillation and an infeed movement are coupled via synchronized actions. In this way, it is possible to program oscillation with infeed at the reversal points which is active on a non-modal basis.

15.2 Asynchronous oscillation

15.2.1 Asynchronous oscillation

Characteristics

The characteristics of asynchronous oscillation are as follows:

- The oscillation axis oscillates backwards and forwards between reversal points 1 and 2 at the specified feedrate until the oscillation movement is deactivated or until there is an appropriate response to a supplementary condition. If the oscillation axis is not positioned at reversal point 1 when the movement is started, then it traverses to this point first.
- Linear interpolation G01 is active for the oscillating axis regardless of the G command currently valid in the program. Alternately, revolutional feedrate G95 can be activated.
- Asynchronous oscillation is active on an axis-specific basis beyond block limits.
- Several oscillation axes (i.e. maximum number of positioning axes) can be active at the same time.
- During the oscillation movement, axes other than the oscillation axis can be freely interpolated. A continuous infeed can be achieved via a path movement or with a positioning axis. In this case, however, there is no interpolative connection between the oscillation and infeed movements.
- If the PLC does not have control over the axis, then the axis is treated like a normal positioning axis during asynchronous oscillation. In the case of PLC control, the PLC program must ensure via the appropriate stop bits of the NC/PLC interface that the axis reacts in the desired way to NC/PLC interface signals. These NC/PLC interface signals also include end of program, operating mode change and single block.

- The oscillating axis can be an input axis for transformations (e.g. inclined axis, see Function Manual "Transformations", Section "Kinematics transformation").
- The oscillation axis can act as the master axis for gantry and coupled motion axes.
  Further information
  G1: Gantry axes (Page 259)
- It is possible to traverse the axis with jerk limitation (SOFT) and/or with knee-shaped acceleration characteristic (as for positioning axes).
- In addition to this, the oscillation movement can be activated in synchronism with the block via the part program.
- The oscillation movement can likewise be started, influenced and stopped from the PLC/ HMI.
- Interpolatory oscillation is not possible (e.g. oblique oscillation).

15.2.2 Influences on asynchronous oscillation

Setting data

The setting data required for oscillation can be set with special language commands in the NC part program, via the HMI and/or the PLC.

Feedrate

The feed velocity for the oscillation axis is selected or programmed as follows:

- The velocity defined for the axis as a positioning axis is used as the feed velocity. This value can be programmed via FA[axis] and has a modal action. If no velocity is programmed, then the value stored in machine data POS_AX_VELO is used (see also Section "P2: Positioning axes (Page 185)").
- When an oscillation movement is in progress, the feed velocity of the oscillation axis can be altered via setting data. It can be specified via the part program and setting data whether the changed velocity must take effect immediately or whether it should be activated at the next reversal point.
- The feedrate can be influenced by the override (axial NC/PLC interface signal and programmable).
- If Dry Run is active, the dry run velocity setting is applied if it is higher than the currently programmed velocity.
  Activation of the dry run feed entered in SD42100 $SC_DRY_RUN_FEED can be controlled with SD42101 $SC_DRY_RUN_FEED_MODE.
  Further information
  V1: Feedrates (Page 125)
- The velocity overlay / path overlay can be influenced by the handwheel (see following table and Section "H1: Manual traversing (Page 633)").
- The oscillation axis can be moved with reversal feed.

Revolutional feedrate

The reversal feed can also be used for oscillation axes.

Reversal points

The positions of the reversal points can be entered via setting data before an oscillation movement is started or while one is in progress.

- The reversal point positions can be entered by means of manual travel (handwheel, JOG keys) before or in the course of an oscillation movement, regardless of whether the oscillation movement has been interrupted or not.

The following applies to alteration of a reversal point position: When an oscillation movement is already in progress, the altered position of a reversal point does not become effective until this point is approached again. If the axis is already approaching the position, the correction will take effect in the next oscillation stroke.

Note

If a reversal point must be altered at the same time as NC/PLC interface signal DB21, … DBX0.3 ("Activate DRF") is set, the handwheel signals are applied both to the DRF offset and to the offset of the reversal point, i.e. the reversal point is shifted absolutely by an amount corresponding to twice the distance.

Stop times

A stop time can be programmed via setting data for every reversal point.

The setting can be changed in the following blocks of the NC program. It is then effectively block synchronous from the next reversal point.

Stop time can be changed asynchronously via setting data. It is then effective from the instant that the appropriate reversal point is next traversed.

The following table explains the motional behavior in the exact stop range or at the reversal point, depending on the stop time input.

Table 15-1 Effect of stop time

Stop time setting	Response
-2	Interpolation continues without wait for exact stop
-1	Wait for coarse exact stop at reversal point
0	Wait for fine exact stop at reversal point
>0	Wait for fine exact stop at reversal point, followed by wait for stop time.

Deactivate oscillation

One of the following options can be set for termination of the oscillation movement when oscillation mode is deactivated:

- Termination of oscillation movement at the next reversal point
- Termination of oscillation movement at reversal point 1
- Termination of oscillation movement at reversal point 2

Following this termination process, sparking-out strokes are processed and an end position approached if programmed.

On switchover from asynchronous oscillation to spark-out and during spark-out, the response at the reversal point regarding exact stop corresponds to the response determined by the stop time programmed for the appropriate reversal point. A sparking-out stroke is the motion to the other reversal point and back (see the table below):

Note

Oscillation with motion-synchronous actions and stop times "OST1/OST2"

Once the set stop times have expired, the internal block change is executed during oscillation (indicated by the new distances-to-go of the axes). The deactivation function is checked when the block changes. The deactivation function is defined according to the control setting for the motional sequence "OSCTRL".

This dynamic response can be influenced by the feed override.

An oscillation stroke may then be executed before the sparking-out strokes are started or the end position approached.

Although it appears as if the deactivation response has changed, this is not in fact the case.

Table 15-2 Operational sequence for deactivation of oscillation

Function	Inputs	Explanation
Deactivation at defined reversal point	Number of sparking-out strokes equals 0, no end position active	The oscillation movement is stopped at the appropriate reversal point
Deactivation with specific number of sparking-out strokes	Number of sparking-out strokes is not equal to 0, no end position is active	After the appropriate reversal point is reached, the number of sparking-out strokes specified in the command are processed.
Deactivation with sparking-out strokes and defined end position (optional)	Number of sparking-out strokes is not equal 0 end position active	After the appropriate reversal point is reached, the number of sparking-out strokes specified in the command are processed, followed by approach to specified end position.
Deactivation without sparking-out strokes, but with defined end position (optional)	Number of sparking-out strokes is equal 0 end position active	After the appropriate reversal point is reached, the axis is traversed to the specified end position.

NC language

The NC programming language allows asynchronous oscillation to be controlled from the part program. The following functions allow asynchronous oscillation to be activated and controlled as a function of NC program execution.

Note

If the setting data is directly written in the part program, then the data change takes effect prematurely with respect to processing of the part program (at the preprocessing time). It is possible to re-synchronize the part program and the oscillation function commands by means of a preprocessing stop (STOPRE).

Further information

Programming Manuals

1) Activate, deactivate oscillation:

- OS[oscillation axis] = 1; Activate oscillation for oscillation axis
- OS[oscillation axis] = 0; Deactivate oscillation for oscillation axis

Note

Every axis may be used as an oscillation axis.

2) End of oscillation:

- WAITP(oscillation axis)
  Positioning axis command – stops block until oscillation axis is at fine stop and synchronizes preprocessing and main run. The oscillation axis is entered as positioning axis again and can then be used normally.
  If an axis is to be used for oscillation, it must be released with a WAITP(axis) command beforehand.
  This also applies if oscillation is initiated from the PLC/HMI. In this case, the WAITP(axis) call is also needed if the axis was programmed beforehand via the NC program. As of SW version 3.2 it is possible to select via machine data $MA_AUTO_GET_TYPE, whether WAITP() shall be performed with programming or automatically.

Note

WAITP effectively implements a time delay until the oscillation movement has been executed. Termination of the movement can be initiated, for example, through a programmed deactivation command in the NC program or via the PLC or HMI by means of deletion of distance-to-go.

3) Setting reversal points:

- OSP1[oscillation axis] = position of reversal point 1
- OSP2[oscillation axis] = position of reversal point 2

A position is entered into the appropriate setting data in synchronism with the block in the main run and thus remains effective until the setting data is next changed.

If incremental traversal is active, then the position is calculated incrementally to the last appropriate reversal point programmed in the NC program.

4) Stopping times at reversal points:

- OST1[oscillation axis] = stop time at reversal point 1 in [s]
- OST2[oscillation axis] = stop time at reversal point 2 in [s]

A stop time is entered into the appropriate setting data in synchronism with the block in the main run and thus remains effective until the setting data is next changed.

The unit for the stop time is identical to the unit selected for the stop time programmed with G04.

5) Setting feedrate:

- FA[axis] = FValue
  Positioning axis infeed.

The feedrate is transferred to the appropriate setting data in synchronism with the block in the main run. If the oscillation axis is moved with reversal feed, the corresponding dependencies must be indicated as described in Description of Functions V1.

6) Setting control settings for sequence of movements:

- OSCTRL[oscillating axis] = (set options, reset options)
  The set options are defined as follows (the reset options deselect the settings):

Table 15-3 Set/reset options

Option value	Meaning
0	Stop at next reversal point on deactivation of the oscillation movement (default). Can only be achieved by resetting option values 1 and 2.
1	Stop at reversal point 1 on deactivation of the oscillation movement
2	Stop at reversal point 2 on deactivation of the oscillation movement
3	On deactivation of oscillation movement, do not approach reversal point unless sparking-out strokes are programmed
4	Approach an end position after spark-out process
8	If the oscillation movement is aborted by delete distance-to-go, the sparking-out strokes must then be executed and the end position approached (if programmed)
16	If the oscillation movement is terminated by deletion of distance-to-go, the programmed reversal point must be approached on deactivation of the oscillation movement.
32	Altered feedrate will only take effect from the next reversal point.
64	If feedrate setting is 0, path overlay is active, or otherwise velocity overlay
128	For rotary axis DC (shortest path)
256	Sparking-out stroke as single stroke
512	First approach start position

Note

The option values 0-3 encode the behavior at reversal points at Power OFF. You can choose one of the alternatives 0-3. The other settings can be combined with the selected alternative according to individual requirements. A + character can be inserted to create a string of options.

Example: The oscillation movement for axis Z must stop at reversal point 1 on deactivation; an end position must then be approached and a newly programmed feedrate take immediate effect; the axis must stop immediately after deletion of distance-to-go.

OSCTRL[Z] = (1+4, 16+32+64)

The set/reset options are entered into the appropriate setting data in synchronism with the block in the main run and thus remain effective until the setting data is next changed.

Note

The control evaluates the reset options, then the set options.

7) Sparking-out strokes:

- OSNSC[oscillation axis] = number of sparking-out strokes

The number of sparking-out strokes is entered into the appropriate setting data in synchronism with the block in the main run and thus remains effective until the setting data is next changed.

8) End position to be approached after deactivation of oscillation:

- OSE[oscillation axis] = end position of oscillation axis

The end position is entered into the appropriate setting data in synchronism with the block in the main run and thus remains effective until the setting data is next changed. Option value 4 is implicitly set so that the set end position is approached.

9) Start position to be approached prior to activation of oscillation:

- OSB [oscillation axis] = start position of oscillation axis

The start position is entered into the appropriate setting data SD43790 $SA_OSCILL_START_POS in synchronism with the block in the main run and thus remains effective until the setting data is next changed. Bit 9 in setting data SD43770 $SA_OSCILL_CTRL_MASK must be set to initiate an approach to the start position. The start position is approached **before reversal point 1**. If the start position coincides with reversal position 1, reversal position 2 is approached next.

As an alternative to programming command OSB, it is also possible to enter the start position directly in setting data SD43790 $SA_OSCILL_START_POS.

All positional information in the setting data and system variables refer to the basic coordinate system (BCS). The positional data for OSB, OSE refer to the workpiece coordinate system (WCS).

No halt time applies when the start position is reached, even if this position coincides with reversal position 1; instead, the axis waits for the exact stop fine signal. Any configured exact stop condition is fulfilled.

If a non-modal oscillation process does not require an infeed motion if the start position coincides with reversal position 1, this option can be configured with another synchronized action (see Section "Non-modal oscillation (starting position = reversal point 1) (Page 608)").

Programming example

An example that contains all the important elements for asynchronous oscillation can be found in Section "Example of asynchronous oscillation (Page 601)".

15.2.3 Asynchronous oscillation under PLC control

Activation

The function can be selected from the PLC using the following setting data in all operating modes except for MDA Ref and JOG Ref.:

SD43780 OSCILL_IS_ACTIVE (switch-on oscillation motion)

Settings

The following criteria can be controlled from the PLC via setting data: Activation and deactivation of oscillation movement, positions of reversal points, stop times at reversal points, feedrate velocity, the options in the reversal points, the number of sparking-out strokes and the end position after deactivation. However, these values can also be set beforehand as a setting data via the HMI user interface directly or via an NC program. These settings remain valid after power ON and the PLC can also start an oscillation movement set in this way directly via setting data OSCILL_IS_ACTIVE (via variable service).

Supplementary conditions

A spindle which must act as an axis to execute an oscillation movement started via the PLC must fulfill the conditions required to allow traversal as a positioning axis, i.e. the spindle must, for example, have been switched to the position control (SPOS) beforehand.

The axes always respond to the following two stop bits - regardless of whether the axis is controlled from the PLC or not:

- DB31, ... DBX28.5 (stop at the next reversal point)
- DB31, ... DBX28.6 (stop along braking ramp)

15.2.4 Special reactions during asynchronous oscillation

With PLC control

The PLC program can assume the control of an oscillation axis via NC/PLC signals. The NC/PLC interface signals also include end of program, operating mode change and single block.

Using the NCU system software, an asynchronous reciprocating axis interpolated by the main run reacts to NC STOP, alarm handling, end of program, program control and RESET.

The axis/spindle is controlled via the axial NC/PLC interface (PLC→NC) by means of IS DB31, ... DBX28.7 (PLC controlled axis) = 1

For further information about axes with PLC control, see Section "P2: Positioning axes (Page 185)".

Without PLC control

If the PLC does not have control over the axis, then the axis is treated like a normal positioning axis (POSA) during asynchronous oscillation.

Delete distance-to-go

Channel-specific delete distance-to-go is ignored.

Axial delete distance-to-go

- Without PLC control: Stop via braking ramp
- With PLC control: No stop (must be initiated from the PLC)

The following applies to **both** cases: After the axis is stopped, if necessary, the appropriate reversal point is approached and the distance-to-go deleted. The sparking-out strokes are then executed and the end position approached. Provided this has been set in OSCILL_CTRL_MASK.

The oscillation movement is then completed.

Note

During grinding, the calipers can be put into action via axial delete distance-to-go.

Emergency stop

The emergency stop completes the oscillation movement that must be restarted.

Reset

The oscillation movement is interrupted and deselected with a braking ramp. The options selected subsequently are not processed (sparking-out strokes, end point approach).

Working area limitation, limit switches

If it is detected during preprocessing that the oscillation movement would violate an active limitation, then an alarm is output and the oscillation movement not started.

If during an active oscillation movement the oscillation axis overtravels a limitation that has been activated in the meantime (e.g. 2nd software limit switch), then the axis is decelerated along a ramp and an alarm output.

⚠ CAUTION
Protection areas No protection areas act for a oscillation movement.

Follow-up mode

There is no difference to positioning axes.

End of program

If the axis is not controlled by the PLC, then the program end is not reached until the oscillation movement is terminated (reaction as for POSA:

Positioning across block boundaries).

If the axis is controlled by the PLC, then it continues to oscillate after program end.

Mode change

The following table shows the operating modes in which oscillation can be implemented. Changeover to an operating mode which allows oscillation does not affect the oscillation movement. Changeover to inadmissible operating modes is rejected with an alarm. It is not possible to traverse an axis in oscillation mode while applying control commands from the NC program or via operator inputs (JOG) simultaneously; an alarm is output if this is attempted. The following applies: The type of movement first started has priority.

Table 15-4 Operating modes which allow oscillation

Operating mode	Allows oscillation
AUTO	Yes
MDA	Yes
MDA Repos	Yes
MDA Teachin	Yes
MDA Ref	No
JOG	Yes
JOG Ref	No
JOG Repos	Yes

Single-block processing

If the axis is not controlled by the PLC, then it responds to a single block in the same way as a positioning axis (POSA), i.e. it continues the movement.

Override

The override is specified by the:

NC/PLC interface

Axial override acts on the oscillation axis.

Programming

The override acts on oscillation axes in the same way as on positioning axes.

Block search

In Block Search the last valid oscillation function is registered and the machine data OSCILL_MODE_MASK is activated (default) accordingly, either directly after NC start (when approaching the start position after block search) or after reaching the start position after block search.

OSCILL_MODE_MASK Bit 0:

0: Oscillation starts after reaching the start position.

1: Oscillation starts immediately after NC start.

REORG

Reversal point 1 is always approached first before oscillation continues.

ASUB

The oscillation movement continues while an ASUB (asynchronous subprogram) is in progress.

15.3 Oscillation controlled by synchronized actions

General procedure

An asynchronous oscillation movement is coupled via synchronized actions with an infeed motion and controlled accordingly.

The following description concentrates solely on the motion-synchronous actions associated with the oscillation function.

Further information

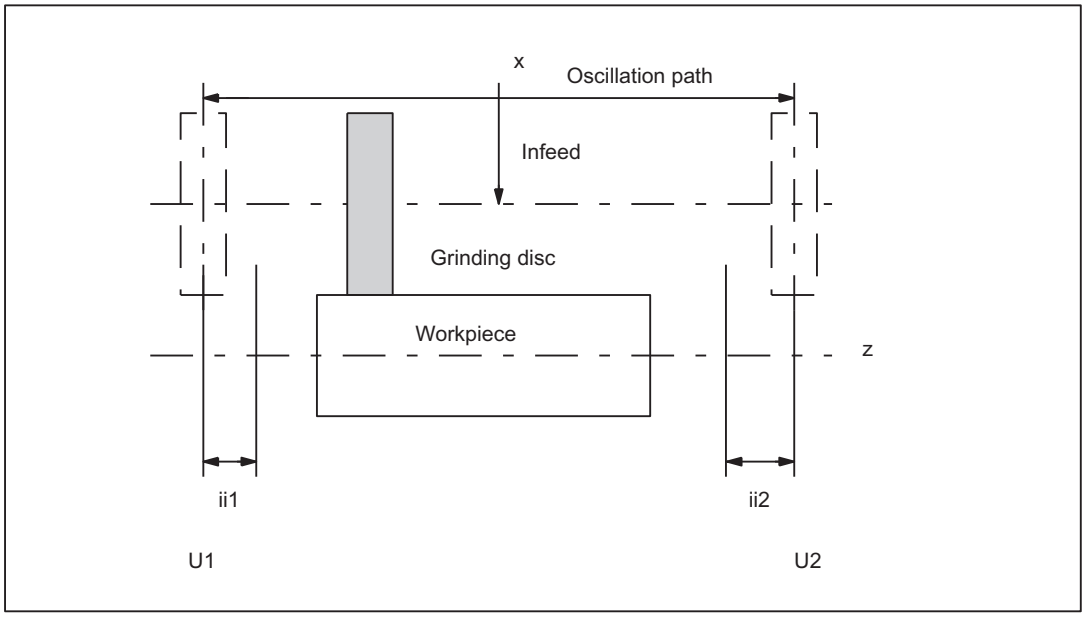
Function Manual Synchronized Actions

Functions

You can implement the following functions with the commands described in the following:

1. Infeed at the reversal point (see Section "Infeed at reversal point 1 or 2 (Page 593)").
2. Infeed at the reversal range (see Section "Infeed in reversal point range (Page 594)").
3. Infeed in both reversal points (see Section "Infeed at both reversal points (Page 595)").
4. Stop the oscillation movement at the reversal point until infeed is terminated (see Section "Stop oscillation movement at the reversal point (Page 596)").

5. Enable oscillation movement (see Section "Oscillation movement restarting (Page 597)").
6. Do not start partial infeed too early (see Section "Do not start partial infeed too early (Page 598)").



U1 Reversal point 1
U2 Reversal point 2
ii1 Reversal range 1
ii2 Reversal range 2

Figure 15-1 Arrangement oscillating axis, infeed axis

Programming

Before the motion block that contains the assignment between the infeed and the oscillating axis (see Section "Assignment of oscillation and infeed axes OSCILL (Page 599)"), the infeed definition (POSP) and the motion-synchronous actions, the parameters for the oscillation must first be defined (see Section "Influences on asynchronous oscillation (Page 581)"):

Using a WAITP [oscillating axis] (see MD30552 $MA_AUTO_GET_TYPE), the axis for oscillation is enabled. This means that the oscillating parameters are simultaneously transferred into the setting data. The symbolic names, e.g. $SA_REVERSE_POS1 can then be used to program the motion-synchronous actions.

Note

For motion-synchronous actions with $SA_REVERSE_POS values, the comparison **values at the time of interpretation** are valid. If setting data is changed afterwards, this has no effect.

For motion-synchronous actions with $AA_REVERSE_POS values, the comparison values within the **interpolation** are valid. This ensures a reaction to the modified reversal positions.

- **Conditions for motion-synchronized actions (frequency WHEN / WHENEVER)**
- Activation through motion block
  - Assign oscillation axis and infeed axes to one another OSCILL
  - Specify infeed response POSP

Note

By setting the axial override in the action component of the motion-synchronized action to zero (`DO $AA_OVR[<axis>]=0`), then the oscillating/infeed axis can be stopped if the associated condition (frequency WHEN / WHENEVER) is fulfilled.

An override of 100% automatically becomes active if the condition is no longer fulfilled.

Main run evaluation

It is possible to compare the synchronization conditions in the Interpolator clock cycle in the main run with the current actual values ($$ variable on the right of comparison conditions). With normal system variable comparison, the expressions are evaluated in the first run.

You can find a complete description of the synchronized actions in:

Further information

Function Manual Synchronized Actions

Examples

Example 1: unchanged reversal positions

For motion-synchronized actions, the reversal positions **$**SA_OSCILL_REVERSE_POS are used at the **preprocessing level**. If the corresponding setting data change, then the modified values are **not active** in the program.

```
Program code                                Comment
...
$SA_OSCILL_REVERSE_POS1[Z]=-10      ; Preprocessing var.: reversal position 1
$SA_OSCILL_REVERSE_POS2[Z]=10       ; Preprocessing var.: reversal position 2
G0 X0 Z0
WAITP(Z)
; Synchronized action 1: Reversal point 1 fallen below ⇒
; Stop infeed axis with override 0%
; Preprocessing var. Reversal position 1 is used
ID=1 WHENEVER $AA_IM[Z] < $SA_OSCILL_REVERSE_POS1[Z] DO $AA_OVR[X]=0
; Synchronized action 2: Reversal point 2 exceeded ⇒
; Stop infeed axis with override 0%
; Preprocessing var. reversal position 2 is used
ID=2 WHENEVER $AA_IM[Z] > $SA_OSCILL_REVERSE_POS2[Z] DO $AA_OVR[X]=0
OS[Z]=1 FA[X]=1000 POS[X]=40        ; Activate oscillation
```

Program code	Comment
OS[Z]=0	; Deactivate oscillation
M30	

Example 2: changing reversal positions

For the motion-synchronized actions, as reversal position, main run variable **$ $**AA_OSCILL_REVERSE_POSx is used. If the corresponding setting data change, then the modified values are **active** in the program.

Program code	Comment
...	
$SA_OSCILL_REVERSE_POS1[Z]=-10	; Preprocessing var.: reversal position 1
$SA_OSCILL_REVERSE_POS2[Z]=10	; Preprocessing var.: reversal position 2
G0 X0 Z0	
WAITP(Z)	
; Synchronized action 1: Reversal point 1 fallen below ⇒	
; Stop infeed axis with override 0%	
; **Main run var.** Reversal position 1 is used	
ID=1 WHENEVER $AA_IM[Z] < **$$**SA_OSCILL_REVERSE_POS1[Z] DO $AA_OVR[X]=0	
; Synchronized action 2: Reversal point 2 exceeded ⇒	
; Stop infeed axis with override 0%	
; **Main run var.** reversal position 2 is used	
ID=2 WHENEVER $AA_IM[Z] > **$$**SA_OSCILL_REVERSE_POS2[Z] DO $AA_OVR[X]=0	
OS[Z]=1 FA[X]=1000 POS[X]=40	; Activate oscillation
OS[Z]=0	; Deactivate oscillation
M30	

15.3.1 Infeed at reversal point 1 or 2

Function

As long as the oscillation axis has not reached the reversal point, the infeed axis does not move.

Application

Direct infeed in reversal point

Programming

Reversal point 1

```
WHENEVER $AA_IM[Z] <> $SA_OSCILL_REVERSE_POS1[Z]
 DO $AA_OVR[X] = 0 $AA_OVR[Z] = 100
```

Reversal point 2

```
WHENEVER $AA_IM[Z] <> $SA_OSCILL_REVERSE_POS2[Z]
```

```
 DO $AA_OVR[X] = 0 $AA_OVR[Z] = 100
```

Explanation of system variables

- `$AA_IM[ Z ]`: Current position of oscillating axis Z in the MCS
- `$SA_OSCILL_REVERSE_POS1[ Z ]`: Position of the reversal point1 of the oscillation axis
- `$AA_OVR[ X ]`: Axial override of the infeed axis
- `$AA_OVR[ Z ]`: Axial override of the oscillation axis

Infeed

The absolute infeed value is defined by the POSP instruction (see Section "Definition of infeeds POSP (Page 599)").

Assignment

The assignment between the oscillation axis and the infeed axis is defined by the OSCILL instruction (see Section "Assignment of oscillation and infeed axes OSCILL (Page 599)").

15.3.2 Infeed in reversal point range

Reversal range 1

Function

No infeed takes place provided the oscillation axis has not reached the reversal range (position at reversal point 1 plus contents of variables ii1). This applies on the condition that reversal point 1 is set to a lower value than reversal point 2. If this is not the case, then the condition must be changed accordingly.

Application

The purpose of this synchronized action is to prevent the infeed movement from starting until the oscillation movement has reached reversal range 1 (see "Figure 15-1 Arrangement oscillating axis, infeed axis (Page 591)").

Programming

```
WHENEVER $AA_IM[Z] > $SA_OSCILL_REVERSE_POS1[Z] + ii1
 DO $AA_OVR[X] = 0
```

Explanation of system variables:

- `$AA_IM[ Z ]`: Current position of oscillating axis Z
- `$SA_OSCILL_REVERSE_POS1[ Z ]`: Position of reversal point 1 of the oscillation axis
- `$AA_OVR[ X ]`: Axial override of the infeed axis
- `ii1`: Magnitude of reversal range (user variable)

Reversal range 2

Function

The infeed axis stops until the current position (value) of the oscillation axis is lower than the position at reversal point2 minus the contents of variable ii2. This applies on condition that the setting for reversal point position 2 is higher than that for reversal point position 1. If this is not the case, then the condition must be changed accordingly.

Application

The purpose of this synchronized action is to prevent the infeed movement from starting until the oscillation movement has reached reversal range 2 (see "Figure 15-1 Arrangement oscillating axis, infeed axis (Page 591)").

Programming

```
WHENEVER $AA_IM[Z] < $SA_OSCILL_REVERSE_POS2[Z] - ii2
 DO $AA_OVR[X] = 0
```

Explanation:

- `$AA_IM[ Z ]`: Current position of oscillating axis Z
- `$SA_OSCILL_REVERSE_POS2[ Z ]`: Position of reversal point 2 of the oscillation axis
- `$AA_OVR[ X ]`: Axial override of the infeed axis
- `ii2`: Magnitude of reversal range 2 (user variable)

Infeed

The absolute infeed value is defined by the POSP instruction (see Section "Definition of infeeds POSP (Page 599)").

Assignment

The assignment between the oscillation axis and the infeed axis is defined by the OSCILL instruction (see Section "Assignment of oscillation and infeed axes OSCILL (Page 599)").

See also

Oscillation controlled by synchronized actions (Page 590)

15.3.3 Infeed at both reversal points

General procedure

The functions described above for infeed at the reversal point and in the reversal range can be freely combined.

Combinations

Infeed at two sides

- Reversal point 1 - reversal point 2
- Reversal point 1 - reversal range 2
- Reversal range 1 - reversal point 2
- Reversal range 1 - reversal range 2

One-sided infeed

- Reversal point 1
- Reversal point 2
- Reversal range 1
- Reversal range 2

(See Section "Infeed at reversal point 1 or 2 (Page 593)" and "Infeed in reversal point range (Page 594)")

15.3.4 Stop oscillation movement at the reversal point

Function

Reversal point 1:

Every time the oscillation axis reaches reversal position 1, it must be stopped by means of the override and the infeed movement started.

Application

The synchronized action is used to hold the oscillation axis stationary until partial infeed has been executed. This synchronized action can be omitted if the oscillation axis need not wait at reversal point 1 until partial infeed has been executed. At the same time, this synchronized action can be used to start the infeed movement if this has been stopped by a previous synchronized action which is still active.

Programming

```
WHENEVER $AA_IM[oscillation axis] ==
$SA_OSCILL_REVERSE_POS1[oscillation axis]
 DO $AA_OVR[oscillation axis] = 0   $AA_OVR[infeed axis] = 100
```

Explanation of system variables

`$AA_IM[ oscillation axis ]`: Current position of oscillation axis

`$SA_OSCILL_REVERSE_POS1[ oscillation axis ]`: Reversal point 1 of the oscillation axis

`$AA_OVR[ oscillation axis]`: Axial override of the oscillation axis

$AA_OVR[infeed axis]: Axial override of the infeed axis

Function

Reversal point 2:

Every time the oscillation axis reaches reversal position 2, it must be stopped by means of the override 0 and the infeed movement started.

Application

The synchronized action is used to hold the oscillation axis stationary until partial infeed has been executed. This synchronized action can be omitted if the oscillation axis need not wait at reversal point 2 until partial infeed has been executed. At the same time, this synchronized action can be used to start the infeed movement if this has been stopped by a previous synchronized action which is still active.

Programming

```
WHENEVER $AA_IM[oscillation axis] ==
$SA_OSCILL_REVERSE_POS2[oscillation axis]
 DO $AA_OVR[oscillation axis] = 0   $AA_OVR[infeed axis] = 100
```

Explanation of system variables

- $AA_IM[oscillation axis]: Current position of oscillation axis
- $SA_OSCILL_REVERSE_POS2[oscillation axis]: Reversal point 2 of the oscillation axis
- $AA_OVR[oscillation axis]: Axial override of the oscillation axis
- $AA_OVR[infeed axis]: Axial override of the infeed axis

15.3.5 Oscillation movement restarting

Function

The oscillation axis is started via the override whenever the distance-to-go for the currently traversed path section of the infeed axis = 0, i.e. partial infeed has been executed.

Application

The purpose of this synchronized action is to continue the movement of the oscillation axis on completion of the part infeed movement. If the oscillation axis need not wait for completion of partial infeed, then the motion-synchronous action with which the oscillation axis is stopped at the reversal point must be omitted.

Programming

```
WHENEVER $AA_DTEPW[infeed axis] == 0
```

```
DO $AA_OVR[oscillation axis]=100
```

Explanation of system variables

- $AA_DTEPW[infeed axis]: axial remaining travel distance for the infeed axis in the workpiece coordinate system (WCS): Path distance of the infeed axis
- $AA_OVR[oscillation axis]: Axial override for oscillation axis

15.3.6 Do not start partial infeed too early

Function

The functions described above prevent any infeed movement outside the reversal point or the reversal range. On completion of an infeed movement, however, restart of the next partial infeed must be prevented.

Application

A channel-specific flag is used for this purpose. This flag is set at the end of the partial infeed (partial distance-to-go == 0) and is deleted when the axis leaves the reversal range. The next infeed movement is then prevented by a synchronized action.

Programming

```
WHENEVER $AA_DTEPW[infeed axis]==0 DO $AC_MARKER[index]=1
```

Reversal point 1

```
WHENEVER $AA_IM[Z] <> $SA_OSCILL_REVERSE_POS1[Z] DO
$AC_MARKER[index]=0
WHENEVER $AC_MARKER[index]==1 DO $AA_OVR[infeed axis]=0
```

Explanation of system variables

- $AA_DTEPW[infeed axis]: axial remaining travel distance for the infeed axis in the workpiece coordinate system (WCS): Path distance of the infeed axis
- $AC_MARKER[index]: Channel-specific marker with index
- $AA_IM[oscillation axis]: Current position of oscillation axis
- $SA_OSCILL_REVERSE_POS1[oscillation axis]: Reversal point 1 of the oscillation axis
- $AA_OVR[infeed axis]: Axial override for infeed axis

15.3.7 Assignment of oscillation and infeed axes OSCILL

Function

One or several infeed axes are assigned to the oscillation axis with command OSCILL. Oscillation motion starts.

The PLC is informed of which axes have been assigned via the NC/PLC interface. If the PLC is controlling the oscillation axis, it must now also monitor the infeed axes and use the signals for the infeed axes to generate the reactions on the oscillation axis via 2 stop bits of the interface.

Application

The axes whose response has already been defined by synchronous conditions are assigned to one another for activation of oscillation mode. The oscillation movement is started.

Programming

OSCILL[oscillation axis] = (infeed axis1, infeed axis2, infeed axis3)

Infeed axis2 and infeed axis3 in brackets plus their delimiters can be omitted if they are not required.

15.3.8 Definition of infeeds POSP

Function

The control receives the following data for the infeed axis:

- Total infeed
- Part infeed at reversal point/reversal point range
- Part infeed response at end

Application

This instruction must be given after activation of oscillation with `OSCILL` to inform the controller of the required infeed values at the reversal points/reversal point ranges.

Programming

POSP[infeed axis] = (end position, part section, mode)

End position: End position for the infeed axis after all partial infeeds have been traversed.

Part section: Part infeed at reversal point/reversal point range

Mode 0: For the last two part steps, the remaining path up to the target point is divided into two equally large residual steps (default setting).

Mode 1: The part length is adjusted such that the total of all calculated part lengths corresponds exactly to the path up to the target point.

15.3.9 External oscillation reversal

For example, keys on the PLC can be used to change the oscillation area or instantaneously reverse the direction of oscillation.

The edge-triggered PLC input signal DB31, ... DBX28.0 (oscillation reversal) is used to brake the current oscillation motion and then traverse in the opposite direction. The braking phase is signaled via the PLC output signal DB31, ... DBX100.2 (oscillation reversal active).

The braking position of the axis can be accepted as the **new reversal position** by means of the PLC signal DB31, ... DBX28.4 (change reversal position).

The PLC input signal DB31, ... DBX28.3 (select reversal point) is ignored, but rather the change affects the last initiated *external oscillation reversal* command.

No change in the reversal points applied via handwheel or JOG keys may be active for the relevant axis. If handwheel or JOG key changes are currently active, display alarm 20081 (Braking position cannot be accepted as reversal position - handwheel active) will be generated. The alarm is automatically reset when the conflict has been eliminated.

Hold time

No stop time is applied for a change of direction due to an *external oscillation reversal*. The axis waits for the exact stop fine signal. Any configured exact stop condition is fulfilled.

Infeed motion

With non-modal oscillation, no infeed movement is performed for a change of direction due to an *external oscillation reversal* as the reversal position has not been reached and consequently the appropriate synchronized action is not fulfilled.

System variables

The braking position can be scanned via system variable $AA_OSCILL_BREAK_POS1, when approach to reversal position 1 is aborted or via

$AA_OSCILL_BREAK_POS2 when approach to reversal position 2 is aborted.

If the relevant reversal point is approached again, the position of the reversal point can be scanned in $AA_OSCILL_BREAK_POS1 or $AA_OSCILL_BREAK_POS2.

In other words, only after an *External oscillation reversal* command is there a difference between the values in $AA_OSCILL_BREAK_POS1 and $AA_OSCILL_REVERSE_POS1 or the values in $AA_OSCILL_BREAK_POS2 and $AA_OSCILL_REVERSE_POS2.

External oscillation reversal can therefore be **detected** by a synchronized action (see Section "Examples (Page 601)").

Special cases

If the PLC input signal "oscillation reversal" is activated as the axis is approaching the start position, the approach movement is aborted and the axis continues by approach interruption position 1.

If the PLC input signal "oscillation reversal" is set during a stop period, the stop timer is deactivated; if exact stop fine has not yet been reached, the axis waits for the exact stop fine reached signal before continuing its motion.

If the PLC input signal "oscillation reversal" is activated as the axis is approaching the end position, the approach movement is aborted and oscillation is terminated.

For an example of the external oscillation reversal command (see Section "Change reversal position via synchronized action with "external oscillation reversal" (Page 611)").

15.4 Marginal conditions

Availability of the "Oscillation" function

The function is an option ("oscillation functions"), which must be assigned to the hardware through the license management.

15.5 Examples

Requirements

The examples given below require components of the NC language specified in the sections entitled:

- Asynchronous oscillation

and

- Oscillation controlled by synchronized actions.

15.5.1 Example of asynchronous oscillation

Task

The oscillation axis Z must oscillate between -10 and 10. Approach reversal point 1 with exact stop coarse and reversal point 2 without exact stop. The oscillation axis feedrate must be 5000. 3 sparking-out strokes must be executed at the end of the machining operation followed by approach by oscillation axis to end position 30. The feedrate for the infeed axis is 1000, end of the infeed in X direction is at 15.

Program section

Program code	Comment
OSP1[Z]=-10	; Reversal point 1
OSP2[Z]=10	; Reversal point 2
OST1[Z]=-1	; Stop time at reversal point 1: Exact stop coarse
OST2[Z]=-2	; Stop time at reversal point 2: without exact stop
FA[Z]=5000	; Infeed for oscillating axis
OSNSC[Z]=3	; three spark-out strokes
OSE[Z]=-3	; End position
OS1 F500 X15	; start oscillation, infeed X axis
	; with infeed 500, feed target 15

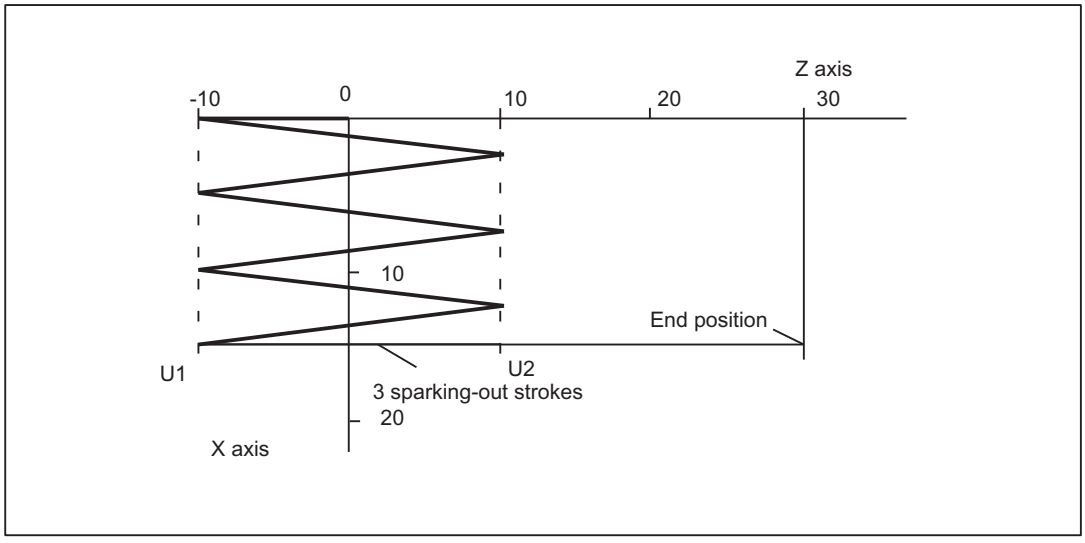


Figure 15-2 Sequences of oscillation movements and infeed, example 1

15.5.2 Example 1 of oscillation with synchronized actions

Task

Direct infeed must take place at reversal point 1; the oscillation axis must wait until the partial infeed has been executed before it can continue traversal. At reversal point 2, the infeed must take place at a distance of -6 from reversal point 2; the oscillation axis must not wait at this reversal point until partial infeed has been executed. Axis Z is the oscillation axis and axis X is the infeed axis (see Section "Oscillation controlled by synchronized actions (Page 590)).

Note

The setting data OSCILL_REVERSE_POS_1/2 are values in the machine coordinate system; therefore comparison is only suitable with $AA_IM[n].

Program section

```
Program code                    Comment
; Example 1: Oscillation with synchronized actions
OSP1[Z]=10 OSP2[Z]=60          ; explain reversal points 1 and 2
OST1[Z]=-2 OST2[Z]=0           ; Reversal point 1: without exact stop
                               ; Reversal point 2: Exact stop fine
FA[Z]=5000 FA[X]=250           ; Feed for oscillating axis, feedrate, infeed axis
OSCTRL[Z]=(1+8+16,0)           ; Switch off oscillation motion at reversal point 1
                               ; After delete DTG spark-out and approach end position
                               ; After DTG, approach relevant reversal position
                               ;
OSNSC[Z]=3                     ; 3 sparking-out strokes
OSE[Z]=0                       ; End position = 0
WAITP(Z)                       ; enable oscillation for the Z axis
;
; motion-synchronous actions:
;
; always, when                  the current position of the oscillating axis in the
;                               Machine Coordinate System
; not equal to                  reversal position 1
; then                          set the marker with index 1 to value 0
;                               (Reset marker 1)
WHENEVER $AA_IM[Z]<>$SA_OSCILL_REVERSE_POS1[Z] DO $AC_MARKER[1]=0
;
; always, when                  the current position of the oscillating axis in the
;                               Machine Coordinate System
; less than                     the beginning of reversal range 2
;                               (here: reversal point 2 -6),
; then                          set the axial override of the infeed axis to 0%.
; and                           set the marker with index 2 to value 0
;                               (Reset marker 2)
WHENEVER $AA_IM[Z]<$SA_OSCILL_REVERSE_POS2[Z]-6 DO $AA_OVR[X]=0 $AC_MARKER[2]=0
;
; always, when                  the current position of the oscillating axis in the
;                               Machine Coordinate System
; equal to                      reversal position 1,
; then                          set the axial override of the oscillation axis to 0%
;
; and                           Set the axial override of the infeed axis to
;                               100% (so that the previous synchronized action
;                               is cancelled!)
WHENEVER $AA_IM[Z]==$SA_OSCILL_REVERSE_POS1[Z] DO $AA_OVR[Z]=0 $AA_OVR[X]=100
;
; always, when                  the distance-to-go of the partial infeed is
; equal to                      0,
```

```
; then                          set the marker with index 2 to value 1
; and                           set the marker with index 1 to value 1
WHENEVER $AA_DTEPW[X]==0 DO $AC_MARKER[2]=1 $AC_MARKER[1]=1
;
; always, when                 the flag with index 2 is
; equal to                      1,
; then                          Set the axial override of the infeed axis to
;                               0%, this prevents a premature infeed
;                               (oscillation axis has not exited the
;                               reversal point 1).
;
WHENEVER $AC_MARKER[2]==1 DO $AA_OVR[X]=0
;
; always, when                 the flag with index 1 is
; equal to                      1,
; then                          Set the axial override of the infeed axis to
;                               0%, this prevents a premature infeed
;                               (oscillation axis has not exited the
;                               reversal point range 2).
; and                           Set the axial override of the infeed axis to
;                               100% ('Start' oscillation)
WHENEVER $AC_MARKER[1]==1 DO $AA_OVR[X]=0 $AA_OVR[Z]=100
;
; if the current position of the oscillating axis in the MCS is
; equal to                      reversal position 1,
; then                          Set the axial override of the infeed axis to
;                               100%
; and                           Set the axial override of the infeed axis to
;                               0% (so that the second synchronized action
;                               is cancelled once!)
WHEN $AA_IM[Z]==$SA_OSCILL_REVERSE_POS1[Z] DO $AA_OVR[Z]=100 $AA_OVR[X]=0
;
;-----------------------------------------
OSCILL[Z]=(X)                 ; Assign axis X to the oscillation axis Z as
POSP[X]=(5,1,1)
                              ; infeed axis, this should
                              ; infeed to end position 5 in steps
                              ; from 1 and the sum of all part lengths should
                              ; give exactly the end position
M30                           ; End of program
```

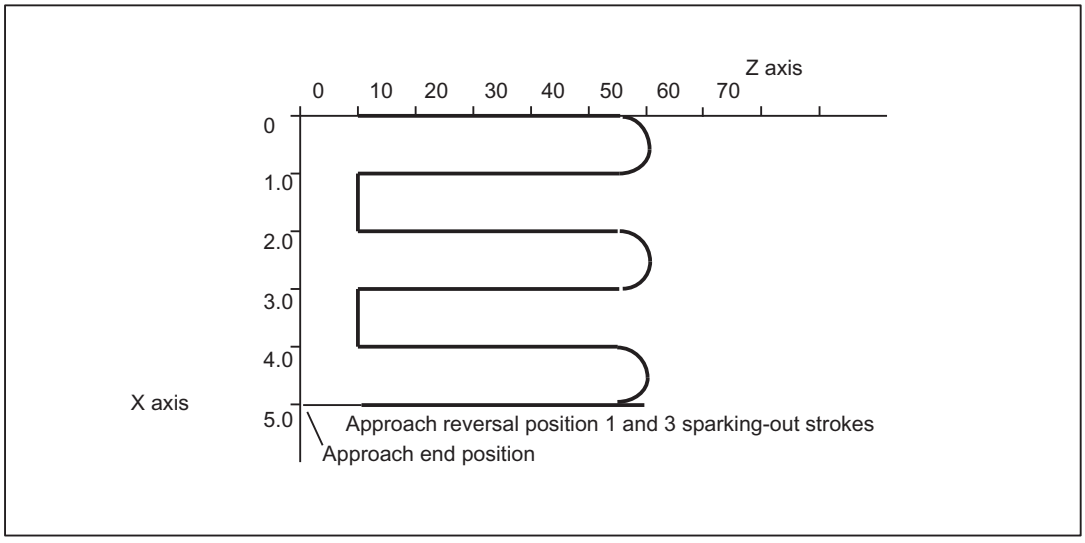


Figure 15-3 Sequences of oscillation movements and infeed, example 1

15.5.3 Example 2 of oscillation with synchronized actions

Task

No infeed must take place at reversal point 1. At reversal point 2, the infeed must take place at distance ii2 from reversal point 2; the oscillation axis must wait at this reversal point until partial infeed has been executed. Axis Z is the oscillation axis and axis X the infeed axis.

Program section

Example 2: Oscillation with synchronized actions

```
Program code                Comment
DEF INT ii2                 ; Define variable for reversal area 2
;
OSP1[Z]=10 OSP2[Z]=60       ; explain reversal points 1 and 2
OST1[Z]=0 OST2[Z]=0         ; Reversal point 1: Exact stop fine
;                             Reversal point 2: Exact stop fine
FA[Z]=5000 FA[X]=100        ; Feed for oscillating axis, feed for infeed axis
OSCTRL[Z]=(2+8+16,1)        ; Switch off oscillation motion at reversal point 2
;                             After delete distance-to-go, spark-out and end position
;                             Approach appropriately after delete distance-to-go
;                             Approach reversal position
OSNSC[Z]=3                  ; 3 sparking-out strokes
OSE[Z]=70                   ; End position = 70
ii2=2                       ; Set reversal point range
WAITP(Z)                    ; permit oscillation for Z axis
;
```

```
; motion-synchronous actions:
; always, when          the current position of the oscillating axis in the
;                       Machine Coordinate System
; less than             the start of reversal area 2
; then                  set the axial override of the feed axis
;                       to 0%
; and                   set the marker with index 0 to value 0
WHENEVER $AA_IM[Z]<$SA_OSCILL_REVERSE_POS2[Z]-ii2 DO $AA_OVR[X]=0 $AC_MARKER[0]=0
;
; always, when          the current position of the oscillating axis in the
;                       Machine Coordinate System
; greater or equal to   reversal position 2
; then                  set the axial override of the oscillating axis
;                       to 0%
WHENEVER $AA_IM[Z]>=$SA_OSCILL_REVERSE_POS2[Z] DO $AA_OVR[Z]=0
;
; always, when          the distance-to-go of the partial infeed is
; equal to              0,
; then                  set the marker with index 0 to value 1
WHENEVER $AA_DTEPW[X] == 0 DO $AC_MARKER[0]=1
;
; always, when          the flag with index 0 is
; equal to              1,
; then                  set the axial override of the infeed axis to 0%
;                       to prevent a new premature infeed
;                       (oscillation axis has not yet exited the reversal point
                        range 2,
;                       but the infeed axis is ready for a
;                       new infeed)
; and                   set the axial override of the oscillation axis to 100%
;                       (this cancels the 2nd synchronized action)
;
WHENEVER $AC_MARKER[0]==1 DO $AA_OVR[X]=0 $AA_OVR[Z]=100
;
OSCILL[Z]=(X)          ; Start the axes
POSP[X]=(5,1,1)
;                       The oscillating axis Z is assigned axis X as
;                       an infeed axis
;                       Axis X should travel to end position 5 in
;                       increments of 1
;
M30
```

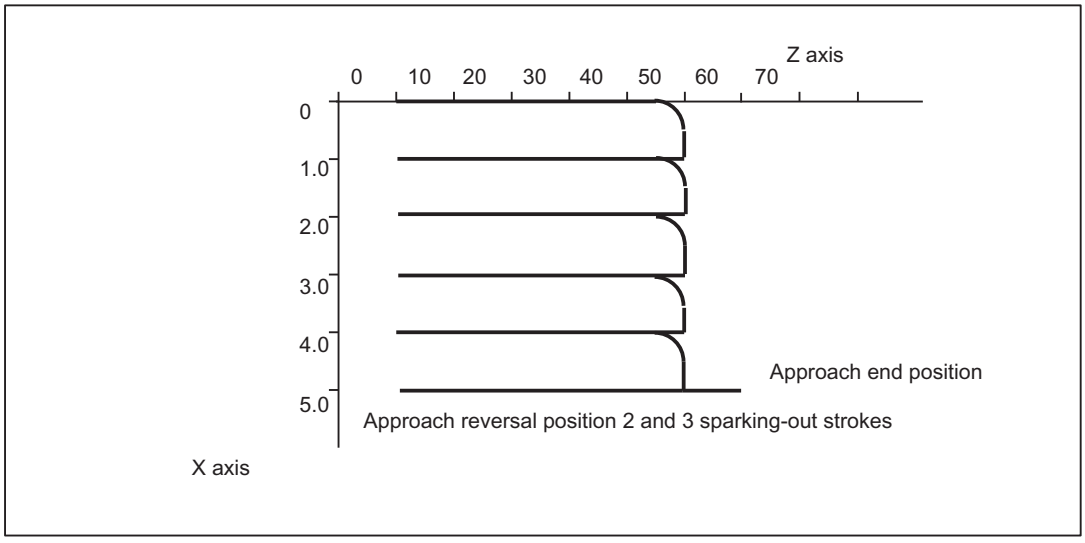


Figure 15-4 Sequences of oscillation movements and infeed, example 2

15.5.4 Examples for starting position

15.5.4.1 Define starting position via language command

Program code	Comment
WAITP(Z)	; enable oscillation for the Z axis
OSP1[Z]=10 OSP2[Z]=60	; explain reversal points 1 and 2
OST1[Z]=-2 OST2[Z]=0	; Reversal point 1: without exact stop
	; Reversal point 2: Exact stop fine
FA[Z]=5000 FA[X]=2000	; Infeed for oscillating axis,
	; feedrate for infeed axis
OSCTRL[Z]=(1+8+16,0)	; Switch off oscillation motion at reversal point 1
	; after delete DTG spark-out and approach end position,
	; After DTG, approach relevant reversal position
	;
OSNSC[Z]=3	; 3 sparking-out strokes
OSE[Z]=0	; End position = 0
OSB[Z]=0	; Start position = 0
OS[Z]=1 X15 F500	; Start oscillation, continuous infeed
OS[Z]=0	; Deactivate oscillation
WAITP(Z)	; wait for completion of the oscillation motion
M30	

Explanation

When the Z axis starts oscillation, it first approaches the starting position (position = 0 in the example) and then begins the oscillation motion between the reversal points 10 and 60. When the X axis has reached its end position 15, the oscillation finishes with 3 sparking out strokes and approach of end position 0.

15.5.4.2 Start oscillation via setting data

```
Program code                                Comment
WAITP(Z)
STOPRE
$SA_OSCILL_REVERSE_POS1[Z]=-10              ;   reversal position 1 = -10
$SA_OSCILL_REVERSE_POS2[Z]=30               ;   reversal position 2 = 30
$SA_OSCILL_START_POS[Z] = -50               ;   Start position = -50
$SA_OSCILL_CTRL_MASK[Z] = 512               ;   Approach start position,
                                            ;   on switch-off, stop at the next
                                            ;   reversal point
                                            ;   do not approach any end position
                                            ;   no sparking-out strokes for DTG
$SA_OSCILL_VELO[ Z ] = 5000                 ;   Infeed for oscillating axis
$SA_OSCILL_IS_ACTIVE[ Z ] = 1               ;   starting
$SA_OSCILL_DWELL_TIME1[ Z ] = -2            ;   without wait for exact stop
$SA_OSCILL_DWELL_TIME2[ Z ] = 0             ;   wait for fine exact stop
STOPRE
X30 F100
$SA_OSCILL_IS_ACTIVE[ Z ] = 0               ;   Stop
WAITP(Z)
M30
```

Description

When the Z axis starts oscillation, it first approaches the starting position (position = -50 in the example) and then begins the oscillation motion between the reversal points -10 and 30. When the X axis has reached its end position 30, the oscillation finishes at the next approached reversal point.

15.5.4.3 Non-modal oscillation (starting position = reversal point 1)

Oscillation with synchronized actions

```
Program code                    Comment
N701                            ; Oscillation with synchronized actions,
                                ; start position == reversal point 1
;
```

```
Program code                                    Comment
N702 OSP1[Z]=10 OSP2[Z]=60                      ; explain reversal points 1 and 2
N703 OST1[Z]=0 OST2[Z]=0                        ; Reversal point 1: Exact stop coarse
                                                ; Reversal point 2: Exact stop fine
N704 FA[Z]=5000 FA[X]=2000                      ; Infeed for oscillating axis,
                                                ; feedrate for infeed axis
N705 OSCTRL[Z]=(1+8+16.0)                       ; switch off oscillation motion at
                                                ; reversal point 1 after DTG spark-out
                                                ; and approach end position
                                                ; After DTG, approach relevant
                                                ; reversal position
                                                ;
N706 OSNSC[Z]=3                                 ; 3 sparking-out strokes
N707 OSE[Z]=0                                   ; End position = 0
N708 OSB[Z]=10                                  ; Start position = 10
N709 WAITP(Z)                                   ; enable oscillation for the Z axis
;
; motion-synchronous actions:
; set marker with index 2 on 1 (initialization)
WHEN TRUE DO $AC_MARKER[2]=1
;
; always, when                                  the marker with index 2 equals 0
;                                               and the current position of the oscillating axis
;                                               does not equal the reversal position 1
; then                                          set the marker with index 1 to 0.
;
WHENEVER ($AC_MARKER[2] == 0) AND $AA_IW[Z]>$SA_OSCILL_REVERSE_POS1[Z])
DO $AC_MARKER[1]=0
; always, when                                  the current position of the oscillating axis
;                                               is less than the start of reversal range 2
;
; then                                          set the axial override of the feed axis
;                                               to 0 and set the marker with index 0 to
;                                               0
WHENEVER $AA_IW[Z]<$SA_OSCILL_REVERSE_POS2[Z]-6 DO $AA_OVR[X]=0 $AC_MARKER[0]=0
;
; always, when                                  the current position of the oscillating axis
;                                               is equal to the reversal position 1,
; then                                          set the axial override of the oscillating axis
;                                               to 0 and set the axial override of the
;                                               infeed axis to 100% (so that the
;                                               previous synchronized action is cancelled!)
WHENEVER $AA_IW[Z]==$SA_OSCILL_REVERSE_POS1[Z] DO $AA_OVR[Z]=0 $AA_OVR[X]=100
;
; always, when                                  the distance-to-go of the partial infeed equals
                                                0
```

```
Program code                          Comment
;
; then                                Set the marker with index 0 to 1 and
;                                     set the marker with index 1 to 1 and
WHENEVER $AA_DTEPW[X]==0 DO $AC_MARKER[0]=1 $AC_MARKER[1]=1
;
; always, when                        the marker with index 0 equals 1,
; then                                set the axial override of the feed axis
;                                     to 0 to prevent a new premature
;                                     infeed!
WHENEVER $AC_MARKER[0]==1 DO $AA_OVR[X]=0
;
; always, when                        the marker with index 1 equals 1,
; then                                set the axial override of the feed axis
;                                     to 0 (to prevent a new premature
;                                     infeed!) and set the
;                                     axial override of the oscillation axis to 100%
;                                     (so that the previous
;                                     synchronized action is canceled!)
WHENEVER $AC_MARKER[1]==1 DO $AA_OVR[X]=0 $AA_OVR[Z]=100
;
; When                                the current position of the oscillating axis
;                                     is equal to the reversal position 1,
; then                                reset the marker with index 2,
;                                     release the first synchronized action (no
;                                     infeed when reaching the start position
;                                     == reversal position 1)
WHEN $AA_IW[Z]==$SA_OSCILL_REVERSE_POS1[Z] DO $AC_MARKER[2]=0
;
;---------------------------------
N750 OSCILL[Z]=(X) POSP[X]=(5,1,1)
; Assign axis X to the oscillation axis Z as infeed axis,
; this should infeed to end position 5
; in substeps of 1 and the sum of all sublengths
; should be exactly the same as the end position.
;
N780 WAITP(Z)                       ; release the Z axis
;
N790 X0 Z0
N799 M30                            ; End of program
```

Description

The starting position matches reversal point 1. The WHEN .... synchronized actions (see above) prevent an infeed when the starting position is reached.

15.5.5 Example of external oscillation reversal

15.5.5.1 Change reversal position via synchronized action with "external oscillation reversal"

```
Program code                    Comment
DEFINE BREAKPZ AS $AA_OSCILL_BREAK_POS1[Z]
DEFINE REVPZ AS $SA_OSCILL_REVERSE_POS1[Z]

WAITP(Z)                        ; enable oscillation for the Z axis
OSP1[Z]=10 OSP2[Z]=60           ; explain reversal points 1 and 2
OSE[Z]=0                        ; End position = 0
OSB[Z]=0                        ; Start position = 0

                                ; At external reversal of oscillation for
                                ; oscillation reversal point 1, adapt this
WHENEVER BREAKPZ <> REVPZ DO $$SA_OSCILL_REVERSE_POS1 = BREAKPZ
OS[Z]=1 X150 F500               ; Start oscillation, continuous infeed
OS[Z]=0                         ; Deactivate oscillation
WAITP(Z)                        ; wait for completion of the oscillation motion
M30
```

15.6 Data lists

15.6.1 Machine data

15.6.1.1 General machine data

Number	Identifier: $MN_	Description
10710	PROG_SD_RESET_SAVE_TAB	Oscillations to be saved from SD
11460	OSCILL_MODE_MASK	Control screen form for asynchronous oscillation

15.6.2 Setting data

15.6.2.1 Axis/spindle-specific setting data

Number	Identifier: $SA_	Description
43700	OSCILL_REVERSE_POS1	Position at reversal point 1
43710	OSCILL_REVERSE_POS2	Position at reversal point 2
43720	OSCILL_DWELL_TIME1	Stop time at reversal point 1
43730	OSCILL_DWELL_TIME2	Stop time at reversal point 2
43740	OSCILL_VELO	Feed velocity of oscillation axis
43750	OSCILL_NUM_SPARK_CYCLES	Number of sparking-out strokes
43760	OSCILL_END_POS	Position after sparking-out strokes/at end of oscillation movement
43770	OSCILL_CTRL_MASK	Control screen form for oscillation
43780	OSCILL_IS_ACTIVE	Oscillation movement ON/OFF
43790	OSCILL_START_POS	Position that is approached after oscillation before reversal point 1, if activated in SD43770:

15.6.3 System variables

15.6.3.1 Main run variables for motion-synchronous actions

Main run variable_read

The following variables are provided for main run variable_read:

$A_IN[<arith. expression>]	digital input (Boolean)
$A_OUT[<arith. expression>]	digital output (Boolean)
$A_INA[<arith. expression>]	analog input (Boolean)
$A_OUTA[<arith. expression>]	analog output (Boolean)
$A_INCO[<arith. expression>]	comparator inputs (Boolean)
$AA_IW[<axial expression>]	actual position PCS axis (real)
$AA_IB[<axial expression>]	actual position BCS axis (real)
$AA_IM[<axial expression>]	Actual position MCS axis (IPO setpoints) (real) With $AA_IM[S1] setpoints for spindles can be evaluated. Modulo calculation is used for spindles and rotary axes, depending on machine data $MA_ROT_IS_MODULO and $MA_DISPLAY_IS_MODULO.
$AA_OSCILL_BREAK_POS1	Breaking position after external oscillation reversal when approaching reversal point 1
$AA_OSCILL_BREAK_POS2	Breaking position after external oscillation reversal when approaching reversal point 2

$AC_TIME	Time from the start of the block (real) in seconds (including the times for the internally generated intermediate blocks)
$AC_TIMES	Time from the start of the block (real) in seconds (without times for the internally generated intermediate blocks)
$AC_TIMEC	Time from the start of the block (real) in IPO steps (including steps for the internally generated intermediate blocks)
$AC_TIMESC	Time from the start of the block (real) in IPO steps (without steps for the internally generated intermediate blocks)
$AC_DTBB	Distance from beginning of block in BCS (Distance to begin, baseCoor) (real)
$AC_DTBW	Distance from beginning of block in PCS (Distance to begin, workpieceCoor) (real)
$AA_DTBB[<axial expression>]	axial distance from beginning of block in BCS (Distance to begin, baseCoor) (real)
$AA_DTBW[<axial expression>]	axial distance from beginning of block in PCS (Distance to begin, workpieceCoor) (real)
$AC_DTEB	Distance to end of block in BCS (Distance to end) (Distance to end, baseCoor) (real)
$AC_DTEW	Distance to end of block in PCS (Distance to end, workpieceCoor) (real)
$AA_DTEB[<axial expression>]	axial distance to end of movement in BCS (Distance to end, baseCoor) (real)
$AA_DTEW[<axial expression>]	axial distance to end of movement in PCS (Distance to end, workpieceCoor) (real)
$AC_PLTBB	Distance from beginning of block in BCS (Path Length from begin, baseCoor) (real)
$AC_PLTEB	Distance to end of block in BCS (Distance to end) (Path Length to end, baseCoor) (real)
$AC_VACTB	Path speed in BCS (Velocity actual, baseCoor) (real)
$AC_VACTW	Path speed in PCS (Velocity actual, workPieceCoor) (real)
$AA_VACTB[<axial expression>]	Axis velocity in BCS (Velocity actual, baseCoor) (real)
$AA_VACTW[<axial expression>]	Axis velocity in PCS (Velocity actual, workPieceCoor) (real)
$AA_DTEPB[<axial expression>]	axial distance-to-go for oscillation infeed in BCS (Distance to end, pendulum, baseCoor) (real)
$AA_DTEPW[<axial expression>]	axial distance-to-go for oscillation infeed in PCS (Distance to end, pendulum, workpieceCoor) (real)
$AC_DTEPB	Path distance-to-go for oscillation infeed in BCS (not P2) (Distance to end, pendulum, baseCoor) (real)
$AC_DTEPW	Path distance-to-go for oscillation infeed in PCS (not P2) (Distance to end, pendulum, workpieceCoor) (real)

$AC_PATHN	(Path parameter normalized) (real) Normalized path parameter: 0 for beginning of block to 1 for end of block
$AA_LOAD[<axial expression>]	Drive utilization
$AA_POWER[<axial expression>]	Drive efficiency in W
$AA_TORQUE[<axial expression>]	Drive torque setpoint in Nm
$AA_CURR[<axial expression>]	Actual current value of axis
$AC_MARKER[<arithmetic_expression>] (int)	Flag variables: Can be used to set up complex conditions in synchronized actions: 8 bit memories (Index 0 - 7) are available. Reset sets the bit memories to 0. Example: WHEN .....DO $AC_MARKER[0]=2 WHEN .....DO $AC_MARKER[0]=3 WHEN $AC_MARKER[0]==3 DO $AC_OVR=50 Can be read and written independently of synchronous actions in the parts program: IF $AC_MARKER == 4 GOTOF SPRUNG
$AC_PARAM[<arithmetic_expression>] (real)	Floating-decimal parameter for synchronous actions. Serves as intermediate saving and evaluation in synchronous actions. 50 Parameters (Index 0 - 49) are available.
$AA_OSCILL_REVERSE_POS1 [<axial expression>] (real)	
$AA_OSCILL_REVERSE_POS2 [<axial expression>] (real)	current oscillation reversal points 1 and 2: The current setting data from $SA_OSCILL_REVERSE_POS1 or $SA_OSCILL_REVERSE_POS2 is read. This enables setting data changes at the reversal positions during active oscillation, i.e. during an active synchronous action.

Conditions

Conditions for motion-synchronous actions are formulated:

Main run variable comparison operator expression

For details, see:
Further information
Function Manual Synchronized Actions

R3: Extended stop and retract 16

16.1 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.

16.2 ESR executed autonomously in the drive

16.2.1 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 axis-specific 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 axis-specific, 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).

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

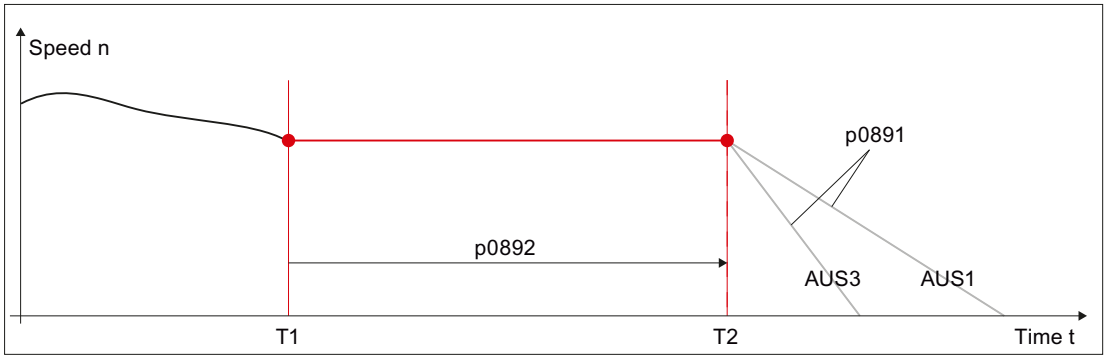
16.2.2 Configuring stopping in the drive

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.



T1 Time at which stopping was triggered

T2 Instant in time after the time that was configured in p0892 has expired

OFF3/OFF1 Braking ramp as a function of p0891

Figure 16-1 Response to drive-integrated stopping

Feedback signal

The stop status is returned to the control (see Section "Feedback of the ESR status (Page 621)").

Further information

For a detailed description of drive parameters, refer to:

SINAMICS S120/S150 List Manual

16.2.3 Configuring retraction in the drive

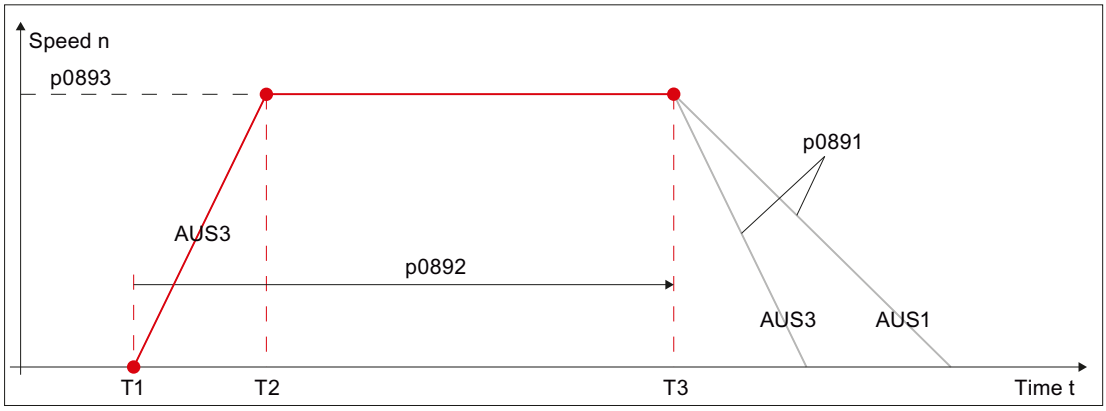
Drive integrated retraction is configured using the following drive parameters:

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) 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 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.



T1 Instant in time that retraction was initiated

T2 Instant in time when the retraction speed specified in p0893 is reached

T3 Instant in time after the time that was configured in p0892 has expired

Figure 16-2 Behavior for drive-integrated retraction

Feedback signal

The retract status is returned to the control (see Section "Feedback of the ESR status (Page 621)").

Further information

For a detailed description of drive parameters, refer to:

SINAMICS S120/S150 List Manual

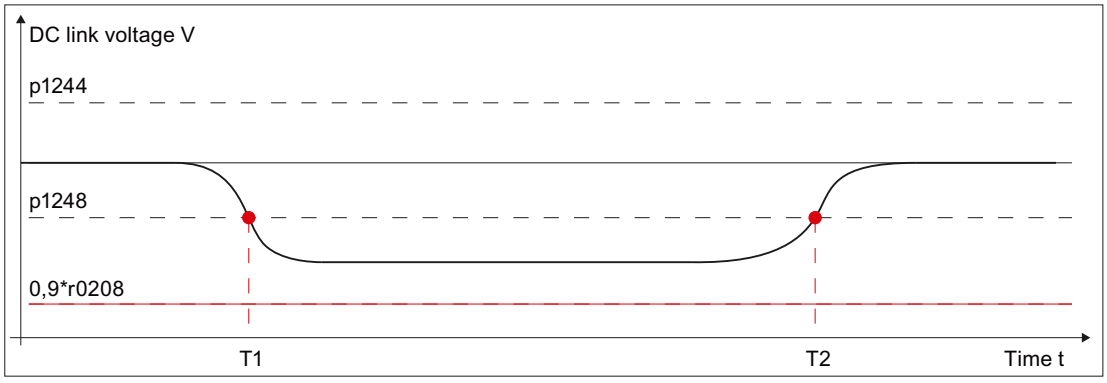
16.2.4 Configuring generator operation in the drive

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.



p1244 Upper DC link voltage threshold

p1248 Lower DC link voltage threshold

r0208 Rated power unit line supply voltage.
The pulses are canceled if the DC link voltage drops below 0.9*r0208

T1 Instant in time at which generator operation was triggered

T2 Instant in time at which generator operation was exited

Figure 16-3 Configuring_drive_integrated_generator_operation

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 for the message: |n_act| < speed threshold 3

Feedback signal

The generator operation status is returned to the control (see Section "Feedback of the ESR status (Page 621)").

Further information

A detailed description of drive parameters and the Vdc control can be found in:

- SINAMICS S120/S150 List Manual
- Function Manual SINAMICS S120 Drive Functions; Servo control > Vdc control

16.2.5 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:

$AA_ESR_ENABLE[<axis>]	
Value	**Meaning**
1	ESR responses enabled in the drive
0	ESR responses inhibited in the drive

16.2.6 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.

Further information

For a detailed description of drive parameters, refer to:

SINAMICS S120/S150 List Manual

16.2.7 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 621)") or to acknowledge the ESR reactions triggered in the drive (see Section "Acknowledge ESR reactions (Page 622)").

Interrelationship between signals

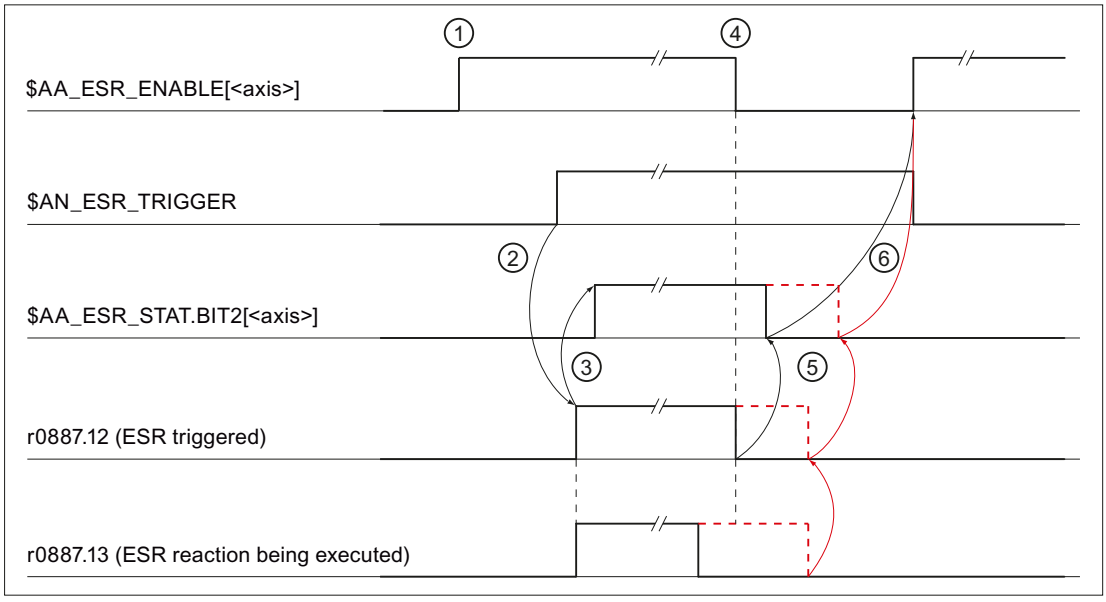
MELDW	System variable	NC/PLC interface signal	Meaning
MELDW.Bit 9 (r0887.12)	$AA_ESR_STAT.Bit 2	DB31, … DBX95.2	ESR: Reaction triggered or generator operation active (r0887.12)
MELDW.Bit 4 (r0056.15)	$AA_ESR_STAT.Bit 3	DB31, … DBX95.1	ESR: DC link undervoltage (p1248)
MELDW.Bit 2 (r2199.0)	$AA_ESR_STAT.Bit 4	DB31, … DBX95.3	ESR, generator operation: Speed lower than minimum (p2161)

16.2.8 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.



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 16-4 Sequence: Trigger and acknowledge drive-integrated ESR reactions

16.2.9 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.

16.2.9.1 Stopping (ESRS)

Syntax

```
ESRS(<access_1>,<stopping time_1>[,...,<axis_n>,<stopping time_n>])
```

Meaning

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 name
<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

16.2.9.2 Retraction (ESRR)

Syntax

```
ESRR(<axis_1>,<retraction distance_1>,<retraction
velocity_1>[,...,<axis_n>,<retraction distance_n>,<retraction
velocity_n>])
```

Meaning

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 name
<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 16-3 Configuring_drive_integrated_generator_operation (Page 620)").

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.

Note

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.

See also

Configuring generator operation in the drive (Page 619)

16.2.9.3 Boundary conditions

Description

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 `ESRS(...)` 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 power-up (POWER ON)

If the drive-autonomous ESR should already be active after the control has powered 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.

Further information

For detailed information about event-controlled program calls, refer to:

Function Manual Basic Functions; Mode group, channel, program mode, reset response, event-driven program calls

16.3 Boundary conditions

Transformation

If NC-controlled extended stopping and retraction is activated for **an**axis involved in an active transformation, then extended stopping and retraction must also be activated for **all other** axes involved in the transformation. Otherwise, when activating extended stopping and retraction, it is possible that the transformation group will ungrouped.

Drive components

In order that the configured stopping or retraction responses can be executed, the drive components involved must be fully functional. Axis-specific servo or drive alarms, which indicate the failure of one of these components, are also implicitly signaling that the configured stopping or retraction response of the axis involved is no longer available or only partially.

Motion-synchronous actions

Motion-synchronous actions are executed in the interpolator clock cycle. Increasing the interpolator clock cycle, e.g. due to a high number of active motion-synchronous actions, results in a coarser time grid for evaluating trigger conditions and triggering responses for extended stopping and retraction.

16.4 Examples

16.4.1 NC-controlled reactions

Task

The A axis is to operate as the generator drive, while the X axis should retract 10 mm at maximum velocity in the event of a fault. The Y and Z axes should stop after a delay of 100 ms so that the X retraction axis has sufficient time to cancel the mechanical coupling.

The example is performed using NC-controlled reactions.

Only the essential parts of the entire functionality are displayed.

Requirements

The following functions must be available:

- Extended stopping and retraction (ESR)
- Static synchronized actions
- Asynchronous subprograms (ASUBs)

Settings

Machine data

Axis-specific operating mode of the ESR

- MD37500 $MA_ESR_REACTION[X] = 21 (NC-controlled retraction axis)
- MD37500 $MA_ESR_REACTION[Y] = 22 (NC-controlled stop axis)
- MD37500 $MA_ESR_REACTION[Z] = 22 (NC-controlled stop axis)

Configuration: NC-controlled extended stop

- MD21380 $MC_ESR_DELAY_TIME1 = 0.1 (duration of the path interpolation in seconds)
- MD21381 $MC_ESR_DELAY_TIME2 = 0.04 (braking duration in seconds)

Programming

Program code	Comment
LFPOS	Axis-specific lift to a position
POLF[X]=IC(10)	Retraction position, incremental
POLFMASK(X)	Enable retraction
; Enable "Extended stop and retract"	
$AA_ESR_ENABLE[X]=1	; X axis
$AA_ESR_ENABLE[Y]=1	; Y axis
$AA_ESR_ENABLE[Z]=1	; Z axis

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
```

16.4.2 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 LFOF	; Fast retraction for thread cutting OFF

Program code	Comment
N130 POLFMASK()	; Disable retraction of all axes

16.4.3 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)	; Disable retraction, axis X Y and
	; Enable retraction, axis Z
N90 Y10	; Retraction response, axial: Z (abs.)
N100 POLFMASK()	; Disable retraction of all axes

16.4.4 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

16.4.5 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
```

16.5 Data lists

16.5.1 Machine data

16.5.1.1 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

16.5.1.2 Axis/spindlespecific machine data

Number	Identifier: $MA_	Description
37500	ESR_REACTION	Reaction definition with extended stop and retract

16.5.2 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

H1: Manual traversing 17

17.1 Overview

Application

Even on modern, numerically controlled machine tools, a facility must be provided that allows the user to traverse the axes manually.

Examples:

- Setting up the machine
  Especially when setting up a new machining program, a manual traversing of the machine axes is required. Where coordinate offset or rotation is selected, manual travel can even be performed in the transformed workpiece coordinate system.
- Retraction of the tool after a program abort
  After a program abort due to a power failure or RESET, the operator must manually retract the tool from its current machining position. The transformations and coordinate systems used for machining must remain active while this is done.

JOG mode

Manual traversing of axes is performed in JOG mode.

If JOG mode is active for the current mode group, this is reported to the PLC via the corresponding interface signal:

DB11 DBX6.2 (BAG1: operating mode JOG)

DB11 DBX26.2 (BAG2: operating mode JOG)

...

DB11 DBX186.2 (BAG10: operating mode JOG)

Further information:
Function Manual Basic Functions; Mode group, channel, program operation, reset response

Display

If JOG mode is active, the basic JOG screen is displayed on the operator interface. This main screen contains values relating to position, feedrate, spindle, and tool.

Manual traversing in the BCS and WCS

The user has the option of traversing axes in the basic coordinate system (BCS) or workpiece coordinate system (WCS).

- Manual traversing in the BCS
  Each axis can be traversed manually.
- Manual traversing in the WCS
  Only the axes declared as geometry axes can be manually traversed in the WCS (channel-specific). Any coordinate offsets or rotations that have been selected remain active.

Manual traversing of several axes

Several axes can be manually traversed at the same time in JOG mode. However, there is no interpolatory relationship.

Control via the PLC interface

Manual traversing is controlled via the PLC interface.

See Section "Control via the PLC interface (Page 636)".

A separate PLC interface exists in each case for machine axes (axis-specific), geometry axes (channel-specific), and orientation axes (channel-specific).

Parameter assignment (general)

For the manual traversing of axes, there are parameterizable properties, which are valid for all JOG movements regardless of the traversing variant and the selected JOG function. These are described under "Parameter assignment (general) (Page 637)".

Traversing variants

The following manual traversing variants are possible in JOG mode:

- Continuous traversing with the traversing keys of the machine control panel in jog or continuous mode
  See Section "Continuous manual traversing (Page 643)".
- Incremental traversing with the traversing keys of the machine control panel in jog or continuous mode
  See Section "Incremental manual traversing (Page 646)".
- Traversing of axes via electronic handwheels
  See Section "Manual traversing using a handwheel (Page 651)".

Manual traversing of the spindle

Spindles can also be traversed manually in JOG mode. Essentially, the same conditions apply as for manual traversing of axes. Spindles can be traversed in JOG mode using the traversing keys continuously or incrementally, in jog or continuous mode, or using the handwheel. The function is selected and activated via the axis-/spindle-specific PLC interface in the same way as for the machine axes. The axis-specific machine data also applies to the spindles.

Special features to be observed during manual traversing of spindles can be found in Section "Manual traversing of the spindle (Page 667)".

Manual traversing of geometry axes

Manual traversing of geometry axes is used for traversing for which transformations and frames have to be active. The geometry axes are traversed in the most recently valid coordinate system.

Special features to be observed during manual traversing of geometry axes (and orientation axes) can be found in Section "Manual traversing of geometry axes/orientation axes (Page 668)".

Additional functions

In addition to various traversing variants, further functions can be selected in JOG mode:

- REF (reference point approach)
- REPOS (repositioning)
- Approach of configured fixed points
  See Section "Approaching a fixed point in JOG (Page 671)".
- Approach of a definable position
  See Section "Position travel in JOG (Page 678)".
- Traverse geometry axes along a circular arc
  See Section "Circular travel in JOG (Page 683)".
- JOG retract (retraction movement in the tool direction after a program abort due to a power failure or RESET)
  See Section "Retraction in the tool direction (JOG retract) (Page 693)".

Use of handwheels in automatic mode

Handwheels can also be used in automatic mode:

- Handwheel override in AUTOMATIC mode with path specification and/or velocity override (Page 703)
- Contour handwheel/path input using handwheel (option) (Page 710)
- Correction of the tool wear for machining in AUTOMATIC mode by means of additional incremental work offset using the handwheel (DRF) (Page 703)

Monitoring

For manual traversing, limitations apply, which are monitored in the controller.

See Section "Monitoring functions (Page 723)".

Switching from JOG mode to AUTOMATIC/MDI mode

It is only possible to switch operating modes from JOG to AUTOMATIC or MDI if all axes in the channel have reached "coarse exact stop".

Further information:
Function Manual Basic Functions; Mode group, channel, program operation, reset response

Commissioning

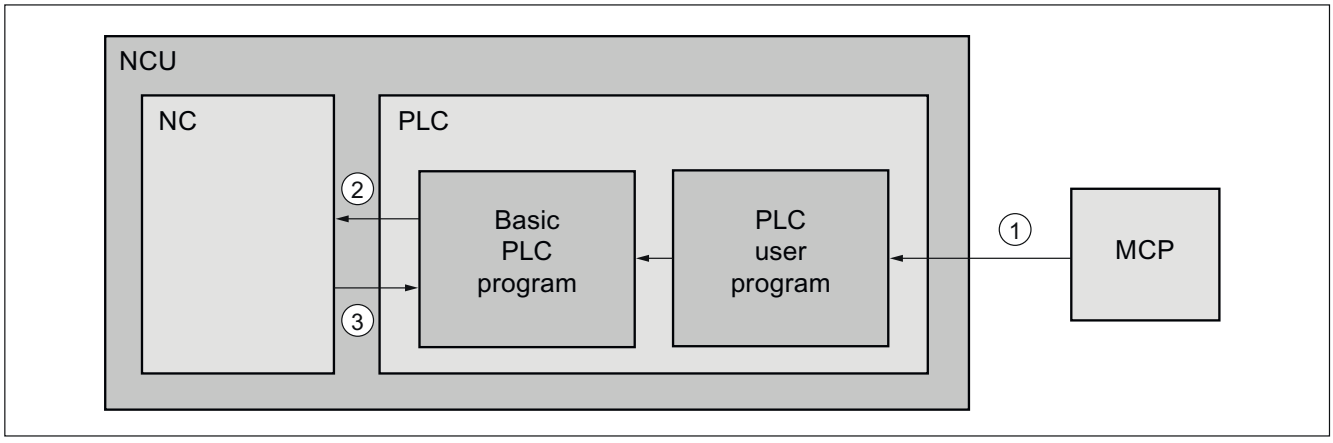
For information on the commissioning of electronic handwheels, see Section "Start-up: Handwheels (Page 724)".

17.2 Control via the PLC interface

The JOG functions in the NC are activated via the PLC interface from the machine control panel (MCP). To do this, the input signals of the machine control panel from the PLC must be transferred to the input interface of the NC in the NC/PLC interface. In this way, the machine manufacturer can simply adapt the functionality of manual traversing to the machine tool through the PLC user program. You can use this, for example, to change the assignment between the direction keys of the machine control panel and the traversing requests to the NC with regard to the machine and geometry axes.

Basic procedure

The following figure shows the basic procedure for the selection of a JOG function from the machine control panel (MCP) to the NC.



① The operator selects, for example, the "Continuous traversing" machine function on the machine control panel for a machine axis.

The input signals of the MCP are transferred cyclically from the basic PLC program in the data blocks of the MCP input interface.

② The PLC user program reads the input signals of the MCP. The input signals can be linked with any other signals in accordance with the current machine or machining situation. Finally, the PLC user program writes the corresponding request signals to the NC in the respective axis-specific NC/PLC interface.

The basic PLC program transfers the request signals in the internal input interface to the NC.

③ After activation of the requested function, the NC writes the feedback in the internal output interface to the PLC. The basic PLC program cyclically transfers the output signals in the respective axis-specific NC/PLC interface

Further information

Detailed information on the configuration and integration of machine control panels in the PLC user program can be found in the Function Manual PLC.

See also

V1: Feedrates (Page 125)

17.3 Parameter assignment (general)

Feedrate behavior

In JOG mode, it is possible to traverse an axis/spindle manually also at a revolutional feedrate (as for G95) specific to the current speed of the master spindle.

The setting is made via the setting data:

SD41100 $SN_JOG_REV_IS_ACTIVE (JOG: revolutional/linear feedrate)

Bit	Value	Meaning
0	0	The behavior of the axis/spindle depends on the setting data: SD43300 $SA_ASSIGN_FEED_PER_REV_SOURCE (revolutional feedrate for positioning axes/spindles) The behavior of a geometry axis on which a frame with rotation acts, or of an orientation axis, depends on the channel-specific setting data: SD42600 $SC_JOG_FEED_PER_REV_SOURCE (control of the revolutional feedrate in JOG)
	1	For a JOG movement, it is always traversed with revolutional feedrate as a function of the master spindle.
1	0	For rapid traverse as well, the axis/spindle, geometry axis or orientation axis is traversed with revolutional feedrate (selection see Bit 0).
	1	For rapid traverse, the axis/spindle, geometry axis or orientation axis is always traversed without revolutional feedrate.
2	0	For manual traversing using the handwheel in the JOG mode, the axis/spindle, geometry axis or orientation axis is traversed with the revolutional feedrate (selection see bit 0).
	1	For manual traversing using the handwheel in the JOG mode, the axis/spindle, geometry axis or orientation axis is always traversed with the revolutional feedrate.
3	0	For manual traversing using the handwheel in the DRF mode, the axis/spindle is traversed with the revolutional feedrate (selection see bit 0).
	1	For manual traversing using the handwheel in the DRF mode, the axis/spindle is always traversed without the revolutional feedrate.

Note

If a master spindle has not been defined and the axis is to be manually traversed at a revolutional feedrate, alarm 20055 is output or alarm 20065 for geometry axes.

Velocity

The system data that determines the velocity of the manual traversing depends on the active feedrate behavior.

Linear feedrate (G94) active

- Machine axes
  The velocity is determined by the following setting data:
  - For linear axes:
    SD41110 $SN_JOG_SET_VELO (axis velocity for JOG)
  - For rotary axes:
    SD41130 $SN_JOG_ROT_AX_SET_VELO (JOG velocity for rotary axes)

  If "0" is entered in SD41110/SD41130, the axis-specific machine data takes effect:
  MD32020 $MA_JOG_VELO (conventional axis velocity)
  The default setting for feedrate velocity unit is mm/min or rpm for rotary axes.
- Spindles
  The velocity applied for the manual traversing of the spindles is defined by the setting data:
  SD41200 $SN_JOG_SPIND_SET_VELO (speed of spindle in JOG mode)
  If "0" is entered in SD41200, the axis-specific machine data takes effect:
  MD32020 $_MA_JOG_VELO (conventional axis velocity)
  The maximum speeds of the active gear stage are considered when manually traversing a spindle.
  Further information
  S1: Spindles (Page 733)
- Geometry axes
  The velocity applied for the manual traversing of geometry axes is defined by the channel-specific machine data:
  MD21165 $MC_JOG_VELO_GEO (conventional velocity for geometry axes)
- Orientation axes
  The velocity applied for the manual traversing of orientation axes is defined by the channel-specific machine data:
  MD21155 $MC_JOG_VELO_ORI (conventional velocity for orientation axes)

Revolutional feedrate (G95) active

The feedrate value (in mm/rev.) is defined by the following setting data:

SD41120 $SN_JOG_REV_SET_VELO (revolutional feedrate of axes in JOG)

If "0" is entered in SD41120, the axis-specific machine data takes effect:

MD32050 $MD_JOG_REV_VELO (revolutional feedrate for JOG)

Rapid traverse override

If the traversing keys/handwheel are also actuated together with the rapid traverse override key, the movement will be made with the configured rapid traverse velocity:

- Machine axes
  - MD32010 $MA_JOG_VELO_RAPID (rapid traverse in jog mode)
  - MD32040 $MA_JOG_REV_VELO_RAPID (revolutional feedrate for JOG with rapid traverse override)
- Geometry axes
  - MD21160 $MC_JOG_VELO_RAPID_GEO (conventional rapid traverse for geometry axes)
- Orientation axes
  - MD21150 $MC_JOG_VELO_RAPID_ORI (conventional rapid traverse for orientation axes)

Feedrate override

The velocity applied for manual traversing can also be influenced using the axis-specific feedrate override switch provided that the following NC/PLC interface signal is set:

DB31, ... DBX1.7 (axis-specific feedrate override active)

Percentages are assigned to the individual feedrate override switch positions via machine data. For a switch position of 0 % the axis is not traversed, provided that 0 is entered in the associated machine data.

The interface signal DB31, ... DBX1.7 (axis-specific feedrate override active) has no meaning for switch position 0 %.

Instead of being set by the feedrate override switch position (gray code), the percentage value (0 % to 200 %) can optionally be set directly by the PLC. Again, the selection is made via machine data.

Further information
V1: Feedrates (Page 125)

Spindle override

The velocity during manual traversing of the spindle can be influenced by the spindle override switch.

Further information
S1: Spindles (Page 733)

Acceleration and jerk

The axis-specific acceleration and jerk limitation values also take effect in JOG mode.

It is also possible to limit acceleration and jerk channel-specifically for manual traversing of geometry and orientation axes. This enables better handling of the kinematics that generate Cartesian motions entirely via rotary axes (robots).

Maximum axis-specific acceleration for JOG motion

The maximum acceleration for manual traversing of an axis is defined in machine data:

MD32300 $MA_MAX_AX_ACCEL [0] (maximum axis-specific acceleration for path motions in the dynamic response mode DYNNORM)

Note

Only the dynamic response mode DYNNORM is always effective for JOG mode.

Basic setting of axis-specific jerk limitation

Acceleration with jerk limitation can be specified as the axis-specific basic setting for JOG mode:

MD32420 $MA_JOG_AND_POS_JERK_ENABLE== TRUE

Maximum axis-specific jerk

For JOG motions with active jerk limitation, the value from the following machine data takes effect as maximum axis-specific jerk:

MD32430 $MA_JOG_AND_POS_MAX_JERK (maximum axis-specific jerk)

Note

MD32430 $MA_JOG_AND_POS_MAX_JERK acts only when the axis-specific jerk limitation in JOG mode has been enabled for the machine axes to be traversed:

MD32420 $MA_JOG_AND_POS_JERK_ENABLE [<axis>] == TRUE

This is also possible by programming `SOFTA(<Axis1>,<Axis2>, ...)` in the part program.

Maximum acceleration when manually traversing spindles

As a spindle often uses many gear stages in speed-control mode and position-control mode, the acceleration associated with the current gear stage is always applied to the spindle in JOG mode.

Further information
S1: Spindles (Page 733)

Maximum acceleration when manually traversing geometry axes

The maximum acceleration when manually traversing geometry axes can be specified for each channel via the machine data:

MD21166 $MC_JOG_ACCEL_GEO [<geometry axis>]

With <geometry axis> = 0, 1, 2

Note

With MD21166 $MC_JOG_ACCEL_GEO [<geometry axis>], there is no direct limitation to MD32300 $MA_MAX_AX_ACCEL.

Note

When a transformation is active, MD32300 $MA_MAX_AX_ACCEL determines the maximum possible axis-specific acceleration.

Maximum jerk when manually traversing geometry axes

The maximum jerk when manually traversing geometry axes in the SOFT acceleration mode (acceleration with jerk limitation) can be specified for each channel via the machine data:

MD21168 $MC_JOG_JERK_GEO [<geometry axis>]

With <geometry axis> = 0, 1, 2

Note

MD21168 $MC_JOG_JERK_GEO acts only when the axis-specific jerk limitation in JOG mode has been enabled for the base machine axes:

MD32420 $MA_JOG_AND_POS_JERK_ENABLE [<axis>] == TRUE

With MD21168 = 0, instead of the channel-specific jerk limitation, the axis-specific limit value from MD32430 $MA_JOG_AND_POS_MAX_JERK is effective.

Maximum jerk when manually traversing orientation axes

The maximum jerk when manually traversing orientation axes can be specified for each channel via the machine data:

MD21158 $MC_JOG_JERK_ORI [<orientation axis>]

For MD21158 to take effect, the channel-specific jerk limitation for the manual traversing of orientation axes must be enabled via the following machine data:

MD21159 $MC_JOG_JERK_ORI_ENABLE == TRUE

See also

Manual traversing of geometry axes/orientation axes (Page 668)

Manual traversing of the spindle (Page 667)

17.4 Continuous manual traversing

17.4.1 Function

For continuous manual traversing, the plus and minus traversing keys are selected to move the relevant axis in the appropriate direction. If both traversing keys are pressed simultaneously, there is no traversing movement, or, if an axis is in motion, it is stopped.

Note

When the control is switched on, axes can be traversed to the limits of the machine because they have not yet been referenced. Hardware limit switches might be triggered as a result. The software limit switches and the working-area limitation are not operative.

Continuous traversing in jog mode

In jog mode, the axis traverses as long as the traversing key is pressed. With the release of the traversing key, the axis is decelerated to standstill. The movement is completed when the parameterized exact stop criterion of the axis is reached.

If the axis reaches a traversing range limit before the traversing key is released (working area limitation, software limit switch, etc.), the axis is stopped at the limit.

Continuous traversing in continuous mode

In continuous mode, traversing of the axis is started by pressing the traversing key. The traversing movement is continued even after the traversing key is released.

⚠ WARNING

Risk of collision

In continuous mode, several axes can be started by pressing the respective traversing key. Any interlocks must be implemented by the user / machine manufacturer via the PLC user program.

The traversing movement can be interrupted and continued by the operator at any time or aborted. If the axis reaches a traversing range limit before canceling the traversing movement (working area limitation, software limit switch, etc.), the axis is stopped at the limit.

Interrupt traversing movement

The operator can interrupt traversing via the operator controls of the machine control panel (MCP) in the following ways:

- Feedrate override = 0%
- Feedrate stop
- NC stop or NC stop axis/spindle

If the cause of the interruption is removed, the axis continues to traverse.

Abort traversing movement

The operator can cancel traversing via the operator controls of the machine control panel (MCP) in the following ways:

- Pressing the same traversing key again
- Pressing the traversing key for the opposite direction
- RESET
- Deselection of the JOG mode by changing the operating mode to AUTOMATIC or MDI

The traversing movement is canceled by the controller:

- If an active traversing range limit is reached (working area limitation, software limit switch, etc.)

  ⚠ CAUTION

  Traversing range limit inactive

  Software limit switches and working-area limitations are only active after referencing of the axis.

- If an alarm occurs with cancellation of the traversing movements

Interface signals

Selection

Continuous traversing in JOG mode must be activated via the PLC interface:

- Machine axes:
  - DB31, ... DBX5.6 (request machine function "Continuous manual traversing")
- Geometry axes:
  - DB21, ... DBX13.6 (geometry axis 1: Request machine function "Continuous manual traversing")
  - DB21, … DBX17.6 (geometry axis 2: Request machine function "Continuous manual traversing")
  - DB21, … DBX21.6 (geometry axis 3: Request machine function "Continuous manual traversing")
- Orientation axes:
  - DB21, ... DBX321.6 (orientation axis 1: Request machine function "Continuous manual traversing")
  - DB21, … DBX325.6 (orientation axis 2: Request machine function "Continuous manual traversing")
  - DB21, … DBX329.6 (orientation axis 3: Request machine function "Continuous manual traversing")

Feedback

Once the continuous procedure takes effect, a feedback is sent to the PLC:

- Machine axes:
  - DB31, ... DBX65.6 (active machine function "Continuous manual traversing")
- Geometry axes:
  - DB21, ... DBX41.6 (geometry axis 1: Active machine function "Continuous manual traversing")
  - DB21, … DBX47.6 (geometry axis 2: Active machine function "Continuous manual traversing")
  - DB21, … DBX53.6 (geometry axis 3: Active machine function "Continuous manual traversing")
- Orientation axes:
  - DB21, ... DBX333.6 (orientation axis 1: Active machine function "Continuous manual traversing")
  - DB21, … DBX337.6 (orientation axis 2: Active machine function "Continuous manual traversing")
  - DB21, … DBX341.6 (orientation axis 3: Active machine function "Continuous manual traversing")

Travel command plus/minus

Once a traversing request for an axis is pending, the interface signal "Travel command minus" or "Travel command plus" is output to the PLC depending on the direction of movement:

- Machine axes:
  - DB31, ... DBX64.6 (travel command "minus") or
  - DB31, ... DBX64.7 (travel command "plus")
- Geometry axis 1:
  - DB21, … DBX40.6 (geometry axis 1: Travel command "minus") or
  - DB21, … DBX40.7 (geometry axis 1: Travel command "plus")
- Geometry axis 2:
  - DB21, … DBX46.6 (geometry axis 2: Travel command "minus") or
  - DB21, … DBX46.7 (geometry axis 2: Travel command "plus")
- Geometry axis 3:
  - DB21, … DBX52.6 (geometry axis 3: Travel command "minus") or
  - DB21, … DBX52.7 (geometry axis 3: Travel command "plus")
- Orientation axis 1:
  - DB21, … DBX332.6 (orientation axis 1: Travel command "minus") or
  - DB21, … DBX332.7 (orientation axis 1: Travel command "plus")

- Orientation axis 2:
  - DB21, … DBX336.6 (orientation axis 2: Travel command "minus") or
  - DB21, … DBX336.7 (orientation axis 2: Travel command "plus")
- Orientation axis 3:
  - DB21, … DBX340.6 (orientation axis 3: Travel command "minus") or
  - DB21, … DBX340.7 (orientation axis 3: Travel command "plus")

Note

A travel command set in JOG mode is canceled if the controller enable or pulse enable for the axis to be traversed is missing. The travel command must therefore be set again after release of the axis.

17.4.2 Parameter assignment

Jog or continuous mode

The selection of jog or continuous mode for the continuous procedure is performed NC-specifically for all axes via the setting data:

SD41050 $SN_JOG_CONT_MODE_LEVELTRIGGRD (jog/continuous mode for JOG continuous)

Jogging mode is the default setting.

17.4.3 Supplementary conditions

Indexing axis

An indexing axis always stops at an indexing position both in jog mode and in continuous mode. In jog mode, the indexing axis traverses to the next indexing position in the direction of travel, for example, when the traversing key is released

17.5 Incremental manual traversing

17.5.1 Function

With incremental traversing, the operator specifies the number of increments to be traversed by the axis via the machine control panel.

In addition to five fixed increment sizes (default setting: INC1, INC10, INC100, INC1000 and INC10000), a variable increment size (INCvar) that can be set via the setting data is also available.

Incremental travel in jogging mode

If the traversing key for the required direction (e.g. +) is pressed, the axis begins to traverse the increment that has been set. If the traversing key is released before the increment has been fully traversed, the movement is interrupted and the axis stops. If the same traversing key is pressed again, the axis traverses the remaining distance until it is zero. Up to this point, the movement can still be interrupted by releasing the traversing key.

Pressing the traversing key for the opposite direction does not have any effect, unless the increment has been fully traversed or the movement has been aborted.

Cancel traversing movement

If the increment should not travel to the end, the traversing movement can be interrupted as follows:

- RESET
- DB31, ... DBX2.2 (delete distance-to-go)

Incremental travel in continuous mode

The axis traverses the entire set increment when the traversing key is pressed (first rising edge). If the same traversing key is pressed again (second rising edge) before the axis has finished traversing the increment, the traversing movement is aborted, i.e. not completed.

Interrupt traversing movement

Behavior as for continuous traversing (see "Continuous manual traversing (Page 643)").

Cancel traversing movement

The traversing movement can be stopped and aborted by means of the following operations or monitoring functions:

- Pressing the same traversing key again
- Pressing the traversing key for the opposite direction
- RESET
- DB31, ... DBX2.2 (delete distance-to-go)
- When the first valid limit is reached

⚠ CAUTION
Traversing range limit inactive Software limit switches and working-area limitations are only activated after reference point approach.

- Deselection or change of the current increment (e.g. change from INC100 to INC10)
- When faults occur (e.g. on cancellation of the servo enable)

⚠ WARNING

Risk of collision

If "continuous" mode is selected, several axes can by started by pressing and releasing the relevant direction key. Any interlocks must be implemented via the PLC!

Note

While an axis is moving, a change of mode from JOG to AUTOMATIC or MDI is not permitted within the control.

Interface signals

Selection

Incremental traversing in JOG mode must be activated via the PLC interface:

- Machine axes:
  - DB31, ... DBX5.0 - 5 (request machine function "INC1" to "INCvar")
- Geometry axes:
  - DB21, ... DBX13.0 - 5 (geometry axis 1: Request machine function "INC1" to "INCvar")
  - DB21, ... DBX17.0 - 5 (geometry axis 2: Request machine function "INC1" to "INCvar")
  - DB21, ... DBX21.0 - 5 (geometry axis 3: Request machine function "INC1" to "INCvar")
- Orientation axes:
  - DB21, ... DBX321.0 - 5 (orientation axis 1: Request machine function "INC1" to "INCvar")
  - DB21, ... DBX325.0 - 5 (orientation axis 2: Request machine function "INC1" to "INCvar")
  - DB21, ... DBX329.0 - 5 (orientation axis 3: Request machine function "INC1" to "INCvar")

Feedback

Once the incremental procedure takes effect, a feedback is sent to the PLC:

- Machine axes:
  - DB31, ... DBX65.0 - 5 (active machine function "INC1" to "INCvar")
- Geometry axes:
  - DB21, ... DBX41.0 - 5 (geometry axis 1: Active machine function "INC1" to "INCvar")
  - DB21, ... DBX47.0 - 5 (geometry axis 2: Active machine function "INC1" to "INCvar")
  - DB21, … DBX53.0 - 5 (geometry axis 3: Active machine function "INC1" to "INCvar")
- Orientation axes:
  - DB21, ... DBX333.0 - 5 (orientation axis 1: Active machine function "INC1" to "INCvar")
  - DB21, ... DBX337.0 - 5 (orientation axis 2: Active machine function "INC1" to "INCvar")
  - DB21, ... DBX341.0 - 5 (orientation axis 3: Active machine function "INC1" to "INCvar")

Travel command plus/minus

Once a traversing request for an axis is pending, the interface signal "Travel command minus" or "Travel command plus" is output to the PLC depending on the direction of movement:

- Machine axes:
  - DB31, ... DBX64.6 (travel command "minus") or
  - DB31, ... DBX64.7 (travel command "plus")
- Geometry axis 1:
  - DB21, … DBX40.6 (geometry axis 1: Travel command "minus") or
  - DB21, … DBX40.7 (geometry axis 1: Travel command "plus")
- Geometry axis 2:
  - DB21, … DBX46.6 (geometry axis 2: Travel command "minus") or
  - DB21, … DBX46.7 (geometry axis 2: Travel command "plus")
- Geometry axis 3:
  - DB21, … DBX52.6 (geometry axis 3: Travel command "minus") or
  - DB21, … DBX52.7 (geometry axis 3: Travel command "plus")
- Orientation axis 1:
  - DB21, … DBX332.6 (orientation axis 1: Travel command "minus") or
  - DB21, … DBX332.7 (orientation axis 1: Travel command "plus")

- Orientation axis 2:
  - DB21, … DBX336.6 (orientation axis 2: Travel command "minus") or
  - DB21, … DBX336.7 (orientation axis 2: Travel command "plus")
- Orientation axis 3:
  - DB21, … DBX340.6 (orientation axis 3: Travel command "minus") or
  - DB21, … DBX340.7 (orientation axis 3: Travel command "plus")

Note

A travel command set in JOG mode is canceled if the controller enable or pulse enable for the axis to be traversed is missing. The travel command must therefore be set again after release of the axis.

17.5.2 Parameter assignment

Fixed increments

The parameter assignment of the fixed increment sizes is performed via NC-specific machine data:

MD11330 $MN_JOG_INCR_SIZE_TAB[1 ... 5] = <number of increments 1 ... 5>

Variable increment

The parameter assignment of the variable increment size is performed via NC-specific setting data:

SD41010 $SN_JOG_VAR_INCR_SIZE = <number of increments>

Distance evaluation of one increment

The distance evaluation of one increment for fixed and variable increment sizes is performed via the axis-specific machine data:

MD31090 $MA_JOG_INCR_WEIGHT = <distance>

Note

Reversal of the direction evaluation

The input of a negative value causes a reversal of the direction evaluation of the traversing keys or the handwheel direction of rotation.

Jog or continuous mode

The selection of jog or continuous mode is performed for the incremental procedure NC-specifically for all axes via the machine data:

MD11300 $MN_JOG_INC_MODE_LEVELTRIGGRD (INC and REF in jog mode)

Jogging mode is the default setting.

17.5.3 Supplementary conditions

Indexing axis

Irrespective of the set increment value, an indexing axis traverses to the next indexing position in the direction of travel after pressing the traversing key.

17.6 Manual traversing using a handwheel

17.6.1 Function

The electronic handwheels (accessories) can be used to simultaneously traverse selected axes manually. The weighting of the handwheel graduations is dependent on the increment-size weighting. Where coordinate offset or rotation is selected, manual travel can even be performed in the transformed workpiece coordinate system.

Commissioning

For information on the commissioning of electronic handwheels, see Chapter "Start-up: Handwheels (Page 724)".

Select

JOG mode must be active. The user must also set the increment INC1, INC10, etc., which applies to manual traversing using a handwheel. As with incremental travel, the required machine function must be set at the PLC interface accordingly.

Traversing

When the electronic handwheel is turned, the associated axis is traversed either in the positive or negative direction depending on the direction of rotation.

Note

If the axis is already being moved using the traversing keys, then manual traversing using the handwheel is not possible.

Traversing distance

The traversing distance produced by rotating the handwheel is dependent on the following factors:

- Number of handwheel pulses received at the interface
- Active increment (machine function INC1, INC10, INC100, ... INCvar)
- Pulse evaluation of the handwheel:
  MD11320 $MN_HANDWH_IMP_PER_LATCH (handwheel pulses per detent position)
  See Section "Parameter assignment (Page 658)".
- Distance of an increment:
  MD31090 $MA_JOG_INCR_WEIGHT (evaluation of an increment for INC/handwheel)
  See Section "Parameter assignment (Page 658)".

Handwheel connection

Up to six handwheels can be connected simultaneously. This means that up to six axes can be traversed by handwheel simultaneously.

Representation of the handwheel number in the NC/PLC interface signals

Depending on the parameter assignment of MD11324, the representation of the handwheel number in the NC/PLC interface signals is **bit-coded** (three handwheels can be represented) or **binary-coded** (six handwheels can be represented). From the HMI, the handwheel number is always transferred bit-coded, independent of the setting in MD11324.

See Chapter "Parameter assignment (Page 658)".

Handwheel assignment

The axis that is moved by turning the handwheel can be set via the via the PLC user interface or via the user interface (HMI). The assignment is linked to the NC/PLC interface through the PLC user program. In this way, several axes can be assigned to one handwheel simultaneously.

Setting via the PLC user interface

The assignment is made using one of the following interface signals:

- Machine axes:
  - DB31, ... DBX4.0 - 2 (activate handwheel)
- Geometry axes:
  - DB21, ... DBX12.0 - 2 (geometry axis 1: Activate handwheel)
  - DB21, ... DBX16.0 - 2 (geometry axis 2: Activate handwheel)
  - DB21, ... DBX20.0 - 2 (geometry axis 3: Activate handwheel)
- Orientation axes:
  - DB21, ... DBX320.0 - 2 (orientation axis 1: Activate handwheel)
  - DB21, ... DBX324.0 - 2 (orientation axis 2: Activate handwheel)
  - DB21, ... DBX328.0 - 2 (orientation axis 3: Activate handwheel)

Setting via the user interface (HMI).

Pressing the "Handwheel" softkey in the JOG-mode basic menu displays the "Handwheel" window. Here, every handwheel can be assigned an axis and the handwheel can be enabled or disabled.

Note

The handwheel assignment is **not** possible via the user interface (HMI) for more than three connected handwheels and a binary-coded representation of the handwheel number in the NC/PLC interface signals.

Handwheel selection by HMI

A separate user interface is provided between the HMI and PLC to allow activation of the handwheel from the user interface. This interface supplied by the basic PLC program for handwheels 1, 2 and 3 contains the following information:

- **Axis number** assigned to the handwheel (if, during the handwheel selection, a machine axis was selected):
  DB10 DBX100.0 - 4 (axis number handwheel 1)
  DB10 DBX101.0 - 4 (axis number handwheel 2)
  DB10 DBX102.0 - 4 (axis number handwheel 3)
- **Channel number** assigned to the handwheel (if during the handwheel selection a geometry axis was selected):
  DB10 DBX97.0 - 3 (channel number geometry axis handwheel 1)
  DB10 DBX98.0 - 3 (channel number geometry axis handwheel 2)
  DB10 DBX99.0 - 3 (channel number geometry axis handwheel 3)

- Additional information on the machine or geometry axis:
  DB10 DBX100.7 (machine axis handwheel 1)
  DB10 DBX101.7 (machine axis handwheel 2)
  DB10 DBX102.7 (machine axis handwheel 3)
- The information that the handwheel is enabled or disabled:
  DB10 DBX100.6 (handwheel 1 selected)
  DB10 DBX101.6 (handwheel 2 selected)
  DB10 DBX102.6 (handwheel 3 selected)

For the specified axis, the basic PLC program sets the associated interface signal either to "0" (disable) or to "1" (enable):

- Machine axes:
  - DB31, ... DBX4.0 - 2 (activate handwheel)
- Geometry axes:
  - DB21, ... DBX12.0 - 2 (geometry axis 1: Activate handwheel)
  - DB21, ... DBX16.0 - 2 (geometry axis 2: Activate handwheel)
  - DB21, ... DBX20.0 - 2 (geometry axis 3: Activate handwheel)

Note

Orientation axes can only be activated via the associated PLC user interface signals:

- DB21, ... DBX320.0 - 2 (orientation axis 1: Activate handwheel)
- DB21, ... DBX324.0 - 2 (orientation axis 2: Activate handwheel)
- DB21, ... DBX328.0 - 2 (orientation axis 3: Activate handwheel)

Travel request

The following NC/PLC interface signal informs the PLC that an axis wants to travel or is travelling:

- Machine axes:
  - DB31, ... DBX64.4 (travel request "minus") or
  - DB31, ... DBX64.5 (travel request "plus")
- Geometry axis 1:
  - DB21, … DBX40.4 (geometry axis 1: Travel request "minus") or
  - DB21, … DBX40.5 (geometry axis 1: Travel request "plus")
- Geometry axis 2:
  - DB21, … DBX46.4 (geometry axis 2: Travel request "minus") or
  - DB21, … DBX46.5 (geometry axis 2: Travel request "plus")
- Geometry axis 3:
  - DB21, … DBX52.4 (geometry axis 3: Travel request "minus") or
  - DB21, … DBX52.5 (geometry axis 3: Travel request "plus")

- Orientation axis 1:
  - DB21, … DBX332.4 (orientation axis 1: Travel request "minus") or
  - DB21, … DBX332.5 (orientation axis 1: Travel request "plus")
- Orientation axis 2:
  - DB21, … DBX336.4 (orientation axis 2: Travel request "minus") or
  - DB21, … DBX336.5 (orientation axis 2: Travel request "plus")
- Orientation axis 3:
  - DB21, … DBX340.4 (orientation axis 3: Travel request "minus") or
  - DB21, … DBX340.5 (orientation axis 3: Travel request "plus")

Note

A travel request is the sum of all sub-movements, i.e. the component from couplings and correction values is also taken into account.

For the method of operation of the "Travel request" function, see Section "Travel request (Page 664)".

Travel command

Depending on the setting in machine data MD17900 $MN_VDI_FUNCTION_MASK (see Section "Parameter assignment (Page 658)"), the following interface signal is output to the PC already when a travel request is present or not until the axis moves:

- Machine axes:
  - DB31, ... DBX64.6 (travel command "minus") or
  - DB31, ... DBX64.7 (travel command "plus")
- Geometry axis 1:
  - DB21, … DBX40.6 (geometry axis 1: Travel command "minus") or
  - DB21, … DBX40.7 (geometry axis 1: Travel command "plus")
- Geometry axis 2:
  - DB21, … DBX46.6 (geometry axis 2: Travel command "minus") or
  - DB21, … DBX46.7 (geometry axis 2: Travel command "plus")
- Geometry axis 3:
  - DB21, … DBX52.6 (geometry axis 3: Travel command "minus") or
  - DB21, … DBX52.7 (geometry axis 3: Travel command "plus")
- Orientation axis 1:
  - DB21, … DBX332.6 (orientation axis 1: Travel command "minus") or
  - DB21, … DBX332.7 (orientation axis 1: Travel command "plus")

- Orientation axis 2:
  - DB21, … DBX336.6 (orientation axis 2: Travel command "minus") or
  - DB21, … DBX336.7 (orientation axis 2: Travel command "plus")
- Orientation axis 3:
  - DB21, … DBX340.6 (orientation axis 3: Travel command "minus") or
  - DB21, … DBX340.7 (orientation axis 3: Travel command "plus")

Note

A travel command set in JOG mode is canceled if the controller enable or pulse enable for the axis to be traversed is missing. The travel command must therefore be set again after release of the axis.

Invert handwheel direction of rotation

The handwheel direction of rotation can be inverted, if the direction of movement of the handwheel does not match the expected direction of motion of the axis. The adaptation can be especially necessary, if a handwheel (HT2, HT8) can be assigned to various axes.

In addition to configuring the particular MD, handwheel direction of rotation inversion can be activated by setting the IS "Invert the handwheel direction of rotation" belonging to the particular axis:

- Machine axes:
  - DB31, ... DBX7.0 (invert handwheel direction of rotation)
- Geometry axes:
  - DB21, ... DBX15.0 (geometry axis 1: Invert handwheel direction of rotation)
  - DB21, ... DBX19.0 (geometry axis 2: Invert handwheel direction of rotation)
  - DB21, ... DBX23.0 (geometry axis 3: Invert handwheel direction of rotation)
- Orientation axes:
  - DB21, … DBX323.0 (orientation axis 1: Invert handwheel direction of rotation)
  - DB21, … DBX327.0 (orientation axis 2: Invert handwheel direction of rotation)
  - DB21, … DBX331.0 (orientation axis 3: Invert handwheel direction of rotation)
- Contour handwheel:
  - DB21, ... DBX31.5 (contour handwheel: Invert handwheel direction of rotation)

Note

The inversion signal should be set in the PLC user program at the same time as the handwheel selection (IS "Activate handwheel").

The acknowledgement that the handwheel direction of rotation has been inverted by the NC is realized for each axis using the IS "Handwheel direction of rotation inversion active":

- Machine axes:
  - DB31, ... DBX67.0 (handwheel direction of rotation inversion active)
- Geometry axes:
  - DB21, ... DBX43.0 (geometry axis 1: Handwheel direction of rotation inversion active)
  - DB21, ... DBX49.0 (geometry axis 2: Handwheel direction of rotation inversion active)
  - DB21, ... DBX55.0 (geometry axis 3: Handwheel direction of rotation inversion active)
- Orientation axes:
  - DB21, ... DBX335.0 (orientation axis 1: Handwheel direction of rotation inversion active)
  - DB21, ... DBX339.0 (orientation axis 2: Handwheel direction of rotation inversion active)
  - DB21, ... DBX343.0 (orientation axis 3: Handwheel direction of rotation inversion active)
- Contour handwheel:
  - DB21, ... DBX39.5 (contour handwheel: Handwheel direction of rotation inversion active)

Note

It is only permissible to change the inversion signal at standstill. If the change is made while motion setpoints are being output by the interpolator, then the signal change is rejected and an alarm is output; further, motion is stopped taking into account the actual acceleration value.

Abort/interruption of the traversing motion

The following NC/PLC interface signals abort the traversing motion. The setpoint/actual-value difference is deleted.

NC/PLC interface signal	Scope	Effect	
DB21, ... DBX7.7 (NC reset)	Geometry axis / machine axis	Abort	
DB21, ... DBX12.0 - 2 (geometry axis 1: Activate handwheel) DB21, ... DBX16.0 - 2 (geometry axis 2: Activate handwheel) DB21, ... DBX20.0 - 2 (geometry axis 3: Activate handwheel)	Geometry axis	1 → 0	Abort
DB21, ... DBX6.2 (delete distance-to-go)	Geometry axis / machine axis	0 → 1	Abort
DB31, ... DBX2.2 (delete distance-to-go / spindle reset)	Geometry axis / machine axis	0 → 1	Abort
DB31, ... DBX4.2 (activate handwheel)	Machine axis	1 → 0	Abort
DB31, ... DBX1.3 (axis/spindle disable)	Geometry axis / machine axis	0 → 1	Abort
DB31, ... DBX1.5 (position measuring system 1)	Geometry axis / machine axis	0 → 1	Abort
DB31, ... DBX1.6 (position measuring system 2)	Geometry axis / machine axis	0 → 1	Abort

For other NC/PLC interface signals (hold signals), the influence of manually traversing using the handwheel (cancellation or interruption of traversing motion) can be set (see Chapter "Parameter assignment (Page 658)").

NC STOP only interrupts the traversing movement. Any setpoint/actual-value difference is retained. The distance-to-go is then traversed using NC START.

See also

Monitoring functions (Page 723)

S1: Spindles (Page 733)

17.6.2 Parameter assignment

Distance or velocity specification

Either the distance or the velocity can be entered via the handwheel:

- Distance specification (default setting)
  The distance specified by the handwheel is traversed and No pulses are lost.
  Limiting the velocity to the maximum permissible value causes the axes to overtravel.
- Velocity specification
  The handwheel only defines the traverse velocity. As soon as the handwheel stops, the axes stop too. Motion is braked immediately if no pulses are supplied from the handwheel in one IPO cycle, thus preventing overtravel by the axes. The handwheel pulses do not supply a path default.

The input mode is set with machine data:

MD11346 $MN_HANDWH_TRUE_DISTANCE (handwheel distance or velocity specification)

Pulse evaluation of the handwheel

The number of pulses that are to be generated per handwheel detent position must be specified for each handwheel:

MD11320 $MN_HANDWH_IMP_PER_LATCH [<n>] (handwheel pulses per detent position)

Note

Input of a negative value results in a reversal of the handwheel direction of rotation.

Increment size

The number of increments to be traversed by the axis per handwheel pulse is determined by the selected increment size.

In addition to five fixed increment sizes (default setting: INC1, INC10, INC100, INC1000 and INC10000), a variable increment size (INCvar) that can be set via the setting data is also available.

Fixed increments

The parameter assignment of the fixed increment sizes is performed via NC-specific machine data:

MD11330 $MN_JOG_INCR_SIZE_TAB[1 ... 5] = <number of increments 1 ... 5>

Variable increment

The parameter assignment of the variable increment size is performed via NC-specific setting data:

SD41010 $SN_JOG_VAR_INCR_SIZE = <number of increments>

Distance evaluation of one increment

The distance evaluation of one increment for fixed and variable increment sizes is performed via the axis-specific machine data:

MD31090 $MA_JOG_INCR_WEIGHT = <distance>

Note

Input of a negative value results in a reversal of the handwheel direction of rotation.

Limitation of the increment size

The machine operator can limit the size of the selected increment:

- For machine axes, using the axis-specific machine data:
  MD32080 $MA_HANDWH_MAX_INCR_SIZE (limiting the selected increment)
- For geometry axes, using the channel-specific machine data:
  MD20620 $MC_HANDWH_GEOAX_MAX_INCR_SIZE (limitation of handwheel increment for geometry axes)
- For orientation axes, using the channel-specific machine data:
  MD20621 $MC_HANDWH_ORIAX_MAX_INCR_SIZE (limitation of handwheel increment for orientation axes)

Representation of the handwheel number in the NC/PLC interface signals

The representation of the handwheel number in the NC/PLC interface signals is defined using machine data:

MD11324 $MN_HANDWH_VDI_REPRESENTATION

Value	Meaning
0	**Bit-coded** representation (**basic setting**) → Three handwheels can be represented.
1	**Binary-coded** representation → Six handwheels can be represented.

Output of the NC/PLC interface signals "travel command plus" / "travel command minus"

The output behavior of the NC/PLC interface signals "travel command plus" / "travel command minus" is specified with the machine data:

MD17900 $MN_VDI_FUNCTION_MASK

Value	Meaning
0	The NC/PLC interface signals "Travel command plus" / "Travel command minus" are already output when a travel request is active (default setting).
1	The NC/PLC interface signals "Travel command plus" / "Travel command minus" are only output when the axis actually moves (i.e. if setpoints are output to the servo).

Velocity

The following axis velocities, effective during JOG, are also used for manually traversing using the handwheel:

- SD41110 $SN_JOG_SET_VELO (axis velocity for JOG)
- SD41130 $SN_JOG_ROT_AX_SET_VELO (axis velocity for rotary axes for JOG mode)
- MD32020 $MA_JOG_VELO (conventional axis velocity)

Note

Because of the limited feedrate, the axis is not able to follow the handwheel rotation synchronously, especially in the case of a large pulse weighting, and therefore overtravels.

Acceleration

As for manual travel (see "Parameter assignment (general) (Page 637) ").

Movement in the opposite direction

The behavior at a reversal of the traversing direction (by turning the handwheel in the opposite direction) can be set in the machine data:

MD11310 $MN_HANDWH_REVERSE (threshold for direction change handwheel)

Value	Meaning
= 0	If the handwheel is moved in the opposite direction, the resulting distance is computed and the calculated end point is approached as fast as possible. If this end point is located before the point where the moving axis can decelerate in the current direction of travel, the unit is decelerated and the end point is approached by moving in the opposite direction. If this is not the case, the newly calculated end point is approached immediately.
> 0	If the handwheel is moved in the opposite direction by at least the number of pulses indicated in the machine data, the axis is decelerated as fast as possible and all pulses received until the end of the interpolation are ignored. This means that another movement takes place only after the axis reaches standstill (setpoint side).

Response at software limit switches, working area limitation

When axes are traversed in JOG mode, they can traverse only up to the first active limitation before the corresponding alarm is output.

Depending on the setting in the machine data:
MD11310 $MN_HANDWH_REVERSE (threshold for direction change, handwheel)
the behavior is then as follows (as long as the axis has still not arrived at the end point from the setpoint side):

- The distance resulting from the handwheel pulses forms a fictitious end point which is used for subsequent calculations.
  If this fictitious end point is, for example, 10 mm behind the limit, these 10 mm must be traversed in the opposite direction before the axis traverses again. If a movement in the opposite direction is to be performed immediately after a limitation is reached, the fictitious distance-to-go can be deleted via delete distance-to-go or deselection of the handwheel assignment.
- All handwheel pulses leading to an end point behind the limitation are ignored. Any movement of the handwheel in the opposite direction leads to an immediate movement in the opposite direction, i.e. away from the limit.

Feedrate behavior

In JOG mode, the feedrate behavior of the axis/spindle also depends on the setting data:

SD41100 $SN_JOG_REV_IS_ACTIVE (JOG: Revolutional/linear feedrate)

SD41100 $SN_JOG_REV_IS_ACTIVE	
Active	An axis/spindle is always traversed with revolutional feedrate MD32050 $MA_JOG_REV_VELO (revolutional feedrate for JOG) or MD32040 $MA_JOG_REV_VELO_RAPID (revolutional feedrate for JOG with rapid traverse override) depending on the master spindle.
Not active	The behavior of the axis/spindle depends on the setting data: SD43300 $SA_ASSIGN_FEED_PER_REV_SOURCE (revolutional feedrate for positioning axes/spindles)
	The behavior of a geometry axis on which a frame with rotation acts, depends on the channel-specific setting data: SD42600 $SC_JOG_FEED_PER_REV_SOURCE (control of the revolutional feedrate in JOG)

Effect of the NC/PLC interface stop signals

The effect of the NC/PLC interface stop signals on manual traversing using a handwheel (abort or interruption of the traversing motion) can be set using machine data:

MD20624 $MC_HANDWH_CHAN_STOP_COND (definition of the handwheel traversing behavior, channel-specific)

MD32084 $MA_HANDWH_STOP_COND (behavior when manually traversing using a handwheel)

NC/PLC interface signal	Scope	MD20624 $MC_HANDWH_CHAN_STOP_COND	
		Bit == 0	Bit == 1
DB11 DBX0.5 (mode group stop)	Geometry axis / machine axis	Interruption until NC start	Abort
DB11 DBX0.6 (mode group stop, axes plus spindle)	Geometry axis / machine axis	Interruption until NC start	Abort
DB21, ... DBX7.3 (NC Stop)	Geometry axis / machine axis	Interruption until NC start	Abort
DB21, ... DBX7.4 (NC stop, axes plus spindle)	Geometry axis / machine axis	Interruption until NC start	Abort
DB21, ... DBX6.0 (feedrate disable)	Geometry axis / machine axis	Interruption	Abort
DB21, ... DBB4 (feed rate override) DB31, ... DBX4.5 (rapid traverse override)	Geometry axes	Override == 0: Interruption if rapid traverse override is not active	Override == 0: Abort if rapid traverse override is not active
DB21, ... DBB5 (rapid traverse override) DB31, ... DBX4.5 (rapid traverse override)	Geometry axes	Rapid traverse override == 0: Interruption if rapid traverse override is active	Rapid traverse override == 0: Abort if rapid traverse override is active
DB21, ... DBX12/16/20.3 (geometry axis 1/2/3: Feedrate stop)	Geometry axis	Interruption	Abort

NC/PLC interface signal	Scope	MD32084 $MC_HANDWH_CHAN_STOP_COND	
		Bit == 0	Bit == 1
DB31, ... DBB0 (feed rate override, axial)	Machine axis, not spindle	Override == 0: Interruption	Override == 0: Abort
DB31, ... DBB19 (spindle override)	Spindle	Override == 0: Interruption	Override == 0: Abort
DB31, ... DBX4.3 (feedrate stop / spindle stop)	Geometry axis / machine axis	Interruption	Abort
DB31, ... DBX2.3 (clamping in progress)	Geometry axis / machine axis	No effect	Abort
DB31, ... DBX2.1 (control system enable)	Geometry axis / machine axis	Interruption	Abort
DB31, ... DBX21.7 (pulse enable)	Geometry axis / machine axis	Interruption	Abort

Note

Removal of the "Controller enable" and "Pulse enable" during traversing results in a rapid stop of the axis and a reset alarm.

Interruption of a traversing motion

When a stop command is issued, the distance-to-go is saved and the handwheel pulses are collected. When the stop condition no longer applies, the resulting distance is traversed.

Abort of a traversing motion

When a stop command is issued, the distance-to-go is deleted and the handwheel pulses are ignored (i.e. not collected) until the stop condition no longer applies.

Behavior in the event of a safe operating stop

Analog to the NC/PLC interface stop signals, the behavior in the event of a stop can also be adjusted by activating the safe operating stop (SBH):

Stop condition	Scope	MD20624 $MC_HANDWH_CHAN_STOP_COND	
		Bit 12 == 0	Bit 12 == 1 (Default setting)
Safe operating stop in the event of active target velocity limit	Machine axis	Interruption	Abort

In the default setting, Bit 12 is set to "1", i.e. in the event of a stop resulting from activation of the safe stop, current movements are aborted and new handwheel pulses ignored.

Prerequisite for an interpolatory stop is an active target velocity limit (i.e., MD36933 $MA_SAFE_DES_VELO_LIMIT > 0).

17.6.3 Travel request

The following examples are intended to illustrate the method of operation of the "Travel request" NC/PLC interface signal.

Example 1: Manual traversing using handwheel with distance input, the stop condition is no criterion for cancellation

If, for manual traversing using a handwheel with distance input (MD11346 $MN_HANDWH_TRUE_DISTANCE == 1 or == 3), an active stop condition is not a criterion for cancellation (see MD32084 $MA_HANDWH_STOP_COND or MD20624 $MC_HANDWH_CHAN_STOP_COND); then, the output of NC/PLC interface signals "Travel request" and "Travel command" corresponds to the behavior shown in the following diagrams.

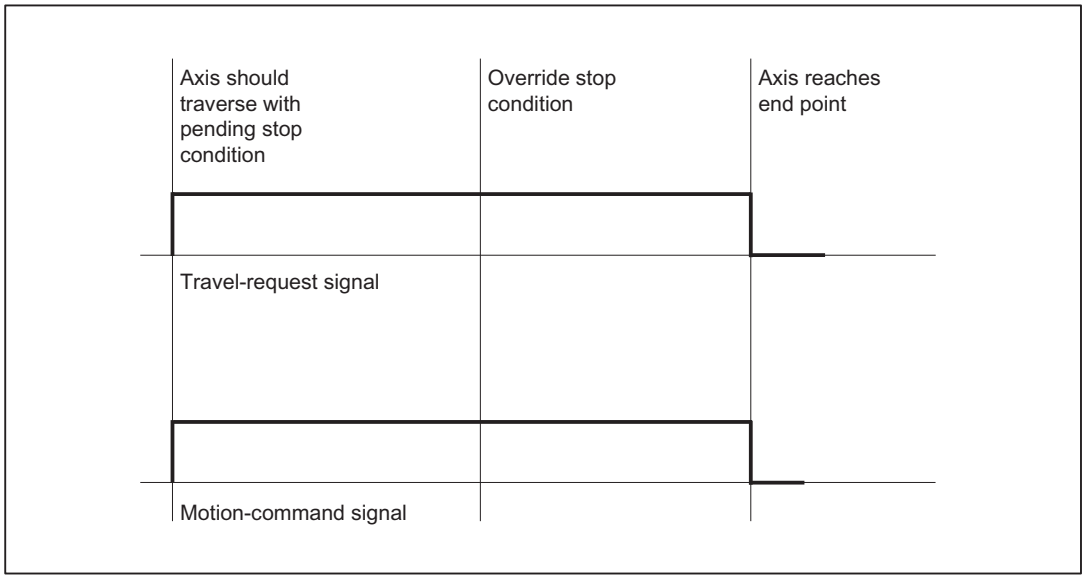


Figure 17-1 Signal-time diagram:Manual traversing using the handwheel with distance input, stop condition is not a cancellation criterionMD17900 $MN_VDI_FUNCTION_MASK bit 0 = 0

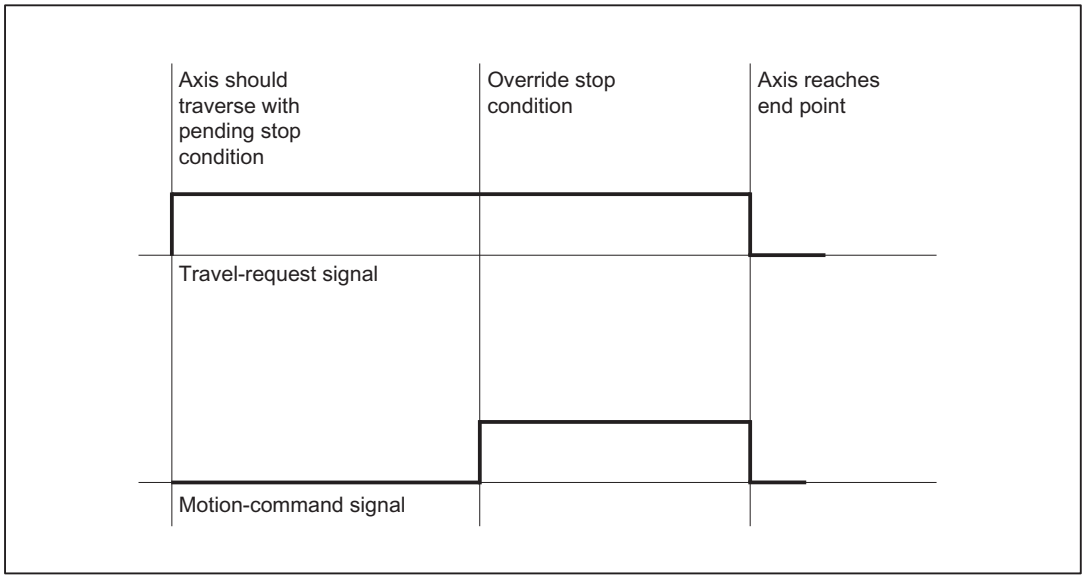


Figure 17-2 Signal-time diagram:Manual traversing using the handwheel with distance input, stop condition is not a cancellation criterionMD17900 $MN_VDI_FUNCTION_MASK bit 0 = 1

Example 2: Manual traversing using a handwheel, stop condition is a cancellation criterion

If a stop condition is selected as cancellation criterion via machine data MD32084 $MA_HANDWH_STOP_COND or MD20624 $MC_HANDWH_CHAN_STOP_COND during manual traversing using a handwheel, **no travel command** is output, **but** the corresponding **travel request** is output.

When the stop condition is removed, the corresponding "Travel request" NC/PLC interface signal is reset, as an abort is present. The stop condition is no longer active, but the axis cannot be traversed as the stop condition has caused an abort.

In addition, either the distance specification (MD11346 $MN_HANDWH_TRUE_DISTANCE == 1 or == 3) is active or the handwheel is moved continuously, i.e. it provides pulses.

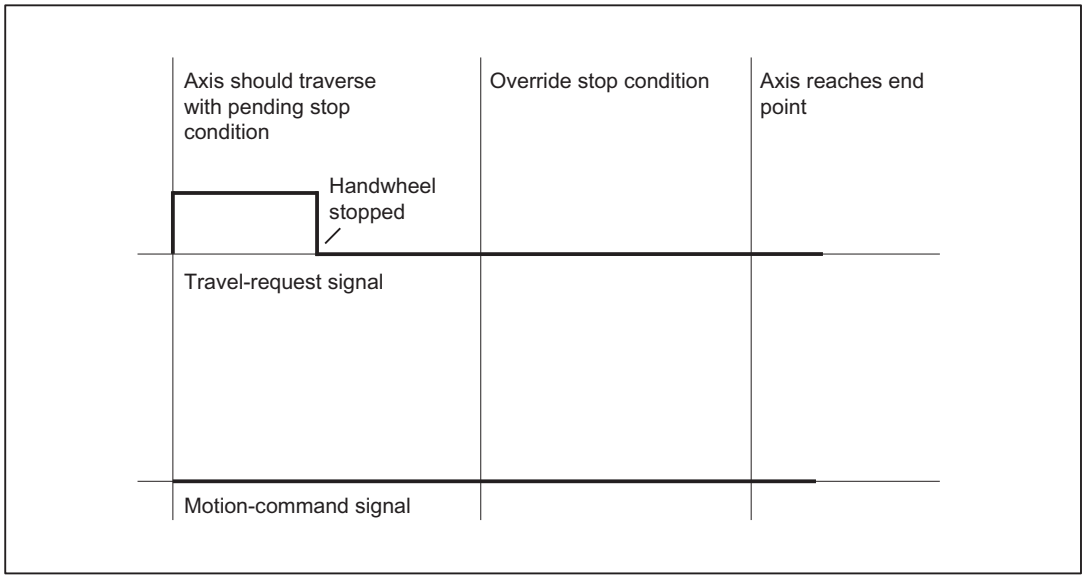


Figure 17-3 Signal-time diagram:Manual traversing using a handwheel, stop condition is a cancellation criterion

If a stop condition is activated during the handwheel travel, the motion is aborted and the "Travel request" and "Travel command" NC/PLC interface signals are reset.

Example 3: Manual traversing using a handwheel with velocity setpoint, stop condition is a cancellation criterion

If the handwheel is no longer moved for velocity specification (MD11346 $MN_HANDWH_TRUE_DISTANCE == 0 or == 2), the "Travel request" NC/PLC interface signal is reset.

The "Travel request" PLC signal is also reset when the handwheel is deselected.

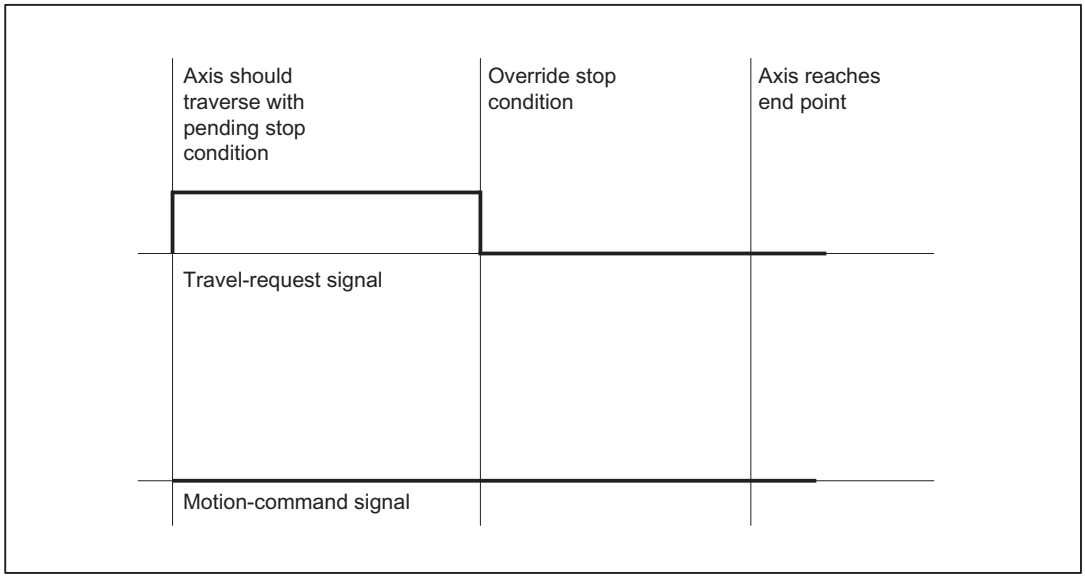


Figure 17-4 Signal-time diagram:Manual traversing using a handwheel with velocity setpoint, stop condition is a cancellation criterion

Supplementary conditions

NC stop

With NC stop present, no travel command and, therefore, no travel request is output. There is an exception with DRF travel: If DRF travel is permitted in the NC stop state via machine data MD20624 $MC_HANDWH_CHAN_STOP_COND (bit 13 == 1), the behavior corresponds to that of manual traversing using a handwheel.

17.7 Manual traversing of the spindle

Spindle mode

Manual traversing of the spindle is possible both in positioning mode (spindle is in position control) and in open-loop control mode.

Velocity

The velocity applied for the manual traversing of the spindles is defined by the general setting data:

SD41200 $SN_JOG_SPIND_SET_VELO (speed of spindle in JOG mode)

If "0" is entered in SD41200, the axis-specific machine data takes effect:

MD32020 $_MA_JOG_VELO (conventional axis velocity)

The maximum speeds of the active gear stage are also considered when manually traversing a spindle.

Further information

S1: Spindles (Page 733)

Spindle override

The velocity during manual traversing of the spindle can be influenced by the spindle override switch.

Acceleration

As a spindle often uses many gear stages in speed-control mode and position-control mode, the acceleration associated with the current gear stage is always applied to the spindle in JOG mode.

Further information

S1: Spindles (Page 733)

Interface signals

When manually traversing the spindles, the interface signals between NC and PLC have the same effect as for the machine axes.

The interface signals DB31, ... DBX60.7 or DBX60.6 (position reached with fine or coarse exact stop) are only set if the spindle is in position control.

Spindle-specific interface signals

In JOG mode, the following interface signals **to** the spindle have no effect:

- DB31, ... DBX17.6 (invert M3/M4)
- DB31, ... DBX18.6 - 7 (oscillation rotation direction, right/left)
- DB31, ... DBX18.5 (oscillation enable)
- DB31, ... DBX16.7 (delete S value)

In JOG mode, the following interface signals **from** the spindle are not set:

- DB31, ... DBX83.7 (clockwise actual direction of rotation)
- DB31, ... DBX83.5 (spindle in setpoint range)

17.8 Manual traversing of geometry axes/orientation axes

Simultaneous travel

Only one geometry axis can be traversed continuously or incrementally at one time using the traversing keys. Where an attempt is made to traverse more than one geometry axis, alarm 20062 "Axis already active" is output. However, three geometry axes can be traversed simultaneously using handwheels 1 to 3.

Transverse axis

If a geometry axis is defined as transverse axis (see MD20100 $MC_DIAMETER_AX_DEF) and radius programming has been selected, the following applies:

- Continuous travel:
  There are no differences when a transverse axis is traversed continuously.
- Incremental travel:
  Only **half the distance** of the selected increment size is traversed. For example, with INC10 the axis only traverses 5 increments when the traversing key is pressed.
- Traversing with the handwheel:
  As for incremental travel, with the handwheel only half the distance is traversed per handwheel pulse.

References:
Function Manual Basic Functions; Transverse Axes (P1)

PLC interface

For geometry axes/orientation axes, there is a separate PLC interface that contains the same signals as the axis-specific PLC interface:

- Geometry axes:
  DB21, ... DBB12 - 23 and
  DB21, ... DBB40 - 56
- Orientation axes:
  DB21, ... DBB320 - 331 and
  DB21, ... DBB332 - 343

Velocity

- Geometry axes
  The velocity applied for the manual traversing of geometry axes is defined by the channel-specific machine data:
  MD21165 $MC_JOG_VELO_GEO (conventional velocity for geometry axes)
- Orientation axes
  The velocity applied for the manual traversing of orientation axes is defined by the channel-specific machine data:
  MD21155 $MC_JOG_VELO_ORI (conventional velocity for orientation axes)

Rapid traverse override

If the traversing keys/handwheel are also actuated together with the rapid traverse override key, the movement will be made with the configured rapid traverse velocity:

- Geometry axes
  - MD21160 $MC_JOG_VELO_RAPID_GEO (conventional rapid traverse for geometry axes)
- Orientation axes
  - MD21150 $MC_JOG_VELO_RAPID_ORI (conventional rapid traverse for orientation axes)

Feedrate override

The channel-specific feedrate override switch is active when manually traversing geometry axes.

Acceleration and jerk

For the manual traversing of geometry axes/orientation axes, the acceleration and jerk can be limited for specific channels. This enables better handling of the kinematics that generate Cartesian motions entirely via rotary axes (robots).

Geometry axes

The maximum acceleration when manually traversing geometry axes can be specified for each channel via the machine data:

MD21166 $MC_JOG_ACCEL_GEO [<geometry axis>]

With <geometry axis> = 0, 1, 2

The maximum jerk when manually traversing geometry axes in the SOFT acceleration mode (acceleration with jerk limitation) can be specified for each channel via the machine data:

MD21168 $MC_JOG_JERK_GEO [<geometry axis>]

With <geometry axis> = 0, 1, 2

Orientation axes

The maximum jerk when manually traversing orientation axes can be specified for each channel via the machine data:

MD21158 $MC_JOG_JERK_ORI [<orientation axis>]

For MD21158 to take effect, the channel-specific jerk limitation for the manual traversing of orientation axes must be enabled via the following machine data:

MD21159 $MC_JOG_JERK_ORI_ENABLE == TRUE

Alarms

Alarm 20060

If the axis is not defined as a geometry axis, alarm 20060, "Axis cannot be traversed as a geometry axis", is output if you attempt to manually traverse it as a geometry axis in JOG mode.

Alarm 20062

Alarm 20062, "Axis already active", is triggered in the case that a geometry axis/orientation axis is manually traversed under the following conditions:

- The axis is already being traversed in JOG mode via the axial PLC interface.
- A frame for a rotated coordinate system is already active and another geometry axis in this coordinate system is traversed in JOG mode with the traversing keys.

17.9 Approaching a fixed point in JOG

17.9.1 Function

The machine user can use the "Approaching fixed point in JOG" function to approach axis positions defined through machine data by actuating the traverse keys of the machine control panel. The traveling axis comes to a standstill automatically on reaching the defined fixed point.

Note

Approaching a fixed point with G75

The process for approaching defined fixed points can also be activated from the part program using the `G75` command.

For further information on fixed point traversing with `G75` see:
Further information
Programming Manual NC Programming; Supplementary commands > Approach fixed point (G75)

Applications

Typical applications are, for example:

- Approaching a basic position before starting an NC program.
- Travel towards tool change points, loading points and pallet change points.

Requirements

- The "Approaching fixed point in JOG" can be activated only in the JOG mode.
  The function cannot be activated if the machine function JOG-REPOS or JOG-REF is active and in JOG in AUTOMATIC.
- The axis to be traversed must be referenced.
- A kinematic transformation may not be active.
- The axis to be traversed may not be a following axis of an active coupling.

Procedure

Procedure in "Approaching fixed point in JOG"

1. Selection of JOG mode
2. Enabling the "Approach fixed point in JOG" function
3. Traversing of the machine axis with traverse keys or handwheel

Activation

After selecting the "Approach fixed point in JOG" function, the PLC outputs the number of the fixed point to be approached binary coded to the NC using the following bits:

DB31, ... DBX13.0 - 2 (JOG approach fixed point)

The NC confirms the activation with the interface signal as soon as the function is effective:

DB31, ... DBX75.0 - 2 (JOG approach fixed point active)

Note

Activation is not possible:

- during an NC reset
- In case of impending emergency stop
- During processing of an ASUP

No alarm message occurs. Delayed activation takes place after closure or after acknowledgement of the active function.

Sequence

The actual traversing is started with the traverse keys or the handwheel in the direction of the approaching fixed point.

The selected machine axis traverses till it comes to an automatic standstill at the fixed point.

The corresponding NC/PLC interface signal is reported on reaching the fixed point with "Exact stop fine":

DB31, ... DBX75.3 - 5 (JOG approach fixed point reached)

This display signal is also signaled if the axis reaches the fixed point position in the machine coordinates system via other methods e.g. NC program, FC18 (for 840D sl) or synchronized action on the setpoint side and comes to a standstill on the actual value side within the "Exact stop fine" tolerance window (MD36010 $MA_STOP_LIMIT_FINE).

Movement in the opposite direction

The response while traversing in the opposite direction, i.e. against the direction of the approaching fixed point, depends on the setting of Bit 2 in the machine data:

MD10735 $MN_JOG_MODE_MASK (settings for the JOG mode)

Traverse in the opposite direction is possible only if the bit is set.

Traverse in the opposite direction is blocked if the bit is not set and the following channel status message is output if an attempt is made with the traverse keys or with the handwheel to traverse in the direction opposite the approaching fixed point:

"JOG: Direction blocked <axis>"

Approaching other fixed point

The axis motion is cancelled and the following alarm is output if a different fixed point is selected while traversing to the fixed point:

Alarm 17812 "Channel %1 Axis %2 fixed point approach in JOG: Fixed point changed"

The message signal DB31, ... DBX75.0 - 2 (JOG - Approaching fixed point active) displays the number of the newly selected fixed point. The JOG traverse must be triggered again to continue traversing.

Note

To avoid the alarm message, the machine user should proceed as follows:

1. Cancel the current traverse movement with residual distance deletion.
2. Activate fixed point approach for another fixed point and start the operation after the axis comes to a standstill.

Withdrawal from fixed point / deactivation

To withdraw from a fixed position, you must deactivate the "Approaching fixed point in JOG" function. This is done by resetting the activation signal to "0".

DB31, ... DBX13.0 - 2 = 0

The message signals DB31, ... DBX75.0 - 2 (JOG - Approaching fixed point active) and DB31, ... DBX75.3 - 5 (JOG - Approaching fixed point reached) are deleted on leaving the fixed point position.

Special case: Axis is already on fixed point

The axis cannot be moved if, while starting the fixed point traverse, the axis is already at the position of the fixed point to be approached. This is displayed through the following channel status message:

"JOG: position reached <axis>"

To withdraw from the fixed position, you must deactivate the "Approaching fixed point in JOG" function.

Special features of incremental travel

If, during incremental travel, the fixed point is reached before the increment is completed, then the increment is considered to have been completed fully. This is the case even when only whole increments are traveled:

MD11346 $MN_HANDWH_TRUE_DISTANCE = 2 or 3

Special features of modulo rotary axes

Modulo rotary axes can approach the fixed point in both directions. The shortest path (DC) is not observed during the travel.

Features of spindles

A spindle changes to the positioning mode on actuating the "Approaching fixed point in JOG" function. The closed loop position control is active and the axis can traverse to the fixed point.

If no zero mark is detected the alarm message in the axis operation is output:

Alarm 17810 "Channel %1 Axis %2 not referenced"

As a spindle must also be a modulo rotary axis at all times, the same conditions apply for direction observation as for modulo rotary axes (refer to the paragraph "Features of modulo rotary axes")

17.9.2 Parameterization

Movement in the opposite direction

The behavior while traversing in the opposite direction - i.e. in the direction opposite to approaching the fixed point - depends on the following setting:

MD10735 $MN_JOG_MODE_MASK, bit 2 (settings for the JOG mode)

Bit	Value	Meaning
2	0	Travel in the opposite direction is **not** possible (basic setting).
	1	Movement in the opposite direction is possible.

Fixed point positions

Up to four fixed point positions can be defined for an axis:

MD30600 $MA_FIX_POINT_POS[0...3] = <fixed point position 1...4>

Number of valid fixed point positions

The number of fixed point positions entered in MD30600 $MA_FIX_POINT_POS that are actually valid, can be defined using:

MD30610 $MA_NUM_FIX_POINT_POS = <number of valid fixed point positions>

Note

Exception: G75

For compatibility reasons, the following parameter assignment is also possible for G75:

MD30610 $MA_NUM_FIX_POINT_POS = 0 (no valid fixed point positions)

it is assumed that there are two valid fixed point positions in MD30600 $MA_FIX_POINT_POS[0] and [1].

Fixed point positions 1 and 2 can be activated via the NC/PLC interface, however they can only be approached via G75.

Axis dynamics

The axis-specific acceleration and the axis-specific jerk for "Approaching fixed point in JOG" are determined by the following machine data:

- When traversing with **traverse keys or handwheel**:
  - MD32300 $MA_MAX_AX_ACCEL [0] (maximum axis-specific acceleration for path motions in the dynamic response mode DYNNORM)
  - MD32430 $MA_JOG_AND_POS_MAX_JERK (maximum axis-specific jerk)

    Note

    MD32430 $MA_JOG_AND_POS_MAX_JERK is only effective when the axis-specific jerk limitation for single-axis movements has been enabled for the machine axes to be traversed:

    MD32420 $MA_JOG_AND_POS_JERK_ENABLE [<axis>] == TRUE

- During traversing via the **G75 part program command**:
  - MD32300 $MA_MAX_AX_ACCEL [0/1] (maximum axis-specific acceleration for path motions in the dynamic response mode DYNNORM/DYNPOS)

    Note

    The type of positioning axis dynamics (DYNNORM or DYNPOS) is defined by the machine data:

    MD18960 $MN_POS_DYN_MODE = <mode>

  - MD32431 $MA_MAX_AX_JERK [0] (maximum axis-specific jerk for path motions in the dynamic response mode DYNNORM)

    Note

    MD32431 $MA_MAX_AX_JERK is only effective when the axis-specific jerk limitation for single-axis movements has been enabled for the machine axes to be traversed:

    MD32420 $MA_JOG_AND_POS_JERK_ENABLE [<axis>] == TRUE

Reference:
Function Manual, Basic Functions; Acceleration (B2)

17.9.3 Programming

System variables

The following system variables that can be read in the part program and in the synchronous actions for the "Approach fixed point" function.

System variable	Description
$AA_FIX_POINT_SELECTED [<Axis>]	Number of fixed point to be approached
$AA_FIX_POINT_ACT [<Axis>]	Number of the fixed point on which the axis is currently located

17.9.4 Supplementary Conditions

Axis is indexing axis

The axis is not traversed and an alarm is output if the axis to be traversed is an indexing axis and the fixed point position to be approached does not match an indexing position.

Frames active

All active frames are ignored. Traversing is performed in the machine coordinate system.

Offset values active

Active offset values (DRF, external zero offset, synchronized action offset $AA_OFF, online tool offset) are also traversed. The fixed point is a position in the machine coordinates system.

An alarm is issued if an offset movement (DRF, external zero offset, synchronized action offset $AA_OFF, online tool offset) is made during a fixed point approach in JOG. The position of the fixed point to be approached in the machine coordinates system is not reached; instead a position that would have been reached without active offset movement is reached. The NC/PLC interface signal DB31, ... DBX75.3-5 corresponding to the fixed point is not output.

Working-area limitations

Working-area limitations (in BCS and WCS) are considered and the axis motion is stopped on reaching the limits.

17.9.5 Application example

Target

A rotary axis (machine axis 4 [AX4]) is to be moved to fixed point 2 (90 degrees) with the "Approaching fixed point in JOG" function.

Parameter assignment

The machine data for the "Approaching fixed point" function of machine axis 4 are parameterized as follows:

MD30610 $MA_NUM_FIX_POINT_POS[AX4] = 4	4 fixed points are defined for machine axis 4.
MD30600 $MA_FIX_POINT_POS[0,AX4] = 0	1. Fixed point of AX4 = 0 degrees
MD30600 $MA_FIX_POINT_POS[1,AX4] = 90	2. Fixed point of AX4 = 90 degrees
MD30600 $MA_FIX_POINT_POS[2,AX4] = 180	3. Fixed point of AX4 = 180 degrees
MD30600 $MA_FIX_POINT_POS[3,AX4] = 270	4. Fixed point of AX4 = 270 degrees

Initial situation

Machine axis 4 is referenced and is at position of 0 degrees. This corresponds to the 1st fixed position and is output through the NC/PLC interface signal:

DB31 DBX75.0 = 1 (bit 0 - 2 = 1)

Approaching fixed point 2

The control is switched into the JOG mode.

The "Approaching fixed point" function is activated at fixed point 2 via the NC/PLC interface signal:

DB31 DBX13.1 = 1 (bit 0 - 2 = 2)

The actuation is confirmed via the NC/PLC interface signal:

DB31 DB75.1 = 1 (bit 0 - 2 = 2)

The plus travel key on the machine control panel is used to travel continuously to approach fixed point 2.

The machine axis 4 stops at the 90 degree position. This is signaled via the NC/PLC interface signal:

DB31 DBX75.4 = 1 (bit 3 - 5 = 2)

17.10 Position travel in JOG

17.10.1 Function

The "Position travel in JOG" function allows the machine operator to specify, via the setting data, a position in the machine coordinate system, to which the selected machine axis can be traversed using the traverse keys or the handwheel.

Applications

The function is used for machine tools that are exclusively operated manually, e.g. to approach drill-holes in precise increments.

Preconditions

- The "Position travel in JOG" function can be activated only in the JOG mode.
  The function cannot be activated if the machine function JOG-REPOS or JOG-REF is active and in JOG in AUTOMATIC.
- The axis to be traversed must be referenced.
- A kinematic transformation may not be active.
- The axis to be traversed may not be a following axis of an active coupling.

Procedure

Procedure for position travel in JOG:

1. Selection of JOG mode
2. Specification of the position to be approached
3. Enabling the "Position travel in JOG" function
4. Traversing of the machine axis with traverse keys or handwheel

Specification of the position to be approached

The position to be approached is entered in the setting data:

SD43320 $SA_JOG_POSITION (JOG position)

Activating the function

The PLC sets the interface signal after the "Position travel in JOG" function has been selected:

DB31, ... DBX13.3 (JOG travel to position)

The NC confirms the activation with the interface signal as soon as the function is effective:

DB31, ... DBX75.6 (JOG JOG travel to position active)

Note

Activation is not possible:

- during an NC reset
- In case of impending emergency stop
- During processing of an ASUP
- if the "approaching fixed point JOG" function is already activated

No alarm message occurs. Delayed activation takes place after closure or after acknowledgement of the active function.

Sequence

Approaching the position is started with the traverse keys or the handwheel.

The axis traverses until it comes to an automatic standstill at the position to be approached.

When the position is reached with "Exact stop fine", the following NC/PLC interface signal is output:

DB31, ... DBX75.7 (JOG position reached)

Approach a different position

If the content of the setting data SD43320 is changed while approaching a position, then the axis movement is canceled and the following alarm is output:

Alarm 17822 "Channel %1 Axis %2 jogging to position: position changed"

The JOG traverse must be triggered again to approach the new position.

Note

To avoid the alarm message, the machine user should proceed as follows:

1. Cancel the current traverse movement with residual distance deletion.
2. Enter a new position in SD43320 after axis has come to a standstill.
3. Restart the JOG motion.

Retraction from the position/deactivation

To retract from the position, the function "Position travel in JOG" must be deactivated. This is done by resetting the activation signal to "0".

DB31, ... DBX13.3 = 0

The deactivation also resets the message signals DB31, ... DBX75.6 and DB31, ... DBX75.7.

Retraction in the opposite direction

The behavior when retracting from the position in the opposite direction depends on the setting of bit 2 in the machine data:

MD10735 $MN_JOG_MODE_MASK (settings for the JOG mode)

Retraction in the opposite direction is possible only if the bit is set.

If the bit is not set, retraction in the opposite direction is blocked, and the following channel status message is output if an attempt is made with the traverse keys or with the handwheel to exit the approached position in the opposite direction:

"JOG: Direction blocked <axis>"

Special case: axis is already at position

The axis cannot be traversed if the axis is already at the position to be approached when starting the JOG motion. This is displayed through the following channel status message:

"JOG: position reached <axis>"

To retract, the function "Position travel in JOG" must be deactivated.

Special features of incremental travel

If, during incremental travel, the position is reached before the increment is completed, then the increment is considered to have been completed fully. This is the case even when only whole increments are traveled.

MD11346 $MN_HANDWH_TRUE_DISTANCE = 2 or 3

Special features of modulo rotary axes

Modulo rotary axes can approach the position in both directions. The shortest path (DC) is not observed during the travel.

Features of spindles

A spindle changes to the positioning mode on actuating the "Position travel in JOG" function. The closed loop position control is active and the axis can be traversed to the position.

If no zero mark is detected the alarm message in the axis operation is output:

Alarm 17810 "Channel %1 Axis %2 not referenced"

As a spindle must also be a modulo rotary axis at all times, the same conditions apply for direction observation as for modulo rotary axes (refer to the paragraph "Features of modulo rotary axes")

See also

Parameter setting (Page 681)

17.10.2 Parameter setting

Consideration of axial frames and tool length compensation

The consideration of axial frames and, if an axis is configured as geometry axis, the tool length compensation, depends on the setting of bit 1 in the machine data:

MD10735 $MN_JOG_MODE_MASK (settings for the JOG mode)

Bit	Value	Meaning
1	0	Axial frames and tool length compensation are **not** considered.
	1	Axial frames and tool length compensation are considered.

Retraction in the opposite direction

The behavior when retracting from a position in the opposite direction depends on the setting of bit 2 in the machine data:

MD10735 $MN_JOG_MODE_MASK (settings for the JOG mode)

Bit	Value	Meaning
2	0	Retraction in the opposite direction is **not** possible (default setting).
	1	Retraction in the opposite direction is possible.

Position to be approached

The position to be approached is entered in the setting data:

SD43320 $SA_JOG_POSITION (JOG position)

17.10.3 Supplementary Conditions

Axis is indexing axis

The axis is not traversed and an alarm is output if the axis to be traversed is an indexing axis and the position to be approached does not match an indexing position.

Frames active

All active frames are ignored. Traversing is performed in the machine coordinate system.

Correction values active

Active correction values (DRF, external zero offset, synchronized action offset $AA_OFF, online tool offset) are also traversed.

An alarm is issued if an offset movement (DRF, external zero offset, synchronized action offset $AA_OFF, online tool offset) is made during the approach of a variable position in JOG. The

position to be approached in the machine coordinate system is not reached; instead a position that would have been reached without active offset movement is reached. The NC/PLC interface signal DB31, ... DBX75.7 is not reported.

Working area limits

Working-area limitations (in BCS and WCS) are considered and the axis motion is stopped on reaching the limits.

17.10.4 Application example

Objective

A rotary axis (machine axis 4 [AX4]) is to be moved to Position 100 degrees with the "Position travel in JOG" function.

Parameter assignment

The position to be approached is specified via the user interface as follows:

```
SD43320 $SA_JOG_POSITION[AX4] = 100 ; position to be approached for machine axis 4 =
                                      100
```

Initial situation

Axis 4 [AX4] is referred and is at Position 0 degree.

Approaching the position

The control is switched to JOG mode and machine axis 4 is selected.

The function "Position travel in JOG" is activated:

DB34 DBX13.3 = 1

The NC confirms the activation as soon as the function is effective:

DB34 DBX75.6 = 1

Approaching the position is started via continuous traversing and the JOG traverse key Plus of the machine control panel.

The axis traverses until it comes to an automatic standstill at the position 100 degrees.

When the position is reached with "Exact stop fine", the following NC/PLC interface signal is set:

DB34 DB75.7 = 1

17.11 Circular travel in JOG

17.11.1 Function

With the "Circular travel in JOG" function, the machine operator can simultaneously traverse the two geometry axes of the active plane along an arc using the traverse keys or handwheel.

Applications

This function is used for machine tools that are exclusively operated manually.

Preconditions

- The "Circular travel in JOG" function can be activated only in the JOG mode.
  The function cannot be activated if the machine functions JOG-REPOS and JOG-REF are active.
- The axes that participate in traversing must be homed.
- The active plane must not be inclined.

Procedure

Procedure for circular travel in JOG:

1. Plane selection
2. Selection of JOG mode
3. Specification of circle parameters
4. Activation of the function "Circular travel in JOG"
5. Traversing of the geometry axes with traverse keys or handwheel

Plane selection

The geometry axes of the active plane are selected by programming one of the following G commands:

G command	1st geometry axis of the plane	2nd geometry axis of the plane
G17	X	Y
G18	Z	X
G19	Y	Z

Specification of circle parameters

At least the following data is required for circular travel in JOG:

- Coordinates of the circle center point
- The maximum circle radius for internal machining or minimum circle radius for external machining
- Machining mode (internal or external machining)

The specifications are made via the user interface in the channel-specific setting data:

SD42690 $SC_JOG_CIRCLE_CENTRE (circle center point)

SD42691 $SC_JOG_CIRCLE_RADIUS (circle radius)

SD42692 $SC_JOG_CIRCLE_MODE (jog circle mode)

For details, see Chapter "Parameter setting (Page 689)".

Activating the function

The PLC sets the interface signal after the "Circular travel in JOG" function has been selected:

DB21, ... DBX30.6 (JOG circular travel)

The NC confirms the activation with the interface signal as soon as the function is effective:

DB21, ... DBX377.6 (JOG circular travel active)

Note

Activation is not possible:

- during an NC reset
- In case of impending emergency stop
- During processing of an ASUP
- If at least one of the participating axes is already being traversed
- If PTP travel is active
- During active Cartesian manual traversing with generic transformation

No alarm message occurs. Delayed activation takes place after closure or after acknowledgement of the active function.

Sequence

After activating the function, the tool can be traversed by means of a continuous or incremental JOG motion using traverse keys or a handwheel:

- For a JOG movement request for the 1st geometry axis of the active plane, traversing is done from the current position via a concentric arc with the same center point as for the defined arc.
- For a JOG movement request for the 2nd geometry axis of the active plane, traversing is done perpendicular to the defined arc. As a result, the circle radius changes.

- Only one movement at a time can be carried out in the active plane: either on the arc or perpendicular to it.
- Regardless of the movements in the active plane, the 3rd geometry axis can be traversed perpendicular to the active plane.

Note

The radius can only be reduced to zero, i.e. to the center point of the circle. If the radius is then enlarged again, traversing occurs back in the direction from where the reducing started.

Note

If the current position is the center point of the circle when the function is activated or after a change to the center point of the circle, then the following applies:

- For a full circle, traversing takes place in the direction of the 2nd geometry axes.
- For a segment of a circle, travel is in the direction of the angle bisector.

Internal or external machining

Depending on the specification in the setting data SD42692 $SC_JOG_CIRCLE_MODE, internal or external machining takes place:

- Internal machining
  - Traversing is only possible **within** the defined circle.
  - The circle radius in SD42691 $SA_CIRCLE_RADIUS is the **maximum** radius.
- External machining
  - Traversing is only possible **outside of** the defined circle.
  - The circle radius in SD42691 $SA_CIRCLE_RADIUS is the **minimum** radius.

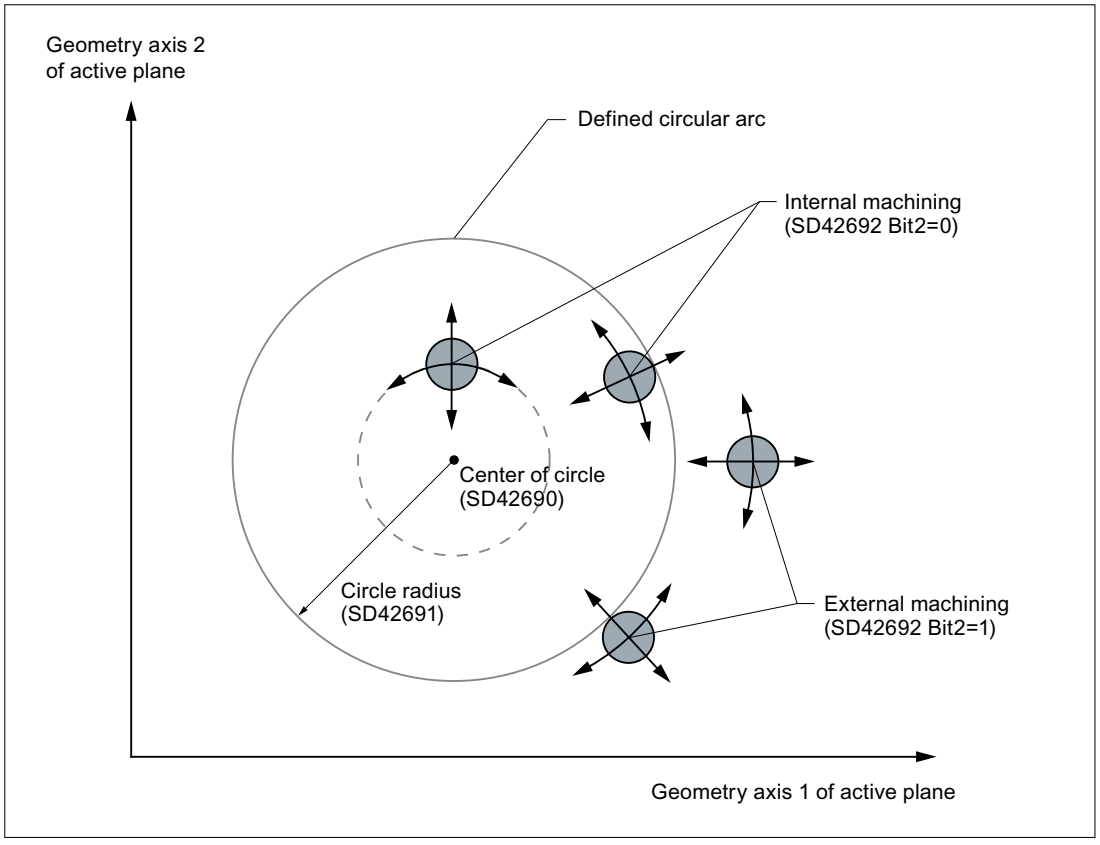


Figure 17-5 Circular travel in JOG: Internal/external machining

Circle segment machining

By specifying a start and end angle, the working area for the 1st and 2nd geometry axis can be limited to a circle segment:

SD42693 $SC_JOG_CIRCLE_START_ANGLE (circle start angle)

SD42694 $SC_JOG_CIRCLE_END_ANGLE (circle end angle)

Furthermore, the direction from the start to the end angle must be specified to permit the unique definition of the circle segment:

SD42692 $SC_JOG_CIRCLE_MODE, bit 0

This bit also specifies the traversing direction along the circular path.

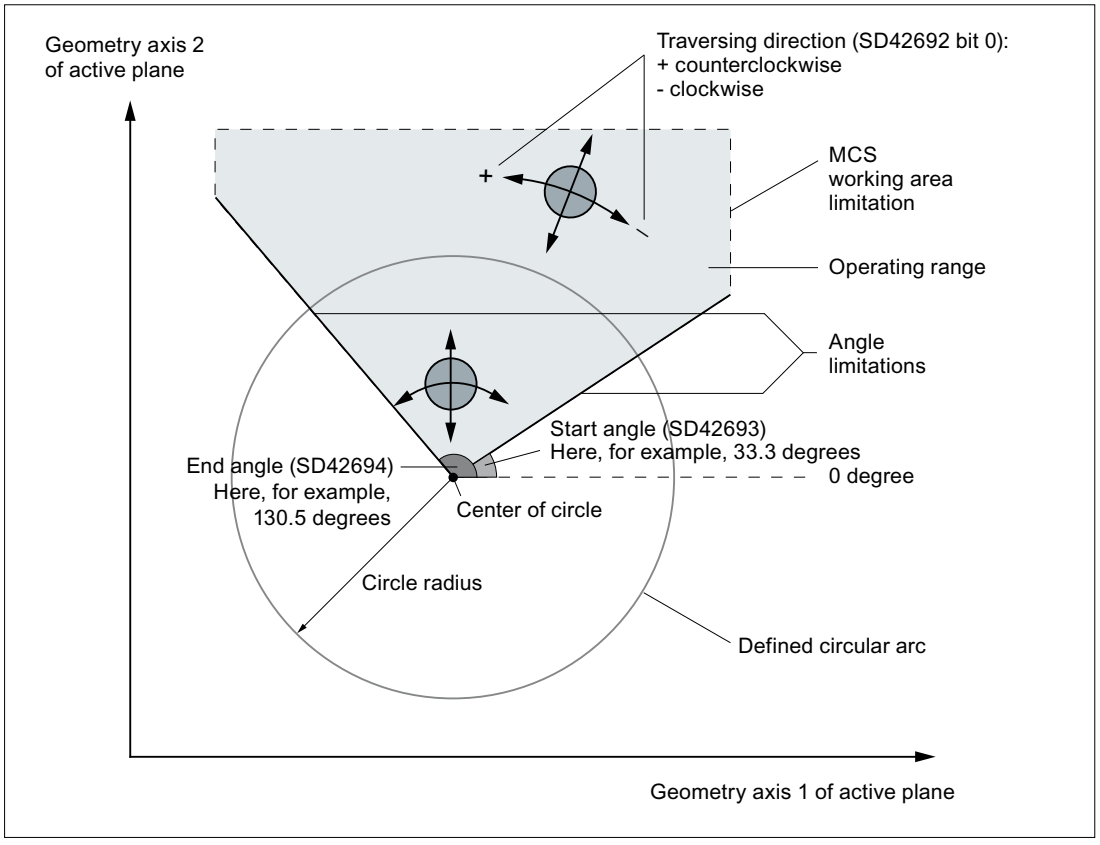


Figure 17-6 Circular travel in JOG: Circle segment machining

Allowance for tool radius

If bit 1 is set in the setting data SD42692 $SC_JOG_CIRCLE_MODE, then the tool radius is considered when monitoring the operating range limits (define circular arc; for circle segment machining also the limits that arise due to the start and end angle):

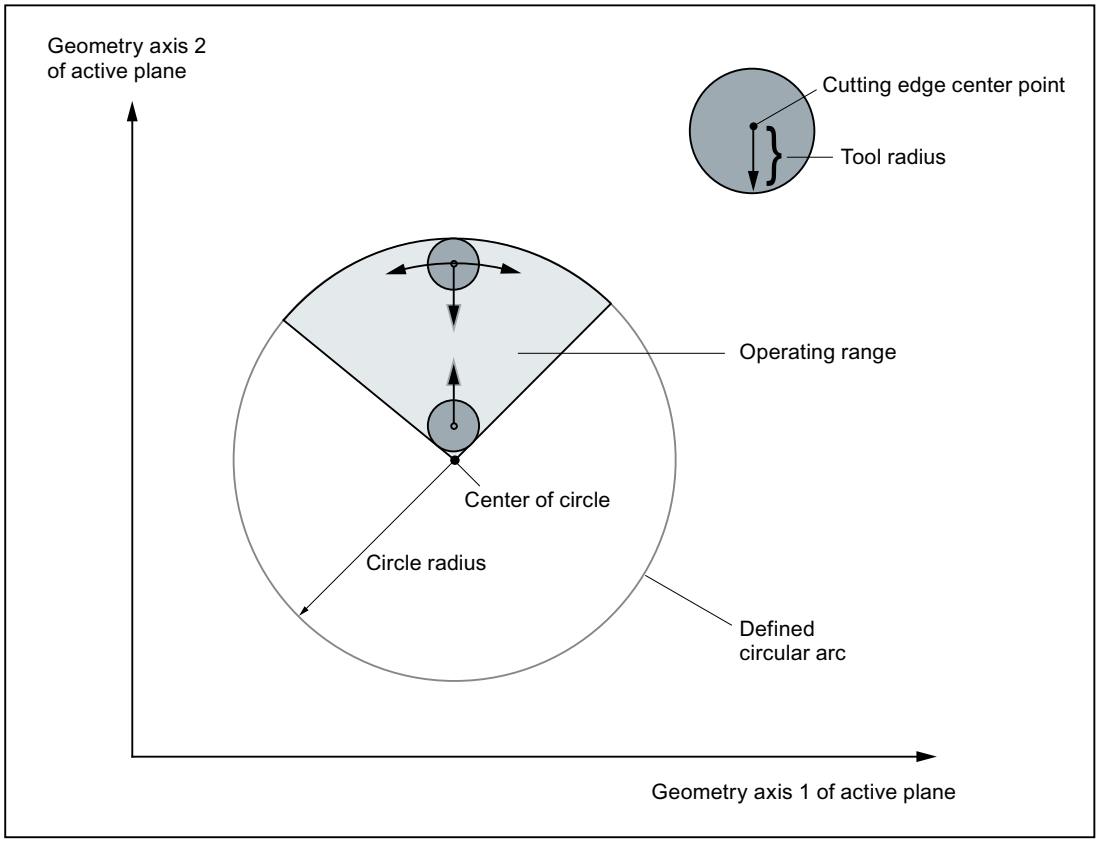


Figure 17-7 Circular travel in JOG: Circle segment internal machining with tool radius compensation

Note

By taking the tool radius into consideration, the smallest possible radius for the internal machining in the vicinity of the center point of the circle is also limited.

Axis monitoring

If the current axis positions of the engaged axes are outside of the workspace limits when the function is activated, then the axes are not traversed and an alarm is output.

Axis positions that are outside of the operating range limits are:

- For internal machining: positions outside of the defined circle
- For external machining: positions within the define circle
- For circle segment machining: Positions outside of the limits that result from the start and end angles (segment borders)
- For active tool radius compensation: Positions too close to limiting circle

- For active tool radius compensation and internal machining: Positions too close to center of circle
- For active tool radius compensation and circle segment machining: Positions too close to segment borders

Changes to the setting data

Changes in the function-relevant setting data SD42690 to SD42694 that are made during an active JOG motion only take effect after ending the active JOG motion by triggering a new JOG motion.

Note

To ensure that changes in the setting data take effect more quickly, the machine operator can proceed as follows:

1. Cancel the current traverse movement with residual distance deletion.
2. Change setting data.
3. Restart the JOG motion.

Abort/deactivation

If during a JOG motion the function "Circular travel in JOG" is deactivated (DB21, ... DBX30.6 = 0), then the JOG motion is canceled and an alarm is output.

Note

To avoid the alarm message, the machine user should proceed as follows:

1. Cancel the current traverse movement with residual distance deletion.
2. Deactivating the function after axis standstill.

17.11.2 Parameter setting

Traversing direction of the second geometry axis

The traversing direction of the second geometry axis can be set via the following machine data:

MD10735 $MN_JOG_MODE_MASK (settings for the JOG mode)

Bit	Value	Meaning
6	0	Traversing of the 2nd geometry axis of the active plane to plus increases the radius, traversing to minus decreases the radius. The traversing movement is independent of whether internal or external machining is active.
7	0	

Bit	Value	Meaning
6	0	Traversing of the 2nd geometry axis of the active plane to plus always takes place in the direction of the limiting circle. I.e. the radius is increased for internal machining and decreased for external machining.
7	1	

Bit	Value	Meaning
6	1	Traversing of the 2nd geometry axis of the active plane to minus always takes place in the direction of the limiting circle. I.e. the radius is increased for internal machining and decreased for external machining.
7	1	

Coordinates of center of circle as radius value

In basic setting, the specified coordinate of an axis is always a diameter value when diameter programming is active. This can be changed for specifying the center of circle coordinates for circular traversing in JOG via the following machine data:

MD20360 $MC_TOOL_PARAMETER_DEF_MASK (definition of the tool parameters)

Bit	Value	Meaning
13	0	For circular travel in JOG, the center of circle coordinate is a diameter value (basic setting).
	1	For circular travel in JOG, the center of circle coordinate is always a radius value.

Center of circle

The center of the circle to be traversed is set in the setting data:

SD42690 $SC_JOG_CIRCLE_CENTRE (circle center point)

The coordinate data refers to the WCS. It can be a diameter or radius value (depending on the setting in the MD20360 Bit 13).

Only the center point coordinates of the geometry axes of the active plane are evaluated, not the coordinates of the geometry axis that is perpendicular to the plane.

Circle radius

The radius of the maximum circle for internal machining or of the minimum circle for external machining is set in the setting data:

SD42691 $SC_JOG_CIRCLE_RADIUS (circle radius)

The radius that is specified refers to the WCS.

Internal or external machining

The machining mode (internal or external machining) is defined in the setting data:

SD42692 $SC_JOG_CIRCLE_MODE (jog circle mode)

Bit	Value	Meaning
2	0	**Internal machining** takes place. The circle radius in SD42691 is the maximum possible radius.
	1	**External machining** takes place. The circle radius in SD42691 is the minimum possible radius.

Circle segment machining

The start and end angles for defining a circle segment that limits the operating range are set in the setting data:

SD42693 $SC_JOG_CIRCLE_START_ANGLE (circle start angle)

SD42694 $SC_JOG_CIRCLE_END_ANGLE (circle end angle)

The angle data refers to the abscissa of the active plane.

The direction from the start to the end angle is specified in the setting data:

SD42692 $SC_JOG_CIRCLE_MODE (jog circle mode)

Bit	Value	Meaning
0	0	The direction from the start to the end angle is counterclockwise.
	1	The direction from the start to the end angle is clockwise.

Note

If the start and end angle are equal to zero, full circles can be traversed without limitation.

Allowance for tool radius

The consideration of the tool radius can be selected via the following setting data:

SD42692 $SC_JOG_CIRCLE_MODE (jog circle mode)

Bit	Value	Meaning
1	0	When monitoring the operating range limits (define circular arc; for circle segment machining also the limits that result from the start and end angle), the tool radius is **not** considered (basic setting).
	1	When monitoring the operating range limits, the tool radius is considered.

17.11.3 Supplementary Conditions

Diameter programming active

During incremental traversing and handwheel traversing of the geometry axes of the active plane, traversing always takes place in the radius programming, even if diameter programming is active for one of the two involved geometry axes.

Axis is not a geometry axis

Both axes of the active plane must be defined as geometry axes:

- Either by configuration of MD20050 $MC_AXCONF_GEOAX_ASSIGN_TAB
- or by programming `GEOAX` in the part program

If the axes of the active plane are not defined as geometry axes, then the axes are not traversed and an alarm is output.

Axis is hirth axis

If an axis involved in the circular path is a hirth axis, then the axis is not traversed and an alarm is output.

Protection zones

Protection zones are not monitored.

17.11.4 Application example

Initial situation

- The geometry axes are configured as follows:

MD20050 $MC_AXCONF_GEOAX_ASSIGN_TAB[0] = 1	1st geo. axis = 1st channel axis
MD20050 $MC_AXCONF_GEOAX_ASSIGN_TAB[1] = 2	2nd geo. axis = 2nd channel axis
MD20050 $MC_AXCONF_GEOAX_ASSIGN_TAB[2] = 3	3rd geo. axis = 3rd channel axis

- G17 is active (basic setting):
  - X axis = 1st geometry axis of active plane
  - Y axis = 2nd geometry axis of active plane
- The selected tool has a radius correction value of 0.02 mm.

- The current axis positions are:
  - X axis: 10
  - Y axis: 10
- The axes are homed.

Parameter assignment

SD42690 $SC_JOG_CIRCLE_CENTRE[AX1] = 10	Center of circle X axis to position 10 mm
SD42690 $SC_JOG_CIRCLE_CENTRE[AX2] = 20	Center of circle Y axis to position 20 mm
SD42691 $SC_JOG_CIRCLE_RADIUS = 20	Circle radius 20 mm
SD42692 $SC_JOG_CIRCLE_MODE = 2	Traversing with the plus key: traversing the circle counter-clockwise Consideration of the tool radius compensation Internal machining
SD42693 $SC_JOG_CIRCLE_START_ANGLE = 0	No start angle active
SD42694 $SC_JOG_CIRCLE_END_ANGLE = 0	No end angle active

Result

Circle inside machining takes place.

With continuous traversing of the X axis using the plus traversing key, a counter-clockwise (G2) circular movement can be initiated with a radius of 10 mm about the center point of the circle (10, 20).

With the traversing of the Y axis, the circle radius can be changed between 0.02 and 19.98.

17.12 Retraction in the tool direction (JOG retract)

17.12.1 Overview

Function

The function "Retraction in the tool direction with the machine function JOG retract", called "JOG retract" in the following, supports the manual tool retraction in the workpiece coordinate system (WCS) after a program abort through a power off of the control or a channel reset in the AUTOMATIC or MDI mode.

In particular, the specific features of the following functions are taken into account:

- Tapping with compensating chuck and speed-controlled spindle with encoder (G33)
- Tapping without compensating chuck and position-controlled spindle (G331, G332)
- Machining with tool orientation with swivel cycle CYCLE800 or orientation transformation

Availability

JOG retract can only be selected if one of the following functions was interrupted as a result of a channel reset or power off:

- Tapping using G33/G331/G332
- Machining using tool type 2xx (drilling tools)

Data to be restored

In order to be able to execute the retraction in the tool direction after a program abort, the following data which was active in the channel before the program abort, is restored:

- Active tool offset
- Active machining plane
- Active tool carrier
- Active transformation data block with transformation parameters
- Data of the thread group with G33 or G331/G332
- Positions of the axes that are involved in the transformation

If this data is completely available after a program abort, it is restored in the channel when the JOG retract machine function is selected. In this case, the workpiece coordinate system (WCS) is aligned by the control in such a way that one of the geometry axes is in the direction of the tool axis. The tool retraction can then be performed manually by traversing this geometry axis.

Note

Data backup

The specified data itself is not backed up, only the references to this data. If the data is changed prior to the selection of JOG retract, the function is performed on the basis of the changed data.

Restrictions

Programs that can be started by interrupt signals are **not** executed in JOG retract.

17.12.2 Parameterization

17.12.2.1 Automatic selection of JOG retract after Power On

After the control has run up (Power On), the channels of a BAG are as standard in the parameterized default mode:

MD10720 $MN_OPERATING_MODE_DEFAULT[<mode group>] = <default mode>

The following NC-specific machine data can be used to set which operating mode is selected after the control run up, if there is retraction data in a BAG channel.

MD10721 $MN_OPERATING_MODE_EXTENDED[<mode group>] = <value>

Value	Meaning
0	Selection of the operating mode corresponding to MD10720 $MN_OPERATING_MODE_DEFAULT
1	Selection of JOG mode, if: DB21, ... DBX377.5 == 1 (retraction data available)

17.12.2.2 Enable of the traversing direction

The retraction movement can be limited to the positive direction of travel or enabled for both travel directions using the following NC-specific machine data:

MD10735 $MN_JOG_MODE_MASK, Bit 8 = <Value>

Value	Meaning
0	Release: Positive traverse direction
1	Release: Positive and negative traverse direction

17.12.2.3 Measuring system status

The current status of the measuring system is displayed via the following axis-specific machine data or must be set to the desired behavior for selection:

MD34210 $MA_ENC_REFP_STATE[<Axis>] = <Value>

Value [1)]	Meaning
Absolute encoder	
2	Encoder is calibrated
Incremental encoder	
1	Automatic referencing: Enabled, but encoder still not referenced
2	Automatic referencing: Encoder is referenced and in exact stop, automatic referencing takes effect at next time encoder activation
3	Restoration of the actual position: The last axis position buffered before switching off is restored, no automatic referencing
1) Only the values relevant for JOG-Retract are stated	

17.12.3 Selection

Function

Requirement

The selection of JOG retract is only possible, if valid retraction data is available for the relevant channel, the channel is in JOG mode and in the "Reset" state:

- DB21, … DBX377.5 == 1 (retraction data available)
- DB11 DBX(n*20+6).2 == 1 (active mode: JOG, with n=0, 1, 2 ... for mode group 1, 2, ...)
- DB21, … DBX35.7 == 1 (channel state: Reset)

Axes and spindles

All active measuring systems of the machine axes involved in the retraction in the tool direction must be in the "referenced" or "restored" state when the control is switched on again (Power On).

A detailed description of the automatic restoration of actual positions after the next start of the control (Power On) can be found in:

Further information
R1: Referencing (Page 291)

Axis interchange

If all the axes and spindles involved in the retraction are not assigned to the channel at the time of selection, an implicit axis interchange is performed for the missing axes.

Coordinate system

With the selection of JOG retract, the workpiece coordinate system (WCS) is set for the traversing of the channel axes. Axes involved in the retraction can only be traversed in the WCS. All axes that are not involved in the retraction can also be traversed in the machine coordinate system (MCS).

Selection options

The selection of JOG retract can be supported through automatic selection of the JOG mode after Power On. The actual selection of JOG retract is performed through manual selection via the user interface or via the PLC user program.

Automatic selection of JOG mode

If the automatic selection of JOG retract is parameterized (see Section: "Automatic selection of JOG retract after Power On (Page 695)") and there is retraction data for a channel after control run-up (Power On), JOG mode is selected for the mode group.

Selection via the user interface

The selection of JOG retract is performed on the user interface via:

"Machine operating area" > "ETC key (">")" > "Retract"

Note

The softkey "Retract" is only displayed if there is retraction data and an active tool.

Selection by PLC user program

The following actions must be performed to select JOG retract by the PLC user program:

- Channel-specific query whether retraction data is available
  DB21, … DBX377.5 == 1 (retraction data available)
- Mode group-specific selection of JOG mode:
  DB11, ... DBX(n*20+6).2 == 1 (active mode: JOG, with n=0, 1, 2 ... for mode group 1, 2, ...)
- Channel-specific selection of JOG retract via the PI service "RETRAC".
  A detailed description of the activation of the PI service "RETRAC" via the function block FB 4 can be found in:
  Further information
  Function Manual PLC
- After confirmation of the selection of JOG retract (see below), select the workpiece coordinate system:
  - DB19.DBX0.7 (current coordinate system: 1 ⇒ WCS, 0 ⇒ MCS)
  - DB19.DBX20.7 (Requirement: Switch over MCS/WCS)

Selection confirmation

After selecting JOG retract, the NC/PLC interface signal is set:

DB21, ... DBX377.4 = 1 (JOG retract active)

Thread cutting (G33 or G331/G332)

If the program execution was aborted during a thread cutting operation (G33 or G331/G332), the axis grouping of tool axis and spindle is restored when JOG retract is selected. The controller parameters and parameter sets for the axes involved are also set in accordance with a programmed thread cutting in the part program.

17.12.4 Tool retraction

General retraction behavior

The tool is retracted by manually traversing the retraction axis (geometry axis) specified when selecting JOG retract in the workpiece coordinate system (WCS). The retraction movement can be performed via the traversing keys of the machine control panel (MCP) or via the handwheel. Retraction is possible within the traversing range limits (working area limitation, software limit switch, etc.).

The retraction movement can be stopped and started again with NC stop / NC start.

The axes and spindles not involved in the tool retraction can be manually traversed as required.

Switching the coordinates systems is possible (MCS ⇔ WCS).

Traversing direction

The retraction movement is only enabled for the positive traversing direction by default. If traversing in the negative direction is also to be possible, this must be explicitly enabled:

MD10735 $MN_JOG_MODE_MASK, bit 8 = 1

Retraction behavior dependent on processing type and tool type

The following retraction motion is possible depending on the machining type and the active tool type:

- Thread cutting (G33) or tapping (G331,G332):
  Irrespective of the tool type, traverse movements are only possible in the retraction axis (tool axis)
- Tool type 2xx (drilling tools): traverse movements are only possible in the retraction axis (tool axis)
- Tool type 4xx (grinding tools): No retraction motion possible
- Tool type 5xx (turning tools): No retraction motion possible

Retraction movement for thread processing

For thread cutting (G33) or tapping (G331,G332) the retraction movement is performed if either the retraction axis or the spindle is moved.

Retraction using the handwheel

If one of the axes involved in the retraction movement is traversed by means of traversing key, the handwheel pulses for other axes involved in the retraction movement are ignored.

If handwheels have been selected for several axes and these are moved, the handwheel pulses are evaluated in the following order:

1. Retraction axis
2. Spindle
3. Axes/spindles not involved in the retraction movement

The pulses of the other handwheels are ignored and only evaluated at standstill of the preceding handwheels.

Locked functions

During JOG retract, the following functions are blocked or are not performed:

- Spindle start via DB31, ... DBX30
- Traversing of a spindle or axis involved in the retraction via function block FC18
- Switchover of a spindle or an axis involved in the retraction to the PLC-controlled axis

- Changing a spindle or axis involved in the retraction to another channel
- Using a spindle or axis involved in the retraction as main run axis (command axis, oscillating axis, FC18 / concurrent axis)

17.12.5 Deselection

JOG retract is deselected channel-specifically via:

- Channel reset: DB21, … DBX7.7 = 1 (reset)
- User interface: "Back" ("<<") softkey

The following basic settings are active:

- MD20110 $MC_RESET_MODE_MASK[<channel>]
- MD20150 $MC_GCODE_RESET_VALUES[<channel>]
- MD20151 $MC_GCODE_RESET_MODE[<channel>]

After the deselection, the channel is in the JOG mode, channel state "Reset".

All channel axes can be moved manually again. The retraction data is retained. It is therefore possible to select JOG retract again.

NC/PLC interface signal

With the deselection of JOG retract, the NC/PLC interface signal is reset:

DB21, ... DBX377.4 = 0 (JOG retract active)

17.12.6 Repeated selection

As long as there is retraction data, JOG retract can always be reselected:

DB21, ... DBX377.5 == 1 (retraction data available)

If selected again, the original retraction data is restored.

Changed axis positions

The channel axes can be traversed in JOG mode while JOG retract is deselected. If JOG retract is selected again, the retraction movement is executed on the basis of the new axis positions.

17.12.7 Continuing machining

AUTOMATIC mode

Before the aborted part program is continued with NC start in AUTOMATIC mode, all machine axes with active measuring systems in the state "restored" or "not referenced" must be referenced.

NOTICE
Possible axis position displacement After the control runs up after a power failure, the axis positions for incremental measuring systems are synchronized or restored corresponding to the setting in machine data. Once the tool has been retracted in JOG retract machine function, axes whose positions have been restored are referenced.

MDI mode and overstore

In the MDI mode and for the overstore function, machining can also be performed, without referencing the axes, with restored positions. To do this, NC start with restored positions must be enabled explicitly for a specific channel:

MD20700 $MC_REFP_NC_START_LOCK = 2

Block search at interruption point

The processing of the part program can be continued from the interruption point via a block search at the interruption point. The last block of the main program level before the interruption is available as interruption point.

A detailed description of the function and for the operation of the block search can be found in:

References

Basic Functions Function Manual; Section "K1: Mode group, channel, program operation, reset response" > "Block search" or "Block search type 5 SERUPRO"

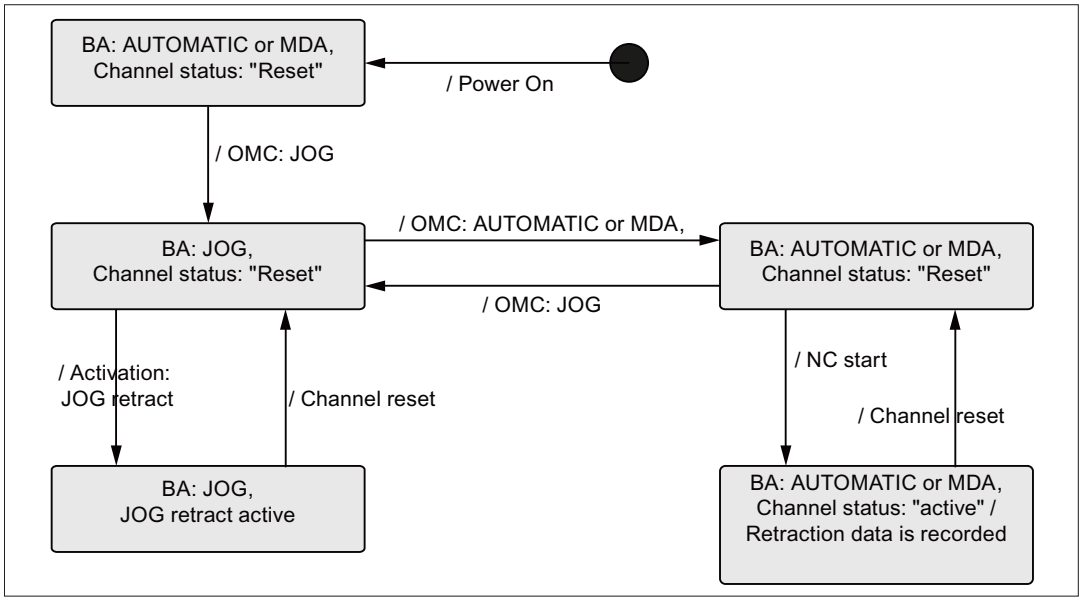
Turning or Milling Operating Manual; Section "Workpiece machining" > "Starting machining at a specific point"

Continue with NC start

With NC start in the AUTOMATIC or MDI mode, the processing of the part program is continued from the selected position. The following actions are performed with regard to JOG retract:

- The retraction data is deleted.
- The NC/PLC interface signal is reset:
  DB21, … DBX377.5 = 0 (retraction data available)
- The current data environment is saved.

17.12.8 State diagram



OM Operating mode

OMC Operating mode change

Figure 17-8 State diagram: JOG retract

17.12.9 System data

The following system data is available for JOG retract:

Meaning	System variable $VA_	NC/PLC interface	OPI variable
Retraction data available	-	DB21, ... DBX377.5	retractState, bit 0
JOG retract active	-	DB21, ... DBX377.4	retractState, bit 1
Retraction axis	-	-	retractState, bits 2 - 3
Position restored, 1st measuring system	$AA_POSRES	DB31, ... DBX71.4	aaPosRes
Position restored, 2nd measuring system	$AA_POSRES	DB31, ... DBX71.5	aaPosRes

References

OPI and system variable

Lists Book 2 List Manual, Section "Variables".

17.12.10 Supplementary conditions

Incremental measuring systems

The user must ensure that machine axes with incremental measuring systems are clamped with sufficient speed in the event of a power failure to prevent a change to the last position, known and saved by the control. Otherwise the assumed position when the control is restarted differs greatly to the actual position of the machine axis. **Drive-autonomous retraction** must also **not** be activated for these axes.

A detailed description of the automatic restoration of actual positions after the next start of the control (Power On) can be found in:

References:
Basic Functions Function Manual, Section "R1 Referencing" > "Automatic restoration of the machine reference"

Drive-autonomous retraction

The "Drive-autonomous retraction" function must **not** be activated for machine axes involved in the retraction movement.

Axis couplings

Axis couplings are **not** restored with the selection of JOG retract.

Tapping using G63

JOG retract is **not** possible for tapping with compensating chuck and speed-controlled spindle without encoder (`G63`).

Transformations

If a transformation is selected through JOG retract, the active measuring systems of all machine axes involved in the transformation must be in the "referenced" or "restored" state.

OEM transformations such as parallel kinematics for hexapods, can **only** be traversed with **referenced** measuring systems. Traversing with restored axis positions is **not** possible.

Tool orientation via directly programmed orientation axes

JOG retract cannot generate a retraction movement in the tool direction if the tool orientation is not performed via NC functions, but through direct programming of the orientation axes.

17.13 Use of handwheels in automatic mode

17.13.1 Handwheel override in automatic mode

17.13.1.1 General functionality

Function

With this function it is possible to traverse axes or to change their velocities directly with the handwheel in automatic mode (Automatic, MDA).

The handwheel override is activated in the NC part program using the NC language elements `FD` (for path axes) and `FDA` (for positioning axes) and is **non-modal**.

With positioning axes, it is possible to activate the handwheel override modally using traverse command `POSA`. When the programmed target position is reached, the handwheel override becomes inactive again.

Additional axes can be traversed simultaneously or using interpolation in the same NC block.

The concurrent-positioning-axes function can also be activated by the PLC user program.

Distinction

Depending on the programmed feedrate, a distinction is made between the following for handwheel override:

- **Path definition**
  Axis feedrate = 0 (FDA = 0)
- **Velocity override**
  Axis feedrate > 0 (FD or FDA > 0)

The table below shows which axis types can be influenced by the "handwheel override in automatic mode" function.

Axes that can be influenced by the "handwheel override in automatic mode" function		
Axis type	**Velocity override**	**Path definition**
Positioning axis	FDA[AXi] > 0 ; acts axially	FDA[AXi] = 0
Concurrent positioning axis	Parameter "Handwheel override active" = 1 and axis feedrate > 0 from FC18	Parameter "Handwheel override active" = 1 and axis feedrate = 0 from FC18
Path axis	FD > 0 ; acts on path velocity	Not possible

Path definition

With axis feedrate = 0 (e.g. `FDA[AXi] = 0`), the traversing movement of the positioning axis towards the programmed target position is controlled entirely by the user rotating the assigned handwheel.

The direction in which the handwheel is turned determines the traversing direction of the axis. The programmed target position cannot be exceeded during handwheel override. The axis can also be moved toward the programmed target position from the opposite direction, whereby the movement is only restricted by the axial position limitations.

A block change occurs when:

- The axis has reached the programmed target position

or

- The distance-to-go is deleted by axial interface signal DB31, ... DBX2.2 (delete distance-to-go).

From this moment on, the path default is deactivated and any further handwheel pulses have no effect.

After this, incrementally programmed positions refer to the point of interruption and not to the last programmed position.

Velocity override

With regard to the velocity override, a distinction is made between axis feedrate and path feedrate.

- **Axis-velocity override** (FDA[AXi] > 0):
  The positioning axis is moved to the target position at the programmed axial feedrate. Using the assigned handwheel, it is possible to increase the axis velocity or to reduce it to a minimum of zero depending on the direction of rotation. The resulting axis feedrate is limited by the maximum velocity. However, the axis cannot be traversed in the opposite direction to that programmed.
  The block change is performed as soon as the axis reaches the programmed target position. This causes the velocity override to be deactivated automatically and any further handwheel pulses have no effect.
  Similarly, this also applies to concurrent positioning axes, where the target position and the velocity are defined by the PLC.
- **Path-velocity override** (FD > 0):
  The path axes programmed in the NC block traverse to the target position at the programmed feedrate. If the velocity override is active, the programmed path velocity is overridden by the velocity generated with the **handwheel of the 1st geometry axis**. The block change is performed as soon as the programmed target position is reached.
  The path velocity is increased or reduced to a minimum of zero depending on the direction of rotation of the handwheel. However, it is not possible to reverse the direction of movement with handwheel override.

Application example

The "Handwheel override in automatic mode" function is frequently used on grinding machines. For example, the user can position the reciprocating grinding wheel on the workpiece using the handwheel (path default). After scratching, the traversing movement is terminated and the block change is initiated (by activating DB31, ... DBX2.2 (delete distance-to-go)).

Preconditions

In order to activate "Handwheel override in automatic mode", the following requirements must have been met:

- A handwheel must be assigned to the axis in question.
- Pulse weighting exists for the assigned handwheel.

Handwheel assignment

The assignment of the connected handwheels to the axes is analogous to the "Manual traversing using a handwheel (Page 651)" via the user interface or via the PLC user interface with one of the following interface signals:

- Machine axes:
  - DB31, ... DBX4.0 - 2 (activate handwheel)
- Geometry axes:
  - DB21, ... DBX12.0 - 2 (geometry axis 1: Activate handwheel)
  - DB21, ... DBX16.0 - 2 (geometry axis 2: Activate handwheel)
  - DB21, ... DBX20.0 - 2 (geometry axis 3: Activate handwheel)

If handwheel override is programmed for an axis to which no handwheel is assigned, a distinction is made between the following cases:

- **For velocity override:**
  The axes traverse at the programmed velocity.
  A self-acknowledging alarm is output (without response).
- **For path definition:**
  No traversing movement is performed because the velocity is zero.
  A self-acknowledging alarm is output (without response).

Note

For the velocity override for path axes, only the **handwheel of the 1st geometry axis** is applied to the path velocity.

Handwheel weighting

The travel path of the axis that is generated by rotating the handwheel by one detent position is dependent on several factors (see Section "Manual traversing using a handwheel (Page 651)"):

- Selected increment size:
  MD11330 $MN_JOG_INCR_SIZE_TAB[5] (increment size for INC/handwheel)
  or
  SD41010 $SN_JOG_VAR_INCR_SIZE (size of the variable increment for JOG)
- Weighting of an increment:
  MD31090 $MA_JOG_INCR_WEIGHT
- Number of handwheel pulses per detent position:
  MD11320 $MN_HANDWH_IMP_PER_LATCH

For example, the axis traverses by 0.001 mm per handwheel detent position if machine function INC1 and the default setting of the above machine data are selected.

In the case of velocity override, the velocity results from the traverse path covered using the handwheel within a certain period of time.

Example

Assumptions:
The operator turns the handwheel with 100 pulses/second.

The selected machine function is INC100.

The default setting is made for the above machine data for handwheel weighting.

⇒ Handwheel traverse path per second: 10 mm
⇒ Velocity override: 0.6 m/min

PLC interface signals

As soon as the handwheel override takes effect, the following interface signals to the PLC are set to signal 1:

- For positioning axes / concurrent positioning axes / command axes / reciprocating axes:
  DB31, ... DBX62.1 (handwheel override active)
- For path axes:
  DB21, ... DBX33.3 (handwheel override active)

For the path input, depending on the traversing direction, the appropriate interface signals are output to the PLC:

- Machine axes:
  - DB31, ... DBX64.6 / 7 (travel command "minus"/"plus")
- Geometry axes:
  - DB21, ... DBX40.6 - 7 (geometry axis 1: Travel command "minus"/"plus")
  - DB21, ... DBX46.6 - 7 (geometry axis 2: Travel command "minus"/"plus")
  - DB21, ... DBX52.6 - 7 (geometry axis 3: Travel command "minus"/"plus")

Limitations

The axial limitations (software limit switch, hardware limit switch, working area limitation) are effective in conjunction with handwheel override. With path default, the axis can be traversed with the handwheel in the programmed traversing direction only as far as the programmed target position.

The resulting velocity is limited by the axial machine data:

MD32000 $MA_MAX_AX_VELO(maximum axis velocity)

NC Stop/override = 0

If the feedrate override is set to 0% or an NC Stop is initiated while the handwheel override is active, the following applies:

- **For path definition:**
  The handwheel pulses arriving in the meantime are summated and stored. If NC Start or the feedrate override > 0%, the saved handwheel pulses become effective (i.e. are traversed). However, if the handwheel is first deactivated [via IS DB21, ... DBX12/16/20.0-2 (geometry axes 1/2/3: Activate handwheel)] then the stored handwheel pulses are deleted.
- **For velocity specification:**
  The handwheel pulses arriving in the meantime are not summated and are not active.

17.13.1.2 Programming and activating handwheel override

General

When the handwheel override is programmed with NC language elements `FD` (for path axes) and `FDA` (for positioning axes), the following points must be observed:

- `FDA` and `FD` function **non-modally.**
  Exception for positioning axes: If traverse instruction `POSA` is programmed, the handwheel override can also act modally because this positioning axis does not affect the block transition.
- When the handwheel override is activated with `FDA` or `FD`, a **target position** must be programmed in the NC block for the positioning axis or for a path axis. When the programmed target position is reached, the handwheel override becomes inactive again.

- It is not possible to program FDA and FD or FA and F in the same NC block.
- The positioning axis must not be an indexing axis.

Positioning axis

Syntax for handwheel override: `FDA[AXi] = [feedrate value]`

Example 1:

Activate velocity override

```
N10 POS[U]=10 FDA[U]=100 POSA[V]=20 FDA[V]=150 . . .
```

POS[U]=10	Target position of positioning axis U
FDA[U]=100	Activate velocity override for positioning axis U; axis velocity of U = 100 mm/min
POSA[V]=20	Target position of positioning axis V (modally)
FDA[V]=150	Activate velocity override for positioning axis V; axis velocity of V = 150 mm/min

Example 2:

Activate path default and velocity override in the same NC block

```
N20 POS[U]=100 FDA[U]= 0 POS[V]=200 FDA[V]=150 . . .
```

POS[U]=100	Target position of positioning axis U
FDA[U]= 0	Activate path default for positioning axis U;
POS [V]=200	Target position of positioning axis V
FDA[V]=150	Activate velocity override for positioning axis V; axis velocity of V = 150 mm/min

Path axis

Syntax for handwheel override: `FD = [feedrate value]`

In order to activate "Handwheel override in automatic mode", the following requirements must have been met:

- Active movement commands from group 1: G01, G02, G03, CIP
- Exact stop active (G60)
- Linear feedrate in mm/min or inch/min active (G94)

These requirements are checked by the control and an alarm is output if any of them is not met.

Example 3:

Activate velocity override

```
N10 G01 X10 Y100 Z200 FD=1500 . . .
```

`X10 Y100 Z200`	Target position of path axes X, Y and Z
`FD=1500`	Activate velocity override for path axes; path velocity = 1500 mm/min

Concurrent positioning axis

The handwheel override for concurrent positioning axes is activated from the PLC via FC18 by setting the appropriate interface signal:

DB31, ... DBX62.1 (handwheel override active)

If the velocity parameter (F_Value) is transferred with the value 0, then the activated handwheel override acts as distance input, i.e. in this case, the feed is not derived from the axial machine data (see also Section "P2: Positioning axes (Page 185)"):

MD32060 $MA_POS_AX_VELO (initial setting for positioning axis velocity)

Further information

Function Manual PLC

17.13.1.3 Special features of handwheel override in automatic mode

Velocity display

The velocity display for handwheel override shows the following values:

- **Set velocity**
  = programmed velocity
- **Actual velocity**
  = resultant velocity including handwheel override

Effect on transverse axes

If the axis is defined as a transverse axis and `DIAMON` is active, the handwheel pulses are interpreted and traversed as diameter values while handwheel override is active.

Dry-run feedrate

With active dry run
DB21, ... DBX0.6 (activate dry-run feedrate) = 1,
the dry-run feedrate is always effective
SD42100 $SC_DRY_RUN_FEED.

In this way, the axis is traversed to the programmed target position at dry-run feedrate without any influence from the handwheel despite the active handwheel override with path default (`FDA[AXi] = 0`), i.e., the path default is ineffective.

DRF active

When "Handwheel override in automatic mode" is activated, it is important to check whether the "DRF" function is active (DB21, ... DBX0.3 = 1).

If this were the case, the handwheel pulses would also cause a DRF offset of the axis. The user must, therefore, first deactivate DRF.

Feedrate override

The feedrate override does not affect the velocity of the movements produced by the handwheel (exception: 0%). It only affects the programmed feedrate.

With path default and fast handwheel movements, the axis may not be able to follow the handwheel rotation synchronously (especially in the case of a large handwheel-pulse weighting), causing the axis to overtravel.

17.13.2 Contour handwheel/path input using handwheel (option)

"Contour handwheel" function

When the function is activated, the path and synchronized axes can be controlled via a handwheel in AUTOMATIC and MDI modes.

Requirements

It is not permissible that fixed feedrate, dry run feedrate, thread cutting, or tapping are selected.

Availability

The "Contour handwheel" function is available as an option requiring a license.

Input mode (path or velocity input)

Either the distance or the velocity can be entered via the handwheel:

- **Path definition**
  The traversing velocity obtained as a result of the path defined by the handwheel pulses each IPO clock cycle, is limited by the control system to the maximum permissible velocity. The handwheel pulses that are not immediately executed are collected, and traversed through at a later point in time. As a consequence, it is possible that the axes **continue to run**. **No pulses are lost**.
- **Velocity specification**
  Based on the handwheel pulses for each IPO clock cycle, the velocity is specified at which the axes will be traversed. Traversing motion is immediately stopped as soon as no pulses are received from the handwheel in an IPO clock cycle. The axes **do not continue to run** .

The input mode is set with machine data:

MD11346 $MN_HANDWH_TRUE_DISTANCE

Path velocity

The path velocity in mm/min **depends** on:

- Handwheel pulses in each IPO clock cycle
- Pulse evaluation of the handwheel using machine data:
  MD11322 $MN_CONTOURHANDWH_IMP_PER_LATCH
- Active increment (INC1, 10, 100, ...)
- Distance weighting of an increment of the first available geometry axis:
  MD31090 $MA_JOG_INCR_WEIGHT

The path velocity is **independent** of:

- Programmed feedrate mode (mm/min, mm/rev.)
- Programmed path velocity
- Rapid traverse velocity for `G0` blocks
- Override (with the exception of 0% override)

Traversing direction

The traversing direction depends on the direction of rotation of the handwheel:

- Clockwise rotation: Traversing the axes in the programmed direction.
  If the block change criterion (IPO end) is reached, the program advances to the next block (response identical to `G60`).
- Counterclockwise: Traversing the axes opposite to the programmed direction.
  In this case, traversing is only possible up to the start of the actual block. Pulses are not collected if the handwheel continues to rotate.

Activation of the function

The function can be activated via interface signals or in the NC program:

- Activation via interface signal
  DB21, ... DBX30.0 - 2
- Activation via the NC program
  Command `FD=0` . The contour handwheel is switched on non-modally. This means in the following block, the programmed velocity `F...` from the block before the contour handwheel again applies when the contour handwheel is activated.

Note

An alarm is displayed if no feedrate was programmed in the previous NC blocks.

"Simulated contour handwheel" function

When the "Contour handwheel" function is active, using the "Simulated contour handwheel" function, the handwheel or the handwheel pulses are simulated.

Activation

The function is activated via interface signal:

DB21, ... DBX30.3 = 1

The feedrate is then no longer specified by the handwheel pulses, but by feedrate `F...` programmed in the NC program.

Specifying the direction of rotation

The direction of rotation of the simulated handwheel is specified using the interface signal:

DB21, ... DBX30.4

Override

When the "Simulated contour handwheel" function is active, the override is effective as for NC program execution.

Stopping traversing motion

If the "Simulated contour handwheel" function is deselected, or the direction of rotation switched over, then the actual traversing motion is stopped along a braking ramp.

Boundary conditions

- **Velocity limiting**
  - When the "Contour handwheel" function is activated via interface signal (DB21, ... DBX30.0 - 2), the path velocity that can be entered via the handwheel is limited to: Max. path velocity = (programmed path velocity `F...`) * (MD12030 $MN_OVR_FACTOR_FEEDRATE[**30**] + reserve for dynamic operations)
  - When the "Contour handwheel" function is activated using command `FD=0` in the NC program, the path velocity that can be entered via the handwheel is limited to the maximum axis velocities defined in machine data (MD32000 $MA_MAX_AX_VELO).
- **Acceleration and velocity limit values**
  When traversing the axes using the "Contour handwheel" function, the axis-specific limit values for acceleration, defined in machine data (e.g. MD32300 $MA_MAX_AX_ACCEL), and velocity (MD32000 $MA_MAX_AX_VELO) are maintained.
- **Interruption of traversing movement**
  If, while traversing using the "Contour handwheel" function, an NC stop is initiated (DB21, ... DBX7.3 or 4), then the function remains selected. The handwheel pulses received during an NC stop are not evaluated. Requirement: MD32084 $MA_HANDWH_CHAN_STOP_COND, bit 2 == 1
- **DRF**
  An active DRF function (DB21, ... DBX0.3) additionally has a path override action.

- **Channel-specific delete distance-to-go**
  Channel-specific delete distance-to-go (DB21, ... DBX6.2) causes the motion, initiated by the contour handwheel, to be canceled. The axes are braked and the program is restarted with the next NC block.
- **Mold making**
  Executing mold making programs using the contour handwheel does not produce a suitable surface quality or constant cutting speed.

17.13.3 DRF offset

17.13.3.1 DRF offset

Function

The "DRF offset" function (differential resolver function) can be used to set an additive incremental zero offset in respect of geometry and auxiliary axes in the basic coordinate system in AUTOMATIC mode via an electronic handwheel.

The handwheel assignment, i.e. the assignment of the handwheel from which the increments for the DRF offset are to be derived, to the geometry or auxiliary axes that are to be moved by this, must be performed via the appropriate machine axes. The appropriate machine axes are those to which the geometry or auxiliary axis is mapped.

The DRF offset is not displayed in the axis actual-value display.

Applications

The DRF offset can be used, for example, in the following application cases:

- Offsetting tool wear within an NC block
  Where NC blocks have very long processing times, it becomes necessary to offset tool wear manually within the NC block (e.g. large surface-milling machines).
- Highly precise offset during grinding
- Simple temperature compensation

Note

The zero offset introduced via the DRF offset is always effective in all modes and after a RESET. It can, however, be suppressed non-modally in the part program.

Velocity reduction

The velocity generated using the handwheel for DRF can be reduced with respect to the JOG velocity:

MD32090 $MA_HANDWH_VELO_OVERLAY_FACTOR (ratio of JOG velocity to handwheel velocity (DRF))

DRF active

DRF must be active to allow the DRF offset to be modified by means of traversal with the handwheel. The following requirements must be fulfilled:

- AUTOMATIC mode
- DB21, ... DBX0.3 (activate DRF) = 1

The DRF offset can be activated/deactivated for specific channels using the "program control" function on the HMI user interface.

The HMI software then sets interface signal:
DB21, ... DBX24.3 (DRF selected) =1.

The PLC program (basic PLC program or user program) transfers this interface signal to interface signal
DB21, ... DBX0.3 (activate DRF) once the corresponding logic operation has been performed.

Control of DRF offset

The DRF offset can be modified, deleted or read:

User:	• Traversing with the handwheel
Part program:	• Reading via axis-specific system variable $AC_DRF[<axis>] • Deleting via parts-program command (DRFOF) for all axes in a channel • Non-modal suppression via parts-program command (SUPA) **Further information** Programming Instructions Fundamentals
PLC user program:	• Reading the DRF offset (axis-specific) **Further information** Function description PLC
HMI user interface:	• Display of the DRF offset (axis-specific)

Note

If DRF offset is deleted, the axis is not traversed!

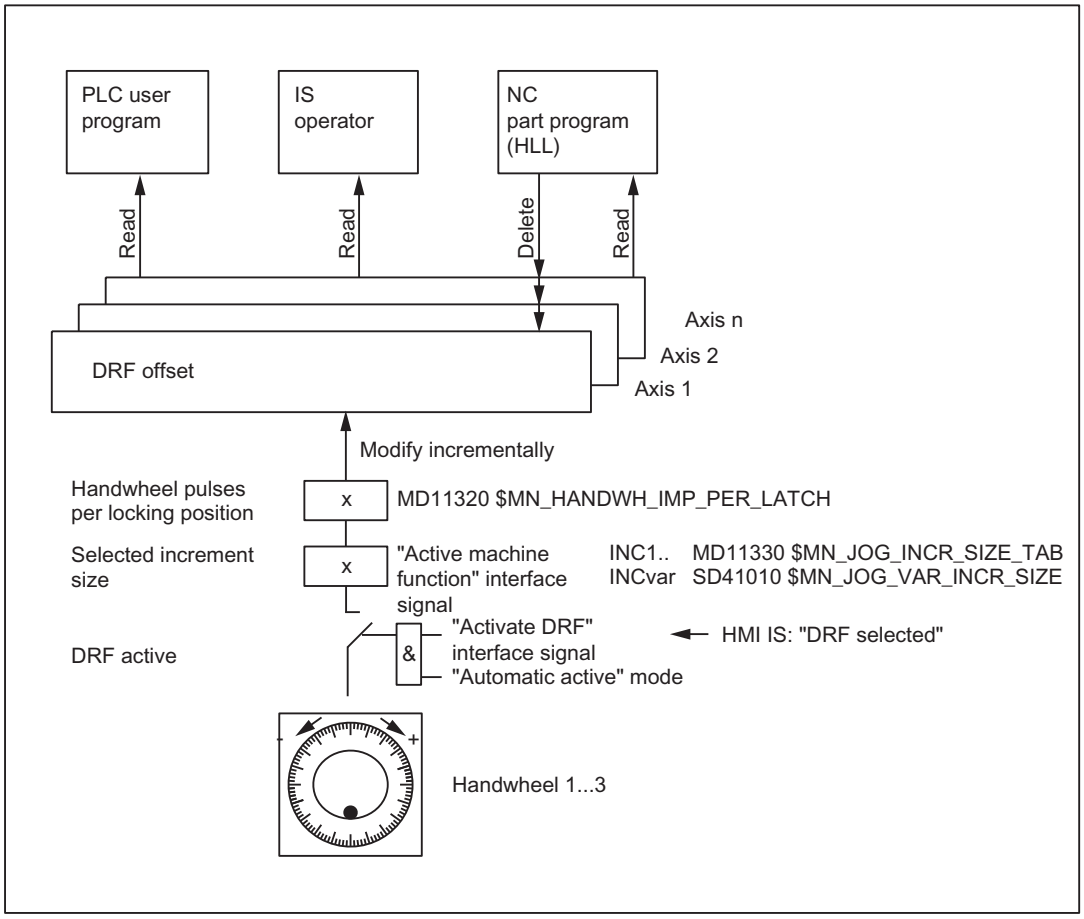


Figure 17-9 Control of DRF offset

Display

The axis actual-position display (ACTUAL POSITION) does not change while an axis is being traversed with the handwheel via DRF. The current axis DRF offset can be displayed in the DRF window.

Reference point approach

In phase 1 of the machine-axis reference point approach, the DRF offset for the corresponding geometry or auxiliary axis is deleted.

During the machine-axis reference point approach, a DRF offset for the corresponding geometry or auxiliary axis cannot be performed simultaneously.

Reset behavior

`PowerOn-Reset`: The DRF offset is deleted.

17.13.3.2 Commissioning

Function

The "DRF offset" function (differential resolver function) can be used to set an additive incremental zero offset in respect of geometry and auxiliary axes in the basic coordinate system in AUTOMATIC mode via an electronic handwheel.

The handwheel assignment, i.e. the assignment of the handwheel from which the increments for the DRF offset are to be derived, to the geometry or auxiliary axes that are to be moved by this, must be performed via the appropriate machine axes. The appropriate machine axes are those to which the geometry or auxiliary axis is mapped.

The DRF offset is not displayed in the axis actual-value display.

Applications

The DRF offset can be used, for example, in the following application cases:

- Offsetting tool wear within an NC block
  Where NC blocks have very long processing times, it becomes necessary to offset tool wear manually within the NC block (e.g. large surface-milling machines).
- Highly precise offset during grinding
- Simple temperature compensation

Note

The zero offset introduced via the DRF offset is always effective in all modes and after a RESET. It can, however, be suppressed non-modally in the part program.

Velocity reduction

The velocity generated using the handwheel for DRF can be reduced with respect to the JOG velocity:

MD32090 $MA_HANDWH_VELO_OVERLAY_FACTOR (ratio of JOG velocity to handwheel velocity (DRF))

DRF active

DRF must be active to allow the DRF offset to be modified by means of traversal with the handwheel. The following requirements must be fulfilled:

- AUTOMATIC mode
- DB21, ... DBX0.3 (activate DRF) = 1

The DRF offset can be activated/deactivated for specific channels using the "program control" function on the HMI user interface.

The HMI software then sets interface signal:
DB21, ... DBX24.3 (DRF selected) =1.

The PLC program (basic PLC program or user program) transfers this interface signal to interface signal
DB21, ... DBX0.3 (activate DRF) once the corresponding logic operation has been performed.

Control of DRF offset

The DRF offset can be modified, deleted or read:

User:	• Traversing with the handwheel
Part program:	• Reading via axis-specific system variable $AC_DRF[<axis>] • Deleting via parts-program command (DRFOF) for all axes in a channel • Non-modal suppression via parts-program command (SUPA) **Further information** Programming Manual Fundamentals
PLC user program:	• Reading the DRF offset (axis-specific) **Further information** Function Manual PLC
HMI user interface:	• Display of the DRF offset (axis-specific)

Note

If DRF offset is deleted, the axis is not traversed!

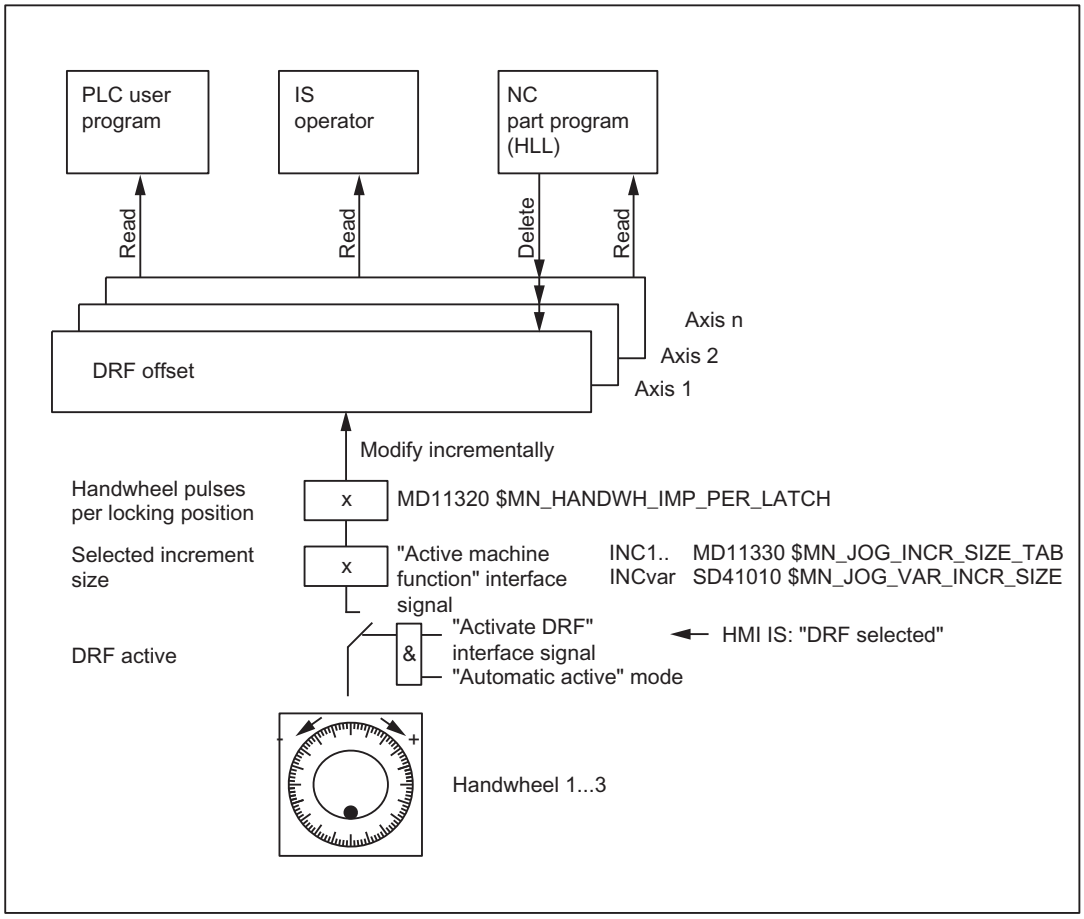


Figure 17-10 Control of DRF offset

Display

The axis actual-position display (ACTUAL POSITION) does not change while an axis is being traversed with the handwheel via DRF. The current axis DRF offset can be displayed in the DRF window.

Reference point approach

In phase 1 of the machine-axis reference point approach, the DRF offset for the corresponding geometry or auxiliary axis is deleted.

During the machine-axis reference point approach, a DRF offset for the corresponding geometry or auxiliary axis cannot be performed simultaneously.

Reset behavior

`PowerOn-Reset`: The DRF offset is deleted.

17.13.3.3 Programming: Deselecting overlays axis-specifically (CORROF)

The following axis-specific overlays (overrides) are deleted with the CORROF procedure:

- Additive work offsets (DRF offsets) set via handwheel traversal
- Position offsets programmed via the $AA_OFF system variable

A preprocessing stop is initiated through the deletion of an overlay (override) value and the position component of the deselected overlaid movement is transferred to the position in the basic coordinate system. Whereby, no axis is traversed.

The position value that can be read via the $AA_IM system variable (current **MCS** setpoint of the axis) **does not change** in the **machine** coordinate system.

The position value that can be read via the $AA_IW system variable (current **WCS** setpoint of the axis) **changes** in the **workpiece** coordinate system because it now contains the deselected component of the overlaid movement.

Note

`CORROF` can be programmed in an **NC program**.

`CORROF` must **not** be programmed in a **synchronized action**.

Syntax

```
CORROF(<Axis>,"<String>"[,<Axis>,"<String>"])
```

Meaning

`CORROF`:	Procedure for the deselection of the following offsets and overlays of an axis: • DRF offset • Position offsets ($AA_OFF)	
	Effective-ness:	Modal
`<Axis>`:	Axis identifier (channel, geometry or machine axis identifier)	
	Data type:	AXIS
`<String>`:	Character string for the definition of the overlay type	
	Data type:	BOOL
	Value	**Meaning**
	`DRF`	DRF offset
	`AA_OFF`	Position offset ($AA_OFF)

Examples

Example 1: Axis-specific deselection of a DRF offset (1)

A DRF offset is generated in the X axis by DRF handwheel traversal. No DRF offsets are operative for any other axes in the channel.

```
Program code                Comment
N10 CORROF(X,"DRF")         ; CORROF has the same effect as DRFOF here.
...
```

Example 2: Axis-specific deselection of a DRF offset (2)

A DRF offset is generated in the X and Y axes by DRF handwheel traversal. No DRF offsets are operative for any other axes in the channel.

```
Program code                Comment
; Only the DRF offset of the X axis is deselected; the DRF offset of the Y axis is
retained.
; With DRFOF, both offsets would have been deselected.
N10 CORROF(X,"DRF")
...
```

Example 3: Axis-specific deselection of a $AA_OFF position offset

```
Program code                                        Comment
; A position offset == 10 is interpolated for the X axis.
N10 WHEN TRUE DO $AA_OFF[X]=10 G4 F5


...
; The position offset of the X axis is deselected: $AA_OFF[X]=0
; The X axis is not traversed.
; The position offset is added to the current position of the X axis.
N80 CORROF(X,"AA_OFF")
...
```

Example 4: Axis-specific deselection of a DRF offset and a $AA_OFF position offset (1)

A DRF offset is generated in the X axis by DRF handwheel traversal. No DRF offsets are operative for any other axes in the channel.

```
Program code                                        Comment
; A position offset of 10 is interpolated for the X axis.
N10 WHEN TRUE DO $AA_OFF[X]=10 G4 F5
...
; Only the DRF offset and the position offset of the X axis are deselected.
; The DRF offset of the Y axis is retained.
N70 CORROF(X,"DRF",X,"AA_OFF")
...
```

Example 5: Axis-specific deselection of a DRF offset and a $AA_OFF position offset (2)

A DRF offset is generated in the X and Y axes by DRF handwheel traversal. No DRF offsets are operative for any other axes in the channel.

Program code	Comment
; A position offset == 10 is interpolated for the X axis.	
N10 WHEN TRUE DO $AA_OFF[X]=10 G4 F5	
...	
; The DRF offset of the Y axis and the position offset of the X axis are deselected.	
; The DRF offset of the X axis is retained.	
N70 CORROF(Y,"DRF",X,"AA_OFF")	
...	

Further information

$AA_OFF_VAL

Once the position offset has been deselected by means of $AA_OFF, system variable $AA_OFF_VAL (integrated distance of axis overlay) for the corresponding axis will equal zero.

$AA_OFF in JOG mode

Also in JOG mode, if $AA_OFF changes, the position offset will be interpolated as an overlaid movement if this function has been enabled via machine data MD 36750 $MA_AA_OFF_MODE.

$AA_OFF in synchronized action

If a synchronized action which immediately resets $AA_OFF (`DO $AA_OFF[<axis>]=<value>`) is active when the position offset is deselected using the `CORROF(<axis>,"AA_OFF")`, then $AA_OFF will be deselected and not reset, and alarm 21660 will be displayed. However, if the synchronized action becomes active later, e.g. in the block after `CORROF`, $AA_OFF will remain set and a position offset will be interpolated.

Automatic channel axis exchange

If an axis that is active in another channel has been programmed for a `CORROF`, it will be fetched into the channel with an axis exchange (requirement: MD30552 $MA_AUTO_GET_TYPE > 0) and the position offset and/or the DRF offset deselected.

17.13.4 Double use of the handwheel

Alarm 14320

The double use of a handwheel for DRF and velocity or distance overlay, including contour handwheel, is suppressed and is displayed using the self-clearing alarm 14320 (Handwheel %1 used twice (%2) in channel %3 axis %4), if, different influences can act on an axis as a result of the handwheel.

This means that an overlaid movement can only be executed when no DRF offset (triggered by the same handwheel) is active for the axes in the basic coordinate system that are involved in the movement, i.e., the DRF movement must have been terminated.

If an overlaid movement has been started, no DRF offset can be started for any of the axes involved that are supplied by the same handwheel. Such a DRF movement is only possible when the movement with overlay has reached its end point or has been aborted by delete distance-to-go or RESET.

If the handwheel override and DRF offset are to be active simultaneously, this is possible with activation of **two** separate handwheels.

Example: Path override

Assumption:
Channel 1 and geometry axis X correspond to machine axis 3 and geometry axis Y corresponds to machine axis 5 and handwheel 2 is selected for the first geometry axis.

If block `X10 Y10 FD=0` is processed in the main run, neither machine axis 3 nor machine axis 5 can be traversed with DRF via handwheel 2. If handwheel 2 is assigned to machine axis 3 while the channel-specific DRF signal is active, then alarm 14320 (Handwheel 2 used twice (8) in channel 1 axis X) is signaled.

If machine axis 3 or machine axis 5 is traversed with DRF using the 2nd handwheel, then motion `X10 Y10 FD=0` cannot be executed and alarm 14320 (handwheel 2 used twice (3) in channel 1 axis X) or alarm 14320 (handwheel 2 used twice (3) in channel 1 axis Y) is signaled.

Example: Velocity override of positioning axis

Assumption:
Channel 1: Channel axis A corresponds to machine axis 4 and handwheel 1 is assigned to this axis.

If block `POS[A]=100 FDA[A]=0` is processed in the main run, machine axis 4 cannot be traversed with DRF. This means that if the channel-specific DRF signal is active, alarm 14320 (Handwheel 1 used twice (6) in channel 1 axis A) is signaled.

If machine axis 4 is traversed with DRF, then no `POS[A]=100 FDA[A]=0` movement can be executed while a DRF movement is being performed. Alarm 14320 (Handwheel 1 used twice (1) in channel 1 axis A) is signaled.

Example: Distance overlay PLC axis (840D sl)

Assumption:
Channel 1: Handwheel 2 is assigned to machine axis 4.

If an axis movement with path override of the 4th machine axis triggered by FC18 is processed in the main run, machine axis 4 cannot be traversed with DRF. This means that if the channel-specific DRF signal is active, alarm 14320 (Handwheel 2 used twice (9) in channel 1 axis A) is signaled.

If machine axis 4 is traversed with DRF, then no axis movement with path override triggered by FC18 can be executed while a DRF movement is being performed. Alarm 14320 (Handwheel 2 used twice (4) in channel 1 axis A) is signaled.

17.14 Monitoring functions

Limitations

For manual traversing and manual traversing using a handwheel, the following limits apply:

- Working area limitation (the axis must be referenced).
- Software limit switches 1 and 2 (axis must be referenced)
- Hardware limit switches

The control ensures that the traversing movement is aborted as soon as the first valid limitation has been reached. Velocity control ensures that deceleration is initiated early enough for the axis to stop exactly at the limit position (e.g. software limit switch). Only when the hardware limit switch is triggered does the axis stop abruptly with "rapid stop".

Alarms are triggered when the various limitations are reached (alarms 16016, 16017, 16020, 16021). The control automatically prevents further movement in this direction. The traversing keys and the handwheel have no effect in this direction.

Note

The software limit switches and working-area limitations are only active if the axis has first been referenced.

If a work offset (DRF offset) via handwheel is active for axes, the software limit switches for these axes are monitored during the main run in JOG mode. This means that the jerk limitation has no effect when the software limit switches are approached. After acceleration in accordance with MD32300 $MA_MAX_AX_ACCEL (maximum axis acceleration) the velocity is reduced at the software limit switch.

For further information on working area limitations and hardware and software limit switches, see:
Further information
Function Manual Monitoring and Compensating; Axis monitoring, protection zones

Retract axis

The axis can be retracted from a limit position by moving it in the opposite direction.

Note

Machine manufacturer

The function for retracting an axis that has approached the limit position depends on the machine manufacturer. Please refer to the machine manufacturer's documentation!

Maximum velocity and acceleration

The velocity and acceleration used when manually traversing are defined by the startup engineer for specific axes using machine data. The control limits the values acting on the axes to the maximum velocity and acceleration specifications.

Further information
B2: Acceleration and jerk (Page 347)

G2: Velocities, setpoint / actual value systems, closed-loop control (Page 31)

17.15 Start-up: Handwheels

17.15.1 General information

In order to operate handwheels of a SINUMERIK control system, they have to be parameterized via NC machine data.

If the handwheels are not directly connected to the control, additional measures are required, e.g. connection via PROFIBUS- or Ethernet-MCP or handwheel module, inserting and configuring the module with SIMATIC STEP 7, HW-Config.

Note

Currently only six handwheels can be parameterized in a SINUMERIK control system.

Connection options

Handwheels can be connected via the following components:

- Ethernet (Page 725) module

Note

Several handwheels, which are connected via different components, can be connected to one SINUMERIK at the same time.

17.15.2 Connection via Ethernet

Parameter assignment

The parameters for handwheels connected via Ethernet modules, e.g. machine control panel "MCP 483C IE", "HT 8", or "HT 2", are assigned in the following NC machine data:

- MD11350 $MN_HANDWHEEL_SEGMENT[< x - 1 >] = 7
  When connected via Ethernet modules, the segment always has to be entered as 7 (Ethernet).
- MD11351 $MN_HANDWHEEL_MODULE[< x - 1 >] = 1
  When connected via Ethernet modules, the module always has to be entered as 1.
- MD11352 $MN_HANDWHEEL_INPUT[< x - 1 >] = y
  with y = 1, 2, 3, ... (handwheel interface at the Ethernet bus)

with x = 1, 2, 3, ... (handwheel number in the NC)

Handwheel interfaces at the Ethernet Bus

The handwheel interfaces at the Ethernet bus are numbered on the basis of the following definitions:

- The sequence of the operator component interfaces is: MCP1, MCP2, BHG
- Each operator component interface has two handwheel interfaces
- Operator components: MCP 483C IE
  A maximum of two handwheels can be connected to an MCP 483C IE via connections X60 and X61 . The assignment of the connections in the operator component interface is:
  - Connection X60: 1st handwheel in the operator component interface MCP1 /MCP2
  - Connection X61: 2nd handwheel in the operator component interface MCP1 /MCP2
- Operator components: HT 8
  The handwheel of the HT 8 is always assigned to the 1st handwheel of operator component interface MCP1 /MCP2 .
- Operator components: HT 2
  The handwheel of the HT 2 is always assigned to the 1st handwheel in the operator component interface BHG .

Operator component interface ->		MCP1		MCP2		BHG	
Handwheel interface [1]		1	2	1	2	1	2
FB1 parameters [2]		MCP1BusAdr		MCP2BusAdr		BHGRecGDNo	
Assignment of the handwheels [3]							
	MCP 483C IE	X60	X61	X60	X61	-	-
	HT 8	x	-	x	-	-	-
	HT 2	-	-	-	-	x	-

Operator component interface ->	MCP1		MCP2		BHG	
Handwheel interface at the Ethernet bus (y) [4] ->	1	2	3	4	5	6
1) Numbering of the handwheel interfaces within an operator component interface 2) Assignment of the operator component to the interface via the corresponding FB1 parameter 3) Assignment of the handwheels of the respective operator components to the handwheel interfaces 4) Numbering of the handwheel interfaces at the Ethernet bus -> MD11352 $MN_HANDWHEEL_INPUT[< x - 1 >] = **y**						

Example

Parameterization of 3 handwheels, connected via the following operator components:

Operator component interface ->	MCP1		MCP2		BHG	
Operator component	HT 8		MCP 483C		HT 2	
FB1 parameters	MCP1BusAdr := 39		MCP2BusAdr := 192		BHGRecGDNo := 40	
Handwheel interface	**x**	-	-	**X61**	**x**	-
Handwheel interface at Ethernet bus ->	**1**	2	3	**4**	**5**	6

Table 17-1 NC machine data for the handwheel assignment

Machine data	Value	Description
		HT 8: Handwheel number in the NC = 1
MD11350 $MN_HANDWHEEL_SEGMENT[0]	7	Segment: Ethernet
MD11350 $MN_HANDWHEEL_MODULE[0]	1	Module: Ethernet
MD11350 $MN_HANDWHEEL_INPUT[0]	1	Handwheel interface at Ethernet bus
		MCP 483C IE: Handwheel number in the NC = 2
MD11350 $MN_HANDWHEEL_SEGMENT[1]	7	Segment: Ethernet
MD11350 $MN_HANDWHEEL_ MODULE [1]	1	Module: Ethernet
MD11350 $MN_HANDWHEEL_ INPUT [1]	4	Handwheel interface at Ethernet bus
		HT 2: Handwheel number in the NC = 3
MD11350 $MN_HANDWHEEL_SEGMENT[2]	7	Segment: Ethernet
MD11350 $MN_HANDWHEEL_ MODULE [2]	1	Module: Ethernet
MD11350 $MN_HANDWHEEL_ INPUT [2]	5	Handwheel interface at Ethernet bus

Table 17-2 FB1 parameters (excerpt)

```
Parameter       Value          Remark
MCPNum          := 2           // Number of connected MCPs
                               // MCP1 = HT 8
MCP1In          ...            // MCP1 parameters...
...             ...
MCP1BusAdr      := 39          // Via switches S1 and S2 on the connecting
                               device
                               // Set "IP address"
                               // MCP2 = MCP 483C IE
```

```
MCP2In          ...             // MCP2 parameters...
...             ...
MCP2BusAdr      := 192          // Via switch S2 on the MCP 483C
                                // Set "IP address"
MCPBusType      := b#16#55      // Bus type: Ethernet
                                // HHU = HT 2
HHU             := 5            // Bus type: Ethernet
BHGIn           ...             // HHU Parameter ...
...             ...
BHGRecGDNo      := 40           // Via switches S1 and S2 on the connecting
                                device
                                // Set "IP address"
```

Filter time

Since the handwheel pulses on the Ethernet bus are not transferred deterministically, filtering (smoothing) of the handwheel pulse transfer process may be necessary for highly dynamic drives. The filter time is parameterized using the following machine data:

- MD11354 $MN_HANDWHEEL_FILTER_TIME[< x - 1 >] = <filter time>
  with x = 1, 2, 3, ... (handwheel number in the NC) and filter time = 0.0 ... 2.0 s

The filter time specifies the time it takes for the handwheel pulses transferred to the control to be sent on to the interpolator for traversing purposes. With a filter time of 0.0 s, the handwheel pulses are sent on to the interpolator within a single interpolation cycle. This can result in jerky axis traversing.

The recommended filter time is: 0.2 – 0.5 s

Zero speed detection

A stationary state is detected by the Ethernet modules to which the handwheel is connected. If a handwheel does not transfer any handwheel pulses for a defined period of time, the module detects this to be a stationary state and transfers it to the NC/PLC interface:

NC/PLC interface signal	Value	Meaning
DB10, DBX245.0	0	Handwheel 1 is operated.
	1	Handwheel 1 is stationary
DB10 DBX245.1	0	Handwheel 2 is operated.
	1	Handwheel 2 is stationary.
DB10 DBX245.2	0	Handwheel 3 is operated.
	1	Handwheel 3 is stationary.
DB10 DBX245.3	0	Handwheel 4 is operated.
	1	Handwheel 4 is stationary.
DB10 DBX245.4	0	Handwheel 5 is operated.
	1	Handwheel 5 is stationary.
DB10 DBX245.5	0	Handwheel 6 is operated.
	1	Handwheel 6 is stationary.

By evaluating the signal, it is possible to reduce the overtravel of an axis traversed via the handwheel, due to the handwheel pulses that have been collected in the control but not yet transferred to the interpolator for traversing purposes. To do this, deletion of the distance-to-go must be triggered for the relevant axis or in the channel when a stationary state is detected:

- DB31,... DBX2.2 = 1 (axial delete distance-to-go)
- DB21,... DBX6.2 = 1 (channel-specific deletion of distance-to-go)

17.16 Data lists

17.16.1 Machine data

17.16.1.1 General machine data

Number	Identifier: $MN_	Description
10000	AXCONF_MACHAX_NAME_TAB[n]	Machine axis name
10720	OPERATING_MODE_DEFAULT	Setting of the operating mode after Power On
10721	OPERATING_MODE_EXTENDED	Extended setting of the operating mode after Power On
10735	JOG_MODE_MASK	Settings for JOG mode
11300	JOG_INC_MODE_LEVELTRIGGRD	INC and REF in inching mode
11310	HANDWH_REVERSE	Defines movement in the opposite direction
11320	HANDWH_IMP_PER_LATCH[n]	Handwheel pulses per locking position
11324	HANDWH_VDI_REPRESENTATION	Coding of the handwheel number (NCK/PLC interface)
11330	JOG_INCR_SIZE_TAB[n]	Increment size for INC/handwheel
11340	ENC_HANDWHEEL_SEGMENT_NR	Third handwheel: Bus segment
11342	ENC_HANDWHEEL_MODULE_NR	Third handwheel: Logical drive number
11344	ENC_HANDWHEEL_INPUT_NR	Third handwheel: Encoder interface
11346	HANDWH_TRUE_DISTANCE	Handwheel path or velocity specification
11350	HANDWHEEL_SEGMENT[n]	Handwheel segment
11351	HANDWHEEL_MODULE[n]	Handwheel module
11352	HANDWHEEL_INPUT[n]	Handwheel connection
11353	HANDWHEEL_LOGIC_ADDRESS[n]	Logical handwheel slot address (STEP 7)
17900	VDI_FUNCTION_MASK	Function mask for VDI signals

17.16.1.2 Channel-specific machine data

Number	Identifier: $MC_	Description
20060	AXCONF_GEOAX_NAME_TAB	Geometry axis in channel
20100	DIAMETER_AX_DEF	Geometry axes with transverse axis functions

Number	Identifier: $MC_	Description
20110	RESET_MODE_MASK	Initial setting after reset / program end
20150	GCODE_RESET_VALUES	Reset position of the G groups
20151	GCODE_RESET_MODE	Reset behavior of the G groups
20360	TOOL_PARAMETER_DEF_MASK	Definition of the tool parameters
20620	HANDWH_GEOAX_MAX_INCR_SIZE	Limitation of the geometry axes
20622	HANDWH_GEOAX_MAX_INCR_VSIZE	Path-velocity override
20624	HANDWH_CHAN_STOP_COND	Definition of the manual traversing behavior using a handwheel, channel-specific
20700	REFP_NC_START_LOCK	NC start disable without reference point
21150	JOG_VELO_RAPID_ORI	Conventional rapid traverse for orientation axes
21155	JOG_VELO_ORI	Conventional velocity for orientation axes
21158	JOG_JERK_ORI	Maximum jerk when manually traversing orientation axes
21159	JOG_JERK_ORI_ENABLE	Jerk limitation for manual traversing of orientation axes activated
21160	JOG_VELO_RAPID_GEO	Conventional rapid traverse for geometry axes
21165	JOG_VELO_GEO	Conventional speed for geometry axes
21166	JOG_ACCEL_GEO	Maximum acceleration when manually traversing geometry axes
21168	JOG_JERK_GEO	Maximum jerk when manually traversing geometry axes

17.16.1.3 Axis/spindlespecific machine data

Number	Identifier: $MA_	Description
30450	IS_CONCURRENT_POS_AX	Default setting at reset: Neutral axis or channel axis
30600	FIX_POINT_POS[n]	Fixed point positions of the axis
30610	NUM_FIX_POINT_POS	Number of fixed point positions of an axis
31090	JOG_INCR_WEIGHT	Weighting of an increment for INC/handwheel
32000	MAX_AX_VELO	Maximum axis velocity
32010	JOG_VELO_RAPID	Rapid traverse in jog mode
32020	JOG_VELO	Axis velocity in job mode
32040	JOG_REV_VELO_RAPID	Revolutional feedrate in JOG mode with rapid traverse override
32050	JOG_REV_VELO	Revolutional feedrate for JOG
32060	POS_AX_VELO	Reset position for positioning-axis velocity
32080	HANDWH_MAX_INCR_SIZE	Limitation of the selected increment size
32082	HANDWH_MAX_INCR_VELO_SIZE	Limitation of the selected increment for velocity override
32084	HANDWH_STOP_COND	Behavior, handwheel travel
32090	HANDWH_VELO_OVERLAY_FACTOR	Ratio of JOG velocity to handwheel velocity (with DRF)
32300	MAX_AX_ACCEL	Maximum axis acceleration
32420	JOG_AND_POS_JERK_ENABLE	Basic setting of axis-specific jerk limitation
32430	JOG_AND_POS_MAX_JERK	Maximum axis-specific jerk for single-axis movements

Number	Identifier: $MA_	Description
32431	MAX_AX_JERK	Maximum axis-specific jerk during path movements
34210	ENC_REFP_STATE	Measuring system status
35130	GEAR_STEP_MAX_VELO_LIMIT[n]	Maximum velocity for gear stage

17.16.2 Setting data

17.16.2.1 General setting data

Number	Identifier: $SN_	Description
41010	JOG_VAR_INCR_SIZE	Size of the variable increment for incremental manual traversing or manual traversing using a handwheel
41050	JOG_CONT_MODE_LEVELTRIGGRD	Jog/continuous mode for continuous manual traversing
41100	JOG_REV_IS_ACTIVE	JOG: Revolutional/linear feedrate
41110	JOG_SET_VELO	Axis velocity in JOG mode
41120	JOG_REV_SET_VELO	Revolutional feedrate of axes in JOG mode
41130	JOG_ROT_AX_SET_VELO	Axis velocity of the rotary axes in JOG mode
41200	JOG_SPIND_SET_VELO	Velocity for spindle in JOG mode

17.16.2.2 Channel-specific setting data

Number	Identifier: $SC_	Description
42690	JOG_CIRCLE_CENTRE	Center of circle
42691	JOG_CIRCLE_RADIUS	Circle radius
42692	JOG_CIRCLE_MODE	Jog circle mode
42693	JOG_CIRCLE_START_ANGLE	Start angle for circular travel in JOG
42694	JOG_CIRCLE_END_ANGLE	End angle for circular travel in JOG

17.16.2.3 Axis/spindle-specific setting data

Number	Identifier: $SA_	Description
43320	JOG_POSITION	JOG position

17.16.3 System variable

17.16.3.1 System variable

Identifier	Description
$AA_POSRES	Axis state "Position restored"

17.16.4 OPI variable

17.16.4.1 OPI variable

Identifier	Description
retractState, bit 0	Retraction data available
retractState, bit 1	JOG retract active
retractState, bits 2 - 3	Retraction axis
aaPosRes	Axis state "Position restored".

S1: Spindles 18

18.1 Brief Description

The primary function of a spindle is to set a tool or workpiece in rotary motion in order to facilitate machining.

Depending on the type of machine, the spindle must support the following functions in order to achieve this:

- Input of a direction of rotation for the spindle (M3, M4)
- Input of a spindle speed (S, SVC)
- Spindle stop, without orientation (M5)
- Spindle stop with orientation / Spindle positioning
  (SPOS, M19 and SPOSA)
- Gear change (M40 to M45)
- Spindleaxis functionality (spindle becomes rotary axis and vice versa)
- Thread cutting (G33, G34, G35)
- Tapping without compensating chuck（G331, G332)
- Tapping with compensating chuck（G63)
- Revolutional feedrate (G95)
- Constant cutting rate (G96, G961, G97, G971)
- Programmable spindle speed limits (G25, G26, LIMS=)
- Position encoder assembly on the spindle or on the spindle motor
- Spindle monitoring for min. and max. speed as well as
  max. encoder limit frequency and end point monitoring of spindle
- Switching the position control (SPCON, SPCOF, M70) on/off
- Programming of spindle functions:
  - From the part program
  - Via synchronized actions
  - Via PLC with FC18 or via special spindle interfaces for simple spindle activation

18.2 Operating modes

18.2.1 Overview

A spindle is an endlessly rotating axis, which can be operated in either the closed-loop position control or closed-loop speed control modes.

Operating modes

- Control operation (closed-loop speed control)
- Oscillating operation (closed-loop speed control)
- Positioning operation (closed-loop position control)
- Synchronous mode
  Further information
  S3: Synchronous spindle (Page 847)
- Tapping without compensating chuck (G331, G332)
  Further information
  Programming Manual Fundamentals; Position commands > Tapping > Tapping without compensating chuck (G331 / G332)
- Axis mode (closed-loop position control)

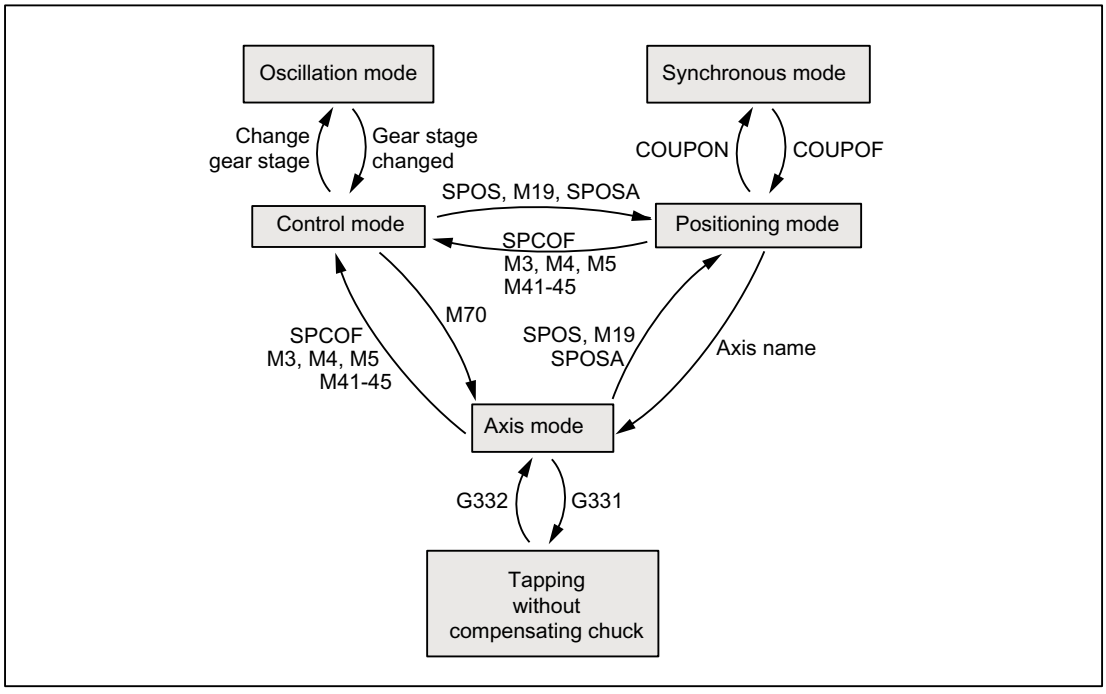
Note

Switchover, axis/spindle mode

A spindle can be switched over between the axis and spindle mode if the same motor is used for axis and spindle operation.

18.2.2 Mode change

Switching between spindle and axis operation can be done as follows:



- Control mode → Oscillation mode
  The spindle changes to oscillation mode if a new gear stage has been specified using automatic gear step selection (M40) in conjunction with a new S value or by M41 to M45. The spindle only changes to oscillation mode if the new gear step is not equal to the current actual gear step.
- Oscillation mode → Control mode
  When the new gear is engaged, the interface signal:
  DB31, ... DBX84.6 (Oscillation mode)
  is reset and the interface signal:
  DB31, ... DBX16.3 (Gear changed)
  is used to go to control mode.
  The last programmed spindle speed (S value) is reactivated.
- Control mode → Positioning mode
  To stop the spindle from rotation (M3 or M4) with orientation or to reorient it from standstill (M5), SPOS, M19 or SPOSA are used to switch to positioning mode.
- Positioning mode → Control mode
  M3, M4 or M5 are used to change to control mode if the orientation of the spindle is to be terminated. The last programmed spindle speed (S value) is reactivated.
- Positioning mode → Oscillation mode
  If the orientation of the spindle is to be terminated, M41 to M45 can be used to change to oscillation mode. When the gear change is complete, the last programmed spindle speed (S value) and M5 (control mode) are reactivated.

- Positioning mode → Axis mode
  If a spindle was stopped with orientation, the assigned axis name is used to program a change to axis mode. The gear step is retained.
- Control mode → Axis mode
  Switching from control mode to axis mode can be also achieved by the programming of M70. In this case, a rotating spindle is decelerated in the same way as for M5, position control activated and the zero parameter set selected.
- Axis mode → Control mode
  To terminate axis mode, M3, M4 or M5 can be used to change to control mode. The last programmed spindle speed (S value) is reactivated.
- Axis mode → Oscillation mode
  To terminate axis mode, M41 to M45 can be used to change to oscillation mode (only if the programmed gear step is not the same as the actual gear step). When the gear change is complete, the last programmed spindle speed (S value) and M5 (control mode) are reactivated.

18.2.3 Control mode

When open-loop control mode?

The spindle is in open-loop control mode with the following functions:

- Constant spindle speed:
  - S... M3/M4/M5 and G93, G94, G95, G97, G971
  - S... M3/M4/M5 and G33, G34, G35
  - S... M3/M4/M5 and G63
- Constant cutting speed:
  - G96/G961 S... M3/M4/M5

The spindle need not be synchronized.

Requirements

A spindle position actual value encoder is absolutely essential for M3/M4/M5 in connection with:

- Revolutional feedrate (G95)
- Constant cutting speed (G96, G961, G97, G971)
- Thread cutting (G33, G34, G35)
- Tapping without compensating chuck (G331, G332)
- Activate position control (SPCON, M70)

A spindle position actual value encoder is **not** required for M3/M4/M5 in connection with:

- Inverse-time feedrate coding (G93)
- Feedrate in mm/min or inch/min (G94)
- Tapping with compensating chuck (G63)

Speed control mode

Speed control mode is particularly suitable if a constant spindle speed is required, but the position of the spindle is not important (e.g. constant milling speed for even appearance of the workpiece surface).

- Speed control mode is activated in the part program with `M3`, `M4`, `M5` or with `SPCOF`.
- The following NC/PLC interface signal is set:
  DB31, ... DBX84.7 (control mode)
- NC/PLC IS:
  DB31, ... DBX61.5 (position controller active)
  is reset if position control is not used.
- Acceleration in speed control mode is defined independently of the gear stage in the machine data:
  MD35200 $MA_GEAR_STEP_SPEEDCTRL_ACCEL
  The value should reflect the physical circumstances, if possible.

Position control mode

Position control is particularly suitable if the position of the spindle needs to be tracked over a longer period or if synchronous spindle setpoint value linkage is to be activated.

- Position control mode is switched on in the part program with: `SPCON(<spindle number>)`
- The following NC/PLC interface signal is set:
  DB31, ... DBX61.5 (position controller active)
- Acceleration in position control mode is defined independent of the gear stage in the machine data:
  MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

Independent spindle reset

The spindle response after a reset or the end of the program (M2, M30) is set with the machine data:

MD35040 $MA_SPIND_ACTIVE_AFTER_RESET (individual spindle reset)

Value	Meaning
0	When the spindle is reset or at the end of the program, the spindle immediately decelerates to a stop at the active acceleration rate. The last programmed spindle speed and direction of rotation are deleted.
1	Upon reset or at the end of the program, the last programmed spindle speed (S-value) and the last programmed direction of spindle rotation (M3, M4, M5) are retained. The spindle is not braked.

If prior to reset or end of program the constant cutting speed (G96, G961) is active, the current spindle speed (in relation to 100% spindle override) is internally accepted as the spindle speed last programmed.

The spindle can only be stopped with the NC/PLC interface signal:

DB31, ... DBX2.2 (delete distance-to-go / spindle reset)

The direction of rotation is deleted in the event of all alarms generating a spindle quick stop. The last programmed spindle speed (S value) is retained. Once the source of the alarm has been eliminated, the spindle must be restarted.

Spindle actual speed display and spindle behavior with G96, G961

DB31, ... DBX61.4 (axis/spindle stationary)

The speed at which the spindle is deemed to be "stationary" is set with the machine data:

MD36060 $MA_STANDSTILL_VELO_TOL

The value should be measured in such a way that the following NC/PLC interface signal is reliably present at a standstill:

DB31,... DBX61.4 (axis/spindle stationary)

If DB31,... DBX61.4 (axis/spindle stationary) is signaled and there is no closed-loop position control active for the spindle, an actual speed of zero is displayed at the operator interface, and zero is read with the system variable $AA_S[<n>].

Spindle response at constant cutting speed G96, G961

- At the start of machining (transition from G0 to Gx) and after NC stop, G60 (exact stop, modal) and G9 (exact stop, non-modal) the system waits until the actual speed has reached the speed setpoint tolerance range before starting the path.
  DB31, ... DBX83.5 (nact = nset)
- The NC/PLC IS:
  DB31, ... DBX83.5 (nact = nset)
  and
  DB31, ... DBX83.1 (setpoint speed limited) are also set to defined values
  even if significant speed changes are specified (transverse axis travels towards position 0).

- If the speed drops below the minimum speed
  or when NC/PLC IS:
  DB31, ... DBX61.4 (axis/spindle stationary)
  is detected, NC/PLC IS:
  DB31, ... DBX83.5 (nact = nset)
  is reset (e.g. for an emergency machine strategy).
- A path operation which has started (G64, rounding), is not interrupted.

In addition, the spindle response is influenced by the following machine data:

MD35500 $MA_SPIND_ON_SPEED_AT_IPO_START (feed enable with spindle in setpoint range).

Spindle behavior at the end of gear stage change

- NC/PLC IS:
  DB31, ... DBX16.3 (gear changed)
  informs the NC that the new gear stage
  (NC/PLC IS DB31, ... DBX16.0-16.2 (actual gear stage A to C))
  applies and oscillation mode is terminated.
  In this case, it does not matter whether NC/PLC IS:
  DB31, ... DBX18.5 (oscillation mode)
  is still set.
  The actual gear stage should correspond to the set gear stage.
  The actual gear stage signaled is relevant for selection of the parameter set.
- Once the gear stage change (GSW) has been acknowledged via the PLC (DB31, ... DBX16.3), the spindle is in speed control mode (DB31, ... DBX84.7 = 1).
  If a direction of rotation (M3, M4, M5 or FC18: "Start spindle rotation") or a spindle speed (S value) was programmed before the gear stage change, then the last speed and direction of rotation will be reactivated after the gear stage change.

18.2.4 Oscillation mode

Oscillation mode is activated for the spindle during the gear step change.

The mode of operation is described in detail in the topic "Gear step change with oscillation mode (Page 794)".

18.2.5 Positioning mode

18.2.5.1 General functionality

When is positioning mode used?

The spindle positioning mode stops the spindle at the defined position and activates the position control, which remains active until it is de-activated.

For the following functions the spindle is in positioning mode:

- SPOS[<n>]=...
- SPOS[<n>]=ACP(...)
- SPOS[<n>]=ACN(...)
- SPOS[<n>]=AC(...)
- SPOS[<n>]=IC(...)
- SPOS[<n>]=DC(...)
- SPOSA[<n>]=ACP(...)
- SPOSA[<n>]=ACN(...)
- SPOSA[<n>]=AC(...)
- SPOSA[<n>]=IC(...)
- SPOSA[<n>]=DC(...) equal to SPOSA[<n>]=...
- M19 or M[<n>]=19

The address extension [<n>] with <n> = spindle number may not apply for the main spindle.

SPOS[<n>]=AC(...)

Spindle positioning to an absolute position (0 to 359.999 degrees). The positioning direction is determined either by the current direction of spindle rotation (spindle rotating) or the distance-to-go.

SPOS[<n>]=IC(...)

Spindle positioning to an incremental position (+/- 999999.99 degrees) in relation to the last programmed position. The positioning direction is defined by the sign of the path to be traversed.

SPOS[<n>]=DC(...)

Spindle positioning across the shortest path to an absolute position (0 to 359.999 degrees). The positioning direction is determined either by the current direction of spindle rotation (spindle rotating) or automatically by the control (spindle stationary).

SPOS[<n>]=...

Same functional sequence as SPOS [<n>]=DC(...).

SPOS[<n>]=ACP(...)

Approaches the position from the positive direction.

When positioning from a negative direction of rotation, the speed is decelerated to zero and accelerated in the opposite direction to execute the positive approach.

SPOS[<n>]=ACN(...)

Approaches the position from the negative direction.

When positioning from a positive direction of rotation, the speed is decelerated to zero and accelerated in the opposite direction to execute the negative approach.

M19 (DIN 66025)

M19 can be used to position the spindle. The position and the position approach mode are read here from the following setting data:

SD43240 $SA_M19_SPOS[<n>] (spindle position for spindle positioning with M19)

SD43250 $SA_M19_SPOSMODE[<n>] (spindle position for spindle positioning with M19)

The positioning options of M19 are identical to those of:

SPOS = <approach mode> <position/path>

M19 is output as an auxiliary function to the NC/PLC interface as an alternative to M3, M4, M5, and M70. The M19 block remains active in the interpolator for the duration of positioning (like SPOS).

Part programs using M19 as a macro (e.g. `DEFINE M19 AS SPOS = 0`) or as a subprogram, continue to remain executable. For the sake of compatibility with previous controls, the internal processing of M19 (NC positions the spindle) can be disabled as shown in the following example:

```
MD22000 $MC_AUXFU_ASSIGN_GROUP[0] = 4          ; Auxiliary function group: 4
MD22010 $MC_AUXFU_ASSIGN_TYPE[0] = "M"         ; Auxiliary function type: "M"
MD22020 $MC_AUXFU_ASSIGN_EXTENSION[0] = 0      ; Auxiliary functions Extension: 0
MD22030 $MC_AUXFU_ASSIGN_VALUE[0] = 19         ; Auxiliary function value: 19
```

Implicitly generated auxiliary function M19

To achieve uniformity in terms of how M19 and SPOS or SPOSA behave at the NC/PLC interface, auxiliary function M19 can be output to the NC/PLC interface in the event of SPOS and SPOSA.

The two following options are available for activating this function:

- Channel-specific activation for all the spindles in the channel via the machine data: MD20850 $MC_SPOS_TO_VDI (Output of M19 to the PLC with SPOS/SPOSA)

Bit	Value	Meaning
0	0	If bit 19 is also set to "0" in the MD35035 $MA_SPIND_FUNCTION_MASK, no auxiliary function M19 is generated in SPOS and SPOSA. This therefore eliminates the acknowledgment time for the auxiliary function.
	1	The auxiliary function M19 is generated and output to the PLC during the programming of SPOS and SPOSA in the part program. The address extension corresponds to the spindle number.

- Spindle-specific and cross-channel activation via the machine data:
  MD35035 $MA_SPIND_FUNCTION_MASK (spindle functions)

Bit	Value	Meaning
19	0	If bit 0 is also set to "0" in the MD20850 $MC_SPOS_TO_VDI, no auxiliary function M19 is generated in SPOS and SPOSA. This therefore eliminates the acknowledgment time for the auxiliary function.
	1	The implicit auxiliary function M19 is generated and output to the PLC during the programming of SPOS and SPOSA. The address extension corresponds to the spindle number.

Note

Activation via MD35035 should be preferred when using a spindle in multiple channels (axis/spindle exchange).

The auxiliary function M19 is implicitly generated if either of the MD configurations = 1.

After activation, the minimum duration of an SPOS/SPOSA block is increased to the time for output and acknowledgment of the auxiliary functions by the PLC.

The properties of the implicitly generated auxiliary function output M19 are "Quick" and "Output during motion". These properties are fixed settings and are independent of the M19 configuration in the auxiliary function-specific machine data (MD..._$M..._**AUXFU**_...).

There is **no** auxiliary function M19 implicitly generated in the case of spindle positioning commands via FC 18.

End of positioning

The positioning can be programmed with:

`FINEA[S<n>]:`	End of motion on reaching "Exact stop fine" (DB31, ... DBX60.7)
`COARSEA[S<n>]:`	End of motion on reaching "Exact stop coarse" (DB31, ... DBX60.6)
`IPOENDA[S<n>]:`	End of motion on reaching "IPO stop"

In addition, an end-of-motion criterion for block changes can be set in the braking ramp (100-0%) with IPOBRKA for single-axis interpolation.

Further information
P2: Positioning axes (Page 185)

Block change

The program advances to the next block if the end-of-motion criteria for all spindles or axes programmed in the current block, plus the block change criterion for path interpolation, are fulfilled. This applies to both part-program and technology-cycle blocks.

SPOS, M19 and SPOSA have the same functionality but differ in their block change behavior:

- SPOS and M19
  The block change is carried out if all functions programmed in the block have reached their end-of-block criterion (e.g. all auxiliary functions acknowledged by the PLC, all axes have reached their end points) and the spindle has completed its positioning motion.
- SPOSA
  The program moves to the next block if all the functions (except for spindle) programmed in the current block have reached their end-of-block criterion. If `SPOSA` is the only entry in the block, block change is performed immediately. The spindle positioning operation may be programmed over several blocks (see WAITS).

Coordination

A coordination of the sequence of motions can be achieved with:

- PLC
- MD configuration
- Programming in the part program

PLC

If the NC/PLC interface signal:
DB31, ... DBX83.5 (spindle in the setpoint range)
is not available, then the channel-specific NC/PLC interface signal:
DB21, ... DBX 6.1 (read-in inhibit)
can be set in order to wait for a spindle to reach a certain position.

MD configuration

Setting:
MD35500 $MA_SPIND_ON_SPEED_AT_IPO_START = 1
is used to perform path interpolation taking the tolerance:
MD35150 $MA_SPIND_DES_VELO_TOL
into account only if the spindle has rotated up to the preselected speed.

The setting
MD35500 $MA_SPIND_ON_SPEED_AT_IPO_START = 2
is used to stop traveling path axes before the start of machining at the last G0.

Machining continues:

- With the next traversing command.
- If the spindle speed is reached.
- When MD35510 $MA_SPIND_STOPPED_AT_IPO_START = 1
  (path feed enable, if spindle stationary).

Programming in the part program

Coordination actions in the part program have the following advantages:

- The part program author can decide at what point in the program the spindle needs to be up to speed, e.g. in order to start machining a workpiece.
- This avoids unnecessary delays.

Coordination in the part program involves programming the `WAITS` command:

`WAITS`:	for main spindle (master spindle)
`WAITS[<n>]`:	for spindles with number <n>
`WAITS[<n>,<m>,...]`:	for several spindles up to the maximum number of spindles

⚠ CAUTION

Coordination error

The part program author must ensure that one of the following maintenance conditions occurs for WAITS.

- Position reached
- Spindle stationary
- Spindle up to programmed speed

In cases where one spindle is used in several channels, the part program author must ensure that WAITS starts at the earliest in the phase in which the spindle from another channel has already started to accelerate or decelerate towards the required new speed or direction of rotation.

The control waits before executing subsequent blocks until:

- Position(s) programmed with `SPOSA` are reached.
- Spindle standstill is reached with M5:
  DB31, ..., DBX 61.4 (spindle stationary)
  taking into account the tolerance:
  MD36060 $MA_STANDSTILL_VELO_TOL
  WAITS is terminated and the next block loaded when the first occurrence of the signal is detected.
- In M3/M4 (speed control mode), the speed in the setpoint range is:
  DB31, ..., DBX83.5 (spindle in setpoint range)
  taking into account the tolerance:
  MD35150 $MA_SPIND_DES_VELO_TOL
  WAITS is terminated and the next block loaded when the first occurrence of the signal is detected.
  This WAITS function applies in the programmed channel.
  WAITS can be used to wait for all spindles known to this channel, although spindles may also have been started in other channels.

Special cases

- **Tolerance for spindle speed:**
  If the machine data setting is:
  MD35150 $MA_SPIND_DES_VELO_TOL = 0
  the NC/PLC interface signal:
  DB31, ... DBX83.5 (spindle in setpoint range)
  is always set to 1.
  WAITS is terminated as soon as the spindle has reached the setpoint-side target after a change in speed or direction (M3/M4).
- **Missing enable signals:**
  If the WAITS function waits for the "Spindle in setpoint range" signal in speed control mode and the spindle stops or fails to rotate because an enable signal (axial feed enable, controller, pulse enable, etc.) is missing, the block is not terminated until the "Spindle in setpoint range" signal is active, once enable signals are being received again.
- **Response to NC and mode-group stop:**
  If an NC or mode-group stop is triggered during WAITS, the wait operation is resumed after the NC start with all the above conditions.

Note

In particular when using spindles across different channels, care should be taken when programming not to start WAITS too early in one channel, i.e. at a time when the spindle in the other channel is still rotating at its "old" speed.
In such cases, the "Spindle in setpoint range" signal is activated and WAITS is stopped too soon.
To prevent this happening, it is strongly recommended to set a WAITM before WAITS.

Feedrate

The positioning speed is configured in the machine data:

MD35300 $MA_SPIND_POSCTRL_VELO (position control switching speed)

The configured positioning speed can be modified by programming or by synchronized actions:
`FA[S<n>]=<value>`

With: `<n>`: Spindle number
`<value>`: Positioning speed in degrees/min

With `FA[S<n>]=0`, the configured speed takes effect.

Acceleration

The accelerations are configured in the machine data:

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL (acceleration in position control mode)

MD35200 $MA_GEAR_STEP_SPEEDCTRL_ACCEL (acceleration in the speed control mode)

The configured dynamic response during positioning can be modified by programming or by synchronized actions:
`ACC[S<n>]=<value>`

With: <n>: Spindle number
<value>: Acceleration as a percentage of the configured acceleration

With ACC[S<n>]=0, the configured acceleration takes effect.

Aborting the positioning process

The positioning action is aborted:

- By the NC/PLC interface signal:
  DB31, ... DBX2.2 (delete distance-to-go / spindle reset).
- With every reset (e.g. operator panel front reset).
- Through NC stop.

The abort response in independent of the machine data:

MD35040 $MA_SPIND_ACTIVE_AFTER_RESET (individual spindle reset)

Special features

The spindle override switch is valid.

18.2.5.2 Positioning from rotation

Initial situation

The spindle can be in speed control mode or in position control mode when positioning starts (SPOS, M19 or SPOSA command in the program).

One must distinguish between the following cases:

Case 1:	Spindle in speed control mode, encoder limit frequency exceeded
Case 2:	Spindle in speed control mode, encoder limit frequency not exceeded
Case 3:	Spindle in position control mode
Case 4:	Spindle speed ‹ Position-control activation speed

Procedure

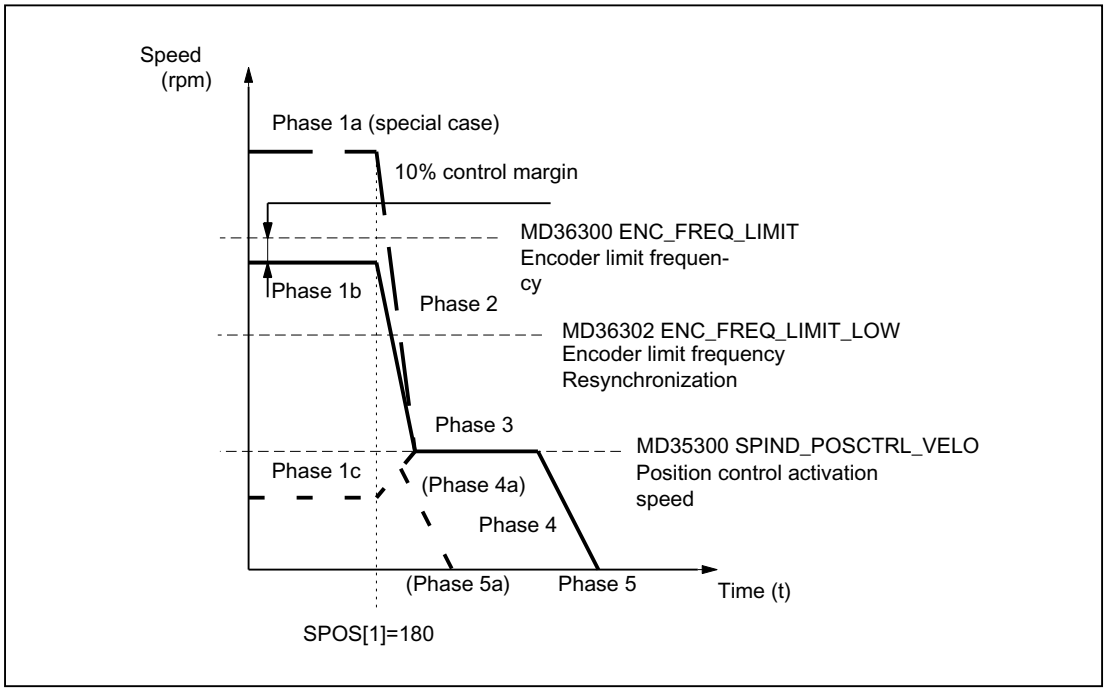


Figure 18-1 Positioning from rotation

Note

The speed arising from the configuration of the encoder limit frequency for the resynchronization of the encoder (MD36302 $MA_ENC_FREQ_LOW) must be greater than the position-control activation speed (MD35300 $MA_SPIND_POSCTRL_VELO).

Phase 1

Positioning from phase 1a:

The spindle is rotating at a higher speed than the encoder limit frequency. The spindle is not synchronized.

Positioning from phase 1b:

The spindle is rotating at a lower speed than the encoder limit frequency. The spindle is synchronized.

Note

If the position control is active, the speed can only amount to 90% of the maximum speed of the spindle or the encoder limit frequency (10% control reserve required).

Positioning from phase 1c:

The spindle rotates at the programmed spindle speed whereby the speed is lower than the configured position-control activation speed:

MD35300 $MA_SPIND_POSCTRL_VELO

The spindle is synchronized.

Phase 2

Spindle speed > Position-control activation speed

When the SPOS, M19 or SPOSA command is activated, the spindle begins to slow down to the position-control activation speed with the configured acceleration:

MD35200 $MA_GEAR_STEP_SPEEDCTL_ACCEL

The spindle is synchronized once the encoder limit frequency threshold is crossed.

Spindle speed < Position-control activation speed

SPOS, M19 or SPOSA are programmed to switch the spindle to position control mode (if it is not already in that mode).

The configured acceleration in position control mode is activated:

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

The travel path to the target point is calculated.

The spindle travels to the programmed end point optimally in terms of time. This means that the end point is approached at the highest possible speed (maximum MD35300 $MA_SPIND_POSCTRL_VELO). Depending on the appropriate secondary conditions, phases 2 - 3 - 4 - 5 or 4a - 5a are executed.

Phase 3

Spindle speed > Position-control activation speed

When the configured position-control activation speed (MD35300 $MA_SPIND_POSCTRL_VELO) is reached:

- Position control is activated (if not already active).
- The distancetogo (to the target point) is calculated.
- There is a switch to the configured configured acceleration in position control mode (or this acceleration is retained):
  MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

Spindle speed < Position-control activation speed

The spindle was accelerated up to the configured position-control activation speed (MD35300 $MA_SPIND_POSCTRL_VELO) to reach the end point. This is not exceeded.

The braking start point calculation detects when the programmed spindle position can be approached accurately at the acceleration configured in position control mode (MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL).

Phase 4

Spindle speed > Position-control activation speed

The spindle brakes from the calculated "braking point" with machine data:
MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL
to the target position.

Spindle speed < Position-control activation speed

At the time identified by the braking start point calculation in phase 3, the spindle brakes to a standstill with the acceleration configured in position control mode (MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL).

Phase 4a:

When the `SPOS` command is activated the proximity of the end point is such that the spindle can no longer be accelerated to the configured position-control activation speed (MD35300 $MA_SPIND_POSCTRL_VELO).

The spindle brakes to a standstill with the acceleration configured in position control mode (MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL).

Phase 5

Spindle speed > Position-control activation speed

Position control remains active and holds the spindle in the programmed position.

Note

The maximum encoder limit frequency of the spindle position actual-value encoder is monitored by the control (it may be exceeded); in position control mode, the setpoint speed is reduced to 90% of the measuring system limit speed.

The following NC/PLC interface signal is set:

DB31, ... DBX83.1 (programmed speed too high)

If "MS limit frequency exceeded" is still pending following a reduction in the setpoint speed, an alarm is output.

Spindle speed < Position-control activation speed (Phase 5, 5a)

The spindle is stationary and it has reached the position. The position control is active and stops the spindle in the programmed position.

If the distance between the spindle actual position and the programmed position (spindle setpoint position) is less than the configured exact stop fine and coarse limits, the following NC/PLC interface signals are set:

DB31, ... DBX60.7 (Position reached with coarse exact stop)

DB31, ... DBX60.7 (Position reached with fine exact stop)

The exact stop limits are defined with the machine data:

MD36010 $MA_STOP_LIMIT_FINE (exact stop fine)

MD36000 $MA_STOP_LIMIT_COARSE (exact stop coarse)

Note

The positioning procedure is considered complete when the end-of-positioning criterion is reached and signaled.

The condition is "Exact stop fine". This applies to `SPOS`, `M19` or `SPOSA` from the part program, synchronized actions and spindle positioning by the PLC using FC 18.

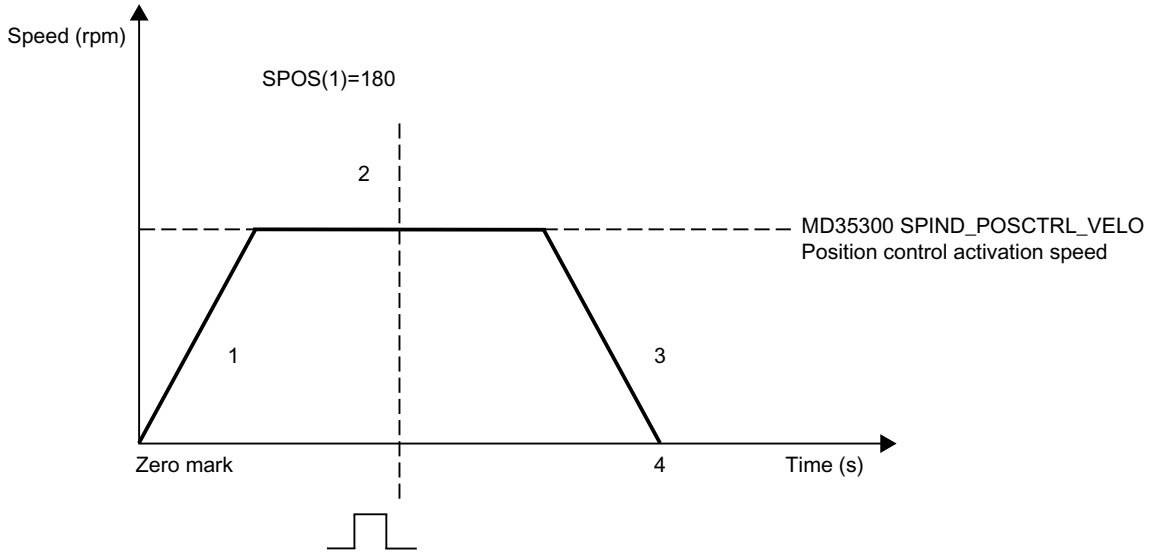
18.2.5.3 Positioning from standstill

Procedure

A distinction is made between two cases with regard to positioning from standstill:

- Case 1: The spindle is not synchronized.
  This is the case if the spindle is to be positioned after switching on the control and drive or after a gear step change (e.g. for a tool change).
  MD31040 $MA_ENC_IS_DIRECT = 0
- Case 2: The spindle is synchronized.
  This is the case if, after switching on the control and drive, the spindle is to be rotated through a minimum of one revolution with M3 or M4 and then stopped with M5 (synchronization with the zero mark) before the first positioning action.

Case 1: Spindle not synchronized



Case 1: Spindle synchronized

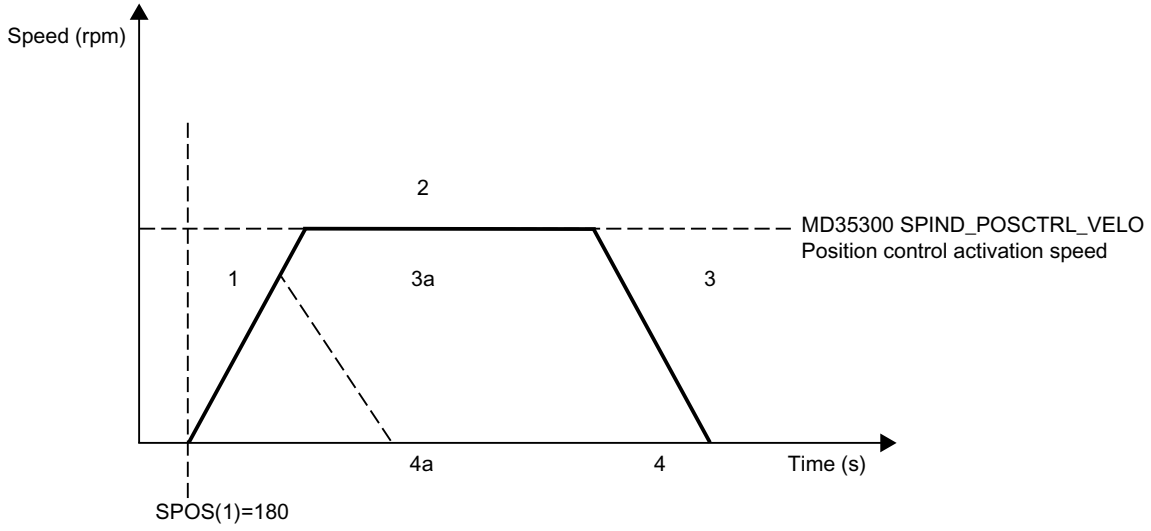


Figure 18-2 Positioning with stationary spindle

Phase 1

Case 1: Spindle not synchronized

With the programming of `SPOS`, `M19` or `SPOSA` the spindle accelerates with the acceleration from the machine data:

MD35200 $MA_GEAR_STEP_SPEEDCTRL_ACCEL (Acceleration in the speed control mode)

This direction of rotation is defined by the machine data:

MD35350 $MA_SPIND_POSITIONING_DIR (Direction of rotation while positioning to standstill)

Exception:

If ACN, ACP, IC is used for positioning, the programmed direction of travel is activated.

The spindle is synchronized at the next zero mark of the spindle position actual-value encoder and switches to the position control mode.

Whether the zero mark is found in the traversed path (except for IC), is monitored:

MD34060 $MA_REFP_MAX_MARKER_DIST (maximum distance to the reference mark)

If the speed defined in machine data:
MD35300 $MA_SPIND_POSCTRL_VELO (Positioning speed)
is reached before the spindle is synchronized, the spindle will continue to rotate at the positioning activation speed (it is not accelerated further).

Case 2: Spindle synchronized

SPOS, M19 or SPOSA will switch the spindle to position control mode.

The acceleration from the following machine data is active:

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL (acceleration in position control mode)

The direction of rotation is defined by the programmed motion (ACP, ACN, IC, DC) or via the pending distance-to-go.

The speed entered in
MD35300 $MA_SPIND_POSCTRL_VELO (position control activation speed)
is not exceeded in the machine data.

The travel path to the end position is calculated.

The spindle travels to the programmed end point optimally in terms of time. This means that the end point is approached at the highest possible speed (maximum MD35300 $MA_SPIND_POSCTRL_VELO). Depending on the appropriate secondary conditions, the phases 1 - 2 - 3 - 4 or 1- 3a - 4a are executed.

Phase 2

Case 1: Spindle not synchronized

When the spindle is synchronized, position control is activated.

The spindle rotates at the maximum speed stored in machine data:
MD35300 $MA_SPIND_POSCTRL_VELO
until the braking start point calculation identifies the point at which the programmed spindle position can be approached accurately with the defined acceleration.

Case 2: Spindle synchronized

To reach the end point, the spindle is accelerated up to the speed defined in machine data:
MD35300 $MA_SPIND_POSCTRL_VELO.

This is not exceeded.

The braking start point calculation identifies when the programmed spindle position can be approached accurately at the acceleration defined in machine data:

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL.

At the point, which is determined by the braking start point calculation in Phase 1, the spindle decelerates to a standstill with the acceleration given in the following machine data:

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

Phase 3

At the point, which is determined by the braking start point calculation in Phase 2, the spindle decelerates to a standstill with the acceleration given in the following machine data:

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

Phase 3a:

When the `SPOS` command is activated the proximity of the end point is such that the spindle can no longer be accelerated up to machine data:
MD35300 $MA_SPIND_POSCTRL_VELO.

The spindle is braked to a standstill with the acceleration given in the following machine data:

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

Phase 4, 4a

The spindle is stationary and it has reached the position. The position control is active and stops the spindle in the programmed position.

NC/PLC IS:
DB31, ... DBX60.6 (Position reached with exact stop coarse)
and
DB31, ... DBX60.7 (Position reached with exact stop fine)
are set if the distance between the spindle actual position and the programmed position (spindle setpoint position) is less than the settings for the exact stop fine and coarse limits.

This is defined in the machine data:

MD36010 $MA_STOP_LIMIT_FINE

MD36000 $MA_STOP_LIMIT_COARSE

Phase 3:

At the point, which is determined by the braking start point calculation in Phase 2, the spindle decelerates to a standstill with the acceleration given in the following machine data:

MD35210 $MA_GEAR_STEP_ POSCTRL_ACCEL

Phase 4:

The spindle is stationary and it has reached the position. The position control is active and stops the spindle in the programmed position.

NC/PLC IS:
DB31, ... DBX60.6 (Position reached with exact stop coarse)
and
DB31, ... DBX60.7 (Position reached with exact stop fine)

are set if the distance between the spindle actual position and the programmed position (spindle setpoint position) is less than the settings for the exact stop fine and coarse limits.

This is defined in the machine data:

MD36010 $MA_STOP_LIMIT_FINE

MD36000 $MA_STOP_LIMIT_COARSE

18.2.5.4 "Spindle in position" signal for tool change

Function

The motion sequence for a tool change, especially for milling machines, mainly comprises positioning the spindle and then the subsequent (for optimization runs, also at the same time) approach to the tool change position with the path axes. In this case, the mandatory requirement is that the spindle is reached before approaching the tool change position.

If the tool change cycle is interrupted by the machine operator (e.g. with an NC stop, NC stop axes plus spindles, mode group stop, etc.), then it must be completely ruled out that the spindle moves into the tool changer at an incorrect position.

This is the reason that for spindle positioning, when the last programmed spindle position is reached with "Exact stop fine" the following NC/PLC interface signal is output to check the position:

DB31, ... DBX85.5 (spindle in position)

Note

The signal is only output for the "Spindle positioning" function.

This includes:

- SPOS, SPOSA and M19 in the part program
- SPOS and M19 in synchronized actions
- Spindle positioning, using FC18
- Spindle positioning via PLC interface (DB31, ... DBX30.4)

Setting the signal

Requirements for output of signal DB31, ... DBX85.5 (spindle in position) are as follows:

- The referenced state of the spindle:
  DB31, ... DBX60.4/5 (referenced/synchronized 1/2) = 1

  Note

  When positioning the spindle, the zero mark is automatically searched for. This is the reason that for an error-free sequence, the referenced signal is always available at the end of positioning movement.

- "Exact stop fine" must have been reached:
  DB31, ... DBX60.7 (exact stop fine) = 1
  Additionally, the last programmed spindle position must have been reached on the setpoint side.

Deleting the signal

When signal DB31, ... DBX60.7 is withdrawn (exact stop fine), then signal DB31, ... DBX85.5 (spindle in position) is also always reset.

Additional properties

- If the spindle is already at the programmed position after a positioning, then the NC/PLC interface signal DB31, ... DBX85.5 (spindle in position) remains set.
- If, after a positioning ("Spindle in position" signal was output) the spindle is traversed, e.g. in the JOG mode, then the NC/PLC interface signal DB31, ... DBX85.5 (spindle in position) is deleted.
  If the spindle is returned to its original position in this mode, then the NC/PLC interface signal DB31, ... DBX85.5 (spindle in position) is set again. The last position selection is kept.

18.2.6 Axis mode

18.2.6.1 General functionality

Functionality

If for certain machining tasks, e.g. on lathes with end-face machining, it is not sufficient to traverse the spindle exclusively under speed control via `M3`, `M4`, `M5` or to position with `SPOS`, `M19` or `SPOSA`, the spindle can be switched to position-controlled axis mode and traversed as a rotary axis.

Examples of rotary axis functions:

- Programming with axis name
- Zero offsets (`G54, G55`, `TRANS`, etc.)
- `G90`, `G91`, `IC`, `AC`, `DC`, `ACP`, `ACN`

- Kinematic transformations (e.g. TRANSMIT)
- Path interpolation
- Traversing as positioning axis

Further information
R2: Rotary axes (Page 221)

Requirements

- The same spindle motor is used for spindle mode and axis mode.
- The same position measurement system or separate position measurement systems can be used for spindle mode and axis mode.
- An actual position value encoder is a mandatory requirement for axis mode.
- The spindle must be referenced for use of the axis mode, e.g. referenced with `G74`. Example:

Program code	Comment
`M70`	`; Switch spindle over to axis mode`
`G74 C1=0 Z100`	`; Reference axis`
`G0 C180 X50`	`; Traverse axis position-controlled`

Configurable M function

The M function used to switch the spindle to axis mode can be configured channel-specifically via the following machine data:

MD20094 $MC_SPIND_RIGID_TAPPING_M_NR

Note

The control detects the transition to axis mode automatically from the program sequence (see "Implicit transition to axis mode (Page 759)"). The explicit programming of the configured M function for switching the spindle to axis mode in the part program is therefore not necessary. However, the M function can continue to be programmed, e.g. to increase the readability of the part program.

Special features

- The feedrate override switch is valid.
- The NC/PLC interface signal does not terminate the axis mode per default:
  DB21, ... DBX7.7 (reset).
- The NC/PLC interface signals:
  DB31, ... DBB16 to DBB19 and DBB82 to DBB91
  are not important if:
  DB31, ... DBX60.0 == 0 (spindle / rotary axis)

- Axis mode can be activated in all gear stages.
  If the actual position value encoder is installed on the motor (indirect measuring system), the positioning and contouring accuracy can vary for the different gear stages.
- The gear stage cannot be changed when the axis mode is active. For this purpose, the spindle must first be switched to control mode with `M41` ... `M45` or `M5`, `SPCOF`.
- In axis mode, the first parameter set is effective (machine data index = zero).
  Further information
  G2: Velocities, setpoint / actual value systems, closed-loop control (Page 31)

Dynamic response

The dynamic limits of the axis apply in axis mode. For example:

- MD32000 $MA_MAX_AX_VELO[<axis>] (maximum axis velocity)
- MD32300 $MA_MAX_AX_ACCEL[<axis>] (maximum axis acceleration)
- MD32431 $MA_MAX_AX_JERK[<axis>] (maximum axial jerk for path motion)

Feedforward control

The feedforward control mode active for the axis is retained.

A detailed description of the "Dynamic feedforward control" function can be found in:

Further information

Function Manual Monitoring and Compensating; Compensations > Dynamic feedforward control (following error compensation)

Example: Resolution switchover for analog actuator

Switching to axis mode

Programming	Comment
SPOS=...	
M5	; Control enable off (from PLC) → is output on PLC
M70	; Switch actuator (from PLC on account of M70) Control enable on (from PLC)
C=...	; NC traverses with axis parameter set

Switching to spindle mode

Programming	Comment
C=...	
M71	; → is output to PLC
	Closed-loop controller enable off (from PLC)
	Switch actuator (from PLC)
	Switched to spindle parameter set (1-5) internally in the NC, controller enable on (from PLC)
M3/4/5 or SPOS=...	; NC traverses with spindle parameter set

Change to spindle mode

The parameter set 1 ... 5 is selected for the active gear stage.

The feedforward control is activated, except for tapping with compensating chuck, if the following applies:

MD32620 $MA_FFW_MODE (feedforward control mode) ≠ 0

Parameter set	Axis mode	Spindle mode
1	Valid	-
2	-	Valid
3	-	Valid
4	-	Valid
5	-	Valid
6	-	Valid
Spindle mode: Parameter set according to the gear stage		

18.2.6.2 Implicit transition to axis mode

Function

The control detects the transition to axis mode automatically from the program sequence and generates the requisite M70 sequence in the control. The situation will dictate which steps are performed. At most, these will include:

1. Stopping the spindle
2. Switching on of the position control, treatment of feedforward control and parameter block changeover
3. Position synchronization of the block preparation (internal preprocessing stop, if necessary)

This function is always active. Explicit programming of M70 in the part program is therefore basically not necessary.

Sequence

Sequence of the implicit transition to axis mode (M70 is not programmed in the part program):

- Transition from speed control mode (M3, M4, M5, SPCOF, ...) to axis mode:
  The transition is detected internally by the control, and an intermediate block is inserted in front of the block which requests the axis mode. The block created contains the M70 functionality. The execution duration for this block is more or less the same as the time required to execute a programmed M70 block. Differences may arise in the event of short switchovers when the spindle is stationary (no braking time) if the implicit generation and output of the auxiliary function M70 to the PLC is dispensed with (see MD35035).
- Transition from positioning mode (M19, SPOS, SPOSA) to axis mode:
  The transition is executed immediately and without the generation of an intermediate block. Configured accordingly (see MD35035), the auxiliary function M70, which is generated implicitly, is output to the PLC when the block in which the spindle has its axis mode is loaded.

Output of auxiliary functions to PLC

The implicit transition to axis mode can be notified to the PLC in the form of an auxiliary function output.

Activation/deactivation

The activation/deactivation of this functionality is performed via the machine data:

MD35035 $MA_SPIND_FUNCTION_MASK (spindle functions)

Bit	Value	Meaning
20	0	No auxiliary function output to the PLC in the case of M70 functionality which is generated in the control.
	1	In the case of M70 functionality which is generated inside the control, the auxiliary function M70 is generated and output to the PLC. The address extension corresponds to the spindle number.

Note

An auxiliary function M70 which is programmed in the part program is always output to the PLC.

Properties

The properties of the implicitly generated auxiliary function output M70 are "Quick" and "Output during motion". These properties are fixed settings and are independent of the M70 configuration in the auxiliary-function-specific machine data (MD..._$M..._**AUXFU**_...).

M70 is only generated once during transition to axis mode. No further M70 auxiliary functions are generated and output in adjacent blocks in which the spindle is operated as an axis. M70 is not implicitly generated and output again until axis mode is exited via, for example, SPOS, M3, M4, M5, SPCOF, etc. and following a renewed transition to axis mode.

Supplementary conditions

Synchronized actions

When the spindle is programmed as an axis in synchronized actions, it is essential to continue making provisions in the application to ensure there are criteria for the transition to axis mode.

If the spindle is in speed control mode, the instruction `M70` or `SPOS` must be programmed prior to programming as an axis. Otherwise alarm signals occur during axis programming.

FC 18

As with synchronized actions, transition to axis mode must also be performed on the application side in FC 18, e.g. through a preparatory positioning instruction. Otherwise, the FC 18 call is acknowledged with an error bit in the FC 18 status word.

No auxiliary function M70 is implicitly generated in the event of transition to axis mode through programming via FC 18.

Examples

Example 1:

Part program: Transition from rotating spindle to axis mode

Configuration: MD35035 $MA_SPIND_FUNCTION_MASK, bit 20 = 1

Program code	Comment
N05 M3 S1000	
N10 ...	
N15 POS[C]=77	: Before loading N15, an M70 intermediate block is generated in which the spindle is stopped, and M70 is output to the PLC.
…	

Example 2:

Part program: Transition from positioning mode to axis mode

Configuration: MD35035 $MA_SPIND_FUNCTION_MASK, bit 20 = 1

Program code	Comment
N05 SPOS=0	
N10 ...	
N15 C77	; Output of the implicit M70 to the PLC, no intermediate block.
…	

Example 3:

Synchronized actions: Transition from spindle positioning mode to axis mode

Configuration: MD35035 $MA_SPIND_FUNCTION_MASK, bit 20 = 1

Program code	Comment
WHEN COND1==TRUE DO SPOS=180	
WHEN COND2==TRUE DO POS[C]=270	; Output of the implicit M70 to the PLC.

Example 4:

Synchronized actions: Transition from speed control mode to axis mode with M70

Configuration: MD35035 $MA_SPIND_FUNCTION_MASK, bit 20 = 1

Program code	Comment
WHEN COND11==TRUE DO M3 S1000	
WHEN COND12==TRUE DO M70	; Output of M70 to the PLC.
WHEN COND13==TRUE DO POS[C]=270	; No generation of an implicit M70 because axis mode already exists.

Example 5:

Synchronized actions: Invalid transition from speed control mode to axis mode

Configuration: MD35035 $MA_SPIND_FUNCTION_MASK, bit 20 = 1

```
Program code                                Comment
WHEN COND21==TRUE DO M3 S1000
WHEN COND22==TRUE DO POS[C]=270             ; Alarm 20141!
```

18.2.7 Initial spindle state

Spindle basic setting

The following machine data is used to specify a spindle mode as basic setting:

MD35020 $MA_SPIND_DEFAULT_MODE

Value	Spindle basic setting
0	Speed control mode, position control deselected
1	Speed control mode, position control activated
2	Positioning mode
3	Axis mode

Time when the spindle basic setting takes effect

The time when the spindle basic setting takes effect is set in the machine data:

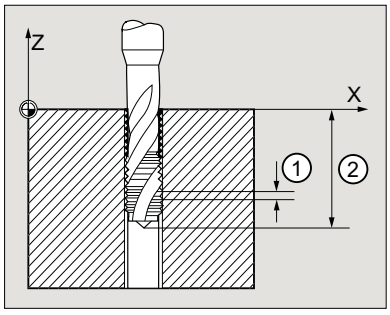
MD35030 $MA_SPIND_DEFAULT_ACT_MASK

Value	Effective time
0	POWER ON
1	POWER ON and program start
2	POWER ON and RESET (M2 / M30)

18.2.8 Tapping without compensating chuck

18.2.8.1 Function

For tapping without compensating chuck, the traversing motion of the linear axis and the spindle are interpolated, closed-loop position controlled. This requires a **position-controlled spindle** with **position measuring system**.



① Thread pitch
② Thread depth

Thread direction of rotation

The thread direction of rotation (right or left-handed thread) is defined by the sign of the pitch:

- Positive pitch → right-handed thread (direction of rotation corresponding to `M3`)
- Negative pitch → left-handed thread (direction of rotation corresponding to `M4`)

Defined spindle start position

When tapping without compensating chuck, a defined spindle start position (`SPOS`) must be programmed in the following cases:

- Threads produced using multiple machining operations
- Threads, for which a defined thread start position is specified

```
SPOS = <start position>
G331 ...
```

18.2.8.2 Programming

For tapping without compensating chuck, using the `G331` and `G332`commands, the following traversing motion is executed:

- `G331`: Tapping in the tapping direction up to the end of thread point
- `G332`: Retraction motion up to the tapping block `G331` with automatic spindle direction of rotation reversal

Syntax

```
G331 <axis> <thread pitch>
G331 <axis> <thread pitch> S...
```

```
G332 <axis> <thread pitch>
```

Meaning

G331:	Tapping The tapped hole is defined by the traversing motion of the axis (drilling depth) and the thread pitch.	
	Effectiveness:	Modal
G332:	Retraction motion when tapping Retraction motion must have the same pitch as when tapping (G331). The direction of rotation of the spindle is reversed automatically.	
	Effectiveness:	Modal
<axis>:	Traversing distance/position of the geometry axis (X, Y or Z) at the end of the thread, e.g. Z50	
<Thread pitch>:	Thread pitch I (X), J (Y) or K (Z): • Positive pitch: Right-handed thread, e.g. K1.25 • Negative pitch: Left-handed thread, e.g. K-1.25	
	Range of values:	±0.001 to ±2000.00 mm/revolution
S...:	Spindle speed The last active spindle speed is used if a spindle speed is not specified.	

Note

Second gear-stage data block

To achieve effective adaptation of spindle speed and motor torque and be able to accelerate faster, a second gear-stage data block for two further configurable switching thresholds (maximum speed and minimum speed) can be preset in axis-specific machine data deviating from the first gear step data block and also independent of these speed switching thresholds. The specifications of the machine manufacturer must be observed.

For further information see the "Axes and Spindles" Function Manual.

Examples

- Example: Tapping with G331 / G332 (Page 765)
- Example: Output the programmed drilling speed in the current gear stage (Page 765)
- Example: Application of the second gear-stage data block (Page 766)
- Example: Speed is not programmed, the gearbox stage is monitored (Page 766)
- Example: Gearbox stage cannot be changed, gearbox stage monitoring (Page 766)
- Example: Programming without SPOS (Page 767)

See also

Configurable gear adaptations (Page 775)

18.2.8.3 Example: Tapping with G331 / G332

Program code	Comment
N10 SPOS[n]=0	; Spindle: Position control mode
	; Start position 0 degrees
N20 G0 X0 Y0 Z2	; Axes: Approach starting position
N30 G331 Z-50 K-4 S200	; Tapping in Z,
	; Pitch K-4 negative =>
	; Direction of spindle rotation: CCW rotation,
	; Spindle speed 200 rpm
N40 G332 Z3 K-4	; Retraction motion in Z,
	; Pitch K-4 negative (counterclock-wise),
	; autom. direction of rotation reversal =>
	; Clockwise spindle direction of rota-tion
N50 G1 F1000 X100 Y100 Z100 S300 M3	; Spindle in spindle operation

18.2.8.4 Example: Output the programmed drilling speed in the current gear stage

Program code	Comment
N05 M40 S500	; Programmed spindle speed: 500 rpm =>
	; Gearbox stage 1 (20 to 1028 rpm)
...	
N55 SPOS=0	; Position the spindle
N60 G331 Z-10 K5 S800	; Tapping
	; Spindle speed 800 rpm => gearbox stage 1

The appropriate gear stage for the programmed spindle speed `S500` with `M40` is determined on the basis of the first gear-stage data block. The programmed drilling speed `S800` is output in the current gear stage and, if necessary, is limited to the maximum speed of the gear stage. No automatic gear-stage change is possible following an `SPOS` operation. In order for an automatic change in gear stage to be performed, the spindle must be in speed-control mode.

Note

If gearbox stage 2 is selected at a spindle speed of 800 rpm, then the switching thresholds for the maximum and minimum speed must be configured in the relevant machine data of the second gear-stage data block (see the examples below).

18.2.8.5 Example: Application of the second gear-stage data block

The switching thresholds of the second gear-stage data block for the maximum and minimum speed are evaluated for G331/G332 and when programming an S value for the active master spindle. Automatic M40 gear-stage change must be active. The gear stage as determined in the manner described above is compared with the active gear stage. If they are found to be different, then the gearbox stage is changed.

Program code	Comment
N05 M40 S500	; Programmed spindle speed: 500 rpm
...	
N50 G331 S800	; Master spindle: Gearbox stage 2 is selected
N55 SPOS=0	; Position the spindle
N60 G331 Z-10 K5	; Tapping
	; Spindle acceleration from second gearbox stage data block 2

18.2.8.6 Example: Speed is not programmed, the gearbox stage is monitored

If no speed is programmed when using the second gearbox stage data block with G331, then the last speed programmed will be used to produce the thread. The gear stage does not change. However, monitoring is performed in this case to check that the last speed programmed is within the preset speed range (defined by the maximum and minimum speed thresholds) for the active gear stage. Otherwise, alarm 16748 is output.

Program code	Comment
N05 M40 S800	; Programmed spindle speed: 800 rpm
...	
N55 SPOS=0	; Position the spindle
N60 G331 Z-10 K5	; Tapping
	; Monitoring the spindle speed, 800 rpm
	; Gearbox stage 1 is active
	; Gearbox stage 2 should be active => Alarm 16748

18.2.8.7 Example: Gearbox stage cannot be changed, gearbox stage monitoring

If the spindle speed is programmed in addition to the geometry in the G331 block when using the second gear-stage data block, if the speed is not within the preset speed range (defined by the maximum and minimum speed thresholds) of the active gear stage, it will not be possible to change gear stages, because the path motion of the spindle and the infeed axis (axes) would not be retained.

As in the example above, the speed and gearbox stage are monitored in the G331 block and alarm 16748 is signaled if necessary.

Program code	Comment
N05 M40 S500	; Programmed spindle speed: 500 rpm =>
	; Gearbox stage 1
...	
N55 SPOS=0	; Position the spindle

Program code	Comment
N60 G331 Z-10 K5 S800	; Tapping
	; Gearbox stage cannot be changed,
	; Monitoring the spindle speed, 800 rpm
	; with gearbox stage data set 1: Gearbox stage 2
	; should be active => Alarm 16748

18.2.8.8 Example: Programming without SPOS

Program code	Comment
N05 M40 S500	; Programmed spindle speed: 500 rpm =>
	; Gearbox stage 1 (20 to 1028 rpm)
...	
N50 G331 S800	; Master spindle: Gearbox stage 2 is selected
N60 G331 Z-10 K5	; Tapping
	; Spindle acceleration from second gearbox stage data block 2

Thread interpolation for the spindle starts from the current position, which is determined by the previously processed section of the part program, e.g. if the gear stage was changed. Therefore, it might not be possible to remachine the thread.

Note

Please note that when machining with multiple spindles, the drill spindle also has to be the master spindle. SETMS(<spindle number>) can be programmed to set the drill spindle as the master spindle.

18.2.8.9 Special case: Direction of rotation reversal via NC/PLC interface signal in the NC program

Normally, a request to invert the direction of rotation of the spindle via the NC/PLC interface signal "Invert M3 / M4" (DB31, ... DBX17.6) is realized by the PLC user program **before** the NC program is started. Only when starting does the NC automatically evaluate the status of the interface signal, and if necessary, change the spindle direction of rotation.

If, while the program is being executed, the spindle direction of rotation must again be changed as a result of the NC/PLC interface signal "Invert M3 / M4" (DB31, ... DBX17.6), then within an NC program, this can only be achieved using the subsequently described principle sequence:

N110 SPCOF or M5	Stop the spindle and switch into the closed-loop speed controlled mode
N111 Mxx or Hxx	Output a user-specific help function to initiate the following actions in the PLC user program: • Set the read-in inhibit in the channel: DB21, ... DBX6.1 = 1 • Invert the interface signal "Invert M3/M4" of the spindle: DB31, ... DBX17.6 • Reset the read-in inhibit in the channel: DB21, ... DBX6.1 = 0
N112 STOPRE	Preprocessing stop

N113 SPOS=IC(0.001)	Incrementally traverse the spindle in the closed-loop position controlled mode. By incrementally traversing the spindle in the closed-loop position controlled mode, the NC again evaluates interface signal DB31, ... DBX17.6 (invert M3/M4). From this block onward, when traversing the spindle, the direction of reversal based on the interface signal is active.
N114 SPOS=0	**Optional**: Position the spindle if, technologically, a defined output position is required, e.g. 0°

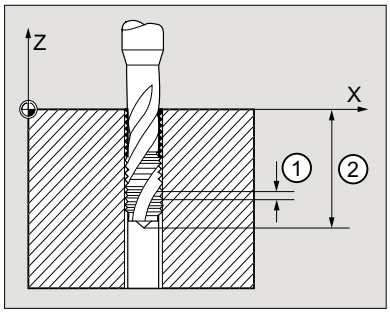
Example

Program code	Comment
N010 G0 X0 Z0	; Initial position of the axes
N020 G1 G94 F1000 Z10	; Traverse the Z axis to the initial position
N030 M3 S100	; Traverse the spindle in the closed-loop speed controlled mode
	; 100 rpm with the spindle direction of rotation that was determined
	; at the start of the program
...	; Any arbitrary machining
N040 M5	; Stop the spindle and switch into the closed-loop speed controlled mode
N050 H10	; Auxiliary function output: H0=10
	; In the PLC user program:
	; - set the read-in inhibit: DB21, ... DBX6.1=1
	; - invert the direction of rotation: DB31, ... DBX17.6
	; - reset the read-in inhibit: DB21, ... DBX6.1=0
N060 STOPRE	; Preprocessing stop
N070 SPOS=IC(0.001)	; Incrementally traverse the spindle in the closed-loop position controlled mode.
N080 SPOS=0	; Position the spindle
N090 G331 Z-10 K1 S500	; Tapping without compensating chuck
N100 G332 Z2 K1	; Tapping without compensating chuck Retraction movement
N110 G1 G94 F1000 Z10	; Retract to the initial position of the Z axis
N120 SPCOF	; Switch the spindle into the closed-loop speed controlled mode
N130 H10	; See above
N140 STOPRE	; Preprocessing stop
N150 SPOS=IC(.001)	; Incrementally traverse the spindle in the closed-loop position controlled mode.
N160 SPOS=0	; Position the spindle
N170 G331 Z-10 K1 S500	; Tapping without compensating chuck
N180 G332 Z2 K1	; Tapping without compensating chuck Retraction movement
N190 G1 G94 F1000 Z10	; Traverse the Z axis to the initial position
...	

18.2.9 Tapping with compensating chuck

18.2.9.1 Function

When tapping with compensating chuck (G63) the spindle is not interpolated with the linear axis, but is operated in the closed-loop speed controlled mode. The feedrate of the linear axis dependent on the spindle speed and the thread pitch must be calculated and explicitly programmed. As the spindle and linear axis are not interpolated together, to compensate spindle speed fluctuations for example, a tapping drill in a length-compensating chuck is required.



Retraction movement

The retraction motion is also programmed with G63 - but with the opposite spindle direction of rotation.

Axis and spindle override value

While the "Tapping **with** compensating chuck" function is selected, 100% is active as axis and spindle override value.

Feedrate

Feedrate F of the linear axis to be programmed is obtained from the product of the spindle speed and the thread pitch of the tapping drill:

F [mm/min] = spindle speed S [rpm] *thread pitch [mm/U]

18.2.9.2 Programming

For tapping with compensating chuck, using the G63 command, the following traversing motion is executed:

- G63: Tapping in the tapping direction up to the end of thread point
- G63: Retraction motion with programmed spindle direction of rotation reversal

Note

After a G63block, the last effective interpolation type G0, G1, G2 is active.

Syntax

```
G63 <axis> <direction of rotation> <speed <feedrate>
```

Meaning

G63:	Tapping with compensating chuck	
	Effective:	Non-modal
<Axis>:	Traversing distance/position of the geometry axis (X, Y or Z) at the end of the thread, e.g. Z50	
<Direction of rotation>:	Direction of spindle rotation: • M3: Clockwise rotation, right-hand thread • M4: Counterclockwise rotation, left-hand thread	
<Speed>:	Maximum permissible spindle speed while tapping, e.g. S100	
<Feedrate>:	Feedrate of the tapping axis F, with F = spindle speed * thread pitch	

Example

Tapping an M5 thread:

- Spindle pitch according to the standard: 0.8 mm/rev
- Spindle speed S: 200 rpm
- Feedrate F = 200 rpm * 0.8 mm/rev = 160 mm/min.

Program code	Comment
N10 G1 X0 Y0 Z2 F1000 S200 M3	; Approach starting point
	; Spindle clockwise direction of rotation, 200 rpm
N20 G63 Z-50 F160	; Tapping with compensating chuck
	; Drilling depth: absolute Z=50mm
	; Feedrate: 160 mm/min
N30 G63 Z3 M4	; Retraction movement: absolute Z=3mm
	; Direction of rotation reversal
	; Spindle with counterclockwise direction of rotation, 200 rpm

18.3 Reference / synchronize

Why synchronize?

In order to ensure that the controller detects the exact position of the spindle when it is switched on, the controller must be synchronized with the position measuring system of the spindle.

The following functions are possible only with a synchronized spindle:

- Thread cutting
- Tapping without compensating chuck
- Axis programming

For further information about synchronizing the spindle, see Section "R1: Referencing (Page 291)".

Why reference?

In order to ensure that the controller detects the exact machine zero when it is switched on, the controller must be synchronized with the position measurement system of the rotary axis. This process is known as referencing. The sequence of operations required to reference an axis is known as search for reference.

Only a referenced axis can approach a programmed position accurately on the machine.

For further information about referencing the rotary axis, see Section "R1: Referencing (Page 291)".

Installation position of the position measurement system

The position measurement systems can be installed as follows:

- directly at the motor in combination with a contactless proximity switch on the spindle as a zero mark encoder
- at the motor using a measuring gearbox in combination with a contactless proximity switch on the spindle as a zero mark encoder
- Directly on the spindle
- on the spindle via a measuring gearbox in combination with a contactless proximity switch on the spindle as a zero mark encoder (only with ratios not equal to 1:1)

Where two position measuring systems are provided, they can be installed either in the same location or separately.

Synchronization procedure

When the spindle is switched on, it can be synchronized as follows:

- The spindle is started with a spindle speed (`S` value) and a spindle rotation (`M3` or `M4`) and synchronized with the next zero mark of the position measurement system or with the next signal from the contactless proximity switch.
- The spindle is to be positioned from standstill using `SPOS`, `M19` or `SPOSA`. The spindle synchronizes with the next zero mark of the position measurement system or with the next proximity switch signal. It is then positioned to the programmed position.

- The spindle can be synchronized from the motion (after M3 or M4) using SPOS, M19 or SPOSA.
  The responses are as follows:
  - With SPOS=<Pos>, SPOS=DC(<Pos>) and SPOS=AC(<Pos>), the direction of motion is retained and the position is approached.
  - With SPOS = ACN(<Pos>) or SPOS = ACP(<Pos>), the position is always approached with a negative or positive direction of motion. If necessary, the direction of motion is inverted prior to positioning.
- Crossing the zero mark in JOG mode by means of direction keys in speed control mode.

Note

It does not make any difference whether the synchronization procedure is initiated from the part program, FC 18 or synchronized actions.

Note

During synchronization of the spindle, all four possible reference point values are effective depending on the measuring system selected. The measurement system offset has the same effect.

The following machine data must be observed:

- MD34080 $MA_REFP_MOVE_DIST
  (Reference point distance / destination point for a distance-coded system)
- MD34090 $MA_REFP_MOVE_DIST_CORR
  (Reference point offset / absolute offset, distance-coded)
- MD34100 $MA_REFP_SET_POS
  (Reference point value, with distance-coded system without any significance)

If a non-referenced spindle with SPOS=IC(...) and a path < 360 degrees is positioned, it may be the case that the zero mark is not crossed and the spindle position is still not synchronized with the zero mark. This can happen:

- After POWER ON
- By setting the axial NC/PLC interface signals:
  DB31, ... DBX17.5 (resynchronize spindle when positioning 2)
  DB31, ... DBX17.4 (resynchronize spindle when positioning 1)

Special features when synchronizing using a contactless proximity switch

The position error caused by the signal delay when using a contactless proximity switch can be corrected internally in the NC by entering a signal runtime compensation.

The signal runtime compensation is set by means of the machine data:

- MD31122 $MA_BERO_DELAY_TIME_PLUS
  (delay time of the contactless proximity switch, positive motion direction)
- MD31123 $MA_BERO_DELAY_TIME_MINUS
  (delay time of the contactless proximity switch, negative motion direction)

The effect depends on the selected referencing mode:

MD34200 $MA_ENC_REFP_MODE = <value>

<Value>	Meaning
2	The position is synchronized without entering a specific starting velocity/speed.
7	As starting velocity to synchronize the position, the velocity from the following machine data is used: MD34040 $MA_REFP_VELO_SEARCH_MARKER (creep velocity) The zero mark is not automatically looked for, it has to be requested explicitly with the 0/1 edge of the NC/PLC interface signal: DB31, ... DBX16.4 / 5 (resynchronize spindle, measuring system 1 / 2)

Note

Generally, no change is required as signal runtimes and signal runtime compensation are preset when the system is delivered.

Referencing sequence

If the spindle is to be programmed in axis mode directly after controller power-up, it must be ensured that the axis is referenced.

When the controller is switched on, the spindle can be referenced (condition is one zero mark per revolution).

For information about the referencing sequence, see Section "R1: Referencing (Page 291)".

The rotary axis is referenced at the same time as the spindle is synchronized (see section "Synchronization procedure") if the position measuring system used for the spindle is also used for the rotary axis.

Position measurement systems, spindle

The spindle can be switched from spindle mode to axis mode (rotary axis) if a single motor is used for spindle mode and axis mode.

The spindle (spindle mode and axis mode) can be equipped with one or two position measurement systems. With two position measurement systems, it is possible to assign one position measurement system to the spindle and the other to the rotary axis, or to assign two position measurement systems to the spindle. Where two position measurement systems are provided, both are updated by the controller, but only one can be active.

The active position measuring system is selected with the NC/PLC interface signal:

DB31, ... DBX1.5 / 6 (position measuring system 1 / 2)

The active position measurement system is required for the following functions:

- Position control of the spindle (SPCON)
- Spindle positioning (SPOS, M19 and SPOSA)
- Thread cutting (G33, G34, G35)
- Tapping without compensating chuck (G331, G332)
- Revolutional feedrate (G95)

- Constant cutting rate (G96, G961, G97, G971)
- Spindle actual speed display
- Axis mode
- Synchronous spindle setpoint coupling

Resynchronizing the position measuring system for the spindle

In the following cases, the spindle position measurement system must be resynchronized:

- The position encoder is on the motor, a contactless proximity switch is mounted on the spindle and the gear stage is changed. Synchronization is triggered internally once the spindle is rotating in the new gear stage (see Synchronization procedure).
- The machine has a selector switch for a vertical and horizontal spindle. Two different position encoders are used (one for the vertical spindle and one for the horizontal spindle), but only one actual value input is used on the controller. When the system switches from the vertical to the horizontal spindle, the spindle must be resynchronized.
  This synchronization is initiated with the NC/PLC interface signal:
  DB31, ... DBX16.4 (resynchronize spindle 1)
  or
  DB31, ... DBX16.5 (resynchronize spindle 2)
  The spindle must be in open-loop control mode.

Restoring the position after a warm restart

For spindles with incremental position measuring systems, it is possible to buffer the actual values after a POWER OFF and after POWER ON, to restore the position last buffered before switching-off, in order that position-dependent functions, e.g. transformations can be restored (see Section "Automatic restoration of the machine reference (Page 332)"). One application, for example, is the tool retraction after a warm restart during processing with tool orientation (see Function Manual Basic Function, Section "Tool retraction after POWER ON with orientation transformation").

The following NC/PLC interface signals display the state of the position measuring system after the position has been restored:

DB31, ... DBX71.4 ("Restored 1") for position measuring system 1

DB31, ... DBX71.5 ("Restored 2") for position measuring system 2

Once the tool has been retracted in JOG mode, axes whose positions have been restored are referenced. As a consequence, signals DB31, ... DBX71.4/5 ("Restored 1/2") are deleted and signals DB31, ... DBX60.4/5 ("Referenced/synchronized 1/2") are set.

Note

If machine data MD20700 $MC_REFP_NC_START_LOCK is set to a value of "2", then an NC start is also possible with "restored" axis positions (in the MDI mode or when overstoring).

18.4 Configurable gear adaptations

18.4.1 Gear stages for spindles and gear change change

Why do we need gear stages?

Gear stages are used on spindles to step down the speed of the motor in order to generate a high torque at low spindle speeds or to step up in order to maintain a high speed.

No. of gear stages

Five gear stages can be configured for each spindle.

The number of used gear stages is defined in machine data:

MD35090 $MA_NUM_GEAR_STEPS

Parameterization of the gear stages

The gear stages 1 to 5 can be parameterized via the following machine data:

Machine data	Meaning
MD35012 $MA_GEAR_STEP_CHANGE_POSITION[<n>]	Gear stage change position
MD35110 $MA_GEAR_STEP_MAX_VELO[<n>]	Maximum speed for automatic gear stage change
MD35120 $MA_GEAR_STEP_MIN_VELO[<n>]	Minimum speed for automatic gear stage change
MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[<n>]	Maximum speed of gear stage
MD35135 $MA_GEAR_STEP_PC_MAX_VELO_LIMIT[<n>]	Maximum speed of gear stage in position control
MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT[<n>]	Minimum speed of gear stage
MD35200 $MA_GEAR_STEP_SPEEDCTRL_ACCEL[<n>]	Acceleration in speed control mode
MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL[<n>]	Acceleration in position control mode
MD35300 $MA_SPIND_POSCTRL_VELO[<n>]	Position control activation speed
MD35310 $MA_SPIND_POSIT_DELAY_TIME[<n>]	Positioning delay time
MD35550 $MA_DRILL_VELO_LIMIT[<n>]	Maximum speed for tapping without compensating chuck

Type of gear stage change

The type of gear stage change is set in machine data:

MD35010 $MA_GEAR_STEP_CHANGE_ENABLE

Bit	Value	Meaning
0	0	The spindle motor is attached to the spindle directly (1:1) or with a non-variable transmission ratio (basic setting). The machine data of the first gear stage is effective.
	1	Spindle motor with up to five gear stages. The gear stage change takes place: • In oscillation mode • At indefinite change position
1	0	Meaning as in Bit 0 = 0.
	1	Meaning as in Bit 0 = 1, however, the gear stage change takes place at the configured spindle position. The change position is set in machine data: MD35012 $MA_GEAR_STEP_CHANGE_POSITION The position is approached in the current gear stage before the gear stage change. If Bit 1 is set, then Bit 0 is ignored!
3	1	The gear stage change dialog between NC and PLC is simulated.
5	1	The second gear stage data set is used while tapping with G331/G332 (see the following paragraph "Second gear stage data set"). The bit must be set for the master spindle used during the tapping.

Requirement for a gear stage change

In principle, the gear stage change is only performed if the requested gear stage is not the same as the active gear stage.

Parameter set selection during gear stage change

The servo parameter set is also changed over with the gear stage if:

MD35590 $MA_PARAMSET_CHANGE_ENABLE = 0 or 1

For further information, see Section "Parameter set selection during gear step change (Page 789)".

Request gear stage change

A gear stage change can be requested:

- In the part program using:
  - `M40 S...`
    Automatic gear stage selection to the programmed speed `S...`
  - `M41` ... `M45`
    Direct selection of gear stages 1 ... 5
  - `M70`
    For MD35014 $MA_GEAR_STEP_USED_IN_AXISMODE = 1 ... 5
    (see "Configurable gear step in M70 (Page 805)")
  - `G331 S...`
    For MD35010 $MA_GEAR_STEP_CHANGE_ENABLE, Bit 5 = 1
- In synchronized actions using:
  - `DO M40 S...`
    Automatic gear stage selection to the programmed speed `S...`
  - `DO M41... M45`
    Direct selection of gear stages 1 ... 5
  - `DO M70`
    For MD35014 $MA_GEAR_STEP_USED_IN_AXISMODE = 1 ... 5
- Through the PLC using the FC18 function block
- In the reset state through description of NC/PLC interface:
  DB31, ... DBX16.0-16.2 (actual gear stage A to C)
  The mechanically active gear stage can be communicated to the NC especially after a POWER ON.

Note

If the spindle motor is attached to the spindle directly (1:1) or with a non-variable transmission ratio (MD35010 = 0), then the M40 and M41 ... M45 auxiliary functions are not relevant to this spindle.

Gear stage change

Gear stage selection between two gear stages with specification of a maximum spindle speed is shown in the example below:

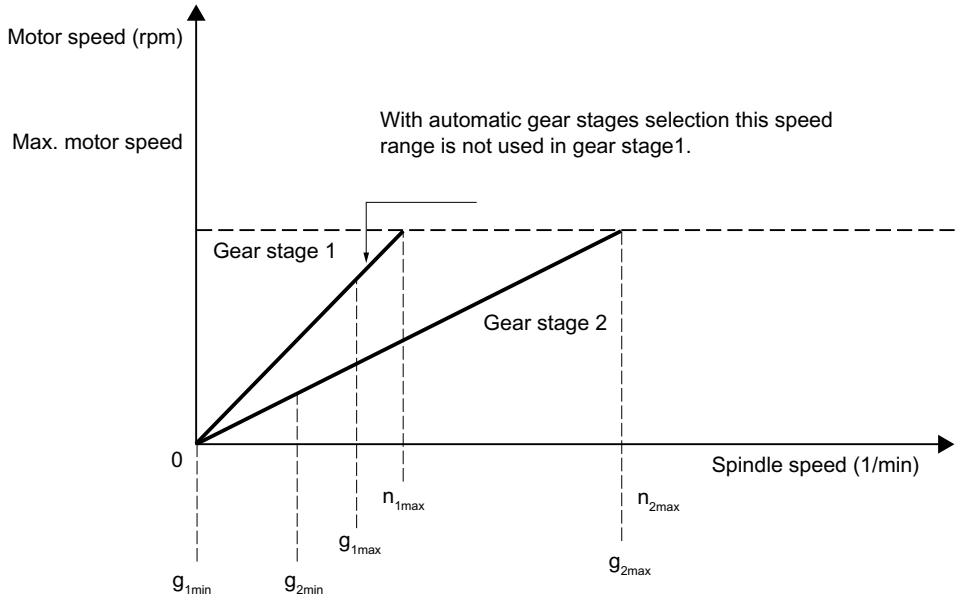


Figure 18-3 Gear stage change with selection between two gear stages

Process sequence of the gear stage change

If the new gear stage is preselected, the following sequence is implemented:

1. Changeover sequence
   The two following NC/PLC interface signals are set:
   DB31, ... DBX82.0-82.2 (setpoint gear stage A to C)
   DB31, ... DBX82.3 (change over gear stage)
   In accordance with the point at which NC/PLC IS:
   DB31, ... DBX18.5 (oscillation speed)
   is set, the spindle decelerates to a standstill at the acceleration for oscillation or at the acceleration for speed control / position control.
   Oscillation can be activated at the latest when the spindle reaches a standstill:
   DB31, ... DBX61.4 (axis/spindle stationary)
   with NC/PLC IS:
   DB31, ... DBX18.5 (oscillation speed).

   In principle, the new gear stage can also be engaged without oscillation
   When the new gear stage is engaged, the following NC/PLC interface signals are set by the PLC program:
   DB31, ... DBX16.0-16.2 (actual gear stage A to C)
   DB31, ... DBX16.3 (gear is changed)

2. End of gear stage change
   The gear stage change is treated as completed (spindle operation type "oscillation mode" is deselected), if the following NC/PLC interface signal is set:
   DB31, ... DBX16.3 (gear is changed)
   The new actual gear stage is changed to the servo and interpolation parameter set when the motor is stationary.
   With NC/PLC interface signal:
   DB31, ... DBX16.3 (gear is changed)
   is used to communicate to the NC that the new gear stage is valid and the oscillation mode can be completed.
   The NC/PLC interface signal:
   DB31, ... DBX82.3 (change gear)
   is reset by the NC,
   which causes the PLC program to reset NC/PLC IS:
   DB31, ... DBX16.3 (gear changed).

   In this case, it does not matter whether NC/PLC IS:
   DB31, ... DBX18.5 (oscillation mode)
   is still set.
   The actual gear stage, which should correspond to the set gear stage, is relevant for selecting the parameter set.
   If this is not the case, then Alarm 22010 :
   MD11410 $MN_SUPPRESS_ALARM_MASK, Bit 3 = 0
   is output.
   Following acknowledgment of gear stage change via the PLC
   with NC/PLC IS:
   DB31, ... DBX16.3 (gear changed)
   the spindle is in speed control mode (DB31, ... DBX84.7).
   Further information
   For further instructions on signal exchange between PLC and NC, see Function Manual Basic Functions; Diverse NC/PLC interface signals and functions.

Second gear stage data set

The automatic gear stage change M40 can be extended by a second configurable gear stage data set.

The second gear stage data set is used **exclusively** in connection with tapping without compensation chuck (G331, G332) so that an effective adjustment of spindle speed and motor torque can be achieved.

The activation is undertaken by setting the following bit for the master spindle:

MD35010 $MA_GEAR_STEP_CHANGE_ENABLE, Bit 5 = 1

The number of used gear stages of the second gear stage data set is defined with the machine data:

MD35092 $MA_NUM_GEAR_STEPS2

The second gear stage block data set is deactivated if:

MD35092 $MA_NUM_GEAR_STEPS2 = 0 (basic setting)

The first gear stage data set then selects the gear stage when M40 is active.

Note

The number of gear stages in the second data set can vary from the first. If no appropriate gear stage is found for a programmed speed for M40, then no gear stage change is carried out (exceptions, see "M40: Automatic gear stage selection for speeds outside the configured switching thresholds (Page 835)").

For more information about a typical program sequence in thread cutting without compensating chuck G331/G332 see:
Further information:
Programming Manual - Fundamentals; Motion Commands

The gear stages 1 to 5 of the second gear stage data set can be parameterized via the following machine data:

Machine data	Meaning
MD35112 $MA_GEAR_STEP_MAX_VELO2[n]	Maximum speed for automatic gear stage change
MD35122 $MA_GEAR_STEP_MIN_VELO2[n]	Minimum speed for automatic gear stage change
MD35212 $MA_GEAR_STEP_POSCTRL_ACCEL2[n]	Acceleration in position control mode

Note

The number of servo parameter sets concerning the mechanical factors remain unchanged. Furthermore, five mechanical gear stages for the spindle and one for the axis operation can be configured.

The speed limitations are configured only once for each gear stage with the following machine data, independently of the different switching thresholds:

Machine data	Meaning
MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[n]	Maximum speed of gear stage
MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT[n]	Minimum speed of gear stage

For tapping without compensating chuck (G331, G332) the speed can be limited to the linear acceleration range of the motor additionally. For this, the maximum speed of the linear motor characteristics range is specified in the following machine data as a function of the gear stage:

MD35550 $MA_DRILL_VELO_LIMIT[n]

Specify gear stage in the part program

Automatic selection with active M40

The gear stage is automatically selected by the control. The gear stage in which the programmed spindle speed (`S...`) is possible is checked in this context. If a gear stage results from this that is not equal to the current (actual) gear stage, then the following NC/PLC interface signals are set:

DB31, ... DBX82.3 (change over gear stage)

DB31, ... DBX82.0-82.2 (setpoint gear stage A to C)

While the appropriate gear stage is being determined, a gear stage change is only requested if the new speed is not within the permissible speed range of the active gear stage.

The speed is limited to the maximum speed of the current gear stage or raised to the minimum speed of the current gear stage and the appropriate NC/PLC interface signal is set:

DB31, ... DBX83.1 (speed setpoint limited)

DB31, ... DBX83.2 (speed setpoint increased)

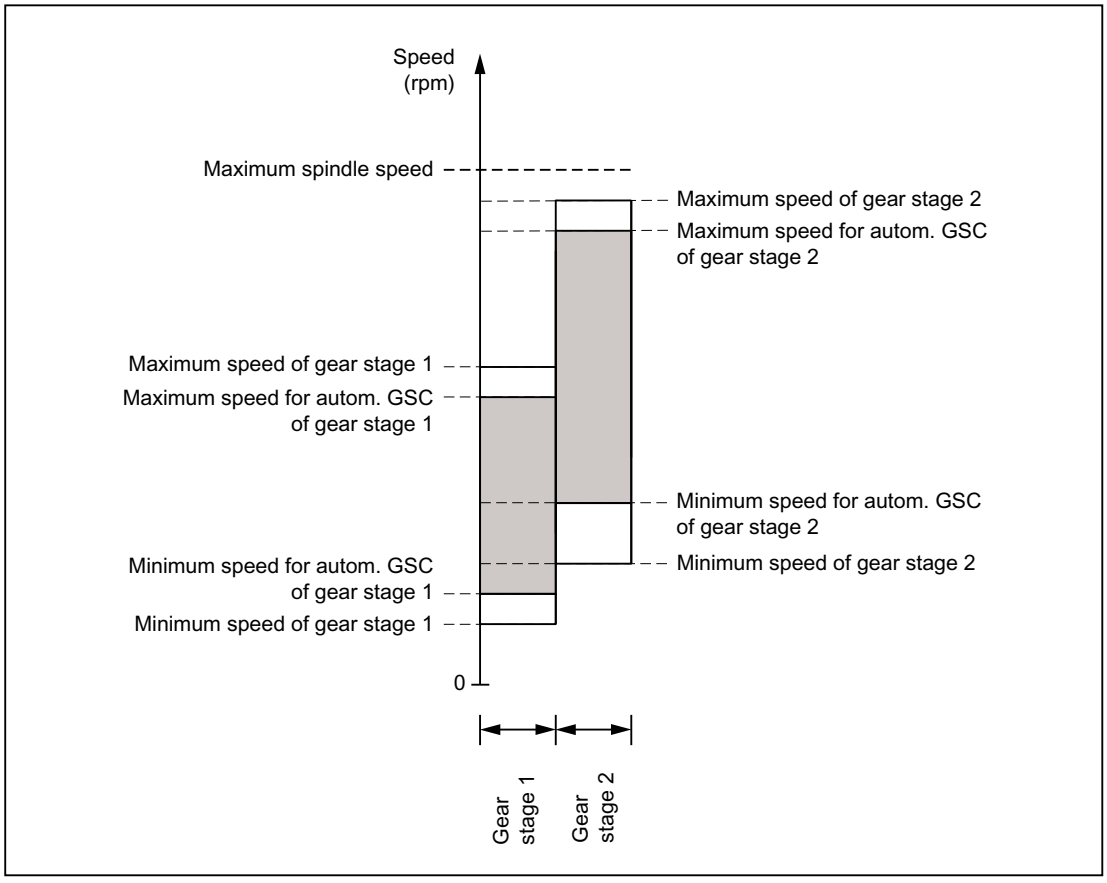


Figure 18-4 Example for two gear stages with overlapping speed ranges for automatic gear stage change (M40)

Note

In the case of M40, the spindle must be in open-loop control mode for automatic gear stage selection with an `S` word. Otherwise the gear stage change is rejected and the following alarm is set:

Alarm 22000 "gear stage change is not possible"

Note

An active reduction gear is not considered in the selection for the automatic gear stage change.

Permanently defining the gear stage with M41 to M45

The gear stage can be permanently defined in the part program with `M41` to `M45`.

If a gear stage is specified via `M41` to `M45` that is not equal to the current (actual) gear stage, then the following NC/PLC interface signals are set:

DB31, ... DBX82.3 (change over gear stage)

DB31, ... DBX82.0-82.2 (setpoint gear stage A to C)

The programmed spindle speed (S...) then refers to this permanently defined gear stage:

- If a spindle speed is programmed and it is higher than the maximum speed of the permanently defined gear stage (MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT), then the speed is decreased to this limit and the following NC/PLC interface signal is set:
  DB31, ... DBX83.1 (speed setpoint limited)
- If a spindle speed is programmed and it is lower than the minimum speed of the permanently defined gear stage (MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT), then the speed is increased to this minimum and the following NC/PLC interface signal is set:
  DB31, ... DBX83.2 (speed setpoint increased)

Block change

When programming the gear stage change in the part program, the gear stage change set remains active until it is aborted by PLC.

This corresponds to the effect as if the following NC/PLC interface signal were set:

DB21, ... DBX6.1 (read-in disable)

Specification of gear stage via PLC with FC18

The gear stage change can also be performed by function block FC18 during a part program, in the reset state or in all operating modes.

If the speed and direction of rotation is specified with FC18, the NC can be requested to select the gear stage as appropriate for the speed. This corresponds to an automatic gear stage change with M40.

The gear stage is not changed if:

- The spindle is positioned via FC18.
- The spindle is traversed in the axis mode.

Further information

For further information on the FC18 function block, see Function Manual PLC.

Specification of a gear stage in synchronized actions

The gear stage change can be requested by synchronized actions using:

- `DO M40 S...`
  Automatic gear stage selection to the programmed speed `S...`
- `DO M41... M45`
  Direct selection of gear stages 1 ... 5
- `DO M70`
  For MD35014 $MA_GEAR_STEP_USED_IN_AXISMODE = 1 ... 5
  (see "Configurable gear step in M70 (Page 805)")

The gear stage is **not** changed if:

- The spindle is positioned via synchronized actions.
- The spindle is traversed in the axis mode.

Note

For further details, please refer to the section "Specification of a gear stage in part program".

Exception:

The block change is not affected by the specification of a gear stage in synchronized actions.

Manual specification of a gear stage

Outside a part program that is running, the gear stage can also be changed without a request from the NC or the machine. This is the case, for example, when a gear stage is changed manually.

To select the appropriate parameter set, the NC must be informed of the current gear stage. To enable this, the control or the part program must be in the reset state.

Boundary conditions

Transfer of the gear stage to the NC is initiated when
NC/PLC IS:
DB31, ... DBX16.0-16.2 (actual gear stage A to C) changes.

These three bits must be set continuously during operation.

Successful transfer is acknowledged with NC/PLC IS:
DB31, ... DBX82.0-82.2 (set gear stage A to C)
to the PLC.

NC/PLC IS:
DB31, ... DBX16.3 (gear changed)
must not be set.

If position control is active when a new gear stage is specified by the PLC with
DB31, ... DBX16.0-16.2, then it is switched off for the duration of this changeover sequence.

NC stop during gear stage change

The spindle cannot be stopped with NC/PLC IS:
DB21, ... DBX7.4 (NC stop)
if:

- The spindle is not yet in oscillation mode for the gear stage change.
- NC/PLC IS:
  DB31, ... DBX16.3 (gear changed)
  is not set.

Note

Options for aborting:

DB31, ... DBX2.2 (delete distance-to-go / spindle reset)
or
DB31, ... DBX16.3 (gear changed)
with corresponding acknowledgment from actual gear stage:
DB31, ... DBX16.0-16.2 (actual gear stage).

Spindle response after a gear stage change

How the spindle behaves once the gear stage has been changed depends on the following initial conditions:

- If the spindle was in the stop state before the gear stage change (M5, FC18: "Stop rotate spindle"), in positioning or axis mode, M5 (spindle stop) is active after completion of the gear stage change.
- If a direction of rotation
  (M3, M4, FC18: "Start spindle rotation"), then the last speed and direction of rotation will become active again after the gear stage change. In the new gear stage, the spindle accelerates to the last spindle speed programmed (`S...`).
- If position control was active before the gear stage change (`SPCON`), then it is reactivated after the gear stage change.
  The next block in the part program can be executed.

Special features

The following points must be observed on gear stage change:

- The gear stage change is not terminated by selecting
  NC/PLC IS:
  DB31, ... DBX20.1 (run-up switchover to V/f mode).

  Setpoint 0 is output.
  The gear stage change is acknowledged as usual
  via the NC/PLC interface signal:
  DB31, ... DBX16.3 (gear is changed)
- The "Ramp-function generator rapid stop" signal must be reset by the PLC before the gear stage change is completed by the PLC.
- The process sequence of the gear stage change is ended during NC reset without any alarm output.
  The gear stage output with NC/PLC IS:
  DB31, ... DBX16.0-16.2 (actual gear stage A to C)
  is applied by the NC.

Star/delta switchover with FC17

Digital main spindle drives can be switched in both directions between star and delta using FC17, even when the spindle is running. This automatic switchover is controlled by a defined logic circuit in FC17 which provides the user with a configurable switchover time for the relevant spindle.

Further information

For further information on the FC17 function block, see Function Manual PLC.

18.4.2 Spindle gear stage 0

Technical background

For machine's where the spindle load gear can be changed over, situations can occur where the gear train between the motor and load (workpiece/tool) is interrupted. This state can occur, e.g. when pressing Reset or Emergency Stop while performing a gear stage change or when the machine is commissioned for the first time while it is being installed. The control must identify this state where the gear train is open and the next gear stage change request must be unconditionally executed.

Function

When the gear is disengaged, the binary-coded value "0" (≙ gear stage 0) is transferred to the NC from the PLC using the interface signal bits DB31, ... DBX16.0-2 (actual gear stage A to C):

DB31, ... DBX16.0-2 = 0

The value is used by the control to identify the state where the gear train is open.

Effects on the gear stage change

Gear stage change in the part program

The actual gear stage signaled from the PLC is read by the NC when starting a part program. If, at this instant in time, a value of "0" is read for the actual gear stage, then the next gear stage change is executed and the gear stage change dialog is performed by the PLC. If a value greater than "0" is read, then already in the program a comparison is made between the requested and active gear stage. If both gear stages are the same, the gear stage is not changed and a possibly programmed path motion is not interrupted.

Gear stage change in synchronized actions, FC18 and DBB30

The actual gear stage signaled from the PLC is always evaluated by the NC when the gear stage is changed. The gear stage is always changed if a value of "0" is read from the NC. When reading a value greater than "0", a comparison is made between the requested and active gear stage. The gear stage is only changed with the PLC if the two values are not equal and the NC/PLC interface signal DB31, ... DBX82.3 (change over gear) is then output.

Boundary conditions

- **Output of DB31, ... DBX16.0-2 = 0**
  When the gear is disengaged, the PLC must enter gear stage 0 in the NC/PLC interface DB31, ... DBX16.0-2 (actual gear stage A to C).
- **Enabling the gear stage change**
  The precondition for a gear stage change after reaching gear stage 0 is the general enable of the gear stage change via via machine data:
  MD35010 $MA_GEAR_STEP_CHANGE_ENABLE (assign parameters to the gear stage change)
  MD35090 $MA_NUM_GEAR_STEPS (number of gear stages set up)
  MD35092 $MA_NUM_GEAR_STEPS2 (2nd gear stage data set: Number of gear stages that have been created) if MD35010 $MA_GEAR_STEP_CHANGE_ENABLE, bit 5 = 1 (tapping without compensating chuck)
- **PLC user program/ POWER ON ASUB**
  The PLC user program or POWER ON ASUB should ensure that when the gear is disengaged (gear stage 0) before spindle motion, a gear stage change request is programmed. For instance, this can be realized with `M41` in the ASUB. Spindle motion such as e.g. in JOG or in axis operation does not generate any gear stage change itself.

Example

Example for the sequence to select the first gear stage after POWER ON

1. POWER ON.
2. The PLC user program determines, in the mechanical environment, the "Gear is disengaged" state.
3. The PLC transfers the "Gear is disengaged" state to the NC by setting:
   DB31, ... DBX16.0-2 = 0
4. Part program start or POWER ON ASUB.

5. N05 (part program, refer below) is executed:
   The gear stage is changed to gear stage 1.
   From the NC:
   - the following NC/PLC-interface signal is set:
     DB31, ... DBX82.3 (change over gear stage)
   - the setpoint gear stage 1 is signaled to the PLC:
     DB31, ... DBX82.0 = 1
     DB31, ... DBX82.1 = 0
     DB31, ... DBX82.2 = 0
6. Mechanical gear stage change, acknowledgement
   If the gear stage is selected, then from the PLC:
   - the following NC/PLC-interface signal is set:
     DB31, ... DBX16.3 (gear is changed)
   - Actual gear stage 1 signaled to the NC:
     DB31, ... DBX16.0 = 1
     DB31, ... DBX16.1 = 0
     DB31, ... DBX16.2 = 0
7. N80 is executed:
   Due to the optimization of the gear stage change frequency in the part program, the gear stage is not changed.

Part program:

Program code	Comment
N05 M41	; Select 1st gear stage
...	
N80 M41	; No gear stage change, if the 1st gear stage is selected.

Configuring data for spindle 1 (AX5):

MD35010 $MA_GEAR_STEP_CHANGE_ENABLE[AX5] = 1 (enable gear stage change)

18.4.3 Determining the spindle gear stage

The actual stage of a spindle can be read using system variables:

- For the display in the user interface, in synchronized actions or with a preprocessing stop in the part program via the system variables:

$VC_SGEAR[<n>]	Currently selected spindle gear stage $VC_SGEAR reads the actual gear stage signaled from the PLC. Value range: 0 ... 5

$AC_SGEAR[<n>]	Active spindle gear stage $AC_SGEAR reads the setpoint gear stage in the main run. Value range: 1 ... 5 The data set for the spindle is activated corresponding to this gear stage.

Note

For a block search, the actual gear stage ($VC_SGEAR[<n>]) can differ from the setpoint gear stage ($AC_SGEAR[<n>]) as, during the block search, no gear stage change takes place. Therefore, using $VC_SGEAR[<n>] and $AC_SGEAR[<n>], it can be interrogated whether a gear stage change should be made after a block search.

- Without preprocessing stop in the part program via system variables:

$P_SGEAR[<n>]	Setpoint gear stage $P_SGEAR reads the gear stage programmed in the part program (`M41` ... `M45`), for `M40` selected, or for `M70`, the configured gear stage.
$P_SEARCH_SGEAR[<n>]	Block search: Gear-specific M function $P_SEARCH_SGEAR contains the last programmed gear stage M function collected with the block search.

18.4.4 Parameter set selection during gear step change

Servo parameter sets

The servo parameter sets 1 to 6 adapt the position controller to the changed properties of the machine during a gear change of the spindle.

Parameter set selection during gear stage change

The gear stage parameter set (interpolation parameters) and, depending on the setting in the following machine data, the servo parameter set are also modified during gear stage change.

MD35590 $MA_PARAMSET_CHANGE_ENABLE (parameter set change possible)

Value	Meaning
0	In-system parameter set selection The parameter sets of the servo are assigned permanently. The following applies: • For axes and spindles in the axis mode, the first parameter set is active in principle. Exception: For G33, G34, G35, G331 and G332, for the axes involved, the parameter set with the following number is activated: Master spindle gear stage + 1 (corresponds to parameter set No. 2 ... 6) • For spindles in the spindle mode, the parameter set is set matching the gear stage.
1	Besides the in-system parameter set selection, there is also the option of an "external" parameter set selection. • By the PLC (DB31, ... DBX 9.0 - 9.2) • Via programming of SCPARA in the part program or in synchronized actions However, the in-system parameter set selection has priority. Note: Value 1 is relevant only to axes.
2	The servo parameter set is specified **exclusively** by the PLC (DB31, ... DBX 9.0 - 9.2) or through the programming of SCPARA in the part program or in synchronized actions (for axes and spindles). The 1st parameter set is selected after POWER ON.

Spindle mode

MD35590 $MA_PARAMSET_CHANGE_ENABLE = 0 or 1

The parameter set is selected according to the gear stage + 1.

The active gear stage is located in:

DB31, ... DBX16.0-16.2 (actual gear stage A to C)

The active parameter set is output in:

DB31, ... DBX69.0-69.2 (controller parameter set A to C)

One set of parameters, with the following assignment, is provided by the NC for each of the five gear stages:

Data set for ...	NC/PLC interface DBX 69.2 / 69.1 / 69.0	Parameter set Number	Parameter set Index [n]
Axis mode	Last active gear stage	1	0
Gear stage 1	001	2	1
Gear stage 2	010	3	2
Gear stage 3	011	4	3
Gear stage 4	100	5	4
Gear stage 5	101 110 111	6	5

Spindle in axis mode

If the spindle is in axis mode, the parameter set index "0" is selected in the servo (note MD35590 $MA_PARAMSET_CHANGE_ENABLE!).

The gear stage change behavior depends on the setting in the machine data:

MD35014 $MA_GEAR_STEP_USED_IN_AXISMODE (gear stage for axis mode in M70)

If there is no gear stage configured for axis mode (MD35014 = 0), no implicit gear stage change takes place in M70 (default setting!). The last gear stage is saved internally and is reactivated with the associated parameter set during the next spindle programming.

If, however, a gear stage is configured for axis mode (MD35014 = 1 ... 5), a gear stage change to gears 1 ... 5 takes place during the execution of M70. When changing from axis mode to spindle mode, the gear stage loaded with M70 remains activated. The gear stage which is activated in spindle mode prior to M70 is not automatically loaded again.

See also "Configurable gear step in M70 (Page 805)."

Load gearbox transmission ratio

It is possible to configure positive or negative **load gearbox factors** for each gear stage and in axis mode.

The setting is undertaken separately for numerator and denominator via the machine data:

MD31050 $MA_DRIVE_AX_RATIO_DENOM[n] (load gearbox denominator)

MD31060 $MA_DRIVE_AX_RATIO_DENOM[n] (load gearbox numerator)

The setting range is the same size for positive and negative load gearbox factors.

It is not possible to enter the value "0".

Note

If an indirect encoder is configured, and the load gearbox transmission ratio changes, then the reference is lost and the NC/PLC interface signal:
DB31, ... DBX60.4/60.5 (referenced/synchronized 1 or 2)
is reset for the relevant measuring system.

Further information

For further information about control and servo parameter set, please refer to:

- G2: Velocities, setpoint / actual value systems, closed-loop control (Page 31)
- Programming Manual Advanced; Programmable servo parameter set

18.4.5 Intermediate gear

Application and functions

A configured intermediate gear can be used to adapt a variety of rotating tools. The intermediate gear on the tool side has a multiplicative effect on the motor/load gearbox.

It is set via machine data:

MD31066 $MA_DRIVE_AX_RATIO2_NUMERA (intermediate gear numerator)

MD31064 $MA_DRIVE_AX_RATIO2_DENOM (intermediate gear denominator)

An encoder on the tool for the intermediate gear
is configured with machine data:
MD31044 $MA_ENC_IS_DIRECT2 (encoder on intermediate gear)
.

A changed parameterization of this machine data can be activated with the "Activate machine data" function either with the aid of the SinuCOM-NC commissioning software or via a softkey on the operator panel (HMI). The existing motor/load gearboxes, on the other hand, are active after POWER ON.

Tool change

If the intermediate gear is changed at the same time as the tool, the user must also reconfigure the transmission ratio of the numerator and denominator via the machine data of the intermediate gear.

Example:

⚠ CAUTION

Engineering error

It remains the task of the user to stop within the appropriate period in order to make changes to the machine data when required and then activate the "Activate machine data" function.

In the case of an installed tool with a transmission ratio of 2:1, a suitable intermediate gear is configured and is activated immediately in the part program with the command NEWCONF.

Program code

```
N05 $MA_DRIVE_AX_RATIO2-NUMERA[AX5] = 2
M10 $MA_DRIVE_AX_RATIO2-DENOM[AX5] = 1
N15 NEWCONF
```

Changeover

Changeover to a new transmission ratio is performed immediately with the aid of the "Activate machine data" function. From a technological viewpoint, the associated mechanical changeover process takes some time, since, in mechanical terms, a different intermediate gear with rotating tool is being installed.

Note

At zero speed, changeover is jerk-free. The user is therefore responsible for taking appropriate precautions.

Applications in which changeover takes place during motion and which require smoothed or soft speed transition can be handled using existing setpoint speed filters.

For further explanations regarding control engineering dependencies, see Section "G2: Velocities, setpoint / actual value systems, closed-loop control (Page 31)".

18.4.6 Nonacknowledged gear step change

Mode change

A gear stage change that has not been acknowledged cannot be interrupted by a change in operating mode (e.g. switchover to JOG).

The switchover is delayed by the maximum period entered in machine data:
MD10192 $MN_GEAR_CHANGE_WAIT_TIME
.

If the gear stage change is not acknowledged within this time, the NC will output an alarm:

Further events

Events that initiate reorganization will also wait until a gear stage change is completed.

The time entered in machine data:
MD10192 $MN_GEAR_CHANGE_WAIT_TIME
determines how long the control waits before executing the gear stage change.
If this time elapses without the gear stage change being completed, the NC responds with an alarm.

The following events have an analog response:

- User ASUP
- Mode change
- Delete distance-to-go
- Axis replacement
- Activate PI user data
- Enable PI service machine data

- Switch over skip block, switch over Dry Run
- Editing in the modes
- Compensation block alarms
- Overstore
- Rapid retraction with G33, G34, G35
- Subprogram level abort, subprogram abort

Response after POWER ON

The active gear stage on the machine can be specified by the PLC after POWER ON and in the RESET state of the NC.

The NC will then select the appropriate parameter set
and check back the NC/PLC interface signals:
DB31, ... DBX82.0-82.2 (set gear stage A to C)
to the PLC.

18.4.7 Gear step change with oscillation mode

The spindle is in the oscillation mode if a new gear stage was defined using M40 (automatic gear stage selection) - or M41 to M45.

When oscillating, the direction of rotation of the spindle motor is continually reversed in short intervals. The oscillating motion of the motor helps to engage a new gear stage, for example. In principle, a new gear stage can also be engaged without oscillation.

The following oscillation types are possible:

- Oscillation controlled by the NC
- Oscillation controlled by the PLC
- Oscillation with FC 18
  Further information
  Function Manual PLC

Commissioning: Machine data

- MD35010 $MA_GEAR_STEP_CHANGE_ENABLE (assign parameters to the gear step change).
- MD35410 $MA_SPIND_OSCILL_ACCEL (acceleration when oscillating)
- MD35430 $MA_SPIND_OSCILL_START_DIR (start direction when oscillating)
- MD35400 $MA_SPIND_OSCILL_DES_VELO (oscillation speed)
- MD35450 $MA_SPIND_OSCILL_TIME_CCW (oscillation time for the M4 direction)
- MD35440 $MA_SPIND_OSCILL_TIME_CW (oscillation time for the M3 direction)

Commissioning: NC/PLC interface signals

- DB31, ... DBX16.0 - 2 (actual gear stage)
- DB31, ... DBX16.3 (gear is changed)
- DB31, ... DBX18.4 (oscillation controlled by the PLC)
- DB31, ... DBX18.5 (oscillation enable)
- DB31, ... DBX18.6 (setpoint direction of rotation, clockwise)
- DB31, ... DBX18.7 (setpoint direction of rotation, counterclockwise)
- DB31, ... DBX61.4 (spindle stationary)
- DB31, ... DBX82.0 - 2 (set gear stage)
- DB31, ... DBX82.3 (change over gear stage)
- DB31, ... DBX83.5 (spindle in setpoint range)
- DB31, ... DBX84.6 (active spindle mode: oscillating mode)

Description of the sequence: Gear stage change

Parameterization: MD35010 $MA_GEAR_STEP_CHANGE_ENABLE, bit 0 = 1

Sequence:

- Deceleration of the spindle.
  The braking action corresponds to an M5 movement.
- NC/PLC interface signals NC → PLC:
  - DB31, ... DBX84.6 == 1 (oscillation mode)
  - DB31, ... DBX82.3 == 1 (changeover gear stage)
  - DB31, ... DBX82.0 - 2 == <set gear stage>
  - DB31, ... DBX61.5 == 0 (position controller active)
- The load gear can now "disengage".
- Oscillation can now be executed from the NC or the PLC (refer below).
- Program execution continues after the feedback signal from the PLC to the NC that the gear stage has been switched over and the new gear stage is active:
- DB31, ... DBX16.3 = 1 (gear stage is changed)
- DB31, ... DBX16.0 - 2 (actual gear stage)
- For measuring systems with indirect encoder (motor encoder) the referencing status is deleted:
  DB31, ... DBX60.4 / 5 = 0

Oscillation controlled by the NC

Requirement

- DB31, ... DBX18.5 = 1 (oscillation enable)

Status feedback signals

- DB31, ... DBX84.6 == 1 (oscillation mode)
- DB31, ... DBX82.3 == 1 (changeover gear stage)
- DB31, ... DBX82.0 - 2 == <set gear stage>
- DB31, ... DBX61.5 == 0 (position controller active)

The load gear can now "disengage".

Start oscillation

The PLC user program can now set the oscillation enable signal.

- DB31, ... DBX18.5 = 1 (oscillation enable)

The motor accelerates with the oscillation acceleration to the oscillation speed in the start direction for the direction of rotation dependent oscillation time:

- MD35410 $MA_SPIND_OSCILL_ACCEL (acceleration when oscillating)
- MD35430 $MA_SPIND_OSCILL_START_DIR (start direction when oscillating)
- MD35400 $MA_SPIND_OSCILL_DES_VELO (oscillation speed)
- MD35450 $MA_SPIND_OSCILL_TIME_CCW (oscillation time for the M4 direction)
- MD35440 $MA_SPIND_OSCILL_TIME_CW (oscillation time for the M3 direction)

Reversing the direction of rotation

After the direction-dependent oscillation time has expired, the motor is braked with the oscillation deceleration and is accelerated in the opposite direction of rotation to the oscillation speed for the other oscillation time.

Oscillation controlled by the PLC

Requirement

- DB31, ... DBX18.4 = 1 (oscillation controlled by the PLC)
- DB31, ... DBX18.5 = 1 (oscillation enable)

Status feedback signals

- DB31, ... DBX84.6 == 1 (oscillation mode)
- DB31, ... DBX82.3 == 1 (changeover gear stage)
- DB31, ... DBX82.0 - 2 == <set gear stage>
- DB31, ... DBX61.5 == 0 (position controller active)

The load gear can now "disengage".

Start oscillation

The PLC user program can now set the oscillation enable signal:

- DB31, ... DBX18.7 = 1 (setpoint direction of rotation, counterclockwise) or DB31, ... DBX18.6 = 1 (setpoint direction of rotation, clockwise)

The motor accelerates with the oscillation acceleration to the oscillation speed in the specified setpoint direction of rotation:

- MD35410 $MA_SPIND_OSCILL_ACCEL (acceleration when oscillating)
- MD35400 $MA_SPIND_OSCILL_DES_VELO (oscillation speed)

Reversing the direction of rotation

The oscillation times and switching over the direction of rotation must be realized in the PLC user program.

End of oscillation mode

On termination of oscillation mode, the spindle returns to the open-loop control mode and automatically changes to the mode defined by `SPCON` or `SPCOF`.

All gear-specific limit values (min./max. speed, etc.) correspond to the parameterized values of the actual gear stage.

Block change

Processing of the active NC program is stopped if the spindle was switched into the oscillation mode (DB31, ... DBX82.3 = 1 (switchover gear stage)).

Processing of the NC program is continued if the gear stage has been switched over (DB31, ... DBX16.3 == 1 (gear stage has been switched over).

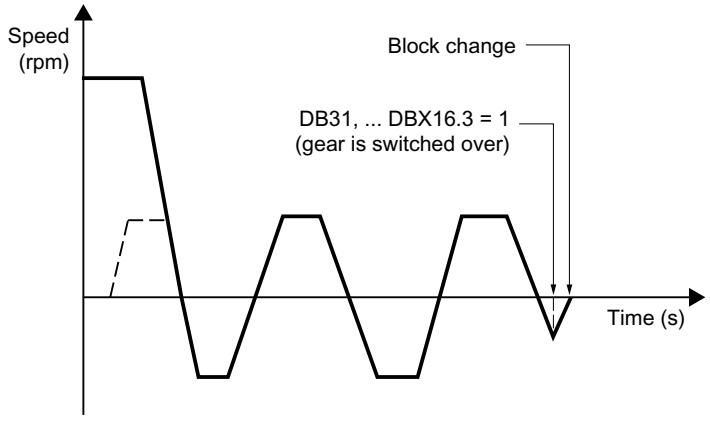
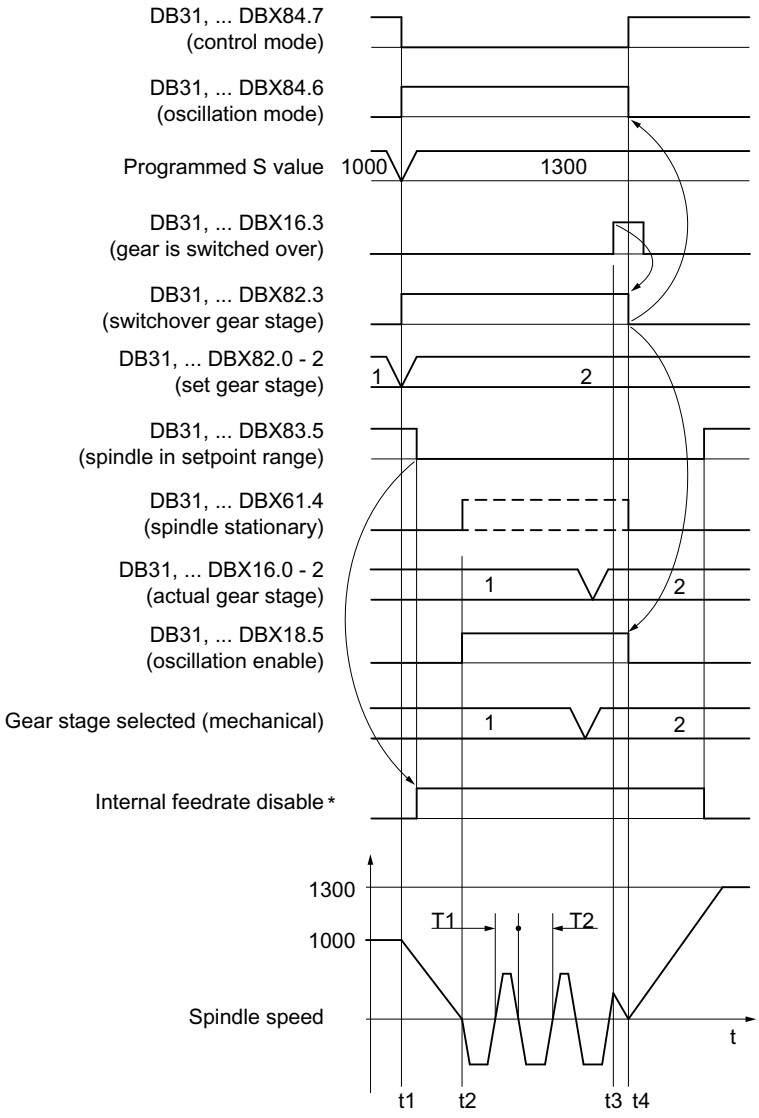


Figure 18-5 Block change following oscillation mode

Example of the signal flow



t_1: As a result of the newly programmed **S value S1300** in the NC program, the NC identifies the necessity to switch over the gear stage (1st → 2nd gear stage), and requests the enable from the PLC:

- NC → PLC: DB31, ... DBX82.3 = 1 (switchover gear stage)

The NC locks program processing (internal feed rate disable *)

t_2: The NC signals **spindle speed == 0** to the PLC:

- DB31, ... DBX61.4 = 1 (spindle stationary)

The PLC user program enables the oscillation mode.

- PLC → NC: DB31, ... DBX18.5 = 1 (oscillation enable)
  Note
  The oscillation enable must be set at the latest at time t_2.

The NC starts to oscillate.

t_3: The gear stage at the machine was switched over.
The PLC user program signals the switchover to the NC and deletes the oscillation mode enable:
- PLC → NC: DB31, ... DBX16.0 - 2 = 2 (actual gear stage)
- PLC → NC: DB31, ... DBX16.3 = 1 (gear stage has been switched over)
- PLC → NC: DB31, ... DBX18.5 = 0 (oscillation enable)

t_4: The NC exits the gear stage switchover by resetting the request to the PLC and the oscillation by switching over the spindle mode from oscillation mode to control mode:
- NC → PLC: DB31, ... DBX82.3 = 0 (switchover gear stage)
- DB31, ... DBX84.6 = 0 (active spindle mode: oscillating mode)
- DB31, ... DBX84.7 = 1 (active spindle mode: control mode)

The NC enables the next block for machining, and accelerates the spindle to the new S value S1300.

*: The internal feed disable is set if:
- Gear stage switchover as a result of programming the S value in the NC program **AND**
- A **machining**block is active (not rapid traverse `G0`)

Note

The internal feed disable is **not** set during a gear stage change from synchronized actions or in the case of specifications via the PLC with FC 18.

Figure 18-6 Gear stage change with stationary spindle

Boundary conditions

- The NC/PLC interface signal does not have to be set in order to brake the spindle:
  DB31, ... DBX4.3 (spindle stop)
  The spindle is brought to a standstill in the control system when a gear stage switchover is requested.
- The gear stage switchover must always be exited by setting the NC/PLC interface signal:
  DB31, ... DBX16.3 = 1 (gear stage is changed)
- Oscillation motion in the NC is stopped when the NC/PLC interface signal is reset.
  DB31, ... DBX18.5 (oscillation enable)
  However, the spindle remains in the "Oscillation mode" mode.
- Spindle synchronization is lost for gear stage switchover and indirect measuring system (MD31050 $MA_ENC_IS_DIRECT = 0 motor encoder).
  Gear stage switchover ⇒ DB31, ... DBX60.4 / 5 = 0 (referenced/synchronized)
  The spindle is again automatically synchronized when the zero mark is passed the next time after a gear stage switchover.

18.4.8 Gear stage change at fixed position

Application and advantages

Machine tools increasingly use standardized spindle drives, firstly to save technological dead time on a gear stage change and secondly to gain the cost benefits of using standardized components.

The "Gear stage change at fixed position" function supports the "directed gear stage change" of load gearboxes that need to be activated in a different way than the NC. The gear stage change can in this case only be performed at a defined spindle position. An oscillation movement as required by conventional load gearboxes is thus no longer necessary.

Sequence for gear stage change at fixed position

The gear stage change at fixed position

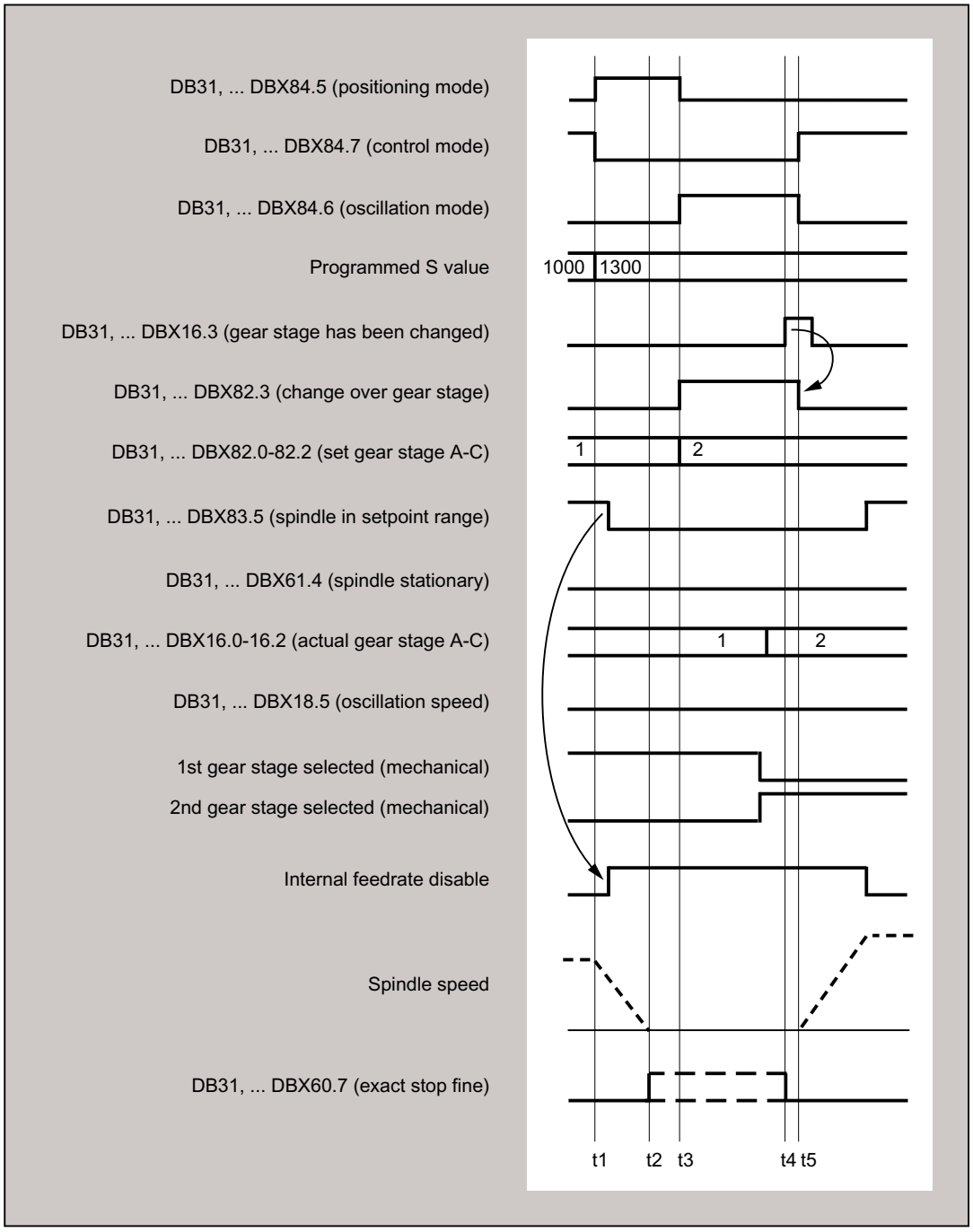
Machine data configuration:
MD35010 $MA_GEAR_STEP_CHANGE_ENABLE = 2
runs the following sequence:

- Positioning of the spindle from standstill or movement to the position configured in machine data:
  MD35012 $MA_GEAR_STEP_CHANGE_POSITION.
  If the gear stage change is performed out of a movement, then the current direction of rotation is maintained. The spindle is in positioning mode during the positioning action.
  NC/PLC IS:
  DB31, ... DBX84.5 (positioning mode)
  is output.
  If no reference is available:
  DB31, ... DBX60.4/5 = 0
  or NC/PLC IS:
  DB31, ... DBX17.4/5 (resynchronize on positioning MS 1/2)
  is set, the positioning action is extended by the time it takes to find the zero mark.
- After reaching the gear stage change position configured in machine data:
  MD35012$MA_GEAR_STEP_CHANGE_POSITION
  the machine waits for the time in machine data:
  MD35310 $MA_SPIND_POSIT_DELAY_TIME
  before switching to oscillation mode,
  and the known gear stage change dialog starts.
- Output of NC/PLC interface signals:
  DB31, ... DBX84.6 (oscillation mode)
  DB31, ... DBX82.3 (change gear)
  DB31, ... DBX82.0-82.2 (set gear stage A to C).
- Position control is not disabled when an active measuring system with indirect encoder (motor encoder) is used:
  MD31040 $MA_ENC_IS_DIRECT = 0
  If a measuring system with a direct encoder (load encoder) is active, position control is deactivated:
  DB31, ... DBX61.5 = 0,
  because the induction flux to the load is interrupted and closed-loop position control is no longer possible.
- If position-controlled operation is not possible, it can be disabled by resetting "Controller enable":
  DB31, ... DBX2.1 = 0
  .

- Mechanical switchover of the gear stage on the machine.
  No oscillation movement is required from the drive.
  NC/PLC IS:
  DB31, ... DBX18.5 (oscillation enable)
  and
  DB31, ... DBX18.4 (oscillation via PLC)
  should **not** be set.
  In principle, oscillation movement is still possible at this point.
- Writing of NC/PLC IS:
  DB31, ... DBX16.0-16.2 (actual gear stage A to C)
  by the PLC.
- After signal:
  DB31, ... DBX16.3 (gear stage changed),
  the last movement to be active is continued, if available.
  For indirect encoders (motor encoders), the referencing status is cleared:
  DB31, ... DBX60.4/5 = 0.
  The spindle is in speed control mode and NC/PLC IS:
  DB31, ... DBX84.7 (open-loop control mode)
  is output.

GSC at fixed position

Typical time sequence for the gear stage change at fixed position:



t_1: With the programming of S1300, NC detects a new gear stage (second gear stage), sets IS DB31, ... DBX84.5 (positioning mode) and blocks processing for the next part program block (= internal feed disable*).

t_2: The spindle is stationary, and exact stop is signaled.

t_3: Gear stage change - wait time

t_4: The new gear stage is engaged. The PLC user transfers the new (actual) gear stage to the NC and sets IS DB31, ... DBX16.3 (gear is changed).

t_5: The NC then retracts IS DB31, ... DBX82.3 (change gear), releases the next part program block for processing, and accelerates the spindle to the new S value (S1300).

* : The internal feed disable is set if:

- The spindle gear stage change has been programmed via the part program **and**
- A processing block is activated (i.e. G0 is not active)

The internal feed disable is **not** set during a gear stage change from synchronized actions or in the case of specifications via the PLC with FC 18.

Figure 18-7 Gear stage change with stationary spindle

gear stage change position MD35012

The gear stage change position is defined in machine data:
MD35012 $MA_GEAR_STEP_CHANGE_POSITION
for each gear stage.

Gear stage change wait time MD35310

After the positioning action the machine waits for the time configured in machine data:
MD35310 $MA_SPIND_POSIT_DELAY_TIME
until gear change request:
DB31, ... DBX84.6 (oscillation mode)
DB31, ... DBX82.3 (change gear)
and
DB31, ... DBX82.0-82.2 (set gear stage A to C)
are output.

Position identifiers / position

The position is always approached via the shortest path (corresponds to DC).

If no reference is available and the spindle is in standstill
(e.g. after Power On), then the direction of travel is determined by the following machine data:
MD35350 $MA_SPIND_POSITIONING_DIR

If an adjustable gear stage change position is required, then this can be achieved by writing the machine data and by a subsequent "Activate machine data".
The change of the MD value can be achieved by the part program or HMI.

If the system is unable to reach the preset position, then alarm 22020 is signaled and the gear stage change dialog between NC and PLC does not take place in order not to destroy the gears. As this alarm is serious, the part program cannot continue and the cause must be eliminated under all circumstances. Experience has shown that the abortion of positioning is usually due to incorrect MD settings or incompatible PLC signals.

Velocity

The positioning velocity is taken from the machine data which is configured depending on the gear stage:

MD35300 $MA_SPIND_POSCTRL_VELO

The NC/PLC interface signals "Spindle override"/"Feedrate override" at
DB31, ... DBX17.0=0: DB31, ... DBB19)
as well as:
DB31, ... DBX17.0=1: DB31, ... DBB0
are effective as normal for positioning.
The positioning speed can be changed proportionally through the program statement `OVRA[Sn]`.

Note

`OVRA[Sn]` is valid modally. After the gear stage change, a value appropriate for the machining should be reset.

The part program statement `FA[Sn]` does not change the positioning speed during gear stage change.

Acceleration

The acceleration values are determined by the machine data which is configured depending on the gear stage:

MD35200 $MA_GEAR_STEP_SPEEDCTRL_ACCEL

and

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

The acceleration can be changed proportionally by programming `ACC[Sn]`.

Note

`ACC[Sn]` is valid modally. After the gear stage change, a value appropriate for the machining should be reset.

Speed-dependent acceleration

The "knee-shaped acceleration characteristic" is effective as in positioning with `SPOS` or FC18.

Jerk

It is currently not possible to limit the change in acceleration.

End of positioning

The transition between the end of the positioning action (DB31, ... DBX84.5)
and the start of oscillation mode (DB31, ... DBX84.6) is defined on
reaching "Exact stop fine" (DB31, ... DB60.7) and the time value entered in machine data:
MD3510 $MA_SPIND_POSIT_DELAY_TIME
.

The determination of the transition condition has an effect firstly on the gear stage change time and secondly on the accuracy of the approach to the preset gear stage change position.

Block change

The block change is stopped and the machining blocks are not started until the gear stage has been changed by the PLC (DB31, ... DBX16.3).

End of gear stage change

Once the gear stage change has been completed, the spindle returns to open-loop control mode and will automatically change to the closed-loop control mode defined by `SPCON` or `SPCOF`.

All gear-specific limit values (min./max. speed of gear stage, etc.) correspond to the check-back values of the actual gear stage.

Supplementary conditions

- The spindle must have at least one measuring system.
- Position-controlled operation must be possible and must have been activated.
- Generally, it must be possible to execute `SPOS` from the part program, from a synchronized action or via FC18: "Start spindle positioning" without errors.

Unless all requirements can be met, the function described cannot be used successfully.

Activation

The function of gear stage change at fixed position
is activated by the configuration:
MD35010 $MA_GEAR_STEP_CHANGE_ENABLE = 2

18.4.9 Configurable gear step in M70

Technical background

In some machines the spindle needs to be in a particular gear stage during axis mode.

Possible reasons:

- Only one optimization (K_V, feedforward control, filter) to suit a gear stage can be found in the servo parameter set for axis mode (index 0). The machine data for this parameter set should not be rewritten.
- There is only one mechanical transmission ratio which, unlike the others, possesses little or no backlash compensation. The spindle can only follow a path motion or transformations (e.g. TRANSMIT) together with other axes in this gear stage.

Function

If the function is activated, a predefined gear stage is loaded automatically during transition to axis mode.

The gear stage change is integrated into the M70 process and occurs after spindle deceleration and before the loading of the servo parameter set with index 0 (note MD35590 $MA_PARAMSET_CHANGE_ENABLE!).

The typical dialog between NC and PLC which occurs during gear stage changes is executed in a similar way to programmed gear stage changes (M41 ... M45) performed.

Requirements

Gear stage changes during transition to axis mode require general enabling of the gear stage change via the machine data:

MD35010 $MA_GEAR_STEP_CHANGE_ENABLE (assign parameters to the gear stage change).

MD35090 $MA_NUM_GEAR_STEPS (number of gear stages set up)

Activation/deactivation

The function is activated/deactivated via the machine data:

MD35014 $MA_GEAR_STEP_USED_IN_AXISMODE (gear stage for axis mode in M70)

Value	Meaning
0	No implicit gear stage change occurs in M70. The current gear stage is retained (default setting!).
1 ... 5	A gear stage change to gear stage 1 ... 5 occurs during the processing of M70.

Supplementary conditions

Gear stage change at fixed position (MD35010 $MA_GEAR_STEP_CHANGE_ENABLE = 2)

The "gear stage change at fixed position" function is supported. The sequence in M70 is then extended by the time it takes to position the spindle. The position is approached at the current gear stage.

Transition to axis mode without programming M70

The control system detects the transition to axis mode automatically from the program sequence (see "Implicit transition to axis mode (Page 759)") and generates the requisite M70 sequence, including the gear stage change, within the control system.

Transition to axis mode with FC 18

Implicit gear stage change is not supported in transition to axis mode with the FC 18 ("Start axis"). This requires the right gear stage to be engaged by the PLC application before switching to axis mode. The gear stage change is also possible with the FC 18 ("Start gear stage change").

Change from axis mode to spindle mode

When changing from axis mode to spindle mode, the gear stage loaded with M70 remains activated. The gear stage which is activated in spindle mode prior to M70 is not automatically loaded again. The servo parameter set is changed from parameter set 1 (index 0) to parameter sets 2 ... 6 (index 1 ... 5) to suit the gear stage (with MD35590 $MA_PARAMSET_CHANGE_ENABLE < 2).

Example

Gear stage 4 should be loaded in the case of spindle transition to axis mode.

Configuration: MD35014 $MA_GEAR_STEP_USED_IN_AXISMODE[<spindle name>] = 4

```
Program code                Comment
N05 M3 S1000

N10 G1 X100 F1000

N15 M70                     ; Gear stage 4 is loaded.
N20 POS[C]=77

N25 ...
```

Note

MD35014 can be changed by the "Activate machine data" function. Thus, the gear stage being loaded can still be changed in the part program before transition to axis mode, if necessary.

18.4.10 Suppression of the gear stage change for DryRun, program test and SERUPRO

Function

For test feed rate (DryRun), program test and SERUPRO, normally, a gear stage change is not required. This is the reason that it can be suppressed for these functions. The corresponding configuration is realized with bits 0 ... 2 in machine data:

MD35035 $MA_SPIND_FUNCTION_MASK

Dry run feedrate (DryRun)	
Bit 0 = 0	When the DryRun function is active - for part program blocks - gear stages are changed with M40, M41 to M45, or programming via FC18 and synchronized actions.
Bit 0 = 1	When the DryRun function is active - for part program blocks - a gear stage change is suppressed with M40, M41 to M45, programming via FC18 and synchronized actions.
Program test and SERUPRO	
Bit 1 = 0	For active program test / SERUPRO function - for part program blocks - gear stages are selected with M40, M41 to M45, programming via FC18 and synchronized actions.

Bit 1 = 1	For active program test / SERUPRO function - for part program blocks - a gear stage change is suppressed with `M40`, `M41` to `M45`, programming via FC18 and synchronized actions.
DryRun, program testing and SERUPRO	
Bit 2 = 0	Gear stage change for programmed gear stage is **not** performed subsequently on REPOS after deselection of functions DryRun, Program Test and SERUPRO.
Bit 2 = 1	Gear stage change for programmed gear stage is performed after deselection of functions DryRun and SERUPRO if possible.

Sequence

If a gear stage change is suppressed, if necessary, the interpolator will limit the programmed spindle speed to the permissible speed range of the active gear stage.

NC/PLC interface signals DB31, ... DBX83.2 (setpoint speed increased) and DB31, ... DBX83.1 (setpoint speed limited) generated as a result of this limit are suppressed.

Monitoring by the PLC program is not necessary during DryRun and in dry run feedrate.

When the gear stage change is suppressed, no new gear stage setpoint (DB31,... DBX82.0-82.2) is output to the PLC.

The gear stage change request DB31, ... DBX82.3 (change gear) is also suppressed.

This ensures that no gear stage change information has to be processed by the PLC program.

Determining the last active gear stage

System variable $P_GEAR returns the gear stage programmed in the part program (which may not have been output to the PLC).

System variable $AC_SGEAR can be used to read the last active gear stage from the part program, synchronized actions and at the user interface.

Behavior after deselection

The DryRun function can be deselected within a running part program. Once it has been deselected, the correct gear stage requested by the part program must be identified and selected.

It cannot be assured that the remainder of the part program will run without errors until the correct gear stage has been activated. Any necessary gear stage change is performed in the system REPOS started on deselection if the spindle is in speed control mode. A complete gear stage change dialog takes place with the PLC and the last programmed gear stage is activated.

If, for REPOS, there is a mismatch between the gear stage programmed in the part program and the actual gear stage supplied via the NC/PLC interface, then no gear stage change takes place.

The same applies to the SERUPRO function.

For further explanations regarding the block search SERUPRO, see:
Further information
G2: Velocities, setpoint / actual value systems, closed-loop control (Page 31)

Boundary conditions

If the gear stage change is suppressed, the output spindle speed moves within the speed range specified by the current gear stage.

The following restrictions apply to the subsequent activation of a gear stage change with REPOS:

- The gear stage change is not activated subsequently if the spindle in the deselection or target block is a command spindle (synchronized action) or PLC spindle (FC18).
- If the gear stage cannot be activated because the spindle is in position or axis mode or a link is active, alarm 22011"Channel%1 block%3 spindle2% Change to programmed gear stage not possible" is signaled.

Example

Gear stage change in DryRun

```
; 1st gear stage for output state activation
N00 M3 S1000 M41                    ; 1st gear stage is selected
M0                                  ; Part program stop

; PI service: Activate dry run feedrate (DryRun)
                                    ; (Configuration)
N10 M42                             ; 2nd gear stage requested, no gear stage change
                                    takes place
N11 G0 X0 Y0 Z0                     ; Position axes
N12 M0                              ; Part program stop

; PI service: Deactivate dry run feedrate (DryRun)
                                    ; REORG and REPOS are performed
                                    ; now the gear stage change to the 2nd gear
                                    stage takes place
N20 G1 Z100 F1000
...
N99 M30                             ; Part program end
```

18.5 Additional adaptations to the spindle functionality that can be configured

The following spindle functions can be activated using the machine data:

MD35035 $MA_SPIND_FUNCTION_MASK, <bit> = <value>

<Bit>	<Value>	Meaning
0 ... 2	1	Activation: Gear stage change behavior for test feedrate (DryRun), program test and SERUPRO See "Suppression of the gear stage change for DryRun, program test and SERUPRO (Page 807)".
4	1	Spindle speed S..., programmed in the NC program, is taken after SD43200 $SA_SPIND_S. This also applies to speed setpoint signals via the PLC user program with FC18 and synchronized actions. See "Function (Page 820)".
5	0	When traversing the spindle in the JOG mode, the currently valid speed setpoint applies. See SD41200 $SN_JOGSPIND_SET_VELO
	1	When traversing the spindle in the JOG mode, the speed parameterized in SD43200 $SA_SPIND_S is effective.
8	1	Cutting velocity S..., programmed in the NC program is taken after SD43202 $SA_SPIND_CONSTCUT_S. This also applies to cutting velocity setpoint signals via the PLC user program with FC18. See "Function (Page 820)".
10	0	SD 43206 $SA_SPIND_SPEED_TYPE is not changed by the NC program or channel settings.
	1	If the spindle is a master spindle, then the feedrate type (active value of the 15th G group) is taken after SD43206 $SA_SPIND_SPEED_TYPE. If the spindle is not a master spindle, then SD43206 $SA_SPIND_SPEED_TYPE remains unchanged. See "Function (Page 820)".
12	Effectiveness of the spindle override for a zero mark search, M19, SPOS, SPOSA = 0:	
	0	The spindle override is**not** active.
	1	The spindle override is active.
19	Response regarding an implicit auxiliary function output M19:	
	0	**AND** MD20850 $MC_SPOS_TO_VDI == 0: For SPOS and SPOSA, auxiliary function M19 is not output to the PLC.
	1	For SPOS and SPOSA, auxiliary function M19 is output to the PLC. The address extension corresponds to the spindle number. See "Positioning mode (Page 739)"
20	Response regarding an implicit auxiliary function output M70:	
	0	An implicit auxiliary function M70 is not output to the PLC. **Note**: A programmed auxiliary function M70 is always output to the PLC.
	1	When transitioning from spindle to axis operation, auxiliary function M70 is output to the PLC. The address extension corresponds to the spindle number. See "Implicit transition to axis mode (Page 759)".

<Bit>	<Value>	Meaning
22	**Deactivating** the axis-specific NC/PLC interface signal **DB31, ... DBX17.6** (invert M3/M4) to reverse the direction of rotation of the spindle for tapping without compensating check (**G331**, **G332**).	
	0	Interface signal DB31, ... DBX17.6 **active**. **Note** • **Standard case**: The value of the interface signal is determined when the NC program starts. When the interface signal subsequently changes, this has no effect within the NC program. • **Special case**: If a change to the interface signal must be taken into account while the NC program is being executed, then this must be implemented as part of the application engineering. See Chapter "Special case: Direction of rotation reversal via NC/PLC interface signal in the NC program (Page 767)".
	1	Interface signal DB31, ... DBX17.6 **not active**.

18.6 Selectable spindles

Function

Regarding channel spindles, the "Switchable spindles" function allows general NC programs to be written, which can be used in different channels. The logical spindle number used in the NC program is converted to the physical spindle number using a spindle number converter. This means for the same spindle number programmed in the NC program, different physical spindles (machine axes) can be traversed in different channels.

The physical spindles loaded or unloaded by "axis replacement" no longer have to be specified explicitly in the part program.

Machine data

Physical spindle number

Each spindle must be assigned to a machine axes using a unique configurable number, the physical spindle number:

MD35000 $MA_SPIND_ASSIGN_TO_MACHAX[<machine axis>] = <physical spindle number>

Activation of the spindle number converter

The spindle number converter is activated on a channel-for-channel basis using machine data:

MD20092 $MC_SPIND_ASSIGN_TAB_ENABLE = TRUE

Setting data

Spindle number converter

To parameterize the channel-specific spindle number converter, each logical spindle number or programmed address extension - with which a spindle (channel spindle) is addressed in the NC program - is assigned a physical spindle number (machine axis).

SD42800 $SC_SPIND_ASSIGN_TAB[<logical spindle number>] = <physical spindle number>

Additional information:

- The logical spindle number of the current master spindle can be read from SD42800 $SC_SPIND_ASSIGN_TAB[**0**] It is not permissible to write to this data. The data is updated when programming `SETMS`.
  Unused spindles are assigned the value 0 in SD42800.
- When an auxiliary function is output, the physical spindle number is output as address extension.
- The spindle number converter is active when programming spindles in NC programs and synchronized actions.
- The spindle number converter is **not** active when spindles are entered at the NC/PLC interface using function block FC18, as in this case the physical spindles are directly addressed.
- System variables affected by the spindle conversion are: $P_S, $P_SDIR, $P_SMODE, $P_GWPS, $AC_SDIR, §AC_SMODE, $AC_MSNUM, $AA_S.

Supplementary conditions

- Switchable channel spindles are **not** a substitute for the Axis replacement function.
- Spindle conversion can be modified by writing to setting data (SD42800 $SC_SPIND_ASSIGN_TAB) via the NC program, PLC user program or PI service. The change takes effect immediately.
- In the spindle number converter, only physical spindles can be used that are assigned to the channel (MD20070 $MC_AXCONF_MACHAX_USED).
  If physical spindles are specified in the spindle number converter, which are presently active in another channel, then depending on the setting in MD30552 $MA_AUTO_GET_TYPE, either the physical spindle is requested for the channel, or alarm Alarm 16105 "Assigned spindle does not exist" is displayed.
- If SD42800 $SC_SPIND_ASSIGN_TAB[<n>] is specified by the PLC or from HMI, then the channel whose table is changed should be in Reset status or the spindle to be changed should not be used in the running part program.
  A synchronized response can be achieved by means of a `STOPRE` preprocessor stop.
- Converting a logical to several physical spindles is not locked. However, with the display of the logical spindle in the user interface, there are ambiguities corresponding to the conversion table.

Example

Assignment, spindle number and machine axis:

```
MD35000 $MA_SPIND_ASSIGN_TO_MACHAX [AX4] = 1
MD35000 $MA_SPIND_ASSIGN_TO_MACHAX [AX5] = 2
MD35000 $MA_SPIND_ASSIGN_TO_MACHAX [AX6] = 3
MD35000 $MA_SPIND_ASSIGN_TO_MACHAX [AX7] = 5
```

Accepting a machine axis in a channel:

```
MD20070 $MC_AXCONF_MACHAX_USED[0] = 4
MD20070 $MC_AXCONF_MACHAX_USED[1] = 5
MD20070 $MC_AXCONF_MACHAX_USED[2] = 6
MD20070 $MC_AXCONF_MACHAX_USED[3] = 7
```

Specifying the master spindle:

```
MD20090 $MC_SPIND_DEF_MASTER_SPIND = 1
```

Spindle number converter

```
MD20092 $MC_SPIND_ASSIGN_TAB_EN-  ; Activate spindle number converter
ABLE=1
SD42800 $SC_SPIND_AS-             ; Master spindle as configured
SIGN_TAB[0]=1
SD42800 $SC_SPIND_AS-             ; Basic setting of the table
SIGN_TAB[1]=1
SD42800 $SC_SPIND_AS-
SIGN_TAB[2]=2
SD42800 $SC_SPIND_AS-
SIGN_TAB[3]=3
SD42800 $SC_SPIND_AS-             ; Logical spindle not assigned
SIGN_TAB[4]=0
; Address extension is 1 output: M1=3 S1=1000
; Spindle 1 (physical master spindle) rotates
M3 S1000
...
; Assignment of logical spindle 1 to physical spindle 5
SD42800 $SC_SPIND_AS-
SIGN_TAB[1]=5

; Assignment of logical spindle 2 to physical spindle 3
SD42800 $SC_SPIND_AS-
SIGN_TAB[2]=3
```

```
; Notice: physical spindle 3 has now been assigned twice.
; When programming logical spindles 2 and 3, physical spindle 3
; is always addressed.
; In the basic machine displays, both spindles rotate.

SETMS(2)                          ; SD42800 $SC_SPIND_ASSIGN_TAB[0] = 2
; Master spindle = address extension = 2, converted spindle number M3 = 5 is output
; The physical spindle configured with number "3" stops.
M5
GET(S4)                           ; Alarm 16105: logical spindle 4
                                  ; cannot be converted
RELEASE(S1)                       ; Channel spindle 1 = physical spindle 5
                                  ; is released.
```

18.7 Programming

18.7.1 Programming from the part program

Programming statements

Statement	Description	
SETMS:	Master spindle is the spindle specified in the following machine data: MD20090 $MC_SPIND_DEF_MASTER_SPIND (position of deletion of the master spindle in the channel)	
SETMS(<n>):	The spindle with the number <n> is the master spindle (may differ from the initial setting: MD20090 $MC_SPIND_DEF_MASTER_SPIND). The master spindle must be defined for the following functions:	
	• G95:	Revolutional feedrate
	• G96 S.../G961 S...:	Constant cutting rate in m/min or feet/min
	• G97/G971:	Cancel G96/G961 and freeze last spindle speed
	• G63:	Tapping with compensating chuck
	• G33/G34/G35:	Thread cutting
	• G331/G332:	Tapping without compensating chuck
	• G4 S...:	Dwell time in spindle revolutions
	• Programming M3, M4, M5, S, SVC, SPOS, M19, SPOSA, M40, M41 to M45, and WAITS without entering the spindle number	
	The current master spindle setting can be retained via RESET / part program end and part program start. The setting is done via the machine data: • MD20110 $MC_RESET_MODE_MASK • MD20112 $MC_START_MODE_MASK	
M3: M<n>=3:	Clockwise spindle rotation for the master spindle Clockwise spindle rotation for spindle number <n>	
M4: M<n>=4:	Counter-clockwise spindle rotation for the master spindle Counter-clockwise spindle rotation for spindle number <n>	
M5: M<n>=5:	Spindle stop without orientation for the master spindle Spindle stop without orientation for spindle number <n>	
S...: S<n>=...:	Spindle speed in rpm for the master spindle Spindle speed in rpm for spindle number <n>	
SVC=...: SVC[<n>]=...:	Cutting rate in m/min or feet/min for the master spindle Cutting rate in m/min or feet/min for the spindle <n>	
SPOS=...: SPOS[<n>]=...:	Spindle positioning for the master spindle Spindle positioning for spindle number <n> The block change is only performed when the spindle is in position.	

Statement	Description
SPOSA=...: SPOSA[<n>]=...:	Spindle positioning for the master spindle Spindle positioning for spindle number <n> The block change is executed immediately. Spindle positioning continues, regardless of further part program processing, until the spindle has reached its position.
SPOS=DC(...): SPOS[<n>]=DC(...): SPOSA=DC(...): SPOSA[<n>]=DC(...):	The direction of motion is retained for positioning while in motion and the position approached. When positioning from standstill, the position is approached via the shortest path.
SPOS=ACN(...): SPOS[<n>]=ACN(...): SPOSA=ACN(...): SPOSA[<n>]=ACN(...):	The position is always approached with negative direction of motion. If necessary, the direction of motion is inverted prior to positioning.
SPOS=ACP(...): SPOS[<n>]=ACP(...): SPOSA=ACP(...): SPOSA[<n>]=ACP(...):	The position is always approached with positive direction of motion. If necessary, the direction of motion is inverted prior to positioning.
SPOS=IC(...): SPOS[<n>]=IC(...): SPOSA=IC(...): SPOSA[<n>]=IC(...):	The travel path is specified. The direction of travel is determined from the sign in front of the travel path. If the spindle is in motion, the direction of travel is inverted as necessary to allow traversing in the programmed direction. If the zero mark is crossed during traversing, the spindle is automatically synchronized with the zero mark if no reference is available or if a new one has been requested via an interface signal.
M19: M[<n>]=19:	Positioning the master spindle to the position in SD43240 Positioning spindle number <n> to the position in SD43240 The block change is only performed when the spindle is in position.
M70: M<n>=70:	Bring spindle to standstill and activate position control, select zero parameter set, activate axis mode for the master spindle for spindle number <n>
SPCON: SPCON(<n>): SPCON(<n>,<m>):	Spindle position control ON for the master spindle for spindle number <n> for spindle numbers <n> and <m>
PCOF: SPCOF(<n>): SPCOF(<n>,<m>):	Spindle position control OFF, activate speed control mode for the master spindle for spindle number <n> for spindle numbers <n> and <m>
FPRAON(S<n>):	Revolutional feedrate for spindle <n> ON, derived from the master spindle
FPRAON(S<n>,S<m>):	Revolutional feedrate for spindle <n> ON, derived from spindle <m> The revolutional feedrate value must be specified with FA[S<m>].
FPRAOF(S<n>):	Revolutional feedrate for spindle <n> OFF
C30 G90 G1 F3600	Rotary axis C (spindle in axis mode) travels to the position 30 degrees at a speed of 3600 degrees/min = 10 rpm

Statement	Description
	Programmable minimum spindle speed limitation
G25 S...:	for the master spindle
G25 S<n>:	for spindle number <n>
	Programmable maximum spindle speed limitation
G26 S...:	for the master spindle
G26 S<n>:	for spindle number <n>
	Programmable maximum spindle speed limitation with G96, G961, G97
LIMS=...:	for the master spindle
LIMS[<n>]=...:	for spindle number <n>
VELOLIM[<spindle>]=...:	Programmable limiting of the configured gear stage dependent maximum speed Using machine data (MD30455 $MA_MISC_FUNCTION_MASK, bit 6), when programming in the part program, it can be set as to whether VELOLIM is effective independent of whether used as spindle or axis (bit 6 = 1) - or is able to be programmed separately for each operating mode (bit 6 = 0). If they are to be separately effective, then the selection is made using the name when programming: • Spindle name S<n> for spindle operating modes • Axis name, e.g. "C", for axis operation The correction value refers to: • Spindles in axis operation (if MD30455 Bit 6 = 0): To the configured maximum axis velocity (MD32000 $MA_MAX_AX_VELO). • Spindles in spindle or axis operation (if MD30455 bit 6 = 1): To the maximum speed of the active gear unit stage (MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[<n>]) For further explanations about the programming of VELOIM, see: **Further information** Programming Manual Advanced
WAITS:	Synchronization command for master spindle The subsequent blocks are not processed until the spindle programmed in a previous NC block with SPOSA has reached its position (with exact stop fine). WAITS after M5: Wait until the spindle is stationary. WAITS after M3/M4: Wait until the spindle reaches its setpoint speed.
WAITS(<n>):	Synchronization command for spindle number <n>
WAITS(<n>,<m>):	Synchronization command for spindle numbers <n> and <m>
FA[S<n>]:	Programming of positioning speed (axial feed) for spindle <n> in [deg/min] With FA[S<n>]=0, the configured value takes effect once more: MD35300 $MA_SPIND_POSCTRL_VELO
OVRA[S<n>]:	Programming of the axial override value for spindle <n> in [%]
ACC[S<n>]:	Programming of the axial acceleration capacity for spindle <n> in [%]

Statement	Description
SPI(<n>):	With SPI(<n>) a spindle number is converted into the data type AXIS according to machine data MD35000 $MA_SPIND_ASSIGN_TO_MACHAX[ ] . SPI is used if axis functions are to be programmed with the spindle. The following commands are possible with SPI: • Frame commands: – CTRANS() – CFINE() – CMIRROR() – CSCALE() • Velocity and acceleration values for following spindles: – FA[SPI(<n>)] – ACC[SPI(<n>)] – OVRA[SPI(<n>)] • System variables: – $P_PFRAME[SPI(<n>),TR]=<value> – $P_PFRAME= CTRANS(X,<axis value>,Y,<axis value>,SPI(<n>),<axis value>) – $P_PFRAME= CSCALE(X,<scale>,Y,<scale>,SPI(<n>),<scale>) – $P_PFRAME=CMIRROR(S<n>,Y,Z) – $P_UBFR=CTRANS(A,10) : CFINE (19,0.1) For further explanations about the programming of SPI, see: **Further information** Programming Manual Advanced
M40: M<n>=40:	Automatic gear stage selection for the master spindle Automatic gear stage selection for spindle number <n>
M41 to M45: M<n>=41 to M<n>=45:	Select gear stage 1 to 5 for the master spindle Select gear stage 1 to 5 for spindle number <n>.

Note

M functions M3, M4, M5, and M70 are not output in DB21, ... DBB194 and DBB202 if a spindle is configured in a channel. These M functions are offered as extended M functions in DB21, ... DBB68 ff. and in the relevant axis DBs, DB31, ... DBB86 ff.

Further information

More detailed explanations for programming the spindle can be found in:

- Programming Manual, Fundamentals

18.7.2 Programming via synchronized actions

Description

M functions M40 to M45 can also be programmed in synchronized actions.

The following must be observed:

- Programming of M40 ... M45 in the part program has no effect on the current status of the automatic gear step change of synchronized actions, and vice versa.
- When programming S values with M40, automatic gear step change is effective separately for synchronized actions and the part program.
- M40 is deactivated after POWER ON.
  The gear step is not adjusted if an S value is specified from a synchronized action.
- An M40 command programmed using synchronized actions always remains active for synchronized actions (modal) and is not reset on reset.
- M41 ... M45 selects first to fifth gear steps in accordance with the programming in the part program.
  An axis replacement is necessary in order to run the function.
  Once the gear step change has been performed, the spindle status is neutral (same response to M3, M4, M5 programming).

Further information

For further explanations regarding the programming of the spindle as well as spindle movements from synchronized actions, refer to:

- Programming Manual Job Planning
- Function Manual Synchronized Actions

18.7.3 Programming spindle controls via PLC with FC18

When the PLC specifies the direction of rotation and speed using FC18, the NC can determine and select a gear stage to match the speed. This is equivalent to the M40 functionality when programming via the part program.

The correct start code must be set when FC18 is called in a PLC user program, in order to activate gear stage selection.

Further information

More detailed explanations regarding the programming of spindle controls by PLC with FC18 can be found in:

- Function Manual PLC

18.7.4 Programming using NC/PLC interface signals

18.7.4.1 Function

Note

The function is only available when using SINUMERIK Operate!

The following jobs/tasks can be issued to a spindle from the PLC user program using the job interface DB31, … DBX30.0 - 4:

- Spindle stop
- Spindle start, clockwise rotation
- Spindle start, counter-clockwise rotation
- Select gear stage
- Spindle-start positioning

Job ID

At the NC side, a spindle job is identified as a result of a positive **edge change** of the appropriate interface signal (0 → 1).

Requirements

At the instant of the spindle job, the following preconditions must be satisfied regarding the channel, program and spindle states.

- Channel state:
  - DB21, ... DBX35.6 == 1 (channel state "interrupted") **OR**
  - DB21, … DBX35.7 == 1 (channel state "reset")
- Program state:
  - DB21, ... DBX35.3 = 1 (program state "interrupted") **OR**
  - DB21, ... DBX35.4 = 1 (program state "interrupted")
- Spindle state
  - ("Channel axis" **OR** "Neutral axis") **AND NO**
  - (positioning motion using FC18 **OR** synchronized action)

Note

Response in the event of an error

In the event of an error, i.e. if a job signal is set during an invalid state, then the corresponding job is ignored **without any feedback signal**.

Channel assignment

A spindle job is processed in that particular channel, which is assigned to the spindle at the instant of the job.

The channel, which is assigned to the spindle, can be determined using the following NC/PLC interface signal:

DB31, ... DBX68.0 - 3 (channel assignment of the NC axis/spindle)

Mode change

The definitions made within the context of the spindle jobs are kept beyond any mode change.

Program start

When an NC program starts in the channel, which is assigned to the spindle, then the actual spindle definitions are kept. Spindle definitions can be changed using NC program commands or using synchronized actions.

See also

M40: Automatic gear stage selection for speeds outside the configured switching thresholds (Page 835)

G2: Velocities, setpoint / actual value systems, closed-loop control (Page 31)

18.7.4.2 Commissioning: Machine data

Automatically accepting spindle definitions in setting data

Using the following machine data settings, spindle definitions are transferred from NC programs, synchronized actions and the spindle control by the PLC (FC18) into the corresponding setting data:
MD35035 $MA_SPIND_FUNCTION_MASK, bit x = <value>

Bit	Value	Meaning
4	1	A **speed setpoint** is entered via NC program, synchronized actions or FC18 is transferred into the following spindle-specific setting data: **SD43200** $SA_SPIND_S[<spindle>] **Note** S values that are not programmed speed values are not transferred. e.g.: • S value at constant cutting speed (G96, G961) • S value for rotation-related dwell time (G4)
	0	**No** speed setpoints are transferred into the setting data.

Bit	Value	Meaning
8	1	A **cutting rate** is entered as setpoint in the following spindle-specific setting data via the NC program, synchronized actions or FC18 if the "Constant cutting rate" function is active (G96, G961, G962): **SD43202** $SA_SPIND_CONSTCUT_S[<spindle>] **Note** S values that are not programmed cutting rate values are not transferred. e.g.: • S value for rotation-related dwell time (G4)
	0	**No** cutting rate setpoints are transferred into the setting data.
10	1	For the **master spindle** the **feedrate type** is transferred into the following setting data: **SD43206** $SA_SPIND_SPEED_TYPE[<spindle>]
	0	**No** feedrate type definitions for the master spindle are transferred into the setting data.

With the spindle definitions transferred into the setting data, the spindle is traversed to execute the spindle jobs.

18.7.4.3 Commissioning: NC/PLC interface signals

Job interface

- DB31, … DBX30.0 (spindle stop)
- DB31, … DBX30.1 (spindle start, clockwise)
- DB31, … DBX30.2 (spindle start, counter-clockwise)
- DB31, ... DBX30.3 (select gear stage)
- DB31, … DBX30.4 (spindle positioning start)

Relevant state signals

- DB21, ... DBX35.3 (program state "interrupted")
- DB21, ... DBX35.4 (program state "canceled")
- DB21, ... DBX35.7 (channel state "reset")
- DB21, ... DBX35.6 (channel state "interrupted")
- DB31, ... DBX68.0 - 3 (channel assignment of the NC axis/spindle)
- DB31, ... DBX 84.0 (constant cutting rate)

Further information

A detailed description of the NC/PLC interface signals can be found in the NC Variables and Interface Signals List Manual.

18.7.4.4 Speed setpoint (SD43200)

When starting a spindle using a spindle job, then the spindle-specific value entered in the setting data becomes active as spindle speed:

SD43200 $SA_SPIND_S[<spindle>] (speed when the spindle is started via the PLC job interface)

Writing a new speed value

In the following situations, a new speed value is written to the setting data:

- A speed setpoint is entered via NC program, synchronized action or FC18
  Supplementary conditions:
  - MD35035 $MA_SPIND_FUNCTION_MASK (Page 821), bit 4 == 1
  - DB31, ... DBX84.0 == 0 (constant cutting rate)
- Writing the setting data to the NC program or synchronized action
- Writing the setting data via HMI (OPI) @@@

Supplementary conditions

Gear stage change

A gear stage change is not initiated if the setpoint speed is out of the speed range of the actual gear stage.

Exceptions, see Chapter "M40: Automatic gear stage selection for speeds outside the configured switching thresholds (Page 835)".

18.7.4.5 Entering a constant cutting rate (SD43202)

When starting a master spindle using a spindle job - and the "Constant cutting rate" spindle speed type is active, then the spindle-specific value entered in the setting data becomes active as constant cutting rate:

SD43202 $SA_SPIND_CONSTCUT_S[<spindle>] (cutting rate when the spindle starts via the PLC job interface)

Writing a new cutting rate of value

In the following situations, a new constant cutting rate of the master spindle is written to the spindle-specific setting data:

- A constant cutting rate is entered via NC program, synchronized action or FC18
  Supplementary conditions:
  - MD35035 $MA_SPIND_FUNCTION_MASK (Page 821), bit 8 == 1
  - DB31, ... DBX84.0 == 1 (constant cutting rate active)
- Writing the setting data to the NC program or synchronized action
- Writing the setting data via HMI (OPI) @@@

System of units

When writing the setting data, the value is interpreted corresponding to the following secondary conditions:

- Writing via NC program or synchronized action:
  - G700 active: feet/min
  - G710 active: m/min
  - G70, G71 active: Depending on the setting in MD10240 $MN_SCALING_SYSTEM_IS_METRIC
- Writing via SINUMERIK Operate:
  - Depending on the setting in MD10240 $MN_SCALING_SYSTEM_IS_METRIC
- Writing via FC18:
  - Function number 10: "constant cutting rate (m/min)"
  - Function number 11: "constant cutting rate (feet/min)"

Reading via system or OPI variable

In the NC program, synchronized action or HMI, the currently entered setpoint for the constant cutting rate can be read using the following system data:

System variable	OPI variable	Meaning
$P_CONSTCUT_S	---	Last programmed cutting rate
$AC_CONSTCUT_S	acConstCutS	Actual constant cutting rate

18.7.4.6 Entering the spindle speed type for the master spindle (SD43206)

Definition of the spindle speed type **for the master spindle** from part program, FC18 or synchronized actions are written to the following setting data from all the usual sources:

SD43206 $SA_SPIND_SPEED_TYPE (spindle speed type for spindle start via PLC interface)

The value range and functionality correspond to the 15th G group (feedrate type).

Permissible values are G values: 93, 94, 95, 96, 961, 97 and 971.

Depending on the setting, for DB31, … DBX30.1/2 (spindle start, clockwise/counter-clockwise) either the speed from SD43200 $SA_SPIND_S or the cutting speed from SD43202 $SA_SPIND_CONSTCUT_S is active:

93, 94, 95, 97 and 971:	The master spindle is started with the speed from SD43200.
96 and 961:	The speed of the master spindle is obtained from the specified cutting rate (SD43202) and the radius of the transverse axis.

18.7.5 External programming (PLC, HMI)

SD43300 and SD42600

The revolutional feedrate behaviour can be selected externally via the axial setting data:
SD43300 $SA_ASSIGN_FEED_PER_REV_SOURCE (Rotational feedrate for spindles)
in JOG operating mode using the channel-specific setting data
SD42600 $SC_JOG_FEED_PER_REV_SOURCE (Revolutional fedrate control in JOG mode)
.

The following settings can be made via the setting data:

>0:	The machine axis number of the rotary axis/spindle from which the revolutional feedrate shall be derived.
-1:	The revolutional feedrate is derived from the master spindle of the channel in which the axis/ spindle is active in each case.
0:	Function is deselected.

FPRAON (S2)

Revolutional feedrate for spindle S2 ON, derived from the master spindle

FPRAON (S2, A)

Revolutional feedrate for spindle S2 ON, derived from axis A.
The revolutional feedrate value must be specified with `FA[Sn]`.

FPRAOF (S2)

Revolutional feedrate for spindle S2 OFF.

SPI(n)

It is also possible to program `SPI(n)` instead of `SPI(Sn)`.

18.8 Spindle monitoring

18.8.1 Permissible speed ranges

The permissible speed range of a spindle results from the parameterized or programmed speed limit values and the active spindle function (G94, G95, G96, G961, G97, G971, G33, G34, G35, G331, G332, etc.).

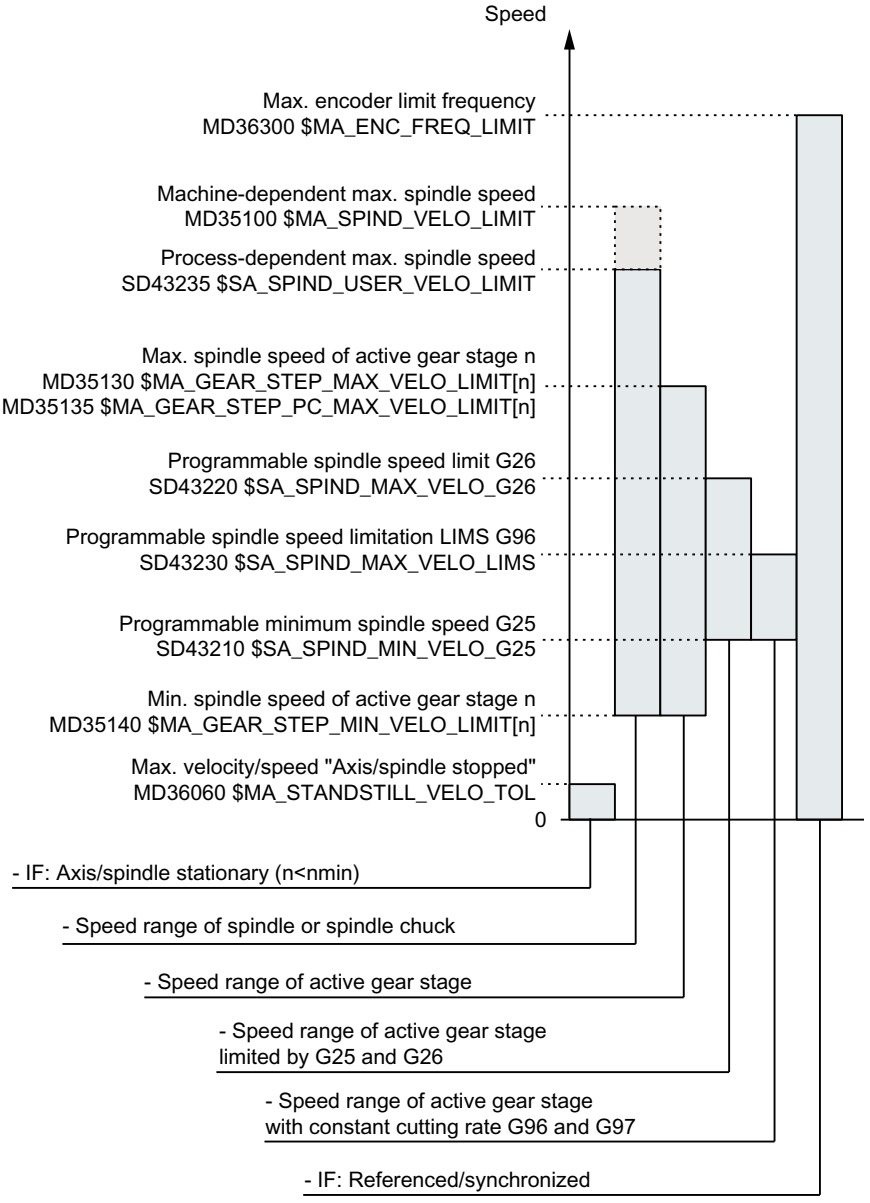


Figure 18-8 Ranges of spindle monitoring functions / speeds

18.8.2 Axis/spindle stationary

Functions such as tool change, open machine doors, path feedrate enable, etc. are only possible at the machine when the spindle is stationary.

Function

The "axis/spindle stationary" state is reached if a setpoint is no longer generated and the spindle actual speed falls below the configured threshold value for "axis/spindle stationary":

MD36060 $MA_STANDSTILL_VELO_TOL (max. velocity/speed for "axis/spindle stationary")

If the spindle has come to a standstill, the following NC/PLC interface signal is set:

DB31, ... DBX61.4 (axis/spindle stationary)

Effectiveness

Monitoring for spindle stop is effective in all spindle modes and in axis mode.

Deactivate path feed

If a spindle is stopped in the open-loop control mode (M5), then path feed is deactivated if the following machine data is set:

MD35510 $MA_SPIND_STOPPED_AT_IPO_START (feedrate enable for spindle stopped)

The path feed is re-enabled if the spindle comes to a standstill.

18.8.3 Spindle in setpoint range

Function

"Spindle in setpoint range" spindle monitoring checks whether:

- The programmed spindle speed is reached.
- The spindle is at a standstill:
  DB31, ... DBX61.4 (axis/spindle stationary) = 1
- The spindle is still in the acceleration or deceleration phase.

In the spindle mode, open-loop control mode, the setpoint speed is compared with the actual speed. If the actual speed deviates by more than the spindle tolerance that can be entered via MD (refer below) then:

- The following axial NC/PLC interface signal is set to "0":
  DB31, ... DBX83.5 (spindle in setpoint range) = 0
- The next machining block is not enabled (depending on the setting in MD35500 $MA_SPIND_ON_SPEED_AT_IPO_START, see "Axis/spindle stationary (Page 827)").

Spindle setpoint speed

The spindle speed setpoint is derived from the programmed speed taking into account the spindle override and the active limits.

If the programmed speed is limited or increased, this is displayed using DB31, ... DBX83.1 (speed setpoint limited) or DB31, ... DBX83.2 (speed setpoint increased) (see also "Minimum / maximum speed of the gear stage (Page 828)"). The means that reaching the tolerance range of the setpoint speed is **not** prevented.

Tolerance range for setpoint speed

The tolerance range of the setpoint speed is defined by the spindle speed tolerance factor:

MD35150 $MA_SPIND_DES_VELO_TOL

Example:

MD35150 $MA_SPIND_DES_VELO_TOL = 0.1

⇒ The spindle actual speed may deviate ±10% from the setpoint speed.

The following NC/PLC interface signal is set to "1" if the spindle actual speed lies within the tolerance range:

DB31, ... DBX83.5 (spindle in setpoint range) = 1

Special case:

If the spindle speed tolerance is set to "0", then DB31, ... DBX83.5 (spindle in the setpoint range) is permanently set to "1" and no path control is performed.

Speed change

Path control only takes place at the start of the traverse block and only if a speed change has been programmed. If the speed tolerance range is exited, e.g. due to an overload, the path movement is not automatically brought to a standstill.

18.8.4 Minimum / maximum speed of the gear stage

Minimum speed

The minimum speed of the gear stage of a spindle is configured in the machine data:

MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT[<n>]

The speed setpoints, generated taking into account the override, do not fall below the minimum speed.

If an S value is programmed, which is less than the minimum speed, the setpoint speed is increased to the minimum speed and the following NC/PLC interface signal is set:

DB31, ... DBX83.2 (speed setpoint increased)

The minimum gear stage speed is effective only in speed mode and can only be undershot by:

- Spindle override 0%
- M5
- S0
- DB31, ... DBX4.3 (spindle stop)
- DB31, ... DBX2.1 (withdraw controller enable)
- DB21, ... DBX7.7 (reset)
- DB31, ... DBX2.2 (delete distance-to-go / spindle reset)
- DB31, ... DBX18.5 (oscillation speed)
- DB21, ... DBX7.4 (NC stop axes plus spindles)
- DB31, ... DBX1.3 (axis/spindle disable)
- DB31, ... DBX16.7 (delete S value)

Maximum speed

The maximum speed of the gear stage of a spindle is configured in machine data:

MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[<n>]

The speed setpoints, generated taking into account the override, are limited to this speed.

The following NC/PLC interface signal is set in the case that the speed is limited:

DB31, ... DBX83.1 (speed setpoint limited)

18.8.5 Diagnosis of spindle speed limitation

System variables

The effective/limiting spindle parameters can be read via the following system variables: The system variables must be indexed with the spindle number and they return values only relevant in the speed control and spindle position modes.

System variable	Meaning
$AC_SMAXVELO[<n>]	Maximum possible spindle speed [RPM].
$AC_SMAXVELO_INFO[<n>]	Identifier for the speed-limiting data element.
$AC_SMINVELO[<n>]	Minimum possible spindle speed [RPM].
$AC_SMINVELO_INFO[<n>]	Identifier for the speed-increasing data element.
$AC_SMAXACC[<n>]	Maximum possible spindle acceleration [rev/s^2].
$AC_SMAXACC_INFO[<n>]	Identifier for the acceleration-limiting data element.
$AC_SPIND_STATE[<n>]	Spindle status in speed-control mode.
<n>: 0, 1, 2, ... maximum spindle number (0: current master spindle of the channel)	

Further information

The detailed description of the system variables can be found in:

List Manual, System Variables

NC/PLC interface signals

The limit or increase of the spindle speed is signaled with the following NC/PLC interface signals:

- DB31, ... DBX83.1 (speed setpoint limited)
- DB31, ... DBX83.2 (speed setpoint increased)

Boundary conditions

Spindle mode

The values delivered by the system variables depend on the spindle mode:

- Speed control mode:
  All system variables deliver current values.
- Positioning mode:
  The system variables $AC_SMAXVELO, $AC_SMAXACC and $AC_SPIND_STATE deliver valid values. The system variables $AC_SMINVELO and $AC_SMINVELO_INFO deliver the data that becomes effective on changing to the speed control mode.
- Axis mode (e.g. if the spindle is used by a transformation TRANSMIT, TRACYL,... or follows a path motion as a special axis):
  The system variable $AC_SPIND_STATE can also be used in the axis mode. Separate system variables are available in the axis mode for dynamic data:
  $AA_VMAXM, $AA_VMAXB and $AA_VLFCT.

SERUPRO block search

The following control response is obtained for a SERUPRO block search:

- The system variable $AC_SMAXVELO / $AC_SMAXACC delivers the maximum representable speed / acceleration.
- $AC_SMAXVELO_INFO and $AC_SMAXACC_INFO deliver the VALUE "0" (no limitation active).
- $AC_SMINVELO and $AC_SMINVELO_INFO deliver data as for normal part program processing.
- $AC_SPIND_STATE returns the states as set for SERUPRO.

Example

The effective/limiting spindle parameters for spindle 1 are written cyclically in the R parameter and displayed as R parameter in the "Parameters" > "User variable" operating area on the HMI.

Program code

```
N05 IDS=1 WHENEVER TRUE DO $R10=$AC_SMAXVELO[1]
```

Program code

```
N10 IDS=2 WHENEVER TRUE DO $R11=$AC_SMAXVELO_INFO[1]
N15 IDS=3 WHENEVER TRUE DO $R12=$AC_SMINVELO[1]
N20 IDS=4 WHENEVER TRUE DO $R13=$AC_SMINVELO_INFO[1]
N25 IDS=5 WHENEVER TRUE DO $R14=$AC_SPIND_STATE[1]
```

See also

Spindle in setpoint range (Page 827)

18.8.6 Maximum spindle speed

Maximum spindle speed: parameterizable machine-related limit value

The maximum machine-dependent spindle speed – for protecting the spindle chuck or the tool, for example – is parameterized using the following machine data:

MD35100 $MA_SPIND_VELO_LIMIT (maximum spindle speed)

The spindle speed is monitored by the NC in the actual values, i.e. taking into account the current gear stage.

Note

Machine manufacturer

It is recommended that changes to the maximum spindle speed in machine data MD35100 only be made when the spindle is stationary. This is particularly important for spindles that are active after NC reset (see also MD35040 SPIND_ACTIVE_AFTER_RESET). Otherwise alarm 22100 may be output.

Further information

- Manufacturer documentation: List Manual, "Detailed Machine Data Description"
- User documentation: Diagnostics Manual

Responses to violation

If the actual speed of a spindle exceeds the parameterized maximum spindle speed by more than the tolerance value,

```
"Actual spindle speed" > MD35100 $MA_SPIND_VELO_LIMIT (maximum
spindle speed) +
MD35150 $MA_SPIND_DES_VELO_TOL (spindle speed tolerance value)
```

the following effects can be seen on the NC side.

- DB31, ... DBX83.0 = 1 (speed limit exceeded)
- Alarm 22100 "Chuck speed exceeded" is displayed
- All axes and spindles of the channel are stopped

Note

Rotary axis / spindle

If a spindle is also temporarily operated as a rotary axis, alarm 22100 "Chuck speed exceeded" is displayed if:

```
"Actual speed of the rotary axis" > MD35100 $MA_SPIND_VELO_LIMIT
(maximum spindle speed) +
MD35150 $MA_SPIND_DES_VELO_TOL (spindle speed tolerance value)
```

Remedy: Adjust the maximum spindle speed to the maximum rotary axis speed during rotary operation:

```
MD35100 $MA_SPIND_VELO_LIMIT = MD32000 $MA_MAX_AX_VELO (maximum
speed of the rotary axis)
```

For the change to take effect a reset must be triggered.

Maximum spindle speed: Configurable process-dependent limit value

The maximum process-related spindle speed is configured through the following setting data:

SD43235 $SA_SPIND_USER_VELO_LIMIT (maximum spindle speed)

Changes can be made by writing the setting data through:

- Programming in an NC program
- Manually through the user interface by the machine operator

Changes become effective immediately.

Limitation of the speed setpoint

Where appropriate, the controller limits programmed spindle speed setpoints to the values specified in the setting data. The limitation is then displayed through the following system variables:

$AC_SMAXVELO_INFO[<Spindle number>] == 21

18.8.7 Maximum encoder limit frequency

⚠ CAUTION

Limit violation

The maximum encoder frequency limit of the actual spindle position encoder is monitored by the control (the limit can be exceeded). It is the responsibility of the machine tool manufacturer to ensure that the configuration of the spindle motor, gearbox, measuring gearbox, encoder and machine data prevents the maximum speed of the actual spindle position encoder being exceeded.

Maximum encoder frequency exceeded.

If the spindle speed reaches a speed (large S value programmed), which exceeds the maximum encoder limit frequency (the maximum mechanical speed limit of the encoder must not be exceeded), the synchronization is lost. The spindle continues to rotate, but with reduced functionality.

With the following functions, the spindle speed is reduced until the active measurement system is operating below the encoder limit frequency again:

- Thread cutting (G33, G34, G35)
- Tapping without compensating chuck (G331, G332)
- Revolutional feedrate (G95)
- Constant cutting rate (G96, G961, G97, G971)
- SPCON (position-controlled spindle operation)

When the encoder limit frequency is exceeded
NC/PLC IS:
DB31, ... DBX60.4 (referenced/synchronized 1)
or
DB31, ... DBX60.5 (referenced/synchronized 2)
are reset for the measurement system in question and NC/PLC IS:
DB31, ... DBX60.2 (encoder limit frequency 1 exceeded)
or
DB31, ... DBX60.3 (encoder limit frequency 2 exceeded)
are set.

If the spindle is in axis mode, the maximum encoder limit frequency must not be exceeded. The maximum velocity (MD32000 $MA_MAX_AX_VELO) must lie below the maximum encoder limit frequency; otherwise, alarm 21610 is output and the axis is brought to a standstill.

Maximum encoder limit frequency undershot

If the maximum encoder frequency limit has been exceeded and the speed subsequently falls below the maximum encoder limit frequency (smaller S value programmed, spindle override switch changed, etc.), the spindle is automatically synchronized with the next zero mark or the next proximity switch signal. The new synchronization will always be carried out for the active position measuring system that has lost its synchronization and whose maximum encoder limit frequency is currently undershot.

Special features

If the following functions are active, the maximum encoder frequency cannot be exceeded:

- Spindle positioning mode, axis mode
- Thread cutting (G33, G34, G35)
- Tapping without compensating chuck G331, G332 (does not apply to G63)
- Revolutional feedrate (G95)
- Constant cutting rate (G96, G961, G97, G971)
- SPCON

18.8.8 End point monitoring

End point monitoring

During positioning (the spindle is in positioning mode), the system monitors the distance from the spindle (with reference to the actual position) to the programmed spindle position setpoint (end point).

For this to work, in machine data:
MD36000 $MA_STOP_LIMIT_COARSE (Exact stop limit coarse)
and
MD36010 $MA_STOP_LIMIT_FINE (Exact stop limit fine)
two limit values can be defined as an incremental path starting from the spindle position setpoint.

Regardless of the two limit values, the positioning of the spindle is always as accurate as defined by the connected spindle measurement encoder, the backlash, the transmission ratio, etc.

Exact stop window dependent on parameter set

Various parameter-set-dependent exact stop windows can be configured.
This makes it possible to work to different levels of accuracy in axis mode and spindle positioning. The exact stop window can be configured separately for each gear step for spindle positioning.

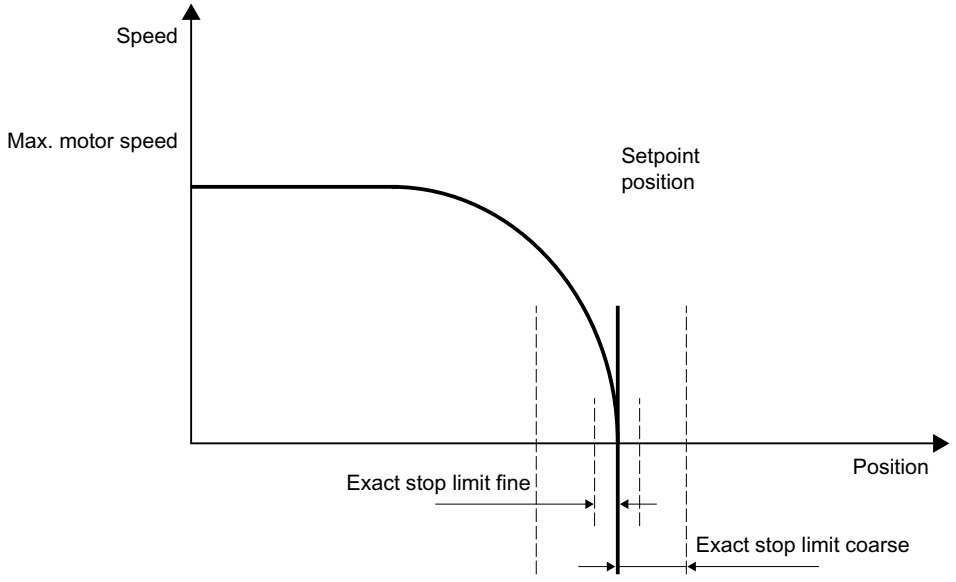


Figure 18-9 Exact stop zones of a spindle

DB31, ... DBX60.7 and DB31, ... DBX60.6 (position reached with exact stop coarse / fine)

The two limit values defined by machine data:
MD36000 $MA_STOP_LIMIT_COARSE (Exact stop limit coarse)
and
MD36010 $MA_STOP_LIMIT_FINE (Exact stop limit fine)
are output to the PLC using NC/PLC IS:
DB31, ... DBX60.7 (Position reached with exact stop coarse)
and
DB31, ... DBX60.6 (Position reached with exact stop fine).

Block change for SPOS and M19

When positioning the spindle with `SPOS` or `M19` the block is changed dependent on end point monitoring with NC/PLC IS:
DB31, ... DBX60.6 (Position reached with exact stop fine).

All other functions programmed in the block must have achieved their end criterion (e.g., all auxiliary functions acknowledged by the PLC).

With `SPOSA`, the block change does not depend on the monitoring of the end point.

18.8.9 M40: Automatic gear stage selection for speeds outside the configured switching thresholds

Function

When M40 is active, an automatic gear stage selection is also made if the programmed spindle speed `S…` lies outside the configured switching thresholds.

In this case, a distinction is made between the following cases:

- **Programmed speed too high**
  The programmed speed is higher than the configured maximum speed of the numerically largest gear stage:
  S... > MD35110 $MA_GEAR_STEP_MAX_VELO[<n>]
  In this case, the **highest gear stage** is selected (according to MD35090 $MA_NUM_GEAR_STEPS).
- **Programmed speed too low**
  The programmed spindle speed is less than the configured minimum speed of the first gear stage:
  S... < MD35120 $MA_GEAR_STEP_MIN_VELO[1]
  In this case, the **first gear stage** is selected.
- **Programmed speed = 0**
  When programming speed 0 (`S0`) the behavior depends on the configuration of the minimum speed of the first gear stage MD35120 $MA_GEAR_STEP_MIN_VELO[1]:
  - If MD35120 $MA_GEAR_STEP_MIN_VELO[1] **= 0** is configured, then when programming `S0`, the **first gear stage** is selected.
  - If MD35120 $MA_GEAR_STEP_MIN_VELO[1] **> 0** is configured, when programming `S0` **no gear stage change** is performed and the last gear stage remains active. This means that it remains possible to stop the spindle with `S0` (instead of `M5`) without initiating a gear stage change.

Effectiveness

Selecting the highest gear stage or the first gear stage for automatic gear stage selection (M40) is active when programming spindle speeds `S...` using the part program, in synchronized actions or when entering via PLC FC18.

For speed programming from the part program for tapping with G331, the behavior is also supported for the second data set to select the gear stage (precondition: MD35010 $MA_GEAR_STEP_CHANGE_ENABLE, bit 5 = 1).

Supplementary conditions

Enabling the gear stage change

The precondition for the function is that the gear stage change is generally enabled via machine data:

MD35010 $MA_GEAR_STEP_CHANGE_ENABLE (assign parameters to the gear stage change).

MD35090 $MA_NUM_GEAR_STEPS (number of gear stages set up)

Example

Automatic gear stage selection M40 is the basic setting after NC reset.

Part program:

Program code	Comment
...	
N15 S3500 M3	; S3500 is greater than MD35110 of the 2nd gear stage. The 2nd gear stage is selected.
...	
N50 S0 M3	; Spindle is stopped, S0 does not request a gear stage change (special handling, S0).
...	
N95 S5 M3	; S5 is less than MD35120 of the 1st gear stage. The 1st gear stage is selected.
...	

Configuring data for spindle 1 (AX5):

MD35010 $MA_GEAR_STEP_CHANGE_ENABLE[AX5] = 1	; Enable gear stage change
MD35090 $MA_NUM_GEAR_STEPS[AX5] = 2	; Number of existing gear stages
MD35110 $MA_GEAR_STEP_MAX_VELO[1,AX5] = 500	; Upper switching threshold for gear stage 1
MD35120 $MA_GEAR_STEP_MIN_VELO[1,AX5] = 10	; Lower switching threshold for gear stage 1
MD35110 $MA_GEAR_STEP_MAX_VELO[2,AX5] = 2000	; Upper switching threshold for gear stage 2
MD35120 $MA_GEAR_STEP_MIN_VELO[2,AX5] = 500	; Lower switching threshold for gear stage 2

18.9 Spindle with SMI 24 (Weiss spindle)

18.9.1 General Information

In order to be able to process the sensor data of the spindle in the control, the sensors must first be connected to I/O modules and transferred to the PLC via fieldbus (PROFIBUS DP or PROFINET I/O).

For a spindle with SMI 24 (Weiss spindle), the sensor data are transferred to the drive using DRIVE-CLiQ and are available there in drive parameters. When using cyclic drive telegram 139, sensor data from the drive are transferred to the control. They are then available there in the following system data:

- System variable
- OPI variables
- NC/PLC interface signals

Requirement

- The spindle is connected to the drive via Sensor Module SMI 24 using DRIVE-CLiQ.
- Drive telegram 139 is configured for the spindle.

Note

Drive telegram 139

In principle, a spindle with Sensor Module SMI 24 can also be operated with another drive telegram. However, sensor data is only transferred to the control using drive telegram 139.

18.9.2 Sensor data

Sensors in the spindle motor

The spindle sensors provide information about the clamping device and the angular position of the motor shaft:

- Analog sensor S1: Clamped state
  Voltage value in the range from 0 - 10 V depending on the position of the draw bar.
- Digital sensor S4: Piston end position
  - 0 = piston not in position
  - 1 = piston is in position, i.e. piston is free to move
- Digital sensor S5: Angular position of the motor shaft
  - 0 = motor shaft not aligned
  - 1 = motor shaft is in position (requirement: The spindle is stationary)

Transmission of sensor data

Sensor data is transferred to the control from sensor module SMI 24 in cyclic drive telegram 139 as process data 11 - 14. Drive telegram 139 is based on drive telegram 136, where sensor data is transferred instead of the data of the 2nd encoder. A detailed description of drive telegram 139 can be found in:

Further information
List Manual SINAMICS S120/S150; Function diagrams - PROFIdrive

System data: Sensor data

Sensor data can be read into the control via the following system data:

Meaning	System variable $VA_	NC/PLC interface DB31, ...	OPI variables	Drive parameters
Sensor configuration	MOT_SENSOR_CONF[<axis>]	DBB132, DBB133	vaMotSensorConf	r5000
Clamped state [1]	MOT_CLAMPING_STATE[<axis>]	DBW134	vaMotClampingState	r5001
Measured value analog sensor S1 [1] [2]	MOT_SENSOR_ANA[<axis>]	DBW136	vaMotSensorAna	r5002
Status digital sensors	MOT_SENSOR_DIGI[<axis>]	DBB138, DBB139	vaMotSensorDigi	r5003
<Axis>: Machine axis name: AX1 ... AXn or spindle name: S1 ... Sm				

1) See Section "Clamping status (Page 840)"

2) Sensor S1: 0 - 10 V
Analog actual value: 10000 increments, resolution 1 mV
Example:
SIMATIC S7 input module: 27648 increments, resolution 0.36 mV
Adaptation factor if you change to a spindle with SMI 24: (10000 incr. * 1 mV) / (27648 incr. * 0.36 mV) = 1.00469393

Detailed system data description

NC/PLC interface signals	NC Variables and Interface Signals List Manual
System variable:	List Manual, System Variables
OPI variables:	List Manual 2; Variables
Drive parameter:	SINAMICS S120/S150 List Manual

18.9.3 Clamped state

Sensor S1 supplies an analog voltage value 0 V - 10 V depending on the position of the clamping device. The voltage value is available in the system data for evaluation of the clamped state on the user side.

Note

The subsequently described evaluation of sensor S1 to generate the state value for the clamped state and limiting the spindle speed are only realized if the following state values are displayed in drive parameter r5000:

- r5000.0 == 1: Sensors available
- r5000.1 == 1: Sensor S1 (clamped state) available
- r5000.10 == 1: State values are generated, speed limits p5043 active

See also Section "Sensor data (Page 838)", paragraph: "System data: sensor data"

State value

To simplify the evaluation, the clamped state in the system data is also available as state value 0 - 11.

A voltage range corresponds to a certain clamped state. The voltage ranges can be set using drive parameter p5041[0...5].

A voltage tolerance can also be set for the voltage ranges using drive parameter p5040.

Note

The voltage range ± voltage tolerance must not overlap.

Speed limits

For the clamped states with state values 3 - 10, speed limit values can be specified using drive parameter p5043[0...6]. In the other clamped states (state values 1, 2 and 11), a limit value of 0 [rpm] permanently applies.

In the various clamped states, the controller limits the spindle speed to the applicable limit.

Note

Changing the speed limits

A change of the speed limits in drive parameter p5043[0...6] is only effective in the controller (limitation of the spindle speed to the new speed limit) after:

- Power-on reset or when the controller is switched-off/switched-on
- Deselecting the "Park" status for the spindle (see Function Manual "Monitoring and Compensating", Section "Parking a machine axis")

Context: State value, voltage range and speed limit

State value [1]	Clamped state	Voltage range [2]		Speed limit
		Upper limit	Lower limit	
0	Sensor S1 not available or state values inactive	---	---	---
1	State initialization running	---	---	[3]
2	Released with signal (error state)	---	p5041[0] + p5040	[3]
3	Released	p5041[0]	p5041[1]	p5043[0]
4	Clamping with tool	---	---	p5043[1]
5	Releasing with tool	---	---	p5043[2]
6	Releasing without tool	---	---	p5043[3]
7	Clamped with tool AND S4 == 0	p5041[2]	p5041[3]	p5043[4]
8	Clamped with tool AND S4 == 1			p5043[4]
9	Clamping without tool	---	---	p5043[5]
10	Clamped without tool	p5041[4]	p5041[5]	p5043[6]
11	Clamped with signal (error state)	p5041[5] - p5040	---	[3]

[1] The state value can be read into the controller using the following system data:
- System variable: $VA_MOT_CLAMPING_STATE[<axis>]
- NC/PLC interface: DB31, ... DBW134
- OPI variables: vaMotClampingState
- Drive parameters: r5001

[2] p5041[0...5]: Voltage threshold values, p5040: Voltage threshold values tolerance

[3] Speed limit permanently set: 0 [rpm]

18.9.4 Additional drive parameters

P5042:Transition time

The following times can be set in drive parameter p5042 for the clamped state identification:

- p5042[0]: Stabilization time for "clamped with tool"
  The clamped state "clamped with tool" must be present in the spindle motor for at least the set stabilization time before the state is signaled to the controller.
- p5042[1]: Maximum time for clamping
  The transition from the "released" state to the "clamped with tool" or "clamped without tool" state may take – as a maximum – the set time.

r5044: Speed limitation from the clamping cycle

The speed limit from p5043[6] which is active in the clamped state "clamped without tool" is displayed in drive parameter r5044.

A value of 65535 means that the speed limit is not active.

18.10 Boundary conditions

18.10.1 Changing control parameters

For spindles that are not in position-controlled mode, machine data changes also take effect when the spindle is not stationary with the NEWCONF command.

In the case of changes to control parameters, speed setpoint jumps may occur when the new values take effect. Control parameters are, for example:

- MD32200 $MA_POSCTRL_GAIN (servo gain factor)
- MD32210 $MA_POSCTRL_INTEGR_TIME (position controller integral time)
- MD32410 $MA_AX_JERK_TIME (time constant for the axial jerk filter)

18.11 Examples

18.11.1 Automatic gear step selection (M40)

Example

To illustrate the contents of the new block search variables:
Assumptions about automatic gear step selection (M40):

S0...500	1. Gear step
S501..1000	2. Gear step
S1001..2000	3. Gear step

Content of system variables:

$P_SEARCH_S	; Collected S value
$P_SEARCH_DIR	; Collected direction of rotation
$P_SEARCH_GEAR	; Collected gear step

Collected	S value:	Direction of rotation:	Gear step:
	; 0/last speed	-5	40/last GS
N05 G94 M40 M3 S1000	; 1000	3	40
N10 G96 S222	: 222	3	40
N20 G97	; f (PlanAxPosPCS)*	3	40
N30 S1500	; 1500	3	40
N40 SPOS=0**	; 1500	-19	40

N50 M19**	; 1500	-19	40
N60 G94 G331 Z10 S300	; 300	-19	40
N70 M42	; 300	-19	42
N80 M4	; 300	4	42
N90 M70	; 300	70	42
N100 M3 M40	; 300	3	40
N999 M30			

* f (PlanAxPosPCS): The speed depends on the current position of the transverse axis in the workpiece coordinate system.

** ($P_SEARCH_SPOS and $P_SEARCH_SPOSMODE are programmed)

18.12 Data lists

18.12.1 Machine data

18.12.1.1 NC-specific machine data

Number	Identifier: $MN_	Description
10192	GEAR_CHANGE_WAIT_TIME	Wait time for acknowledgment of a gear stage change during reorganization
10714	M_NO_FCT_EOP	M function for spindle active after `RESET`
12060	OVR_SPIND_IS_GRAY_CODE	Spindle override Gray-coded
12070	OVR_FACTOR_SPIND_SPEED	Evaluation of the spindle override switch
12080	OVR_REFERENCE_IS_PROG_FEED	Override reference velocity
12082	OVR_REFERENCE_IS_MIN_FEED	Defines the reference of the path override
12090	OVR_FUNCTION_MASK	Selection of override specifications

18.12.1.2 Channelspecific machine data

Number	Identifier: $MC_	Description
20090	SPIND_DEF_MASTER_SPIND	Initial setting for master spindle on channel
20092	SPIND_ASSIGN_TAB_ENABLE	Enabling/disabling of spindle converter
20850	SPOS_TO_VDI	Output of M19 to the PLC with SPOS/SPOSA
22400	S_VALUES_ACTIVE_AFTER_RESET	S function active after RESET

18.12.1.3 Axis/spindlespecific machine data

Number	Identifier: $MA_	Description
30300	IS_ROT_AX	Rotary axis
30310	ROT_IS_MODULO	Modulo conversion
31044	ENC_IS_DIRECT2	Encoder on intermediate gear
31050	DRIVE_AX_RATIO_DENOM	Load gear denominator
31060	DRIVE_AX_RATIO_NUMERA	Load gear numerator
31064	DRIVE_AX_RATIO2_DENOM	Intermediate gear denominator
31066	DRIVE_AX_RATIO2_NUMERA	Intermediate gear numerator
31070	DRIVE_ENC_RATIO_DENOM	Measuring gear denominator
31080	DRIVE_ENC_RATIO_NUMERA	Measuring gear numerator
31122	BERO_DELAY_TIME_PLUS	BERO delay time in plus direction
31123	BERO_DELAY_TIME_MINUS	BERO delay time in minus direction
32200	POSCTRL_GAIN	K_V factor
32800	EQUIV_CURRCTRL_TIME	Equivalent time constant of the current control loop for feedforward control
32810	EQUIV_SPEEDCTRL_TIME	Equivalent time constant of the speed control loop for feedforward control
32910	DYN_MATCH_TIME	Time constant for dynamic response adaptation
34040	REFP_VELO_SEARCH_MARKER	Reference point creep velocity
34060	REFP_MAX_MARKER_DIST	Monitoring of zero mark distance
34080	REFP_MOVE_DIST	Reference point distance / destination point for distance-coded system
34090	REFP_MOVE_DIST_CORR	Reference point offset / absolute offset, distance-coded
34100	REFP_SET_POS	Reference point value
34200	ENC_REFP_MODE	Referencing mode
35000	SPIND_ASSIGN_TO_MACHAX	Assignment of spindle to machine axis
35010	GEAR_STEP_CHANGE_ENABLE	Type of gear stage change
35012	GEAR_STEP_CHANGE_POSITION	Gear stage change position
35014	GEAR_STEP_USED_IN_AXISMODE	Gear stage for axis mode with M70
35020	SPIND_DEFAULT_MODE	Basic spindle setting
35030	SPIND_DEFAULT_ACT_MASK	Activate basic spindle setting
35035	SPIND_FUNCTION_MASK	Setting of spindle-specific functions
35040	SPIND_ACTIVE_AFTER_RESET	Spindle active after reset
35090	NUM_GEAR_STEPS	Number of set gear stages
35092	NUM_GEAR_STEPS2	2nd gear stage data set: Number of set gear stages
35100	SPIND_VELO_LIMIT	Maximum spindle speed
35110	GEAR_STEP_MAX_VELO[n]	Maximum speed for automatic gear stage change
35112	GEAR_STEP_MAX_VELO2[n]	2nd gear stage data set: Maximum speed for automatic gear stage change
35120	GEAR_STEP_MIN_VELO[n]	Minimum speed for automatic gear stage change

Number	Identifier: $MA_	Description
35122	GEAR_STEP_MIN_VELO2[n]	2nd gear stage data set: Minimum speed for automatic gear stage change
35130	GEAR_STEP_MAX_VELO_LIMIT[n]	Maximum speed of gear stage
35135	GEAR_STEP_PC_MAX_VELO_LIMIT[n]	Maximum speed of gear stage in position control
35140	GEAR_STEP_MIN_VELO_LIMIT[n]	Minimum speed of gear stage
35150	SPIND_DES_VELO_TOL	Spindle speed tolerance
35160	SPIND_EXTERN_VELO_LIMIT	Spindle speed limitation from PLC
35200	GEAR_STEP_SPEEDCTRL_ACCEL[n]	Acceleration in speed control mode
35210	GEAR_STEP_POSCTRL_ACCEL[n]	Acceleration in position control mode
35212	GEAR_STEP_POSCTRL_ACCEL2[n]	2nd gear stage data set: Acceleration in position control mode
35220	ACCEL_REDUCTION_SPEED_POINT	Speed limit for reduced acceleration
35230	ACCEL_REDUCTION_FACTOR	Reduced acceleration
35300	SPIND_POSCTRL_VELO	Position control activation speed
35350	SPIND_POSITIONING_DIR	Positioning direction of rotation for a non-synchronized spindle
35400	SPIND_OSCILL_DES_VELO	Oscillation speed
35410	SPIND_OSCILL_ACCEL	Oscillation acceleration
35430	SPIND_OSCILL_START_DIR	Oscillation start direction
35440	SPIND_OSCILL_TIME_CW	Oscillation time for M3 direction
35450	SPIND_OSCILL_TIME_CCW	Oscillation time for M4 direction
35500	SPIND_ON_SPEED_AT_IPO_START	Feed enable with spindle in setpoint range
35510	SPIND_STOPPED_AT_IPO_START	Feed enable with stationary spindle
35550	DRILL_VELO_LIMIT[n]	Maximum speeds for tapping
35590	PARAMSET_CHANGE_ENABLE	Parameter set specification possible from PLC
36060	STANDSTILL_VELO_TOL	Threshold velocity "Axis/spindle stationary"
36200	AX_VELO_LIMIT	Threshold value for velocity monitoring.

18.12.2 Setting data

18.12.2.1 Channelspecific setting data

Number	Identifier: $SC_	Description
42600	JOG_FEED_PER_REF_SOURCE	Revolutional feedrate control in JOG mode
42800	SPIND_ASSIGN_TAB	Spindle number converter
42900	MIRROR_TOOL_LENGTH	Mirror tool length offset
42910	MIRROR_TOOL_WEAR	Mirror wear values of tool length compensation
42920	WEAR_SIGN_CUTPOS	Mirror wear values of machining plane

Number	Identifier: $SC_	Description
42930	WEAR_SIGN	Invert sign of all wear values
42940	TOOL_LENGTH_CONST	Retain the assignment of tool length components when changing the machining plane (G17 to G19)

18.12.2.2 Axis/spindle-specific setting data

Number	Identifier: $SA_	Description
43200	SPIND_S	Speed for spindle start via PLC interface
43202	SPIND_CONSTCUT_S	Cutting rate for spindle start via PLC interface
43206	SPIND_SPEED_TYPE	For spindle start via PLC interface
43210	SPIND_MIN_VELO_G25	Progr. Spindle speed limiting G25
43220	SPIND_MAX_VELO_G26	Progr. Spindle speed limiting G26
43230	SPIND_MAX_VELO_LIMS	Progr. spindle speed limiting G96/G961
43235	SPIND_USER_VELO_LIMIT	Maximum spindle speed
43240	M19_SPOS	Spindle position for spindle positioning with M19
43250	M19_SPOSMODE	Spindle positioning approach mode for spindle positioning with M19
43300	ASSIGN_FEED_PER_REF_SOURCE	Rotational feedrate for positioning axes/spindles

S3: Synchronous spindle

19

19.1 Brief description

19.1.1 Function

The "Synchronous spindle" function can be used to couple two spindles with synchronous position or speed. One spindle is defined as leading spindle (LS), the second spindle is then the following spindle (FS).

Speed synchronism:	$n_{FS} = k_{Ü} * n_{LS}$	with $k_{Ü} = 1, 2, 3, ...$
Position synchronism:	$\varphi_{FS} = \varphi_{LS} + \Delta\varphi$	with $0° \leq \Delta\varphi < 360°$

Possible applications

Rear side machining

One application option is, for example, the reverse side machining in a double-spindle lathe with on-the-fly transfer of the workpiece from the position-synchronous LS to the FS (or vice versa), without having to decelerate down to standstill.

Multi-edge machining (polygonal turning)

The "Synchronous spindle" function provides the basis for multi-edge machining (polygonal turning) through specification of an integer gear ratio $k_{Ü}$ between LS and FS.

Number of FS

The number of FS's that can be operated synchronously to an LS is only restricted by the performance capability of the NC used. In principle, any number of FS can be coupled simultaneously to an LS in arbitrary channels of the NC.

2 pairs of synchronous spindles can be active simultaneously in each NC channel.

Definition

The assignment of FS to LS pair of synchronous spindles can be parameterized channel-specifically via machine data or flexibly defined via part program commands.

Selecting/de-selecting

Part program commands are used to select/deselect the synchronous operation of a pair of synchronous spindles.

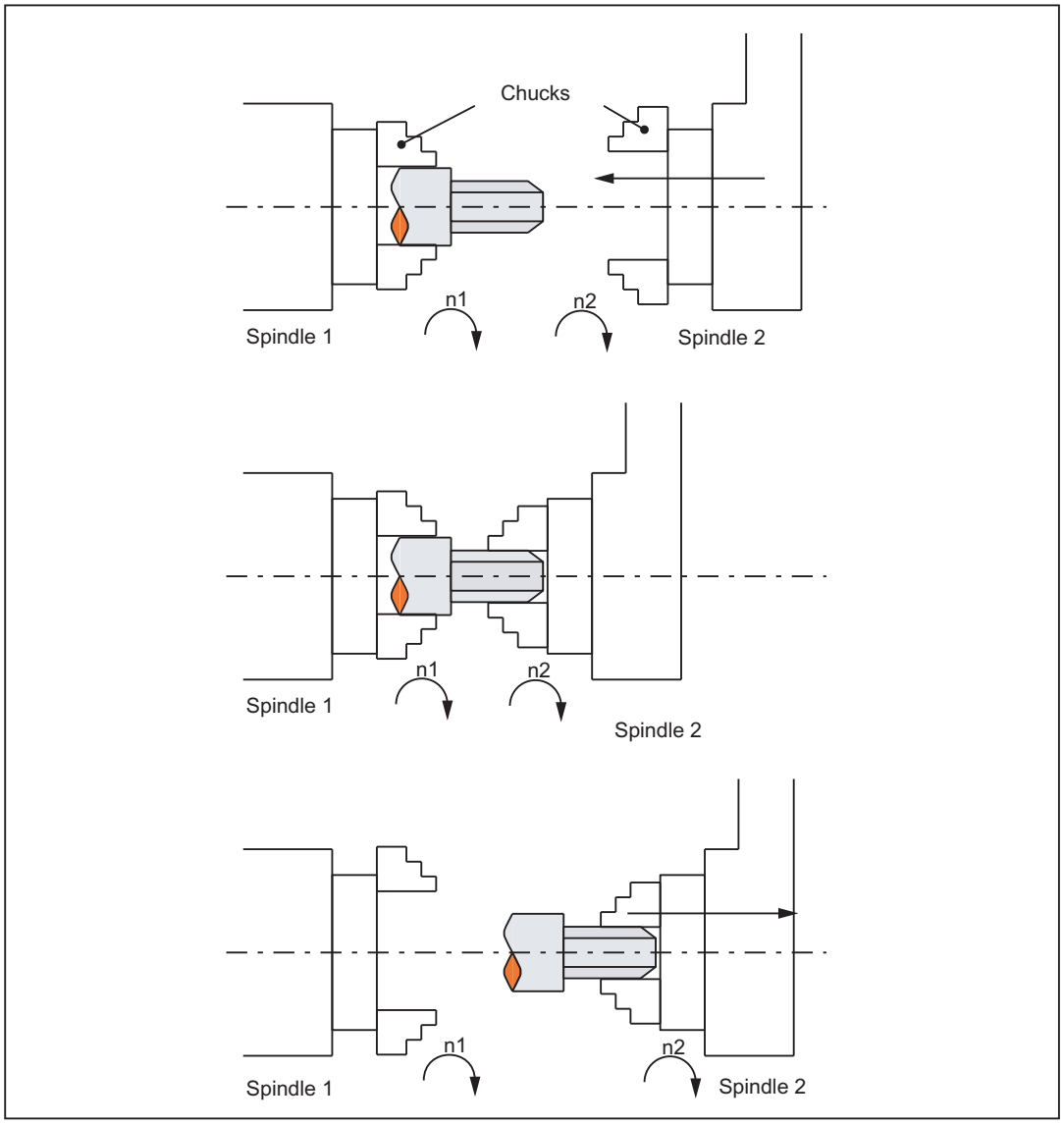


Figure 19-1 Synchronous operation: On-the-fly workpiece transfer from spindle 1 to spindle 2

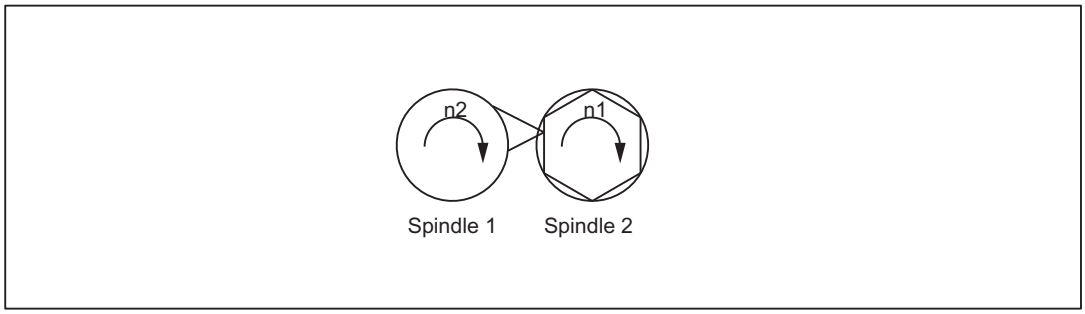


Figure 19-2 Synchronous operation: Polygonal turning

19.1.2 Synchronous mode

Description

<axial expression>:	can be: - Axis name - Spindle name
<Axis name>:	C (if spindle has the name "C" in axis operation.)
<spindle name>:	Sn, SPI(n) where n = spindle number
<Spindle number>:	1, 2, ... according to the spindle number defined in MD35000 $MA_SPIND_ASSIGN_TO_MACHAX
(FS, LS, offset):	LS = Leading Spindle, FS = Following Spindle, Offset = read programmable offset of following spindle using system variables
$P_COUP_OFFS[Sn]	Programmed position offset of the synchronous spindle

Synchronous spindle pair

Synchronous operation involves a following spindle (FS) and a leading spindle (LS), referred to as the **synchronous spindle pair**. The following spindle imitates the movements of the leading spindle when a coupling is active (synchronous operation) in accordance with the defined functional interrelationship.

Synchronous mode

Synchronous mode (also referred to as “Synchronous spindle operation”) is another spindle operating mode. Before synchronous mode is activated, the following (slave) spindle must have been switched to position control. Synchronous operation is activated for the following spindle when the coupling is activated. As soon as the coupling is deactivated, the following spindle switches back to open-loop control mode.

As soon as synchronous operation is active for the following spindle, the following interface signal is reported to the PLC:

IS "Synchronous mode" (DB31, ... DBX84.4) = 1.

Number of synchronous spindles

It is possible to couple several following spindles to one leading spindle. The number of following spindles on this leading spindle depends on the respective version of the appropriate software versions.

Any number of following spindles in any channels of one NCU or a different NCU can be coupled to this leading spindle.

Note that one spindle is always the master and the number of couplings results from the number of axes less the master.

Options in synchronous mode

The following functions are available for synchronous mode:

- FS and LS turn at the same speed
  ($n_{FS} = n_{LS}$; transformation ratio $k_Ü = 1$)
- Rotation in the same or opposite direction between LS and FS
  (can be defined positively or negatively using transformation ratio $k_Ü$)
- Following and leading spindles rotate at different speeds
  ($n_{FS} = k_Ü \cdot n_{LS}$; transformation ratio $k_Ü \neq 1$)
  Application: Polygonal turning
- Settable angular position between FS and LS ($\varphi_{FS} = \varphi_{LS} + \Delta\varphi$)
  The spindles run at synchronous speed with a defined angular offset between FS and LS (position synchronous coupling).
  Application: Shaped workpieces
- Activation of synchronous operation between LS and FS can take place when the spindles are in motion or at standstill.
- The full functionality of the open-loop and position control modes is available for the leading spindle.
- When synchronous mode is not active, the FS and LS can be operated in all other spindle modes.
- The transformation ratio can also be altered when the spindles are in motion in active synchronous mode.
- With synchronous spindle coupling switched on, the offset of the FS to the LS (overlaid movement) can be altered.

Coupling options

Synchronous spindle couplings can be defined as

- Permanently configured via channel-specific machine data
  (hereinafter referred to as **"permanently configured coupling"**) as well as
- Freely defined using language commands (COUP...) in the part program
  (hereafter referred to as **"user defined coupling"**)

. The following variants are possible:

1. A fixed configuration for a coupling can be programmed via machine data. In addition, a second coupling can be freely defined via the part program.
2. No coupling is configured via machine data. In this case, the couplings can be user-defined and parameterized via the part program.

Separate following spindle interpolator

The separate **following spindle interpolator** allows a number of following spindles from different channels or from another NCU to be coupled as defined by the user to a single leading spindle. The following spindle interpolator is

- COUPON or COUPONC activated and
- COUPOF or COUPOFS deactivated

and is always located in the channel in which the COUPON, COUPONC command has been programmed for the following spindle. If the following spindle to be activated was previously programmed in another channel, COUPON/COUPONC initiates an axis replacement and fetches the spindle into its own channel.

Certain synchronous spindle functions can be controlled from the PLC by means of coupling-specific axial VDI interface signals. These signals act exclusively on the slave spindles and do not affect the leading spindle (see Section "Controlling synchronous spindle coupling via PLC (Page 859)").

Definition of synchronous spindles

Before synchronous operation is activated, the spindles to be coupled (FS, LS) must be defined.

This can be done in two ways depending on the application in question:

1. Permanently configured coupling:
   Machine axes that are to function as the following spindle (FS) and leading spindle (LS) are defined in channel-specific MD21300 $MC_COUPLE_AXIS_1[n].
   The machine axes programmed as the LS and FS for this coupling configuration cannot be altered by the NC part program.
   If necessary, the coupling parameters can be modified with the NC part program.
2. User-defined coupling:
   Couplings can be created and altered in the NC part program with language command "COUPDEF(FS, LS, ...)". If a new coupling relationship is to be defined, it may be necessary to delete an existing user-defined coupling beforehand (with language command COUPDEL(FS, LS)).
   The axis names (Sn, SPI(n)) for the following and leading spindles must be programmed with FS and LS for every language command COUP..., thus ensuring that the synchronous spindle coupling is unambiguously defined.
   The valid spindle number must then be assigned axis-specific machine data of a machine axis:
   MD35000 $MA_SPIND_ASSIGN_TO_MACHAX.

IS "Following spindle active" (DB31, ... DBX99.1) and IS "Leading spindle active" (DB31, ... DBX99.0) indicate to the PLC for each machine axis whether the axis is active as a leading or following spindle.

The LS can be programmed either via a part program, PLC or also using synchronized actions.

Transformation ratio

The transformation ratio is programmed with separate numerical values for numerator and denominator (transformation ratio parameters). It is therefore possible to specify the transformation ratio very exactly, even with rational numbers.

In general:

$k_Ü$ = transformation ratio parameter for numerator Transformation ratio parameters for denominator = $Ü_{numerator} : Ü_{denominator}$

The value range of the transformation ratio parameter ($Ü_{numerator}$, $Ü_{denominator}$) is virtually unlimited in the control.

The transformation ratio parameters for the coupling configured via machine data can be defined in channel-specific SD42300: COUPLE_RATIO_1[n]. In addition, the transformation ratio can be altered with language command `COUPDEF`(FS, LS, $Ü_{numerator}$, $Ü_{denominator}$,...). The values entered in the setting data is not overwritten in this case (default settings).

The transformation ratio for the coupling defined via the NC part program can only be input with language command `COUPDEF` (...).

The new transformation ratio parameters take effect as soon as the `COUPDEF` command has been processed.

For further programming commands for synchronous spindle couplings, please see "Programming of synchronous spindle couplings" Section .

Coupling characteristics

The following characteristics can be defined for every synchronous spindle coupling:

- **Block change behavior**
  The condition to be fulfilled for a block change can be defined on activation of synchronous operation or on alteration of the transformation ratio or the defined angular offset when the coupling is active:
  - Block change takes place immediately
  - Block change in response to "Fine synchronism"
  - Block change in response to "Coarse synchronism"
  - Block change for IPOSTOP (i.e. after setpoint-end synchronism)
  - Check of the synchronism conditions at an arbitrary moment with `WAITC`.
- **Type of coupling** between FS and LS
  The position setpoint or the actual position value of the leading spindle can be used as the reference value for the following spindle. The following coupling types can therefore be selected:
  - Setpoint coupling (DV)
    Use in position controlled operation. The control dynamic response of both spindles should coincide as far as possible. Preferably, the setpoint coupling should be used.
  - Actual value coupling (AV)
    Application if no position control of the LS is possible or with great deviation of the control characteristics between FS and LS. The setpoints for the FS are derived from the actual values of the LS. The quality of synchronism is worse with a varying spindle speed than with the setpoint coupling.
  - Speed coupling (VV)
    Internally, the velocity coupling is a setpoint coupling. The requirements for FS and LS are lower. Position control and measuring systems are not required for FS and LS. The position offset between FS and LS is undefined.

The selection of the relevant coupling characteristics for the **configured coupling** is made using machine data (see Section "Configuration (Page 871)") and for the **user defined coupling** using the `COUPDEF` language command (see Section "Definition (COUPDEF) (Page 865)").

In addition, coupling characteristics Type of coupling and Block change response can be altered for the permanently configured coupling by means of language command `COUPDEF`.

Further information

Programming Manual Advanced Synchronous spindle

Change protection for coupling characteristics

The channel-specific MD21340 $MC_COUPLE_IS_WRITE_PROT_1 is used to define whether or not the configured coupling parameters Transformation ratio, Type of coupling and Block change response can be altered by the NC part program:

0: Coupling parameters can be altered by the NC part program via command `COUPDEF`

1: Coupling parameters cannot be altered by the NC part program. Attempts to make changes will be rejected with an alarm message.

Superimposed motion

In synchronous operation, the synchronous spindle copies the movement of the leading spindle in accordance with the programmed transformation ratio.

At the same time, the synchronous spindle can also be traversed with overlay so that the LS and FS can operate at a specific angular position in relation to one another.

The overlaid traversing movement of the FS can be initiated in various ways:

- Programmable position offset of FS for AUTOMATIC and MDA:
  - The `COUPON` and `SPOS` language commands can be used for active synchronous operation to change the position reference between FS and LS (see Section "Selecting synchronous mode for a part program (Page 856)").
- Manual position offset of FS:
  - In JOG (JOG continuous or JOG incremental) mode
    Superimposition of FS using the handwheel or with plus or minus traversing keys when synchronous operation is active.
  - In AUTOMATIC and MDA modes
    Superimposition of FS with handwheel using DRF offset

As soon as the FS executes the overlaid traversing movement, IS "Overlaid movement" (DB31, ... DBX98.4) is set to the 1 signal.

The overlaid movement is executed optimally in terms of time at the maximum possible FS speed with `COUPON`. With an offset change by means of SPOS, the positioning velocity can be specified with FA[Sn] and manipulated by an override (can be selected through IS "Feedrate override valid for spindle" DB31, ... DBX17.0).

Note

For more information about specifying the position speed with FA[Sn]:

Further information

Positioning mode (Page 739)

Setpoint correction

The setpoint correction of the system variable $AA_COUP_CORR[Sn] impacts on all subsequent following spindle programming in the same way as a position offset and corresponds to a DRF offset in the MCS.

Example: establish correction value

If a coupling offset of 7° has been programmed using COUPON(....,77) and if a mechanical offset of 81° has come about as a result of closing the workpiece support fixture, a correction value of 4° is calculated:

The system variables return the following values for the following spindle:

$P_COUP_OFFS[S2] ; programmed position offset = 77°

$AA_COUP_OFFS[S2] ; setpoint position offset = 77°

$VA_COUP_OFFS[S2] ; actual value position offset approx. 77°

$AA_COUP_CORR[S2] ; correction value = 4°

19.1.3 Prerequisites for synchronous mode

Conditions on selection of synchronous mode

The following conditions must be fulfilled before the synchronous spindle coupling is activated or else alarm messages will be generated.

- The synchronous spindle coupling must have been defined beforehand (either permanently configured via machine data or according to user definition via part program using `COUPDEF`).
- The spindles to be coupled must be defined in the NC channel in which the coupling is activated.
  Channel-spec. MD20070 $MC_AXCONF_MACHAX_USED
  Axis-spec. MD35000 $MA_SPIND_ASSIGN_TO_MACHAX
- The following spindle must be assigned to the NC channel in which the coupling is activated.
  Default setting with axis-specific MD30550 AXCONF_ASSIGN_MASTER_CHAN
- The following applies to setpoint and actual value couplings (DV, AV):
  FS and LS must at least have a position measuring system for recording positions and position controls must be started up.

Note

When position control is activated, the maximum setpoint speed of the LS is automatically limited to 90% (control reserve) of the maximum speed. The limitation is signaled via IS "Setpoint speed limited" (DB31, ... DBX83.1).

Further information

S1: Spindles (Page 733)

- The following applies to setpoint couplings (DV):
  To ensure more accurate synchronization characteristics, the LS should be in position control mode (language instruction `SPCON`) before the coupling is activated.
- Before selecting the synchronous mode, the gear stage necessary for FS and LS must be selected. In synchronous mode, gear stage changeover and therefore oscillation mode are not possible for FS and LS. Upon request, an alarm message is generated.

Cross-channel coupling

The LS can be located in any channel.

- The LS can be exchanged between channels by means of "Axis exchange".
- When several following spindles are coupled to one leading spindle, the dynamic response of the coupling is determined by the weakest response as a function of the coupling factor. The acceleration rate and maximum speed are reduced for the leading spindle to such a degree that none of the coupled following spindles can be overloaded.
- The following spindle is always located in the channel in which the coupling has been activated using `COUPON` or `COUPONC`.

See also

R1: Referencing (Page 291)

19.1.4 Selecting synchronous mode for a part program

Activate coupling COUPON, COUPONC

Language command `COUPON` activates the coupling in the part program between the programmed spindles with the last valid parameters and thus also activates synchronous mode. This coupling may be a fixed configuration or user-defined. The leading spindle and/or following spindle may be at standstill or in motion at the instant of activation.

Certain preconditions must be fulfilled before synchronous operation can be activated (see Section "Prerequisites for synchronous mode (Page 855)").

The `COUPONC` command adopts the previous programmed direction of spindle rotation and spindle speed for the following and leading spindle in the part program. It is not possible to specify an angular offset.

COUPON activation variants

Two different methods can be selected to activate synchronous mode:

1. Fastest possible activation of coupling with **any angular reference** between leading and following spindles.
   COUPON(FS, LS)
2. Activation of coupling with a **defined angular offset** POS_{FS} between leading and following spindle. With this method, the angular offset must be programmed on selection.
   COUPON(FS, LS, POS_{FS})

Block change behavior

Before synchronous operation is selected, it must be determined under what conditions the block change must occur when the synchronous mode is activated (see Section "Definition (COUPDEF) (Page 865)").

Determine current coupling status

The $AA_COUP_ACT[<axial expression>] axial system variable can be used in the NC part program to specify the current coupling status for the specified axis/spindle (see Section "Axial system variables for synchronous spindle (Page 869)"). As soon as the synchronous spindle coupling is active for the following spindle, bit 2 must be "1" when read.

Change defined angular offset

Language commands `COUPON` and `SPOS` allow the defined angular offset to be changed while synchronous mode is active. The following spindle is positioned as an overlaid movement at the angular offset programmed with POS_{FS}. During this time, the IS "overlaid movement" (DB31, ... DBX98.4) is set.

Angular offset POS_{FS}

The defined angular offset POS_{FS} must be specified as an absolute position referred to the zero degrees position of the leading spindle in a positive direction of rotation.

The "0° position" of a position-controlled spindle is calculated as follows:

- From the zero mark or proximity sensor signal of the measurement system and
- From the reference values saved using axis-specific machine data:
  MD34100 $MA_REFP_SET_POS, reference point value,
  of no significance with interval-coded systems.
  MD34080 $MA_REFP_MOVE_DIST reference point distance/target point
  with interval-coded systems,
  MD34090 $MA_REFP_MOVE_DIST_CORR reference point offset / absolute offset with interval coding.

Range of POS_{FS}: 0 ... 359.999°.

Further information

R1: Referencing (Page 291)

Read current angular offset

Using axial system variables, it is possible to read the current position offset between the FS and LS in the NC part program. A distinction is made between:

- Current position offset of setpoint between FS and LS
  $AA_COUP_OFFS [<axis name for FS>]
- Current position offset of actual value between FS and LS
  $VA_COUP_OFFS [<axis name for FS>]

(Explanation of <axis name>, see Section "Synchronous mode (Page 849)")

Activation after power ON

Synchronous mode can also be activated with non-referenced/synchronized FS or LS (IS "Referenced/synchronized 1 or 2" DB31, ... DBX60.4 or DBX60.5 = 0). In this case, a warning message is displayed.

Example:

LS and FS are already coupled in a friction lock via a workpiece after power ON.

19.1.5 Deselecting the synchronous mode for the part program

Open coupling (COUPOF, COUPOFS)

Synchronous mode between the specified spindles is canceled by the parts program instruction COUPOF. Three variants are possible.

If synchronous mode is canceled between the specified spindles using COUPOF, then it is irrelevant whether this coupling is permanently configured or user defined. The leading and following spindles can be at standstill or in motion when synchronous operation is deactivated.

On switching off the synchronous mode with COUPOF, the following spindle is put into **control mode**. The originally programmed S-word is no longer valid for the FS, the following spindle can be operated like any other normal spindle.

When the coupling is opened with COUPOF, a block preprocessing stop STOPRE is generally initiated internally in the control.

The COUPOFS instruction can be used to open a coupling either as quickly as possible with a stop and no position data or with a stop at the programmed position.

COUPOF variants

Three different methods can be used to deselect synchronous mode with COUPOF:

1. Deactivation of coupling as quickly as possible
   The block change is enabled immediately.
   COUPOF(FS, LS)
2. A coupling is not deselected until the following spindle has crossed the programmed deactivation position POS_{FS}.
   The block change is then enabled.
   COUPOF(FS, LS, POS_{FS})
3. A coupling is not deselected until the following spindle and leading spindle has crossed the programmed deactivation positions POS_{FS} and POS_{LS}.
   The block change is then enabled.
   COUPOF(FS, LS, POS_{FS} , POS_{LS})

POS_{FS}, POS_{LS}

Deactivation positions POS_{FS} and POS_{LS} match the actual positions of FS and LS respectively referred to the defined reference point value.

Range of POS_{FS}, POS_{LS}: 0 **...** 359.999°.

Further information
R1: Referencing (Page 291)

COUPOF during the motion

If synchronous mode is deselected while the spindles are in motion with COUPOF, the following spindle continues to rotate at the current speed (n_{FS}). The current speed can be read with system variable $AA_S in the NC parts program.

The following spindle can then be stopped from the parts program with M05, SPOS, SPOSA or from the PLC with the appropriate interface signal.

COUPOFS with stop of following spindle

Opening a synchronous spindle coupling is extended by a stop of the following spindle:

- Deactivating a coupling as quickly as possible and opening a coupling as quickly as possible.
  The block change is then enabled.
  COUPOFS(FS, LS)
- Opening the coupling with stop of following spindle at the programmed position. The block change is then enabled.

Boundary condition:

COUPOFS(FS, LS) and COUPOFS(FS, LS, POS_{FS}) have no meaning if a coupling was active.

19.1.6 Controlling synchronous spindle coupling via PLC

Controlling following spindle via PLC

Using the coupling-specific, axial VDI interface signals, it is possible to control synchronization motions for the following spindle from the PLC program. This offers the option of utilizing the PLC to disable, suppress or restore a synchronization motion for the following spindle specified by offset programming.

These signals have no effect on the leading spindle. The following coupling-specific VDI signal (PLC → NC) is available:

IS **"Disable synchronization" (DB31, ... DBX31.5)**

"Disable synchronization"

The synchronization motion for the following spindle is suppressed using the axial signal IS "Disable synchronization" (DB31, ... DBX31.5).

When the main run advances to a block containing part program instruction COUPON (FS, LS, offset), the following interface signal is evaluated for the following spindle:

IS "Disable synchronization" (DB31, ... DBX31.5).

- For IS "Disable synchronization" (DB31, ... DBX31.5) = 0, the position offset is traversed through as before.
- For IS "Disable synchronization" (DB31, ... DBX31.5) = 1, only the continuous velocity synchronism is established. The following spindle does not execute any additional movement.
  The coupling then responds analogously to a programmed `COUPON(<FS>,<LS>)`.

Special features

For the IS "Disable synchronization" (DB31, ... DBX31.5) offset motion of the following spindle cannot be controlled that was generated as follows:

- SPOS, POS
- Synchronized actions
- FC18 (for 840D sl)
- JOG

These functions are controlled by VDI signal IS "Feedrate stop/Spindle stop" (DB31, ... DBX4.3).

Synchronized state reached

Whenever a state of synchronism has been reached, the following two VDI signals are set regardless of whether synchronization has been disabled or not:

IS "Synchronism coarse" (DB31, ... DBX98.1) and

IS "Synchronism fine" (DB31, ... DBX98.0)

Further block changes after COUPON are not prevented by suppression of synchronization.

Example

Block change behavior after `COUPON`

Program code	Comment
	; IS "Disable synchronization"
	; set (DB31, ... DBX31.5) = 1 for S2
N51 SPOS=10 SPOS[2]=10	; positions correspond to an offset
	; of 0°
N52 COUPDEF(S2,S1,1,1,"FINE","DV")	
N53 COUPON(S2,S1,77)	; actual offset of 0 degrees is retained
	; no following spindle movement,
	; VDI signals
	; IS "Synchronism coarse"
	; (DB31, ... DBX98.1) and
	; IS "Synchronism fine"
	; (DB31, ... DBX98.0)

Program code	Comment
	; are set and the block change
	; is enabled.
N54 M0	
N57 COUPOF(S2,S1)	
N99 M30	

Reset and recovery

Resetting the IS "Disable synchronization" (DB31, ... DBX31.5) has no effect on the following spindle offset. If the offset motion of the following spindle has been suppressed by the VDI interface signal, then the offset is not automatically applied when the VDI signal is reset.

Synchronization is recovered as follows:

- By repeating the part program operation COUPON (FS, LS, offset) with IS "Disable synchronization" (DB31, ... DBX31.5) = 0.
  COUPON (FS, LS, offset) can be written e.g. in an ASUB.
- By setting the IS "Resynchronize" (DB31, ... DBX31.4) = 1

Read offset

The following system variables can be used to read three different position offset values of the following spindle from the part program and synchronized actions. The variable $P_COUP_OFFS[Sn] is only available in the part program.

Description	NC variable
Programmed position offset of the synchronous spindle	$P_COUP_OFFS[Sn]
Position offset of synchronous spindle, setpoint end	$AA_COUP_OFFS[Sn]
Position offset of synchronous spindle, actual value end	$VA_COUP_OFFS[Sn]

"Feedrate stop/spindle stop"

By configuring bit 4 in MD30455 MISC_FUNCTION_MASK, the behavior of the axial IS **"Feedrate stop/Spindle stop" (DB31, ... DBX4.3)** is defined for the following spindle.

Bit 4 = 0 compatibility method:

Canceling feed enable for the following spindle decelerates the coupling assembly.

Bit 4 = 1:

Feedrate enable refers only to the interpolation component (SPOS,..) and does not affect the coupling.

Note

Other configuration options for axis functions using MD30455 $MA_MISC_FUNCTION_MASK:

Further information

R2: Rotary axes (Page 221)

19.1.7 Monitoring of synchronous operation

Fine/coarse synchronism

In addition to conventional spindle monitoring operations, synchronous operation between the FS and LS is also monitored in synchronous mode.

IS "Fine synchronism" (DB31, ... DBX98.0) or IS "Coarse synchronism" (DB31, ... DBX98.1) is transmitted to the PLC to indicate whether the current position (AV, DV) or actual speed (VV) of the following spindle lies within the specified tolerance window.

When the coupling is switched on, the signals "Coarse synchronism" and "Fine synchronism" are updated when setpoint synchronism is reached.

The size of the tolerance windows is set with machine data of the FS. Reaching of the synchronism is influenced by the following factors:

- AV, DV: Position variance between FS and LS
- VV: Difference in speed between FS and LS

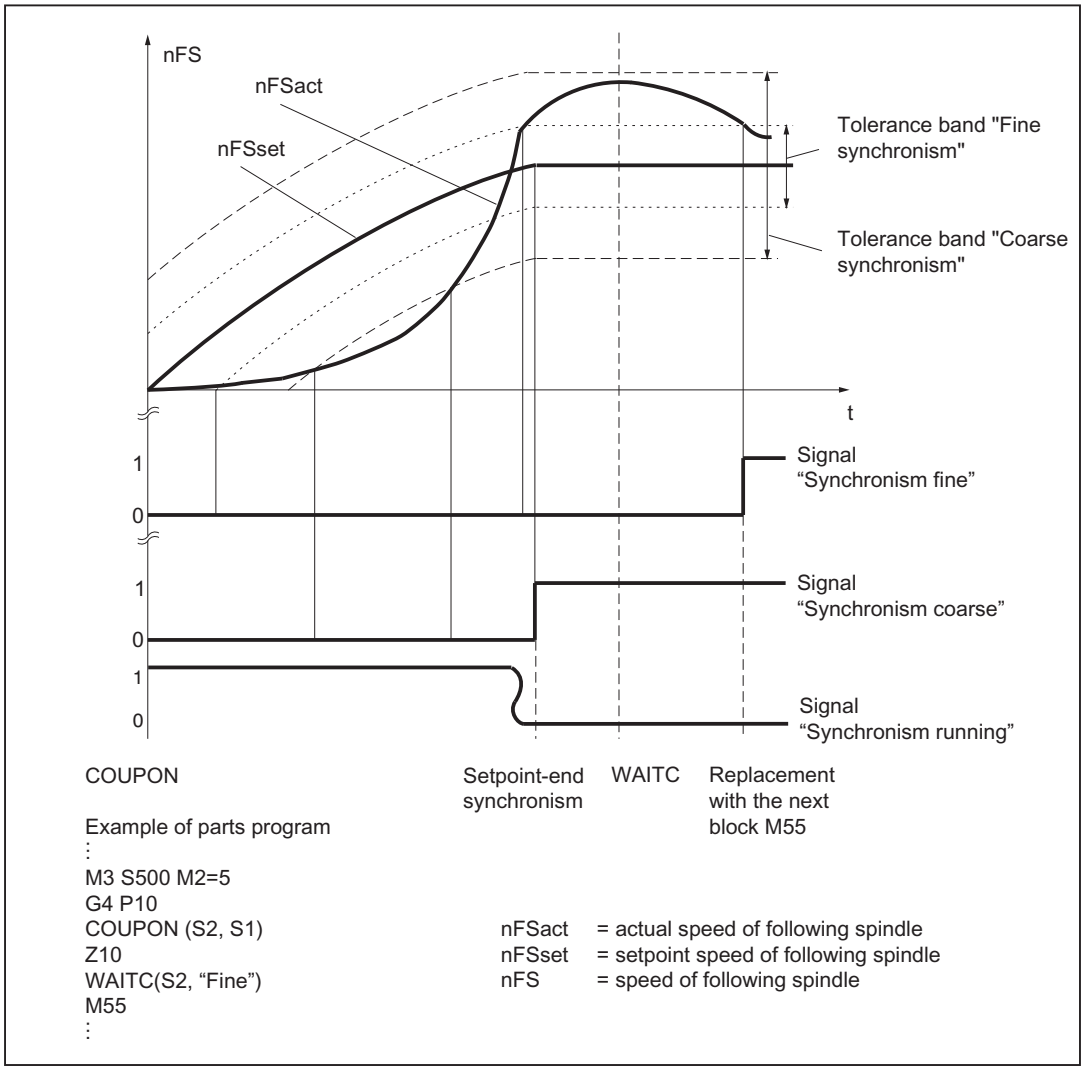


Figure 19-3 Synchronism monitoring with `COUPON` and synchronism test mark `WAITC` with synchronization on a turning leading spindle

Threshold values

The relevant position or velocity tolerance range for the following spindle in relation to the leading spindle must be specified in degrees or 1 rev/min.

- Threshold value for "Coarse synchronism"
  axis spec. MD37200: AV, DV: COUPLE_POS_TOL_COARSE
  MD37220: VV: COUPLE_VELO_TOL_COARSE
- Threshold value for "Fine synchronism"
  axis spec. MD37210: AV,DV: COUPLE_POS_TOL_FINE
  MD37230: VV: COUPLE_VELO_TOL_FINE

Speed/acceleration limits

In synchronous mode, the speed and acceleration limit values of the leading spindle are adjusted internally in the control in such a way that the following spindle can imitate its movement, allowing for the currently selected gear stage and effective speed ratio, without violating its own limit values.

For example, the LS is automatically decelerated to prevent the FS from exceeding the maximum speed in order to maintain synchronism between the spindles.

19.2 Programming

Coupling commands with leading-spindle programming

Command	Function
`COUPDEF(FS, LS, ...)`	Define coupling or change for configured coupling
`COUPON(FS, LS, POSFS)`	Switch the coupling on
`COUPONC(FS, LS)`	Activate coupling with transfer of the currently effective speed of the following spindle
`COUPOF(FS, LS, POSFS, POSLS)`	Switch the coupling off
`COUPOFS(FS, LS, POSFS)`	Deactivate coupling with stop of the following spindle
`COUPDEL (FS, LS)`	Delete coupling
`COUPRES (FS, LS)`	Reactivate configured coupling data

Coupling commands without leading-spindle programming

Command	Function
`COUPOF(FS, POSFS, POSLS)`	Switch the coupling off
`COUPOFS(FS, POSFS)`	Deactivate coupling with stop of the following spindle
`COUPDEL(FS)`	Delete coupling
`COUPRES(FS)`	Reactivate configured coupling data

See also

Definition (COUPDEF) (Page 865)

Switch the coupling (COUPON, COUPONC, COUPOF) on and off (Page 868)

19.2.1 Definition (COUPDEF)

Programmable couplings

The number of couplings can be programmed as often as desired depending on the axes available. This number results from the number of axes/spindles less one for the master. Furthermore, one coupling can also be configured via machine data as in earlier SW versions.

Permanently configured coupling

The coupling characteristics and transformation ratio for a permanently configured synchronous spindle coupling can be altered by the NC part program provided that they are not write-protected. The machine axes for LS and FS cannot be changed.

Define new couplings

Language command "COUPDEF" can be used to create new synchronous spindle couplings (user-defined) and to modify the parameters for existing couplings.

When the coupling parameters are fully specified, the following applies:

COUPDEF(FS, LS, $T_{numerator}$, $T_{denominator}$, block change behavior, coupling type)

The synchronous spindle coupling is unambiguously defined with FS and LS

The other coupling parameters must only be programmed when they need to be changed. The last valid status remains applicable for non-specified parameters.

The individual coupling parameters are explained below:

- **FS, LS**: Spindle name for following and leading spindles
  e.g.: S1, SPI(1), S2, SPI(2)
  The valid spindle number must be assigned in the axis-specific MD35000 $MA_SPIND_ASSIGN_TO_MACHAX of a machine axis.
- **$T_{numerator}$, $T_{denominator}$**: Transformation ratio parameters for numerator and denominator
  The transformation ratio is specified in the form of numeric values for numerator and denominator (see Section "Synchronous mode (Page 849)").
  The numerator must always be programmed. If no denominator is specified, then its value is always assumed to be "1.0".
- **Block change behavior**
  This parameter allows you to select when the block change should take place when synchronous operation is selected:

NOC: Block change is enabled immediately

FINE: Block change in response to "Fine synchronism"

COARSE: Block change in response to "Coarse synchronism"

IPOSTOP: Block change for IPOSTOP (i.e. after setpoint-end synchronism)

The block change response is specified as a character string (i.e. with quotation marks).

The block change response can be specified simply by writing the letters in bold print. The remaining letters can be entered to improve legibility of the part program but they are not otherwise significant.

If no block change response is specified, then the currently selected response continues to apply.

With the programmable synchronism test markers **WAITC**, the replacement with new blocks is delayed until the parameterized synchronism is reached.

- **Coupling type**

DV (Desired Values): Setpoint coupling between FS and LS

AV (Actual Values): Actual value coupling between FS and LS

VV (Velocity Values): Speed coupling between FS and LS

If no coupling type is specified, then the currently selected type continues to apply.

Note

The coupling type may only be changed when synchronous operation is deactivated!

Examples

COUPDEF (SPI(2), SPI(1), 1.0, 1.0, "FINE", "DV")

COUPDEF (S2, S1, 1.0, 4.0)

COUPDEF (S2, SPI(1), 1.0)

Default settings

The following default settings apply to user-defined couplings:

- $Ü_{Numerator}$=1.0
- $Ü_{Denominator}$ =1.0
- Block change response = **IPOSTOP** (block change enabled with setpoint synchronism)
- Type of coupling = **DV** (setpoint coupling)

Delete couplings

Language command "`COUPDEL`" is used to delete user-defined couplings.

COUPDEL (FS, LS)

Note

COUPDEL impacts on an active coupling, deactivates it and deletes the coupling data. Alarm 16797 is therefore meaningless.

The following spindle adopts the last speed. This corresponds to the behavior associated with COUPOF(FS, LS).

Activate original coupling parameters

Language command "COUPRES" can be used to re-activate the configured coupling parameters.

COUPRES (FS, LS)

The parameters modified using COUPDEF (including the transformation ratio) are subsequently deleted.

Language command "COUPRES" activates the parameters stored in the machine and setting data (configured coupling) and activates the default settings (user-defined coupling).

Programmable block change

It is possible to mark a point in the NC program using the "WAITC" language command. The system waits at this point for fulfillment of the synchronism conditions for the specified FS and delays changes to new blocks until the specified state of synchronism is reached (see "Figure 19-3 Synchronism monitoring with COUPON and synchronism test mark WAITC with synchronization on a turning leading spindle (Page 863)").

WAITC (FS)

Advantage: The time between activating the synchronous coupling and reaching synchronism can be used in a meaningful way, technologically speaking.

Note

Basically, it is always possible to write WAITC. If the spindle indicated is not active as FS, the command for this spindle is without effect.

If no synchronism condition is indicated, the check is always performed for the synchronism condition programmed/configured on the respective coupling, at least for the setpoint synchronism.

Examples:

```
WAITC(S2),
:
WAITC(S2, "Fine"),
:
WAITC(S2, ,S4, "Fine")
```

Stop and block change

If "Stop" has been activated for the cancellation period of the axis enables for the leading or following spindle, then the **last** setpoint positions with the setting of the axis enables from the servo drive are approached again.

Commands `COUPON` and `WAITC` can influence the block change behavior. The block change criterion is defined using `COUPDEF` or via the MD21320 $MC_COUPLE_BLOCK_CHANGE_CTRL_1.

19.2.2 Switch the coupling (COUPON, COUPONC, COUPOF) on and off

Activate synchronous mode

Language command `COUPON` is used to activate couplings and synchronous mode.

Two methods by which synchronous operation can be activated are available:

1. **COUPON(FS, LS)**
   Fastest possible activation of synchronous operation with any angular reference between the leading and following spindles.
2. **COUPON(FS, LS, POS_{FS})**
   Activation of synchronous operation with a defined angular offset POS_{FS} between the leading and following spindles. This offset is referred to the zero degrees position of the leading spindle in a positive direction of rotation. The block change is enabled according to the defined setting. Range of POS_{FS}: 0 **...** 359.999 degrees.
3. **COUPONC(FS, LS)**
   When activating with `COUPONC`, the previous programming of M3 S... or M4 S... is adopted. A difference in speed is transferred immediately. An offset position cannot be programmed.

By programming `COUPON`(FS, LS, POS_{FS}) or SPOS when synchronous operation is already active, the angular offset between LS and FS can be changed.

Deactivate synchronous mode

Three different methods can be selected to deactivate synchronous mode:

1. **COUPOF(FS, LS)**
   Fastest possible deactivation of synchronous operation. The block change is enabled immediately.
2. **COUPOF(FS, LS, POS_{FS})**
   Deselection of synchronous operation after deactivation position POS_{FS} has been crossed. Block change is not enabled until this position has been crossed.
3. **COUPOF(FS, LS, POS_{FS}, POS_{LS})**
   Deselection of synchronous operation after the two deactivation positions POS_{FS} and POS_{LS} have been crossed. Block change is not enabled until **both** programmed positions have been crossed.
   Range of POS_{FS}, POS_{LS}: 0 **...** 359,999°.

If continuous path control (G64) is programmed, a non-modal stop is generated internally in the control.

Examples:

```
COUPDEF (S2, S1, 1.0, 1.0, "FINE, "DV")
:
COUPON (S2, S1, 150)
:
COUPOF (S2, S1, 0)
:
COUPDEL (S2, S1)
```

1. **COUPOFS(FS, LS)**
   Deactivating a coupling with stop of the following spindle. Block change is performed as quickly as possible with immediate block change.
2. **COUPOFS(FS, LS, POS_{FS})**
   After the programmed deactivation position that refers to the machine coordinate system has been crossed, the block change is not enabled until the deactivation positions POSFS have been crossed.
   Value range 0 ... 359.999°.

19.2.3 Axial system variables for synchronous spindle

Determining current coupling status

The current coupling status of the following spindle can be read in the NC part program with the following axial system variable:

$AA_COUP_ACT[<axial expression>]

For explanation of <axial expression>, see Section "Synchronous mode (Page 849)".

Example:

$AA_COUP_ACT[S2]

The value read has the following significance for the following spindle:

Byte = 0:	No coupling active
Bit 2 = 1:	Synchronous spindle coupling active
Bit 2 = 0:	Synchronized spindle coupling is not active

Read current angular offset

The current position offset between the FS and LS can be read in the NC part program by means of the following axial system variables:

- Setpoint-based position offset between FS and LS:
  $AA_COUP_OFFS[<axial expression>]
- Actual-value-based position offset between FS and LS:
  $VA_COUP_OFFS[<axial expression>]

Example:

$AA_COUP_OFFS[S2]

If an angular offset is programmed with COUPON, this coincides with the value read after reading the setpoint synchronization.

Reading the programmed angular offset

The position offset last programmed between the FS and LS can be read in the NC part program by means of the following axial system variables:

$P_COUP_OFFS[<axial expression>]

Note

After cancellation of the servo enable signal when synchronous operation and follow-up mode are active, the position offset applied when the controller is enabled again is different to the originally programmed value.

$P_COUP_OFFS only returns the value originally programmed. $AA_COUP_OFFS and $VA_COUP_OFFS return the current value. The programmed offset can be recreated with NST DB31, ... DBX31.4 (resynchronization).

19.2.4 Automatic selection and deselection of position control

Behavior in speed control mode

In DV coupling mode, program instructions COUPON, COUPONC and COUPOF, COUPOFS are used to activate and/or deactivate position control for the leading spindle as required. If there are several following spindles on the leading spindle, then in speed-controlled mode, the **first** DV **activates** coupling position control for the leading spindle and the **last** DV coupling **deactivates** coupling position control for the leading spindle if SPCON is not programmed.

The leading spindle does not need to be located in the same channel as the following spindle.

Automatic selection with COUPON and COUPONC

Depending on the coupling type, the effect of COUPON and COUPONC on the position control for synchronous operation is as follows:

Coupling type	DV	AV	VV
Following spindle FS	Position control ON	Position control ON	No action
Leading spindle LS	Position control On [1]	No action	No action

[1] The position control is activated by a COUPON and COUPONC instruction if **at least one** following spindle has been coupled to it with coupling type DV.

Automatic deselection with COUPOF and COUPOFS

Depending on the coupling type, the effect of COUPOF and COUPOFS on the position control is as follows:

Coupling type	DV	AV	VV
Following spindle FS	Position control OFF [2]	Position control OFF [2]	No action [2]
Leading spindle LS	Position control OFF [3]	No action	No action

[2] COUPOF and COUPOFS without position specification

Speed control mode is activated for the following spindle. Positioning mode is activated with COUPFS with a stop position. Position control is **not deactivated** if the following spindle was located in position-controlled spindle mode using SPCON or COUPFS was programmed with position.

[3] With COUPOF position control is **deactivated** if there are no more couplings of the DV coupling type for this leading spindle. Position control **is retained** if the leading spindle is in positioning mode or axial mode **or** was in position-controlled spindle mode using SPCON.

19.3 Configuration

Note

One synchronous-spindle coupling can be configured for each channel.

Table 19-1 Machine data

Number	Name: $MC_	Function
MD21300	COUPLE_AXIS_1[<n>]	Machine axes of the synchronous-spindle coupling: • <n> = 0: Machine axis number of the following spindle • <n> = 1: Machine axis number of the leading spindle Machine axis numbers in accordance with: MD20070 $MC_AXCONF_MACHAX_USED (machine axes in the channel) Machine axis numbers == 0: No coupling configured. The following system data is then not relevant. **Note**: The machine axes configured for the synchronous-spindle coupling **cannot** be changed using program commands.
MD21320	COUPLE_BLOCK_CHANGE_CTRL_1	Block change release after activating the synchronous operation [1]: • Immediately • On reaching "Synchronism fine" • On reaching "Synchronism coarse" • On reaching "Synchronism setpoint" **Note:** No change protection[1], the block change release can be changed with the COUPDEF command.

Number	Name: $MC_	Function
MD21310	COUPLING_MODE_1	Coupling type [1]: • Actual value coupling • Setpoint value coupling • Speed coupling **Note:** No change protection[1], the coupling type can be changed for **deactivated** coupling with the COUPDEF command.
MD21330	COUPLE_RESET_ MODE_1	Behavior of the synchronous-spindle coupling with regard to NC Start, NC Stop and Reset.
MD21340	COUPLE_IS_WRITE_ PROT_1	Change protection for coupling
1) See MD21340 $MC_COUPLE_IS_WRITE_PROT_1		

Table 19-2 Setting data

Number	Name: $SC_	Function
SD42300	COUPLE_RATIO_1[<n>]	Speed transmission ratio: FS / LS = numerator / denominator [1]: • <n> = 0: Numerator (FS) • <n> = 1: Denominator (LS) **Note:** No change protection[1], the transmission ratio can be changed with the COUPDEF command.
1) See MD21340 $MC_COUPLE_IS_WRITE_PROT_1		

19.3.1 Response of the synchronous-spindle coupling for NC Start

The behavior of the synchronous-spindle coupling during NC Start depends on the setting in the following machine data:

Configured synchronous-spindle coupling

Response	MD21330 $MC_COUPLE_RESET_MODE_1
Maintain coupling	Bit 0 = 0
Deselect coupling	Bit 0 = 1
Activate configured data	Bit 5 = 1
Switch the coupling on	Bit 9 = 1

Programmed synchronous-spindle coupling

Response	MD20112 $MC_START_MODE_MASK
Maintain coupling	Bit 10 = 0
Deselect coupling	Bit 10 = 1

19.3.2 Behavior of the synchronous-spindle coupling for reset

The behavior of the synchronous operation for reset and at program end depends on the setting in the following machine data:

Configured synchronous-spindle coupling

Response	MD21330 $MC_COUPLE_RESET_MODE_1	MD20110 $MC_RESET_MODE_MASK
Maintain coupling	Bit 1 = 0	Bit 0 = 1
Deselect coupling	Bit 1 = 1	Bit 0 = 1
Activate configured data	Bit 6 = 1	Bit 0 = 1

Programmed synchronous-spindle coupling

	MD20110 $MC_RESET_MODE_MASK
Maintain coupling	Bit 0 = 1, Bit 10 = 1
Deselect coupling	Bit 0 = 1, Bit 10 = 0

19.4 Special features

19.4.1 Special features of synchronous mode in general

Control dynamics

When using the setpoint coupling, the position control parameters of FS and LS (e.g. K_V factor) should be matched with one another. If necessary, different parameter blocks should be activated for speed control and synchronized mode. The control parameters of the following spindle may differ from position control, feedforward control and parameter block, as also in the uncoupled case, set using MD30455 $MA_MISC_FUNCTION_MASK (see Section "Special points regarding start-up of a synchronous spindle coupling (Page 886)").

Feedforward control

Due to the improved control system dynamic response it provides, feedforward control for the following and leading spindles in synchronous mode is **always active.**

It can, however, be deselected for FS and LS with axis-specific MD32620 $MA_FFW_MODE. If MD32620 $MA_FFW_MODE is set to zero, there are function limitations. Position control can no longer be activated in motion with SPCON. SPOS, M19 or SPOSA are therefore not possible. The NC part program cannot deactivate the feedforward control for LS and FS with FFWOF.

The feedforward control mode (speed or torque feedforward control) is defined in axis-specific MD32620 $MA_FFW_MODE defined (see Function Manual "Monitoring and Compensating", Section "K3: Compensations").

Speed and acceleration limits

The speed and acceleration limits of the spindles operating in synchronous mode are determined by the "weakest" spindle in the synchronous spindle pair. The current gear stages, the programmed acceleration and, for the leading spindle, the effective position control status (On/Off) are taken into account for this purpose.

The maximum speed of the leading spindle is calculated internally in the control taking into account the speed ratio and the spindle limitations of the following spindle.

Multiple couplings

If the system detects that a coupling is already active for an FS and LS when the synchronous mode is activated, then the activation process is ignored and an alarm message is generated.

Examples of multiple couplings:

- A spindle is acting as the FS for several LS
- Coupling cascade (an FS is an LS of an additional coupling)

Number of configurable spindles per channel

Every axis in the channel can be configured as a spindle. The number of axes per channel depends on the control version.

Cross-channel setpoint coupling

- Cross-channel synchronous spindle couplings can be implemented with no additional restrictions for DV, AV, and VV.
- Any number of following spindles, corresponding to the number of all spindles minus one spindle for the master, in any channels on an NCU can be coupled to one leading spindle.

Start synchronous mode using ASUB

When the PLC starts an ASUB, in the AUTOMATIC or MDI modes, synchronous operation can be switched on and off – or terminated.

Further information

Function Manual Basic Functions; Mode group, channel, program mode, reset response

Response to alarms

In the case of an alarm, which occurs during synchronous operation, and acts as alarm response "withdraw control enable" and "activate follow-up mode" in the control, the ongoing control behavior is the same as the behavior due to NC/PLC interface signals:

- DB31, ... DBX2.1 = 0 (controller enable)
- DB31, ... DBX1.4 = 1 (follow-up mode)

(See Section "Synchronous mode and NC/PLC interface signals (Page 877)")

By resynchronizing via the NC/PLC interface signal:

DB31, ... DBX31.4 = 0 → 1 (resynchronization)

If the programmed offset is restored (see Section "Restore synchronism of following spindle (Page 875)").

Block search when synchronous operation is active

Note

When synchronous operation is active for a block search, then it is recommended that only block search type 5, "Block search via program test" (SERUPRO), is used.

See also

S1: Spindles (Page 733)

19.4.2 Restore synchronism of following spindle

Causes for a positional offset

When the coupling is reactivated after the drive enable signals have been canceled, a positional offset can occur between the leading and following spindles if follow-up mode is activated. A positional offset can be caused by:

- A part has been clamped or both spindles have been turned manually (machine area is open, drives are disconnected from supply).
- After the spindle enable signals are canceled, the two spindles coast to standstill at different speeds if they are not mechanically coupled.
- A drive alarm occurs (internal follow-up mode):
  DB31, ... DBX61.3 (follow-up mode active) = 1
  When the alarm is cleared, the NC must not trigger any synchronization motion.
- A synchronization was not executed due to a synchronization lock of the following spindle:
  DB31, ... DBX29.5 (Disable synchronization)

Basic procedure

If the following and leading spindles have fallen out of synchronism, or failed to synchronize at all, synchronism can be restored between them by the following measures:

1. Set the axis enable signals and cancel synchronization disable signal if this has been set.
2. Start following spindle resynchronization with the NC/PLC interface signal:
   DB31, ... DBX31.4 (resynchronization)
   Only after the resynchronization process is complete can the setpoint-end synchronism be fully restored.
3. Wait until the coupled spindles have synchronized.

Enable resynchronization

Setting the enabling signals closes the coupling at the current actual positions. The two following NC/PLC interface signals are set:

DB31, ... DBX98.1 (coarse synchronism)

DB31, ... DBX98.0 (fine synchronism)

The following **requirements** must be fulfilled for resynchronization to work:

- The axis enabling signal must be set for the following spindle.
- The PLC must not set any synchronization disables for the following spindle:
  DB31, ... DBX31.5 (Disable synchronization)

Resynchronize following spindle

Resynchronization is started for the relevant following spindle and commences as soon as the low-high edge of following interface signal is detected:

DB31, ... DBX31.4 (resynchronization)

The NC acknowledges the detection of the edge by outputting the NC/PLC interface signal:

DB31, ... DBX99.4 (synchronization running)

The interface signal "Synchronization running" is reset if:

- synchronization of the following spindle has been completed up to the stage at which there is synchronism at the setpoint end.
- the NST DB31, ... DBX31.4 (resynchronization) was reset.

Response of synchronous signals during additional movements for the following spindle

The superimposed component is calculated to establish the synchronism signals.

Example

Program code	Comment
N51 SPOS=0 SPOS[2]=90	
N52 OUPDEF(S2,S1,1,1,"FINE","DV")	
N53 COUPON(S2,S1,77)	
N54 M0	; Offset=77°, "coarse", "fine" synchronous run signals exist.
N55 SPOS[2]=0 FA[S2]=3600	; Difference in speed, synchronism signals "coarse", "fine" are reported
N56 M0	; (note tolerances, see above)
	; Offset=0°, "coarse", "fine" synchronous run signals exist.
N60 M2=3 S2=500	; difference in speed, synchronism signals "coarse", "fine" are reported.
	; offset undefined, synchronism signals "coarse", "fine" are reported.

Program code	Comment
N65 M0	; (Note tolerances, see above)

Note

The axis enable signals can be canceled to interrupt a movement overlaid on the following spindle (e.g. SPOS). This component of the movement is not affected by IS "NC/PLC interface signal" DB31, ... DBX31.4 (resynchronization), but is restored by the REPOS operation.

Supplementary condition

IS DB31, ... DBX31.4 (resynchronization) has any effect only if there is a **defined offset position** between the following spindle and leading spindle.

This is the case following `COUPON` with offset positions such as `COUPON(...,77)` or `SPOS`, `SPOSA`, `M19` for the following spindle with a closed coupling..

19.4.3 Synchronous mode and NC/PLC interface signals

Note

During synchronous operation, the effect of the associated interface signal for the leading (LS) or following spindle (FS) on the coupling must always be considered.

Spindle override (DB31, ... DBB19)

Only the spindle override value of the LS is active in synchronous operation.

Spindle disable (DB31, ... DBX1.3)

LS	FS	coupling	Response
0	0	off	Setpoints are output
0	1	off	No setpoint output for FS
1	0	off	No setpoint output for LS
1	1	off	No setpoint output for LS and FS
0	0	ON	Setpoints are output
0	1	ON	Spindle disable not effective for FS
1	0	ON	Spindle disable also effective for FS, no setpoint output
1	1	ON	No setpoint output for LS and FS

Controller enable (DB31, ... DBX2.1)

LS: Resetting the "controller enable" during synchronous operation

If the controller enable of the LS is reset during synchronous operation for active **setpoint** coupling, a control-internal switching is made to the **actual value** coupling. If the controller enable is reset while the LS is traversing, the LS is stopped and an alarm issued. Synchronous operation remains active.

LS and FS: Selection of synchronous operation without controller enable

If the "controller enable" for LS and FS is not set for selected synchronous operation, it will be activated. LS and FS, however, are not traversed until controller enable for LS **and** FS has been granted.

LS and FS: Setting the "Controller enable"

The position assumed by a spindle with setting the "controller enable" depends on DB31, ... DBX1.4 == <value> (follow-up operation):

<value>	Assumed spindle position
0	Position when controller enable is canceled
1	Current position

Note

It is recommended for synchronous spindles during a block search, to write the DB31, ... DBX2.1 interface signal (controller enable) always together for FS **and** LS. If this is not done, the block search, for example, stops after a FS machining because the controller enable of the LS is not pending.

Note

Synchronism error

If the DB31, ... DBX2.1 interface signal (controller enable) is canceled for the FS after Spindle Stop without the coupling being deactivated beforehand, then any synchronism error resulting from external intervention will not be compensated when the "controller enable" is activated again. This causes any programmed angular relationship between FS and LS to be lost.

The angular relationship can be restored by resynchronizing: DB31, ... DBX31.4 = 1 (resynchronization)

Follow-up mode (DB31, ... DBX1.4)

For follow-up operation, the set position is regularly set to the current actual position:

DB31, ... DBX1.4 == 1 (follow-up operation) **AND** DB31, ... DBX2.1 == 0 (controller enable)

⇒ Cyclic: Set position = actual position

Note

DB31, ... DBX1.4 (follow-up operation) is relevant only for DB31, ... DBX2.1 == 0 (controller enable)

Position measuring system 1/2 (DB31, ... DBX1.5 and 1.6)

Switchover of the position measuring system for the FS and LS is possible during synchronous operation. The coupling is retained.

Note

It is recommended to switchover the position measuring system for FS and LS only for deselected synchronous operation.

Delete distance-to-go / spindle reset (DB31, ... DBX2.2)

LS: Setting spindle reset during synchronous operation

The setting of spindle reset causes the LS to be braked to standstill with the parameterized acceleration. Synchronous operation remains active.

Any superimposed movement, other than together with `COUPON` / `COUPONC`, will be completed as fast as possible.

Spindle stop (feed stop) (DB31, ... DBX4.3)

LS and FS: Setting spindle stop during synchronous operation

The setting of "spindle stop" for FS or LS causes both spindles to be braked synchronous to standstill. Synchronous operation remains active.

Resetting spindle stop

Once "spindle stop" has been reset for both spindles, reacceleration is made to the last valid speed setpoint.

Application example

Bring FS and LS to a standstill when a protection door is opened during synchronous operation.

Signal characteristic for LS and FS:

1. Stop: DB31, ... DBX4.3 = 1 (spindle stop)
2. Waiting for standstill: DB31, ... DBX61.4 == 1
3. Stopping: DB31, ... DBX2.1 = 0 (controller enable)

Delete S value (DB31, ... DBX16.7)

LS: Delete S value during synchronous operation

If "delete S value" is set, the LS is braked to a standstill using a ramp. Synchronous operation remains active.

FS: Delete S value during synchronous operation

The control interface signal does not have any function for the FS in synchronous operation.

Resynchronize spindle 1/2 (DB31, ... DBX16.4 and 16.5)

LS: Resynchronizing the position measuring system during synchronous operation

Note

It is recommended to resynchronize the position measuring system of the LS only for deselected synchronous operation.

Resynchronize (DB31, ... DBX31.4)

FS: Restoring the programmed angular offset

If synchronism between FS and LS is lost or not performed, the programmed angular offset can be restored.

- Requirement: DB31, ... DBX31.4 = 1 (resynchronization)
- Acknowledgment: DB31, ... DBX99.4 == 1 (synchronization running)

Traverse keys for JOG (DB31, ... DBX4.6 and 4.7)

The "plus and minus traversing keys" for JOG are **not disabled** in the control for the FS in synchronous operation, i.e. the FS executes a superimposed motion if one of these keys is pressed.

Note

If superimposed traversing movements are to be precluded, they must be locked out by measures in the PLC user program.

NC Stop axes plus spindles (DB21, ... DBX7.4)

"NC Stop axes plus spindles" in synchronous operation decelerates the coupled spindles in accordance with the selected dynamic response. They continue to operate in synchronous mode.

NC Start (DB21, ... DBX7.1)

(See Section "Response of the synchronous-spindle coupling for NC Start (Page 872)")

Note

NC Start after NC Stop does not deselect synchronous operation.

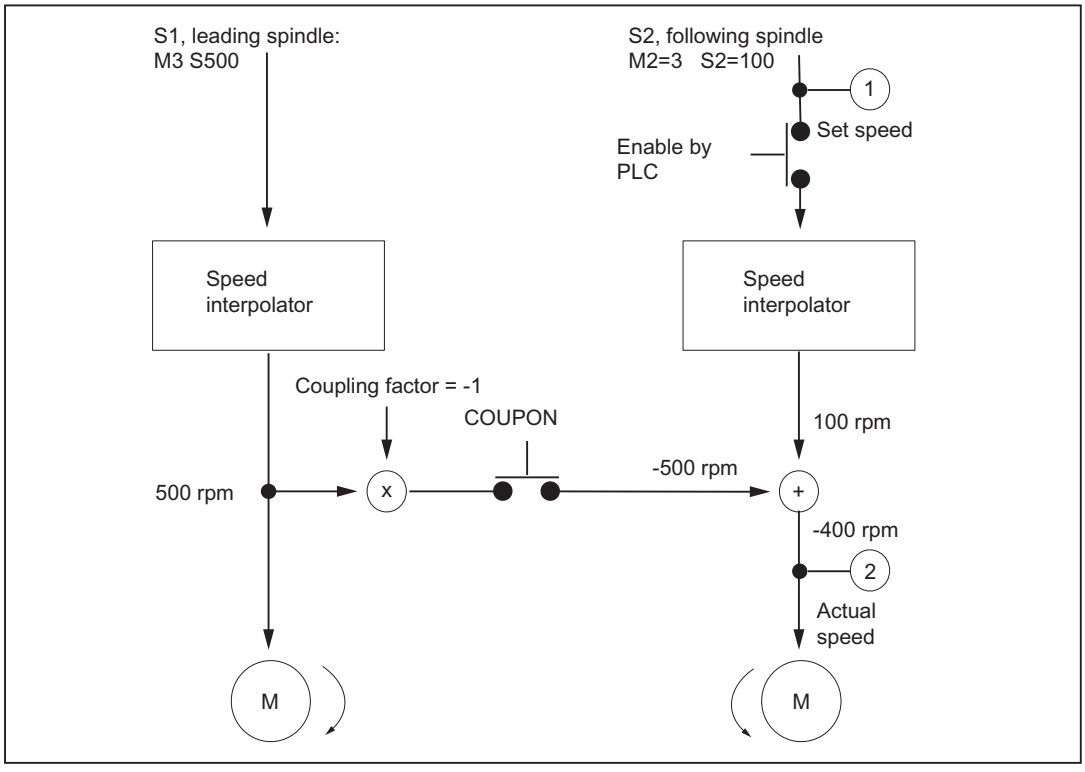
19.4.4 Differential speed between leading and following spindles

When does a differential speed occur?

A differential speed develops, e.g. with turning machine applications, when two spindles oppose one other. Through the signed addition of two speed sources, a speed component is derived from the leading spindle via the coupling factor. In addition to this, for the following spindle a speed (S...) and a direction of rotation (M3/M4) can be programmed.

Generally, to achieve synchronous operation, a coupling factor is used with a value of '-1'. This sign reversal then results in a differential speed for the following spindle as compared to an additional programmed speed.

This typical behavior in relation to the NC is illustrated in the following diagram.



Example

Program code	Comment
N01 M3 S500	; S1 rotates in the positive direction with 500 rpm
	; the master spindle is spindle 1
N02 M2=3 S2=300	; S2 rotates in the positive direction with 300 rpm
N05 G4 F1	
N10 COUPDEF(S2,S1,-1)	; Coupling factor -1:1
N11 COUPON(S2,S1)	; Activate coupling. The speed of the following spindle S2 results from the speed of the leading spindle S1 and the coupling factor.
N26 M2=3 S2=100	; Programming the differential speed, S2 is the following spindle.

Application

Manufacturing operations with positioned leading spindle and rotating tools require exact synchronism with the counter spindle which then functions like a following spindle. A turret rotating about the following spindle allows parts to be machined with different tool types.

The following diagram shows an application in which the tool is positioned parallel to the main spindle.

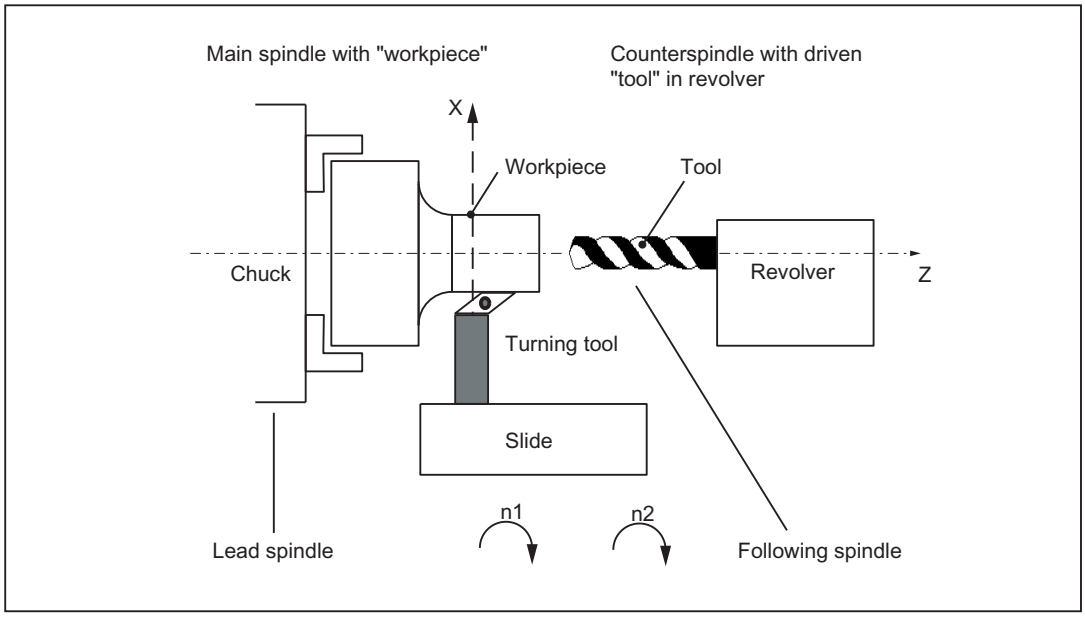


Figure 19-4 Single-slide lathe with revolver around the Z axis

Preconditions

Basic requirements for differential speed programming:

- Synchronous spindle functionality is required.
- The dynamic response of the following spindle must be at least as high as that of the leading spindle. Otherwise, the system may suffer from reduced quality, for example, rigid tapping without a compensating chuck G331/G332.
- The differential speed must be programmed in the channel in which the following spindle is also configured. The leading spindle can be programmed in a different channel.
- The differential speed must be enabled for the following spindle by the PLC via IS "Enable overlaid movement" (DB31, ... DBX26.4). If the enable signal has not been set, alarm 16771 "Channel% Following axis% Overlaid movement not enabled" is output. This alarm is cleared when IS "Enable overlaid movement" (DB31, ... DBX26.4) is set or the coupling is opened.

Note

Positioning motion such as SPOS, SPOSA, M19 and axis motion do not have to be enabled.

Note

The differential speed does not therefore affect the coupling process.

The following or leading spindle cannot change gear stages while a coupling is active.

Activate coupling with COUPONC

When the coupling is activated, the following spindle is accelerated, as before, to the leading spindle speed through application of the coupling factor. If the following spindle is already rotating (M3, M4) when the coupling is activated, it continues with this motion after coupling.

Deactivate coupling

If the coupling is deactivated, the following spindle continues to rotate at the speed corresponding to the sum of both speed components. The spindle behaves as if it had been programmed with the speed and direction transferred from the other spindle. When deactivating, there are no differences to the previous behavior.

Differential speed

A differential speed results from the **renewed** programming of the following spindle (in the example, S2=...) or M2=3, M2=4 in speed control mode **during** an active synchronous spindle coupling or by adopting the speed of the following spindle for COUPONC.

Condition:

Speed S... must also be re-programmed with direction of rotation M3 or M4. Otherwise alarm 16111 "Channel% Block% Spindle% No speed programmed" is displayed.

Read offsets of following spindle

The current offset always changes when a differential speed is programmed. The current offset can be read at the setpoint end with $AA_COUP_OFFS[Sn] and at the actual value end with $VA_COUP_OFFS[Sn].

The last offset programmed returns the variable $P_COUP_OFFS[Sn].

Display differential speed

The programmed difference component is displayed as the speed setpoint for the programmed differential speed (in our example, corresponds to 100 rpm).

The actual speed refers to the motor speed. In the example, the actual speed is 500 rev/min * (-1) + 100 rpm = -400 rev/min.

IS, NC to the PLC

Following spindle in speed-controlled operation

The IS "Spindle in setpoint range" (DB31, ... DBX83.5) is set for the following spindle by the NC if the programmed speed difference (see previous example, N26 with M2=3 S2=100) is reached. If a differential speed is programmed and not enabled by the PLC, this VDI interface signal is not set.

Even if a differential speed has been programmed, the following spindle remains under position control if this is required by the coupling.

Note

The axial VDI interface signal NC → PLC IS "Superimposed motion" (DB31, ... DBX98.4) is set when the differential speed programming creates setpoints in addition to the coupling setpoints.

Actual direction of rotation clockwise (DB31, ... DBX83.7)

IS "Actual direction of rotation clockwise" (DB31, ... DBX83.7) refers to the resulting motor direction.

IS PLC to the NC

Influence on following spindle via PLC interface

The effect of the axial VDI interface signals on the following spindle with differential speed in speed control mode is described below:

Delete distance to go / Spindle Reset (DB31, ... DBX2.2)

The programmed differential speed and direction can be terminated by IS "Delete distance-to-go / Spindle Reset" (DB31, ... DBX2.2). To delete the programmed speed only, it is possible to set IS "Delete S value" (DB31, ... DBX16.7).

Resynchronize spindle 1/2 (DB31, ... DBX16.4 and 16.5)

The IS "Resynchronize spindle 1/2" (DB31, ... DBX16.4/16.5) are **not** locked. Any positional offset is not compensated automatically by the coupling.

Invert M3/M4 (DB31, ... DBX17.6)

IS "Invert M3/M4" (DB31, ... DBX17.6) only inverts the additional programmed speed component for the following spindle.

The motion component generated by the synchronous spindle coupling remains unaffected.

Spindle override (DB31, ... DBB19)

The "Spindle override" VDI interface (DB31, ... DBB19) only affects the additional programmed speed component for the following spindle. If the spindle override switch is transferred to all axial inputs, then any change in the spindle override value is applied **twice** to the following spindle:

- once indirectly by a change in speed for the leading spindle and
- once in the programmed component of the following spindle.

The offset value can be adjusted accordingly in the PLC program.

Deselecting the coupling

If the coupling is deactivated, the following spindle continues to rotate at the speed corresponding to the sum of both speed components. The motion transition upon coupling deselection is at continuous speed.

With `COUPOF` , the spindle behaves as if it had been programmed with the speed and direction transferred from the other spindle. In the example, this would be M4 S400.

When `COUPOFS` is programmed, the following spindle is decelerated to standstill from the current speed.

Activate additional functions

The following spindle can also be a master spindle. In this case, it is capable of additional functions.

- Rotational feedrate with G95, G96 and G97. With G96 S2=... the "constant cutting speed" can be activated for the following spindle.
  The speed dependent on the position of the transverse axis is the setpoint speed for the speed interpolator of spindle 2 and is therefore included in the total speed of S2.
- Rigid tapping without compensating chuck with G331, G332.

19.4.5 Behavior of synchronism signals during synchronism correction

Effect of synchronism correction

New 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.

19.4.6 Delete synchronism correction and NC reset

Variable $AA_COUP_CORR[Sn] returns the value zero for different situations in which the synchronism correct is deleted:

- Once a synchronized spindle coupling has been activated for the following in question with COUPON(..)/COUPONC(..), an existing synchronism correction is adopted in the setpoint position.
- A synchronism correction active during NC reset but not at the parts program end is adopted in the setpoint position. This does not affect the synchronism signals.
- At M30, an existing synchronism correction is retained
- At the user end, the correction value can also be deleted at any early point by describing the variable $AA_COUP_CORR with the **value zero**. The synchronism correction is removed immediately and using a ramp with reduced acceleration rate if larger values are involved.

19.4.7 Special points regarding start-up of a synchronous spindle coupling

Spindle start-up

The leading and following spindles must be started up initially like a normal spindle. The corresponding procedure is described in:
Further information
Commissioning Manual CNC: NC, PLC, drive
S1: Spindles (Page 733)

Requirements

The following parameters must then be set for the synchronous spindle pair:

- The machine numbers for the leading and following spindles
  (for permanently configured coupling with channel-specific machine data MD21300 $MC_COUPLE_AXIS_1[n])
- The required coupling mode (setpoint, actual value or speed coupling)
  (for permanently configured coupling with channel-specific machine data MD21310 $MC_COUPLING_MODE_1[n])

- The gear stage(s) of FS and LS for synchronous operation
- The following coupling properties are still applicable for permanently configured synchronous spindle coupling:
  - Block change response in synchronous spindle operation:
    MD21320 $MC_COUPLE_BLOCK_CHANGE_CTRL_1
  - Coupling cancellation response:
    MD21330 $MC_COUPLE_RESET_MODE_1
  - Write-protection for coupling parameters:
    MD21340 $MC_COUPLE_IS_WRITE_PROT_1
  - Transformation parameters for synchronous spindle coupling:
    SD42300 $SC_COUPLE_RATIO_1[n]

Command behavior of FS and LS for setpoint coupling

In order to obtain the best possible synchronism in **setpoint couplings**, the FS and LS must have the same dynamic response as the response to setpoint changes. The axial control loops (position, speed and current controllers) should each be set to the **optimum** value so that variances can be eliminated as quickly and efficiently as possible.

The **dynamic response adaptation** function in the setpoint branch is provided to adapt different axis dynamic responses without loss of control quality (see also Function Manual "Monitoring and Compensating", Section "K3: Compensations"). The following control parameters must each be set optimally for the FS and LS:

- K_V factor (MD32200 $MA_POSCTRL_GAIN)
- Feedforward control parameters
  MD32620 $MA_FFW_MODE
  MD32610 $MA_VELO_FFW_WEIGHT
  MD32650 $MA_AX_INERTIA
  MD32800 $MA_EQUIV_CURRCTRL_TIME
  MD32810 $MA_EQUIV_SPEEDCTRL_TIME

Behavior during loss of synchronism:

- Axis-specific MD32620 $MA_FFW_MODE

We recommend setting the **feedforward control mode** of the following axis to speed feedforward control with Tt balancing MD32620 = 3.

This feedforward control mode can be further optimized for a more secure balancing process by changing the axis-specific machine data:

Machine data	Meaning
MD32810 EQUIV_SPEEDCTRL_TIME	Equivalent time constant speed control loop for feedforward control
MD37200 COUPLE_POS_TOL_COURSE	Threshold value for "Coarse synchronism"
MD37210 COUPLE_POS_TOL_FINE	Threshold value for "Fine synchronism"
MD37220 COUPLE_VELO_TOL_COURSE	Velocity tolerance 'coarse'
MD37220 COUPLE_VELO_TOL_FINE	Velocity tolerance 'fine'

In such cases, higher threshold values for the synchronism signals and larger position and/or speed tolerances result in more stable results.

Dynamic response adaptation

To obtain a good control behavior, FS and LS must have the same dynamic response. The following error for FS and LS must be equal at any given speed. For dynamically different spindles, a matching via the dynamic response adaptation can be achieved in the setpoint branch. The difference of the equivalent time constants between the dynamic "weakest" spindle to the associated other spindle must be entered as time constant of the dynamic response adaptation.

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.

- Equivalent time constant LS: MD32810 $MA_EQUIV_SPEEDCTRL_TIME[<LS>] = 5 ms
- Equivalent time constant FS: MD32810 $MA_EQUIV_SPEEDCTRL_TIME[<FS>] = 3 ms
- Time constant of dynamic response adaptation for FS: MD32910 $MA_DYN_MATCH_TIME[<FS>] = 5 ms - 3 ms = 2 ms
- Activation of the dynamic response adaptation for FS: MD32900 $MA_DYN_MATCH_ENABLE[<FS>] = 1

The following error for FS and LS is identical for correctly set dynamic response adaptation: Operating area "Diagnostics" > "Service axes"

For the optimization, it may be necessary to adjust servo gain factors or feedforward control parameters slightly.

Position control parameter sets

In the case of spindles, each gear stage is assigned a position-control parameter set. These parameter sets can be used to adapt the dynamic response for LS and FS in synchronous operation. This requires that a different gear stage is used for speed and positioning operation and synchronous operation. The associated gear stage must be switched before activating the associated operating mode.

Control parameters

The following control parameters must be set identically for the FS and LS:

- MD33000 $MA_FIPO_TYPE (fine interpolator type)
- MD32400 $MA_AX_JERK_ENABLE (axial jerk limitation)
- MD32402 $MA_AX_JERK_MODE (filter type for axial jerk limitation)
- MD32410 $MA_AX_JERK_TIME (time constant for the axial jerk filter)
- MD32412 $MA_AX_JERK_FREQ (blocking frequency of the axial jerk filter)
- MD32414 $MA_AX_JERK_DAMP (damping of the axial jerk filter)
- MD32420 $MA_JOG_AND_POS_JERK_ENABLE (release jerk limitation)
- MD32430 $MA_JOG_AND_POS_MAX_JERK (axial jerk)

FS: Automatic parameterization of the control parameters

The control parameters of the following spindle can be set as follows using this machine data:

MD30455 $MA_MISC_FUNCTION_MASK

Bit 5=0: Synchronous spindle coupling, following spindle:

Position control, feedforward control and parameter block are set for the following spindle.

Bit 5=1: Synchronous spindle coupling:

The control parameters of the following spindle are set as in an uncoupled scenario.

Further information
G2: Velocities, setpoint / actual value systems, closed-loop control (Page 31)

Automatic transfer of the LS parameters for synchronous operation

For the automatic dynamic response adaptation for FS and LS, for synchronous operation, the parameter values for the position control, feedforward control and parameter records of the FS can be transferred from the LS:

MD30455 $MA_MISC_FUNCTION_MASK, Bit 5

Separate dynamic response for spindle and axis operations

In spindle and axis operations, dynamic programming FA, OVRA, ACC and VELOLIMA can be set separately from one another with the following MD:

MD30455 $MA_MISK_FUNCTION_MASK Bit 6=0

Assignment is undertaken by the programmed axis or spindle name. For example, in spindle operation, VELOLIMA[S1]=50 therefore only reduces the maximum speed to 50% and in axis operation, VELOLIMA[C]=50 only reduces the maximum speed to 50%.

If, for example, VELOLIMA[S1]=50 and VELOLIMA[C]=50 are to have the same effect as before with this machine data, the programming of FA, OVRA, ACC and VELOLIM have an effect regardless of the programmed names:

MD30455 $MA_MISK_FUNCTION_MASK Bit 6=1

Knee-shaped acceleration characteristic

For the leading spindle, the effect of a knee-shaped acceleration characteristic on the following spindle is identified by the following axis-specific machine data:

- MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT (speed for reduced acceleration) and
- MD35230 $MA_ACCEL_REDUCTION_FACTOR (reduced acceleration).

If MD35242 $MA_ACCEL_REDUCTION_TYPE is present, it is also used to configure the type of acceleration reduction. Otherwise a hyperbolic drop in acceleration is assumed.

If the dynamic response of a following spindle is lower than that of the leading spindle when the coupling factor is taken into account, the leading spindle dynamic response is reduced to the required level while the coupling is active.

The acceleration should be **constant** over the entire speed range for the following spindle. However, if a knee-shaped acceleration characteristic is also stored in the above-mentioned machine data for the following spindle, this is only taken into account when the spindles are coupled in. The setpoints of the following spindle are applied for the specified knee-shaped acceleration characteristic.

Further information
B2: Acceleration and jerk (Page 347), knee-shaped acceleration characteristic

Actual value coupling

In an actual value coupling (AV), the drive for the FS must be considerably more dynamic than the leading spindle drive. The individual drives in an actual value coupling are also set optimally according to their dynamic response.

An actual value coupling should only be used in exceptional cases.

Speed coupling

The velocity coupling (VV) corresponds internally to a setpoint coupling (DV), but with lower dynamic requirements of the FS and LS. A position control servo loop is not required for the FS and/or LS. Measuring systems are not required.

Threshold values for coarse/fine synchronism

After controller optimization and feedforward control setting, the threshold values for coarse and fine synchronism must be entered for the FS.

- Threshold value for "Coarse synchronism"
  Axis-spec. MD7200: AV, DV: COUPLE_POS_TOL_COARSE
  MD37220: VV: COUPLE_VELO_TOL_COARSE
- Threshold value for "Fine synchronism"
  Axis-spec. MD37210: AV, DV: COUPLE_POS_TOL_FINE
  MD37230: VV: COUPLE_VELO_TOL_FINE

The values of the FS must be calculated according to the accuracy requirements of the machine manufacturer, and the PLC interface must be checked via the service display.

Angular offset LS/FS

If there must be a defined angular offset between the FS and LS, e.g. when synchronous operation is activated, the "zero degree positions" of the FS and LS must be mutually adapted. This can be done with the following machine data:

- MD34100 $MA_REFP_SET_POS
- MD34080 $MA_REFP_MOVE_DIST
- MD34090 $MA_REFP_ MOVE_DIST_CORR

Further information
R1: Referencing (Page 291)

Service display for FS

In the "Diagnostics" operating area, when commissioning in the synchronous mode, the following values are displayed for the following spindle:

- Actual deviation between setpoints of FS and LS
  Value displayed: Position offset in relation to leading spindle (setpoint)
  (value corresponds to angular offset between FS and LS that can be read with axis variable $AA_COUP_OFFS in the part program)
- Actual deviation between actual values of FS and LS
  Value displayed: Position offset in relation to leading spindle (actual value)

Further information
Operating Manual

19.5 Boundary conditions

Availability of the "synchronous spindle" function

The function is an option ("synchronous spindle/multi-edge turning" or the corresponding optional version of the generic coupling), which must be assigned to the hardware via the license management.

Note

For information on the different versions of generic coupling, see:
Further information
M3: Coupled axes (Page 419)

19.6 Examples

Programming example

Program code	Comment
	; Leading spindle = master spindle = spindle 1
	; Following spindle = spindle 2
N05 M3 S3000 M2=4 S2=500	; Master spindle rotates at 3000 rpm
	; FS: 500 rpm
N10 COUPDEF (S2, S1, 1, 1, "No", "Dv")	; Def. of coupling, can also
	; be configured
N70 SPCON	; Include leading spindle in the position control
	; (setpoint value coupling).

Program code	Comment
N75 SPCON(2)	; Bring following spindle into closed-loop position control
N80 COUPON (S2, S1, 45)	; On-the-fly coupling to offset position = 45 degrees
N200 FA [S2] = 100	; Positioning speed = 100 degrees/min
N205 SPOS[2] = IC(-90)	; Traverse with 90° overlay in negative direction
N210 WAITC(S2, "Fine")	; Wait for "fine" synchronism
N212 G1 X.., Y.. F...	; Machining
N215 SPOS[2] = IC(180)	; Traverse with 180° overlay in positive direction
N220 G4 S50	; Dwell time = 50 revolutions
	; of master spindle
N225 FA [S2] = 0	; Activate configured velocity (MD)
N230 SPOS[2] = IC (-7200)	; 20 rev. with configured velocity
	; in neg. direction
N350 COUPOF (S2, S1)	; On-the-fly decoupling, S = S2 = 3000
N355 SPOSA[2] = 0	; Stop FS at zero degrees
N360 G0 X0 Y0	
N365 WAITS(2)	; Wait for spindle 2
N370 M5	; Stop FS
N375 M30	

19.7 Data lists

19.7.1 Machine data

19.7.1.1 NC-specific machine data

Number	Identifier: $MN_	Description
10000	AXCONF_MACHAX_NAME_TAB	Machine axis name

19.7.1.2 Channelspecific machine data

Number	Identifier: $MC_	Description
20070	AXCONF_MACHAX_USED	Machine axis number valid in channel
21300	COUPLE_AXIS_1	Definition of synchronous spindle pair
21310	COUPLING_MODE_1	Type of coupling in synchronous spindle mode

Number	Identifier: $MC_	Description
21320	COUPLE_BLOCK_CHANGE_CTRL_1	Block change behavior in synchronous spindle operation
21330	COUPLE_RESET_MODE_1	Coupling abort behavior
21340	COUPLE_IS_WRITE_PROT_1	Coupling parameters are write-protected

19.7.1.3 Axis/spindlespecific machine data

Number	Identifier: $MA_	Description
30455	MISK_FUNCTION_MASK	Axis functions
30550	AXCONF_ASSIGN_MASTER_CHAN	Reset position of channel for axis change
32200	POSCTRL_GAIN	Servo gain factor (K_V factor)
32400	AX_JERK_ENABLE	Axial jerk limitation
32410	AX_JERK_TIME	Time constant for axial jerk filter
32420	JOG_AND_POS_JERK_ENABLE	Initial setting for axial jerk limitation
32430	JOG_AND_POS_MAX_JERK	Axial jerk
32610	VELO_FFW_WEIGHT	Feedforward control factor for speed feedforward control
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
34080	REFP_MOVE_DIST	Reference point approach distance
34090	REFP_MOVE_DIST_CORR	Reference point offset
34100	REFP_SET_POS	Reference point value
35000	SPIND_ASSIGN_TO_MACHAX	Assignment of spindle to machine axis
37200	COUPLE_POS_TOL_COARSE	Threshold value for "Coarse synchronism"
37210	COUPLE_POS_TOL_FINE	Threshold value for "Fine synchronism"
37220	COUPLE_VELO_TOL_COARSE	Speed tolerance "coarse" between leading and following spindles
37230	COUPLE_VELO_TOL_FINE	Speed tolerance "fine" between leading and following spindles

19.7.2 Setting data

19.7.2.1 Channelspecific setting data

Number	Identifier: $SC_	Description
42300	COUPLE_RATIO_1	Transmission parameters for synchronous spindle operation

19.7.3 System variables

System variable	Description
$P_COUP_OFFS[following spindle]	Programmed offset of the synchronous spindle
$AA_COUP_OFFS[following spindle]	Position offset for synchronous spindle (setpoint)
$VA_COUP_OFFS[following spindle]	Position offset for synchronous spindle (actual value)

For exhaustive explanations of the system variables, see:
Further information
List Manual System Variables

Appendix

A

A.1 List of abbreviations

A	
O	Output
ADI4	(Analog drive interface for 4 axes)
AC	Adaptive Control
ALM	Active Line Module
ARM	Rotating induction motor
AS	Automation system
ASCII	American Standard Code for Information Interchange: American coding standard for the exchange of information
ASIC	Application-Specific Integrated Circuit: User switching circuit
ASUB	Asynchronous subprogram
AUXFU	Auxiliary function: Auxiliary function
STL	Statement List
UP	User Program

B	
OP	Operating Mode
BAG	Mode group
BCD	Binary Coded Decimals: Decimal numbers encoded in binary code
BERO	Contact-less proximity switch
BI	Binector Input
BICO	Binector Connector
BIN	BINary files: Binary files
BIOS	Basic Input Output System
BCS	Basic Coordinate System
BO	Binector Output
OPI	Operator Panel Interface

C	
CAD	Computer-Aided Design
CAM	Computer-Aided Manufacturing
CC	Compile Cycle: Compile cycles
CEC	Cross Error Compensation
CI	Connector Input
CF Card	Compact Flash Card

C	
CNC	Computerized Numerical Control: Computer-Supported Numerical Control
CO	Connector Output
CoL	Certificate of License
COM	Communication
CPA	Compiler Projecting Data: Configuring data of the compiler
CRT	Cathode Ray Tube: picture tube
CSB	Central Service Board: PLC module
CU	Control Unit
CP	Communication Processor
CPU	Central Processing Unit: 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

D	
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
DDS	Drive Data Set: Drive data set
DIN	Deutsche Industrie Norm
DIO	Data Input/Output: Data transfer display
DIR	Directory: 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: Differential revolver function (handwheel)
DRIVE-CLiQ	Drive Component Link with IQ
DRY	Dry Run: Dry run feedrate
DSB	Decoding Single Block: Decoding single block
DSC	Dynamic Servo Control / Dynamic Stiffness Control
DW	Data Word
DWORD	Double Word (currently 32 bits)

E	
I	Input
EES	Execution from External Storage
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
ENC	Encoder: Actual value encoder
EnDat	Encoder interface
EPROM	Erasable Programmable Read Only Memory: Erasable, electrically 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

F	
FB	Function Block (PLC)
FC	Function Call: Function Block (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
FIPO	Fine interpolator
FPU	Floating Point Unit: Floating Point Unit
CRC	Cutter Radius Compensation
FST	Feed Stop: Feedrate stop
FBD	Function Block Diagram (PLC programming method)
FW	Firmware

G	
GC	Global Control (PROFIBUS: Broadcast telegram)
GDIR	Global part program memory
GEO	Geometry, e.g. geometry axis
GIA	Gear Interpolation dAta: Gear interpolation data
GND	Signal Ground
GP	Basic program (PLC)
GS	Gear Stage
GSD	Device master file for describing a PROFIBUS slave

G	
GSDML	Generic Station Description Markup Language: XML-based description language for creating a GSD file
GUD	Global User Data: Global user data

H	
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

I	
IBN	Commissioning
ICA	Interpolatory compensation
IM	Interface Module: Interconnection module
IMR	Interface Module Receive: Interface module for receiving data
IMS	Interface Module Send: Interface module for sending data
INC	Increment: Increment
INI	Initializing Data: Initializing data
IPO	Interpolator
ISA	Industry Standard Architecture
ISO	International Standardization Organization

J	
JOG	Jogging: Setup mode

K	
K_V	Gain factor of control loop
K_p	Proportional gain
$K_Ü$	Transformation ratio
LAD	Ladder Diagram (PLC programming method)

L	
LAI	Logic Machine Axis Image: Logical machine axes image
LAN	Local Area Network
LCD	Liquid Crystal Display: Liquid crystal display
LED	Light Emitting Diode: Light-emitting diode
LF	Line Feed

L	
PMS	Position Measuring System
LR	Position controller
LSB	Least Significant Bit: Least significant bit
LUD	Local User Data: User data (local)

M	
MAC	Media Access Control
MAIN	Main program: Main program (OB1, PLC)
MB	Megabyte
MCI	Motion Control Interface
MCIS	Motion Control Information System
MCP	Machine Control Panel: Machine control panel
MD	Machine Data
MDA	Manual Data Automatic: Manual input
MDS	Motor Data Set: Motor data set
MSGW	Message Word
MCS	Machine Coordinate System
MM	Motor Module
MPF	Main Program File: Main program (NC)
MCP	Machine control panel

N	
NC	Numerical Control: Numerical control with block preparation, traversing range, etc.
NCU	Numerical Control Unit: NC hardware unit
NRK	Name for the operating system of the NC
IS	Interface Signal
NURBS	Non-Uniform Rational B-Spline
WO	Work Offset
NX	Numerical Extension: Axis expansion board

O	
OB	Organization block in the PLC
OEM	Original Equipment Manufacturer
OP	Operator Panel: Operating equipment
OPI	Operator Panel Interface: Interface for connection to the operator panel
OPT	Options: 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 control
PCMCIA	Personal Computer Memory Card International Association: Plug-in memory card standardization
PCU	PC Unit: PC box (computer unit)
PG	Programming device
PKE	Parameter identification: Part of a PIV
PIV	Parameter identification: Value (parameterizing 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: Positioning motor
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: Point-To-Point
PUD	Program global User Data: Program-global user variables
PZD	Process data: Process data part of a PPO

Q	
QEC	Quadrant Error Compensation

R	
RAM	Random Access Memory: Read/write memory
REF	REFerence point approach function
REPOS	REPOSition 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

R	
RP	R Parameter, arithmetic parameter, predefined user variable
RPA	R Parameter Active: Memory area in the NC for R parameter numbers
RPY	Roll Pitch Yaw: Rotation type of a coordinate system
RTLI	Rapid Traverse Linear Interpolation: Linear interpolation during rapid traverse motion
RTS	Request To Send: Control signal of serial data interfaces
RTCP	Real Time Control Protocol

S	
SA	Synchronized Action
SBC	Safe Brake Control: Safe Brake Control
SBL	Single Block: Single block
SBR	Subroutine: Subprogram (PLC)
SD	Setting Data
SDB	System Data Block
SEA	Setting Data Active: Identifier (file type) for setting data
SERUPRO	SEarch RUn by PROgram test: Block search, program test
SFB	System Function Block
SFC	System Function Call
SGE	Safety-related input
SGA	Safety-related output
SH	Safe standstill
SIM	Single in Line 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	Sub Routine File: Subprogram (NC)
PLC	Programmable Logic Controller
SRAM	Static RAM (non-volatile)
TNRC	Tool Nose Radius Compensation
SRM	Synchronous Rotary Motor
LEC	Leadscrew Error Compensation
SSI	Serial Synchronous Interface: Synchronous serial interface
SSL	Block search
STW	Control word
GWPS	Grinding Wheel Peripheral Speed
SW	Software
SYF	System Files: System files
SYNACT	SYNchronized ACTion: Synchronized Action

T	
TB	Terminal Board (SINAMICS)
TCP	Tool Center Point: Tool tip
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: 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

U	
UFR	User Frame: Work offset
SR	Subprogram
USB	Universal Serial Bus
UPS	Uninterruptible Power Supply

V	
VDI	Internal communication interface between NC and PLC
VDI	Verein Deutscher Ingenieure [Association of German Engineers]
VDE	Verband Deutscher Elektrotechniker [Association of German Electrical Engineers]
VI	Voltage Input
VO	Voltage Output
FDD	Feed Drive

W	
SAR	Smooth Approach and Retraction
WCS	Workpiece Coordinate System
T	Tool
TLC	Tool Length Compensation
WOP	Workshop-Oriented Programming
WPD	Workpiece Directory: Workpiece directory
TRC	Tool Radius Compensation
T	Tool
TO	Tool Offset
TM	Tool Management
TC	Tool change

X	
XML	Extensible Markup Language

Z	
WOA	Work Offset Active: Identifier for work offsets
ZSW	Status word (of drive)

Index

$

$A_OUT[n], 413
$AA_ACC, 178
$AA_ACT_INDEX_AX_POS_NO, 250
$AA_BRAKE_CONDB, 554
$AA_BRAKE_CONDM, 554
$AA_BRAKE_STATE, 554
$AA_COUP_ACT, 426, 869
$AA_COUP_CORR, 562
$AA_COUP_CORR_DIST, 563
$AA_COUP_OFFS, 869
$AA_ESR_ENABLE, 621
$AA_FGREF, 155
$AA_FGROUP, 155
$AA_FIX_POINT_ACT, 676
$AA_FIX_POINT_SELECTED, 676
$AA_IN_SYNC, 460
$AA_LEAD_P, 460
$AA_LEAD_P_TURN, 460
$AA_LEAD_SP, 460
$AA_LEAD_SV, 460
$AA_LEAD_V, 460
$AA_MOTEND, 179, 208
$AA_PROG_INDEX_AX_POS_NO, 250
$AA_S, 738
$AA_SCPAR, 180
$AA_SYNC, 460, 528
$AA_SYNCDIFF, 527
$AA_VLFCT, 830
$AA_VMAXB, 830
$AA_VMAXM, 830
$AC_FGROUP_MASK, 155
$AC_PATHACC, 358, 370
$AC_PATHJERK, 369, 370
$AC_RETPOINT, 231
$AC_SGEAR, 789, 808
$AC_SMAXACC, 829
$AC_SMAXACC_INFO, 829
$AC_SMAXVELO, 829
$AC_SMAXVELO_INFO, 829
$AC_SMINVELO, 829
$AC_SMINVELO_INFO, 829
$AC_SPIND_STATE, 829
$AN_ESR_TRIGGER, 621
$P_COUP_OFFS, 870
$P_FGROUP_MASK, 156
$P_GEAR, 808
$P_SEARCH_SGEAR, 789
$P_SGEAR, 789
$PA_FGREF, 155
$PA_FGROUP, 156
$SC_IS_SD_MAX_PATH_ACCEL, 356
$SC_IS_SD_MAX_PATH_JERK, 368
$SC_SD_MAX_PATH_ACCEL, 356
$SC_SD_MAX_PATH_JERK, 368
$TC_DPNT, 129
$VA_COUP_OFFS, 869
$VA_SYNCDIFF, 527
$VA_TORQUE_AT_LIMIT, 111
$VC_SGEAR, 788

A

ACC, 178, 817
Acceleration mode, 571
ACCLIMA[FA], 570
ACN, 816
ACP, 816
Activation methods, 856
Activation of coupling, 856
Actual value display, 463
Actual-value acquisition, 49
Actual-value correction, 51
Actual-value processing, 57
Adapting the motor/load ratios, 55
Alarm 10653, 262
ALF, 142
Angular offset POSFS, 857
Applied position difference, 83
Approach fixed point
  in JOG, 671
  with G75, 671
ASCALE, 234
ATRANS, 234
Autonomous singleaxis operations
  NCK reactions, 215
  PLC actions, 214
Axes
  for auxiliary movements, 185
Axis
  -interpolator, 192
Axis assignment, 440
Axis disable, 158
Axis interchange, 549
Axis interchange via PLC, 214

Axis types
For positioning axes, 189

B

Basic system, 39
Block change
Positioning axis type 1, 205
Positioning axis type 2, 206
Block change criterion, 512
Block-related limit (FOC), 111
BRISK, 351
BRISKA, 351

C

Cam signals
Hardware assignment, 412
linked output, 404
Minus, 412
Plus, 412
Separate output, 400
Timer-controlled output, 413
Center of circle
for circular travel in JOG, 690
CFC, 127
CFCIN, 127
CFTCP, 127
Channel, 188
Circle radius
for circular travel in JOG, 690
Circle segment machining, 686
Circular travel
in JOG, 683
Closed-loop control, 69
COARSEA, 202
Computational resolution, 34
Concurrent positioning axes
start from the PLC, 214
Constant cutting rate, 131
Constant speed, 132
Continuous duty, 647
Continuous manual travel, 643
Continuous operation, 643
Contour handwheel, 710
Control direction, 57
Control dynamics, 463
Convex thread, 146
Coordinate reference, 511
Coordinate system, 421
CORROF, 719
Coupled axes, 420
Coupled axis grouping, 420
Deactivating, 424
Definition and switch on, 424
Coupled motion, 419
Activating/deactivating, 422
Control system dynamics, 428
Distance-to-go, 423
Dynamics limit, 427
Interface signals, 425
Programming, 424
Coupled motion axis as leading axis, 421
coupling
Deactivating, 505
Define new, 865
Fixed configuration, 865
Coupling factor, 508
Counter, 508
Denominator, 509
Coupling module, 495
Creating, 500
Delete, 501
Coupling options, 850
Coupling reference, 510
Coupling rule, 495, 508
Coupling type, 459
Coupling types, 543
CP-BASIC, 492
CPBC, 512
CP-COMFORT, 492
CPDEF, 500
CPDEF+CPLA, 502
CPDEL, 501
CPDEL+CPLA, 504
CPFMOF, 519
CPFMON, 518
CPFMSON, 516
CPFPOS, 514
CPFRS, 511
CPLDEF, 502
CPLDEL, 503
CPLDEN, 509
CPLINSC, 525
CPLINTR, 524
CPLNUM, 508
CPLOF, 507
CPLON, 506
CPLOUTSC, 526
CPLOUTTR, 525
CPLPOS, 515
CPLSETVAL, 510
CPMALARM, 540

CPMBRAKE, 535
CPMPRT, 523
CPMRESET, 520
CPMSTART, 522
CPMVDI, 536
CPOF, 505
CPOF+CPFPOS, 520
CPON, 505
CPRES, 548
CPSETTYPE, 544
CPSYNCOP, 529
CPSYNCOP2, 532
CPSYNCOV, 530
CPSYNFIP, 529
CPSYNFIP2, 532
CPSYNFIV, 530
Curve segment, 430
Curve table, 509
Curve tables, 429
  Activation, 445
  Behavior in operating modes, 446
  Deactivation, 445
  Delete, 433
  Insufficient memory, 432
  Interface signals, 447
  Memory optimization, 432
  Memory type, 431
  Multiple use, 446
  Overwrite, 433
  Programming, 435
  Standard memory type, 434
  Tool radius compensation, 431
  Transformations, 451

D

DB 31, ...
  DBX1.4, 271
  DBX101.5, 270
  DBX2.1, 271
  DBX29.5, 271, 285
  DBX60.6, 835
  DBX60.7, 835
  DBX83.1, 749
DB10
  DBX100.0 - 4, 653
  DBX100.6, 654
  DBX100.7, 654
  DBX101.0 - 4, 653
  DBX101.6, 654
  DBX101.7, 654
  DBX102.0 - 4, 653
  DBX102.6, 654
  DBX102.7, 654
  DBX110.0 - 113.7, 412
  DBX114.0 - 117.7, 412
  DBX97.0 - 3, 653
  DBX98.0 - 3, 653
  DBX99.0 - 3, 653
DB11
  DB6.2, 633
  DBX(n*20+6).2, 696
  DBX0.1, 292, 294
  DBX0.2, 292, 294
  DBX0.7, 294, 295, 296
  DBX1.2, 292, 294
  DBX186.2, 633
  DBX26.2, 633
DB11, ...
  DBX5.2, 268
DB19
  DBX0.7, 697
  DBX20.7, 697
DB21, ...
  DBB194, 818
  DBB202, 818
  DBB4, 159
  DBB5, 159
  DBB68ff., 818
  DBX0.3, 425, 582, 710, 714, 716
  DBX0.6, 164, 709
  DBX1.0, 294
  DBX12.3, 158
  DBX15.0, 656
  DBX16.3, 158
  DBX19.0, 656
  DBX20.3, 158
  DBX23.0, 656
  DBX24.3, 714, 716
  DBX24.6, 164
  DBX29.0, 171
  DBX29.1, 171
  DBX29.2, 171
  DBX29.3, 171
  DBX30.0 - 2, 711
  DBX30.4, 712
  DBX30.6, 684
  DBX31.5, 656
  DBX318.6, 164
  DBX323.0, 656
  DBX327.0, 656
  DBX33.0, 295
  DBX33.3, 706
  DBX331.0, 656

DBX335.0, 657
DBX339.0, 657
DBX343.0, 657
DBX36.2, 293, 295
DBX377.4, 697, 699
DBX377.5, 697, 699
DBX377.6, 684
DBX39.5, 657
DBX4.3, 158
DBX40.5, 654
DBX40.7, 645, 649, 655
DBX43.0, 657
DBX46.5, 654
DBX46.7, 645, 649, 655
DBX49.0, 657
DBX52.5, 654
DBX52.7, 645, 649, 655
DBX55.0, 657
DBX6.0, 157
DBX6.1, 743, 783
DBX6.6, 159
DBX6.7, 159
DBX7.1, 881
DBX7.4, 785, 829, 880
DBX7.7, 294, 295, 296
DB21, ... DBX0.3, 712
DB21, ... DBX30.0 - 2, 712
DB21, ... DBX30.3, 712
DB21, ... DBX6.2, 713
DB21, ... DBX7.3, 712
DB21, ... DBX7.4, 712
DB21, …
DBX1.0, 272
DBX12.0 - 2, 653, 654, 705
DBX13.0 - 5, 648
DBX13.6, 644
DBX16.0 - 2, 705
DBX16.0 - 2, 653, 654
DBX17.0 - 5, 648
DBX17.6, 644
DBX20.0 - 2, 653, 654, 705
DBX21.0 - 5, 648
DBX21.6, 644
DBX320.0 - 2, 653, 654
DBX321.0 - 5, 648
DBX321.6, 644
DBX324.0 - 2, 653, 654
DBX325.0 - 5, 648
DBX325.6, 644
DBX328.0 - 2, 653, 654
DBX329.0 - 5, 648
DBX329.6, 644
DBX33.0, 268
DBX332.4, 655
DBX332.5, 655
DBX332.6, 645, 649, 655
DBX332.7, 645, 649, 655
DBX333.0 - 5, 649
DBX333.6, 645
DBX336.4, 655
DBX336.5, 655
DBX336.6, 646, 650, 656
DBX336.7, 646, 650, 656
DBX337.0 - 5, 649
DBX337.6, 645
DBX340.4, 655
DBX340.5, 655
DBX340.6, 646, 650, 656
DBX340.7, 646, 650, 656
DBX341.0 - 5, 649
DBX341.6, 645
DBX35.7, 696
DBX377.5, 696
DBX40.4, 654
DBX40.6, 645, 649, 655
DBX41.0 - 5, 649
DBX41.6, 645
DBX46.4, 654
DBX46.6, 645, 649, 655
DBX47.0 - 5, 649
DBX47.6, 645
DBX52.4, 654
DBX52.6, 645, 649, 655
DBX53.0 - 5, 649
DBX53.6, 645
DB31
DBX84.0, 823
DB31, ...
DBB0, 160
DBB1.5, 330
DBB1.6, 330
DBB101.4, 285, 286
DBB101.6, 285, 286
DBB101.7, 285, 286
DBB19, 161, 877, 885
DBB2.1, 330
DBB60.4, 316, 325, 327
DBB60.5, 316, 325, 327
DBB68ff., 818
DBX 99.1, 471
DBX0.0, 425
DBX0.1, 425
DBX0.2, 425
DBX0.3, 425

DBX0.4, 425
DBX0.5, 425
DBX0.6, 425
DBX0.7, 425
DBX1.1, 99
DBX1.2, 95
DBX1.3, 95, 425, 829, 877
DBX1.4, 280, 319, 320, 426, 874, 878
DBX1.5, 50, 271, 280, 426, 447, 463, 773, 879
DBX1.6, 50, 271, 280, 426, 447, 463, 773, 879
DBX1.7, 160, 161, 640
DBX100.2, 600
DBX101.2, 261, 279, 287
DBX101.3, 261, 279, 287
DBX101.4, 268, 279, 287
DBX101.5, 262, 268, 279, 281, 287
DBX101.6, 279, 287
DBX101.7, 279, 287
DBX103.0, 562
DBX103.4, 533
DBX103.5, 533
DBX12.4, 225
DBX12.7, 301, 303
DBX13.0 - 2, 672
DBX13.0-2, 673
DBX13.3, 678, 679
DBX16.0 - 2, 795
DBX16.0 - DBX16.2, 739, 779, 784, 785, 786, 790, 801
DBX16.3, 735, 739, 779, 784, 785, 786, 795, 797, 799, 801, 805
DBX16.4, 774, 880, 885
DBX16.5, 774, 880, 885
DBX16.7, 829, 880, 884
DBX17.0, 804
DBX17.4 - DBX17.5, 800
DBX17.6, 885
DBX18.4, 801
DBX18.5, 739, 779, 801, 829
DBX2.0, 411
DBX2.1, 95, 280, 319, 320, 425, 426, 800, 874, 878
DBX2.2, 704, 705, 738, 746, 785, 829, 879, 884
DBX2.4 - DBX2.7, 308
DBX20.1, 786
DBX21.7, 99
DBX26.4, 472, 475, 883
DBX28.0, 600
DBX28.1, 195
DBX28.2, 195
DBX28.3, 600
DBX28.4, 600
DBX28.5, 587
DBX28.6, 195, 587
DBX28.7, 587
DBX29.4, 269, 279, 285, 287
DBX29.5, 269, 279, 875
DBX3.1, 94, 98, 99
DBX3.2, 171
DBX3.3, 171
DBX3.4, 171
DBX3.5, 171
DBX30.0, 822
DBX30.1, 822
DBX30.2, 822
DBX30.3, 822
DBX30.4, 822
DBX31.4, 861, 870, 875, 878
DBX31.5, 859, 876
DBX31.6, 560
DBX31.7, 564
DBX4.0, 425
DBX4.1, 425
DBX4.2, 425
DBX4.3, 425, 469, 799, 829, 860, 879
DBX4.6, 292, 880
DBX4.7, 292, 880
DBX4.7/4.6, 264
DBX6.2, 195
DBX60.2, 833
DBX60.3, 833
DBX60.4, 280, 293, 294, 295, 296, 308, 314, 315, 318, 319, 320, 833, 857
DBX60.4 - DBX60.5, 791, 795, 799, 800, 801
DBX60.5, 280, 293, 294, 295, 296, 308, 314, 315, 318, 319, 320, 833, 857
DBX60.6, 196, 742, 749, 754, 755
DBX60.7, 196, 749, 754, 755, 804
DBX61.3, 319, 320, 875
DBX61.4, 319, 738, 739, 744, 779, 827
DBX61.5, 737, 795, 796, 800
DBX62.0, 411
DBX62.1, 706, 709
DBX62.4, 94, 98, 99
DBX62.5, 99, 102
DBX63.0, 194, 197
DBX63.1, 194
DBX63.2, 194, 196, 197
DBX64.5, 654
DBX64.6, 280
DBX64.7, 196, 280, 645, 649, 655
DBX65.0 - 5, 649
DBX65.6, 645
DBX67.0, 657

DBX69.0 - DBX69.2, 790
DBX7.0, 656
DBX7.7, 829
DBX71.4, 774
DBX71.5, 774
DBX74.4, 227
DBX75.0 - 2, 672
DBX75.3 - 5, 672
DBX75.6, 679
DBX75.7, 679
DBX76.6, 248
DBX82.0 - DBX82.2, 779, 781, 782, 784, 794, 795, 796, 800, 803
DBX82.3, 779, 781, 782, 795, 796, 797, 800, 803
DBX83.1, 738, 781, 783, 828, 829, 855
DBX83.2, 781, 783, 828
DBX83.5, 738, 739, 743, 744, 745, 827, 884
DBX83.7, 884
DBX84.4, 849
DBX84.5, 800, 804
DBX84.6, 735, 795, 796, 800, 803, 804
DBX84.7, 737, 779, 801
DBX85.5, 755
DBX98.0, 470, 471, 528, 558, 860, 862, 876
DBX98.1, 470, 471, 528, 558, 860, 862, 876
DBX98.4, 471, 854, 857, 884
DBX98.5, 472
DBX98.6, 472
DBX99.0, 426, 851
DBX99.1, 426, 851
DBX99.2, 563
DBX99.3, 472
DBX99.4, 460, 470, 471, 876
DB31, ... DBX1.1, 95
DB31, ... DBX1.4, 426
DB31, ... DBX62.5, 95
DB31, ... DBX96.5, 342
DB31, …
DBX4.0 - 2, 653, 654, 705
DBX5.0 - 5, 648
DBX5.6, 644
DBX64.4, 654
DBX64.6, 645, 649, 655
DBX9.3, 180
DB31, ... DBB20, 342
DB31, ... DBB21, 342
DB31, ... DBB92, 342
DB31, ... DBB93, 342
DB31, ... DBB94, 342
DB31, ... DBB95, 342
DB31, ... DBX2.1, 342
DB31, ... DBX61.3, 342
DB31, ... DBX96.5, 342
DB31, ...";"DBX2.2, 647
DC, 816
Definition
EG axis group, 473
Dependent coupled motion axis, 421
DIACYCOFA, 122
Diagnosing and optimizing utilization of resources, 447
DIAM90, 121
DIAM90/DIAM90A[AX], 117
DIAM90A, 122
DIAMCHAN, 122
DIAMCHANA, 122
DIAMCYCOF, 121
DIAMCYCOF/DIACYCOFA[AX], 117
Diameter programming
Axis-specific, 122
Channel-specific, 121
DIAMOF, 121
DIAMOFA, 122
DIAMON, 121
DIAMON/DIAMONA[AX], 116
DIAMONA, 122
Differential speed, 883
DILF, 142
Display resolution, 34
DITE, 140
DITS, 140
DRF, 713, 716
DRIVE, 386
DRIVEA, 386
Dry run feedrate, 163
Dynamic limits, 569
Dynamic programming in spindle/axis operations, 889
Dynamic response adaptation, 70

E

EG axis group
- activating, 474
- Deactivating, 478
- defining, 473
- delete, 479
EG axis groupings, 467
Electronic gearbox, 464, 465
System variables, 481
Encoder directly at the tool, 56
End-of-motion criterion
For single axes, 179
with block search, 211

Extended monitoring
Gantry axes, 262

F

FA, 156, 817
FB, 175
FC18, 212
FCUB, 172
Feed override, 211, 212
Feedrate
for chamfer/rounding, 173
Inverse-time (G93), 128
Linear (G94), 128
-override, 159
Path feedrate F, 126
Revolutional (G95), 129
Tooth, 129
types, 125
types (G93, G94, G95), 128
Feedrate disable
Channel-specific, 157
Feedrate override
Programmable, 163
Feedrates
Dry run feedrate, 163
Feedrate disable, 157
Feedrate/spindle stop, 157
Filter circuits
Switchable, 72
Switching over, 74
Filter type
Double moving average calculation, 376
Fine interpolation, 69
FINEA, 202
Fixed feedrates, 170
Fixed point positions, 674
FL, 127
FLIN, 172
FNORM, 172
FOC, 92
FOCOF, 92
FOCON, 92
Following axis overlay, 496
Following axis/spindle module, 502
Following spindle
Resynchronization, 876
Following spindle interpolator, 851
FPO, 172
FPRAOF, 816
FPRAON, 816
FRC, 173
FRCM, 173
FXS, 92
FXS-REPOS, 103
FXST, 92
FXSW, 92
FZ, 129

G

G25, 817
G26, 817
G33, 135
G331, 151
G332, 151
G335, 146
G336, 146
G63, 153, 769
G70, 45
G700, 45
G71, 45
G710, 45
G74, 296
G75, 671
G93, 128
G94, 128
G95, 129
G96, 131
G961, 131
G97, 132
G971, 132
Gantry axes
Differences in comparison with coupled motion, 283
Gear stage
in M70, 806
Manual entry, 784
Specification by PLC, 783
Gear stage change
Automatic, 835
Gear stages, 775

H

Handwheel
Assignment, 652
Distance specification, 658
Path definition, 710
Selection of HMI, 653
Traversal in JOG, 651
Velocity specification, 658, 710

Handwheel connection
Ethernet, 725
Handwheel override in AUTOMATIC mode
Path definition, 704
Velocity override, 704
Hirth gearing, 247

I

IC, 816
Incremental manual travel, 646
Incremental traversing, 650
Independent coupled motion axis, 421
Indexing axes
Coded position, 248
Programming, 248
Indexing axis
System of units, 243
Indexing positions
Number, 243
Indexing positions table, 242
Infeed, 579
Input resolution, 34
Intermediate gear, 56, 792
Interpolation, 463
Interpolation functions, 217
Interpolator
path, 192
Shaft, 192
Inverse-time feedrate (G93), 128
IPOBRKA, 202
IPOENDA, 202
IR, 146
IS feedrate stop / spindle stop, 861

J

Jerk
-increase, velocity-dependent jerk, 372
JOG
Approach fixed point, 671
Jog mode, 643, 647
JOG retract, 693
JR, 146

K

Keywords, 497
KR, 146

L

Leading axes
Defining, 502
Delete, 503
Leading axis
Activating, 506
Switch off, 507
LFOF, 142
LFON, 142
LFPOS, 142
LFTXT, 142
LFWP, 142
Limit
velocity, for path axes, 127
LIMS, 817
Linear feedrate (G94), 129
Logging, 463

M

M19, 741, 816
M3, 815
M4, 815
M40, 818, 835
M41, 818
M42, 818
M43, 818
M44, 818
M45, 818
M5, 815
M70, 816
Machine zero, 291
Main run
-axes, 178
Master value
-coupling, 453
-object, 453
Offset, 524
Scaling, 524
Master value coupling
Behavior in operating modes, 461
Interface signals, 463
Switch off, 458
Switch on, 457
Master-slave changeover with G96, G961, 134
Maximum axis acceleration, 570
Maximum axis velocity, 570
MD10050, 331
MD10070, 331

MD10192, 793
MD10200, 32, 33, 35, 43
MD10210, 32, 33, 35, 65, 67, 235
MD10220, 37
MD10230, 37
MD10240, 39, 41, 47, 181
MD10250, 42
MD10260, 42, 45, 409
MD10270, 43, 243, 409
MD10290, 44
MD10292, 44
MD10450, 409
MD10460, 410, 417
MD10461, 410, 416
MD10470, 412
MD10471, 412
MD10472, 412
MD10473, 412
MD1048, 415
MD10480, 413, 415
MD10485, 407, 414, 415
MD10651, 146
MD10704, 164
MD10710, 133, 141
MD10720, 695
MD10721, 695
MD10722, 214
MD10735, 674, 680, 681, 689, 695, 698
MD10900, 243
MD10910, 242
MD10920, 243
MD10930, 242
MD10940, 244
MD1103, 106
MD1104, 106
MD1105, 106
MD11220, 43
MD11300, 292, 651
MD11310, 661
MD11320, 652, 658
MD11322, 711
MD11324, 660
MD11330, 650, 659, 706
MD11346, 658, 673, 680, 710
MD11410, 136, 422, 541, 779
MD11410 $MN_SUPPRESS_ALARM_MASK Bit31, 470
MD11550, 152
MD11660 $MN_NUM_EG, 467
MD12000, 161
MD12010, 161
MD12020, 159
MD12030, 160, 712
MD12040, 159
MD12050, 160
MD12060, 161
MD12070, 162
MD12080, 162
MD12082, 160
MD12090, 152
MD12100, 162
MD12200, 171
MD12202, 171
MD12204, 171
MD1230/1231, 106
MD17900, 660
MD18400, 430, 433
MD18402, 430, 433
MD18403, 430, 433
MD18404, 430, 433
MD18406, 430, 433
MD18408, 430, 433
MD18409, 430, 433
MD18410, 430, 433
MD18450, 495
MD18452, 495
MD18960, 200, 350, 366
MD20070, 855
MD20090, 815
MD20092, 811
MD20100, 118, 132, 134
MD20110, 121, 423, 456, 458, 461, 462, 699
MD20112, 121, 423, 456, 458, 461, 462
MD20150, 44, 120, 181, 351, 366, 386, 699
MD20151, 699
MD20172, 172
MD20200, 174
MD20201, 174
MD20360, 119, 690
MD20500, 352, 353
MD20600, 365
MD20602, 361, 362
MD20610, 355
MD20620, 659
MD20621, 659
MD20624, 120, 662, 667
MD20700, 326, 700
MD20750, 133
MD20850, 741
MD20900, 434, 440
MD20905, 434, 437, 439, 448
MD21150, 236
MD21155, 639, 669
MD21158, 389, 642, 670

MD21159, 389, 642, 670
MD21160, 640, 669
MD21165, 639, 669
MD21166, 388, 641, 670
MD21168, 389, 642, 670
MD21220, 166
MD21230, 166
MD21300, 851, 886
MD21310, 886
MD21320, 868, 887
MD21330, 887
MD21340, 853, 887
MD21380, 628
MD21381, 628
MD22000, 741
MD22010, 741
MD22020, 741
MD22030, 741
MD22240, 157
MD22410, 157, 163, 181, 572
MD27100, 120
MD30100, 51
MD30110, 52, 341
MD30120, 52
MD30130, 51, 52, 454, 457, 459
MD30132, 454
MD30200, 49, 59
MD30210, 52, 60
MD30220, 52, 60
MD30230, 53, 60, 341
MD30240, 50, 51, 53, 60
MD30242, 51, 54, 60, 317
MD30244, 60
MD30250, 60, 322
MD30260, 60
MD30270, 60
MD30300, 59, 61, 65, 67, 181, 227, 234
MD30310, 59, 224, 227, 228, 235
MD30320, 59, 225, 235
MD30330, 59, 229, 230, 315
MD30340, 59, 224, 225, 229, 230, 316
MD30350, 51
MD30450, 282
MD30455, 316, 555, 561, 817, 861, 873, 889
MD30460, 118, 122, 213, 216
MD30500, 242, 247
MD30503, 225
MD30505, 247
MD30550, 293, 855
MD30552, 191
MD30600, 674
MD31000, 61, 65, 67
MD31010, 61
MD31020, 61, 65, 67, 331
MD31025, 65
MD31030, 59, 61, 65
MD31040, 61, 65, 67, 83, 751, 800
MD31044, 56, 61, 792
MD31050, 56, 61, 62, 65, 67, 791, 799
MD31060, 56, 61, 62, 65, 67, 791
MD31064, 56, 59, 792
MD31066, 56, 59, 792
MD31070, 61, 65, 67
MD31080, 61, 65, 67
MD31090, 44, 650, 652, 659, 706, 711
MD31122, 772
MD31123, 772
MD31200, 44
MD32000, 31, 59, 127, 569, 707, 712, 833
MD32000 $MA_MAX_AX_VELO, 472
MD32010, 640
MD32020, 639, 660, 667
MD32040, 200, 241, 640, 662
MD32050, 200, 241, 639, 662
MD32060, 177, 191, 199, 709
MD32080, 659
MD32084, 662, 712
MD32090, 713, 716
MD32100, 58, 274
MD32200, 70, 275, 887
MD32210, 85
MD32220, 85
MD32250, 58
MD32300, 177, 178, 200, 350, 359, 360, 361, 362, 388, 569, 641, 712, 723
$MA_MAX_AX_ACCEL, 472
MD32300 $MA_MAX_AX_ACCEL, 472
MD32310, 360, 361
MD32320, 178
MD32400, 275, 377, 888
MD32402, 275, 377, 888
MD32407, 378
MD32408, 378
MD32409, 378
MD32410, 275, 377, 888
MD32411, 377
MD32412, 377
MD32420, 275, 352, 366, 367, 388, 641, 888
MD32430, 200, 275, 366, 388, 641, 888
MD32431, 200, 365, 371
MD32432, 372
MD32433, 360
MD32434, 359, 360, 371
MD32435, 371

MD32439, 373
MD32610, 275, 887
MD32620, 78, 275, 873, 887
MD32630, 78
MD32640, 80
MD32650, 275, 887
MD32711, 45
MD32800, 275, 887
MD32810, 78, 275, 276, 887
MD32890, 81, 82
MD32900, 70, 276
MD32910, 70, 80, 276, 888
MD32950, 83
MD33000, 275
MD34000, 305, 315
MD34010, 292, 301, 303, 304, 313, 324
MD34020, 301, 304
MD34040, 266, 303, 304, 305, 313, 773
MD34050, 303
MD34060, 306, 314, 315, 753
MD34070, 266, 307, 314
MD34080, 270, 276, 286, 307, 857, 890
MD34090, 60, 270, 276, 286, 307, 310, 311, 323, 325, 326, 857, 890
MD34092, 305
MD34093, 306
MD34100, 264, 266, 268, 270, 272, 287, 308, 314, 324, 326, 857, 890
MD34102, 316, 317
MD34104, 319
MD34110, 272, 294
MD34200, 311, 317, 319, 324, 327, 773
MD34210, 321, 322, 323, 324, 325, 326, 695
MD34300, 313
MD34320, 60, 61, 310
MD34330, 266, 270, 272, 328
MD34990, 127
MD35000, 811, 818, 851, 855
MD35010, 776, 780, 795, 800, 805
MD35012, 775, 800, 803
MD35014, 806
MD35020, 762
MD35030, 762
MD35035, 742, 807, 810
MD35040, 738
MD35090, 775
MD35092, 780
MD3510, 804
MD35100, 31, 831
MD35110, 775, 836
MD35112, 780
MD35120, 775, 836
MD35122, 780
MD35130, 775, 781, 783, 829
MD35135, 775
MD35140, 134, 135, 775, 781, 783, 828
MD35150, 303, 743, 745, 828, 831
MD35200, 737, 745, 748, 752, 775, 804
MD35210, 737, 745, 748, 749, 753, 754, 775, 804
MD35212, 780
MD35220, 380, 384, 889
MD35230, 380, 384, 889
MD35240, 383
MD35242, 384, 889
MD35300, 331, 745, 747, 748, 753, 754, 775, 804, 817
MD35310, 775, 800, 803
MD35350, 753, 803
MD35400, 794, 796, 797
MD35410, 794, 796, 797
MD35430, 794, 796
MD35500, 739, 743
MD35510, 743, 827
MD35550, 775, 781
MD35590, 776, 790
MD36000, 750, 754, 755, 834, 835
MD36010, 749, 754, 755, 834, 835
MD36012, 271
MD36030, 271
MD36040, 112
MD36042, 112
MD36060, 738, 744, 827
MD36100, 226, 227
MD36110, 226, 227
MD36120, 226
MD36130, 226
MD36210, 58
MD36300, 330, 331
MD36302, 747
MD36500, 50, 272
MD36510, 50
MD36600, 557
MD36933, 663
MD37002, 99
MD37010, 94
MD37012, 99
MD37020, 96, 106
MD37030, 95
MD37040, 95
MD37060, 94, 98, 99
MD37080, 110
MD37100, 276, 284
MD37110, 261, 266, 277, 284, 287, 288
MD37120, 261, 274, 284, 287, 288

MD37130, 261, 274, 284, 288
MD37135, 262
MD37140, 264, 269, 281, 282
MD37150, 262, 270
MD37160, 447, 456
MD37200, 529, 864, 887
MD37200
$MA_COUPLE_POS_TOL_COARSE, 470, 482
MD37200
$MN_COUPLE_POS_TOL_COARSE, 470, 471
MD37202, 531
MD37210, 529, 864, 887
MD37210 $MA COUPLE_POS_TOL_FINE, 471
MD37210 $MA_COUPLE_POS_TOL_FINE, 470, 482
MD37212, 532
MD37220, 529, 864, 887
MD37230, 529, 864, 890
MD37500, 198, 628
MD37510, 197, 198
MD37511, 197, 198
MD37550 $MA_EG_VEL_WARNING, 472
MD37560 $MA_EG_ACC_TOL, 472
MD38020, 378
MD43108, 488
MD7200, 890
MD9004, 35
MD9011, 35
Mean value generation
Double, 376
Measuring systems, 50, 291
MELDW
Bit9, 622
Memory configuration
Curve tables, 430
Generic coupling, 495
Minus
-output cam, 399
Modal activation (FOCON/FOCOF), 110
Modulo 360, 223
Modulo leading axis, 446
Monitoring
Synchronous operation, 527
Motion behavior, 217
Motor/load gear, 56

N

Nonacknowledged gear stage change, 793

O

Offset
of a lead value, 524
Operating range limits
for circular travel in JOG, 688
OS, 584
OSB, 586
OSCILL, 592
Oscillating, 579
asynchronous, 579
continuous infeed, 579
with synchronized actions, 602
Oscillating axis, 579
OSCTRL, 585, 586
OSE, 586
OSNSC, 586
OSP, 584
OST, 585
Output cam
-pair, 399
-positions, 408
-range, 399
-signals, 399
OVR, 163
OVRA, 163, 817

P

p0888, 616, 618, 619
p0891, 617, 618
p0892, 617, 618
p0893, 618
p1240, 619
p1248, 620
p2161, 620
Parameter set
For axes, 76
Parking, 296
Path
feedrate F, 126
-interpolator,
velocity, maximum, 127
Path definition using handwheel, 710
Path feedrate, 32
PCOF, 816
Permanently configured coupling, 850
Phase filter, 81
Physical quantities, 36
PI controller, 571

PLC axes, 212
PLC axis
axes under exclusive PLC control, 213
permanently assigned PLC axis, 213
start via FC18, 215
PLC-controlled axis
Control response to MD30460 bits 6 and 7, 216
Control system response, 216
Plus
-output cam, 399
POLF, 142
POLFMASK, 142
POLFMLIN, 142
Position control loop, 69
Position offset
In synchronous spindles, 875
Position switching signals, 399
Lead/delay times, 410
Positioning accuracy, 34
Positioning axes, 33, 186
Axis types, 189
Axis-specific signals, 213
Channel-specific signals, 212
Concurrent, 191, 212
Dry run feedrate, 217
Maximum number, 201
Tool offset, 203
Types, 189
Positioning axis dynamic response, 200
Position-time cams, 416
POSP, 592
PRESETON, 296

R

Radius-related data, 117
Rapid stop, 557
Rapid traverse
-override, 159
Reaction to Stop, 456
Read offset, 861
Reference axis
for G96 / G961 / G962, 115
Reference point approach, 715, 718
Referencing
in rotary absolute encoders, 328
with incremental measurement system, 299
Referencing methods, 291
replacement zero mark, 328
REPOS offset, 102
Response to setpoint changes, 887
Restrictions, 439
Resynchronization, 876
Retract, 615
Retraction
Direction for thread cutting, 143
Reversal points, 579
Revolutional feedrate (G95), 129
Rotary axes
Absolute dimension programming, 232
Absolute programming, 229
Axis addresses, 221
Commissioning, 234
Feedrate, 222
Incremental programming, 231, 233
Mirroring, 236
Modulo 360, 223
Modulo conversion, 232
Software limit switch, 236
Units of measurement, 222

S

S..., 815
Scaling
of a lead value, 524
SCC, 121
SCPARA, 180
SD41010, 650, 659, 706
SD41050, 240, 646
SD41100, 130, 200, 223, 241, 638, 662
SD41110, 639, 660
SD41120, 639
SD41130, 235, 639, 660
SD41200, 639, 667
SD41500, 408
SD41501, 408
SD41502, 408
SD41503, 408
SD41504, 408
SD41505, 408
SD41506, 408
SD41507, 408
SD41520, 410, 417
SD41521, 410, 416
SD41522, 410
SD41523, 410
SD41524, 410
SD41525, 410
SD41526, 411, 417
SD41527, 411, 416
SD42010, 139, 141
SD42100, 164, 217, 581, 709
SD42101, 165, 217, 581

SD42300, 852, 887
SD42500, 355, 356
SD42502, 356
SD42510, 367, 368
SD42512, 367, 368
SD42600, 130, 200, 203, 241, 662, 825
SD42690, 684, 690
SD42691, 684, 690
SD42692, 684, 686, 688, 691
SD42693, 686, 691
SD42694, 686, 691
SD42800, 812
SD43100, 459
SD43102, 454
SD43104, 454
SD43106, 454
SD43108, 454
SD43200, 821, 823
SD43202, 822, 823
SD43206, 822, 824
SD43210, 134, 135
SD43220, 133
SD43230, 133
SD43235, 832
SD43240, 741, 816
SD43250, 741
SD43300, 130, 200, 203, 241, 662, 825
SD43320, 678, 681
SD43400, 226
SD43410, 226
SD43420, 226
SD43430, 226
SD43500, 106
SD43510, 106
SD43520, 106
SD43600, 211
SD43770, 586
SD43790, 586
Separate following spindle interpolator, 851
SERUPRO ASUP, 102
Servo gain factor, 70
SETMS, 815
Setpoint output, 49
Setpoint system, 49
Several transverse axes
  Axis interchange, 122
Shutting down, 615
Simulated master value, 453
Simulation axes, 50
Single axes
  Applications, 195
  Axis control by PLC, 193
  Extended retract numerically controlled, 198
  Extended stop numerically controlled, 197
Single block
  Positioning axis type 1, 217
  Positioning axis type 2, 217
  Positioning axis type 3, 217
Single-axis dynamic response, 176
SOFT, 365
SOFTA, 367
Software
  -output cam, 399
Sparking-out strokes, 579
SPCOF, 816
SPCON, 816
Specify gear stage, 777
Speed
  feedforward control, 77
Speed control loop, 69
Speed setpoint adjustment, 58
Speed setpoint output, 57
SPI, 818
Spindle
  -gear stage 0, 786
  in master value coupling, 456
  -override, 159, 161
  -setpoint speed, 828
  speed limits, 829
  -speed, maximum, 829
  -speed, minimum, 828
Spindle disable, 158
Spindle speed, 33
Spindle speed limitation with G96, G961, 133
SPOS, 739, 815
SPOSA, 816
Star/delta switchover with FC17, 786
Starting value, 440
Status of coupling, 426, 463
SVC, 815
Switching accuracy
  of the cam signals, 413
Synchronism
  Correction, 562
Synchronism difference, 470
  - scanning, 472
Synchronization, 571
  Following axis, 465
Synchronization mode, 516
Synchronized state reached, 860
Synchronous mode, 849
  Deactivate, 868
  Knee-shaped acceleration characteristic, 889

Synchronous operation
Deviation, 558
-difference, 527
-monitoring, 527
Synchronous operation monitoring
stage 2, 531
Synchronous position
Following axis, 514
Leading axis, 515
Synchronous spindle
Position offset, 875
System of units, 39, 45
System variables, 499

T

Tapping
With compensating chuck, 769
Temporary curve table, 432
Thread
cutting, 142
Thread cutting G33, 135
Tolerance
-factor for spindle speed, 828
Tooth feedrate, 129
TRAANG, 451
TRAILOF, 424
TRAILON, 424
TRANS, 234
Transverse axis, 115
Dimensions, 121
Travel command, 655
Travel request, 654
Travel to fixed stop, 91
Block search, 101
Contour monitoring, 105
Deselection, 99
Fixed stop is reached, 95
Function abort, 104
Monitoring window, 96
Positioning axes, 105
Selection, 94
Traversing keys, 643
Traversing movement release, 328
Traversing range limits, 225
Traversing ranges, 33

U

User-defined coupling, 850

V

Velocities, 31
VELOLIM, 817
VELOLIMA[FA], 570
Vertical axes, 105
Virtual leading axis, 453

W

WAITC, 513
WAITP, 203
Oscillating axis, 591
WAITS, 817