

SIEMENS

SINUMERIK

SINUMERIK ONE Monitoring and compensating

Function Manual

Introduction	1
Fundamental safety instructions	2
Geometric machine modeling	3
Collision avoidance, internal	4
Protection areas	5
Axis monitoring functions	6
Contour tunnel monitoring (option)	7
Compensations	8
Appendix	A

Valid for

control
SINUMERIK ONE

CNC software version 6.22

07/2023

A5E48053702B AG

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.

Table of contents

1	Introduction	9
1.1	About SINUMERIK	9
1.2	About this documentation	9
1.3	Documentation on the internet	12
1.3.1	Documentation overview SINUMERIK ONE	12
1.3.2	Documentation overview SINUMERIK operator components	12
1.4	Feedback on the technical documentation	13
1.5	mySupport documentation	13
1.6	Service and Support.....	14
1.7	Using OpenSSL	15
1.8	Compliance with the General Data Protection Regulation	15
2	Fundamental safety instructions.....	17
2.1	General safety instructions.....	17
2.2	Warranty and liability for application examples	17
2.3	Security information	17
3	Geometric machine modeling	19
3.1	Function description	19
3.1.1	Features	19
3.1.2	Automatic tool protection areas	22
3.2	Commissioning	25
3.2.1	General	25
3.2.1.1	Overview	25
3.2.1.2	Structure of the system variables.....	25
3.2.1.3	Color chart.....	27
3.2.2	Machine data	28
3.2.2.1	Maximum number of protection areas.....	28
3.2.2.2	Maximum number of protection area elements for machine protection areas.....	28
3.2.2.3	Maximum number of protection area elements for automatic tool protection areas.....	28
3.2.2.4	Maximum number of NC/PLC interface signals for the preactivation of protection areas.....	28
3.2.2.5	Maximum number of triangles for machine protection areas.....	29
3.2.2.6	Maximum number of triangles for automatic tool protection areas.....	29
3.2.2.7	Creation mode for automatic tool protection areas	29
3.2.3	System variables: Protection areas	30
3.2.3.1	Overview	30
3.2.3.2	\$NP_PROT_NAME.....	30
3.2.3.3	\$NP_CHAIN_ELEM.....	31
3.2.3.4	\$NP_PROT_TYPE	32
3.2.3.5	\$NP_1ST_PROT	33
3.2.3.6	\$NP_PROT_COLOR.....	34

3.2.3.7	\$NP_PROT_D_LEVEL	36
3.2.3.8	\$NP_BIT_NO	37
3.2.3.9	\$NP_INIT_STAT	38
3.2.3.10	\$NP_INDEX	39
3.2.4	System variables: Protection area elements for machine protection areas	41
3.2.4.1	Overview	41
3.2.4.2	\$NP_NAME	42
3.2.4.3	\$NP_NEXT	43
3.2.4.4	\$NP_NEXTP	44
3.2.4.5	\$NP_COLOR	46
3.2.4.6	\$NP_D_LEVEL	47
3.2.4.7	\$NP_USAGE	48
3.2.4.8	\$NP_TYPE	49
3.2.4.9	\$NP_FILENAME	54
3.2.4.10	\$NP_PARA	59
3.2.4.11	\$NP_OFF	60
3.2.4.12	\$NP_DIR	61
3.2.4.13	\$NP_ANG	63
3.2.5	System variables: Protection area elements for automatic tool protection areas	64
3.2.6	Boundary conditions	65
4	Collision avoidance, internal	67
4.1	Function description	67
4.1.1	Options	67
4.1.2	Characteristics	67
4.1.3	Reaction of the control to a risk of collision	69
4.1.4	Status diagram protection area	72
4.1.5	Tools	73
4.1.6	Boundary conditions	76
4.2	Commissioning	78
4.2.1	General	78
4.2.1.1	Overview	78
4.2.1.2	Structure of the system variables	78
4.2.2	Option data	79
4.2.3	Machine data	80
4.2.3.1	Collision tolerance	80
4.2.3.2	Safety clearance	80
4.2.3.3	Maximum memory space	80
4.2.3.4	Maximum number of collision pairs	81
4.2.3.5	Length of the preview	81
4.2.3.6	Length of the braking the ramp	82
4.2.3.7	Protection levels for collision avoidance On/Off	82
4.2.4	System variables	83
4.2.4.1	Overview	83
4.2.4.2	\$NP_COLL_PAIR	83
4.2.4.3	\$NP_SAFETY_DIST	85
4.2.5	Extend system variables	86
4.2.5.1	Overview	86
4.2.5.2	Status data	86
4.2.5.3	Braking distance estimations	87
4.3	Programming	88

4.3.1	Check for collision pair (COLLPAIR)	88
4.3.2	Request recalculation of the machine model of the collision avoidance (PROTA)	88
4.3.3	Setting the protection zone status (PROTS)	89
4.3.4	Determining the clearance of two protection zones (PROTD)	90
4.4	Example	92
4.4.1	Specifications	92
4.4.2	Part program of the machine model	96
5	Protection areas	105
5.1	Function	105
5.2	Commissioning	112
5.2.1	Machine data	112
5.3	Programming	113
5.3.1	Defining protection zones (CPROTDEF, NPROTDEF)	113
5.3.2	Activating/deactivating protection zones (CPROT, NPROT)	116
5.3.3	Checking for protection zone violation, working area limitation and software limit switches (CALCPOSI)	120
5.4	Special situations	129
5.4.1	Temporary enabling of protection zones	129
5.4.2	Behavior in the AUTOMATIC and MDI modes	130
5.4.3	Behavior in JOG mode	131
5.5	Boundary conditions	134
5.6	Example	135
5.6.1	Protection zone on a lathe	135
5.6.2	Protection zone definition in the part program	136
5.6.3	Protection zone definition with system variables	137
5.6.4	Activating protection zones	144
6	Axis monitoring functions	145
6.1	Contour monitoring	145
6.1.1	Contour error	145
6.1.2	Following-error monitoring	146
6.2	Positioning, zero speed and clamping monitoring	148
6.2.1	Correlation between positioning, zero-speed and clamping monitoring	148
6.2.2	Positioning monitoring	149
6.2.3	Zero-speed monitoring	150
6.2.4	Parameter set-dependent exact stop and standstill tolerance	151
6.2.5	Clamping monitoring	152
6.2.5.1	Function	152
6.2.5.2	Machine data	152
6.2.5.3	NC/PLC interface signals	153
6.2.5.4	Fault responses	154
6.2.5.5	"Automatic stop to release the clamping" clamping function	154
6.2.5.6	"Time-optimized release of the clamping" clamping function	156
6.2.5.7	"Automatic stop to set the clamping" clamping function	157
6.2.5.8	Supplementary conditions	159
6.3	Speed-setpoint monitoring	162
6.4	Actual-velocity monitoring	163

6.5	Measuring system monitoring	164
6.5.1	Encoder-limit-frequency monitoring	166
6.5.2	Plausibility check for absolute encoders	167
6.5.3	Customized error reactions.....	170
6.6	Limit switch monitoring	172
6.6.1	Hardware limit switch	172
6.6.2	Software limit switch	173
6.7	Working area limitation monitoring	176
6.7.1	General	176
6.7.2	Working area limitation in BCS	178
6.7.3	Working area limitation in WCS/SZS	180
6.7.4	Example: Working area limitation in WCS/SZS	183
6.8	Parking a machine axis.....	186
6.9	Parking the passive position measuring system.....	188
6.9.1	Function	188
6.9.2	Supplementary conditions	192
6.9.3	Example: Changing an attachment head for a direct position measuring system	193
6.9.4	Example: Changing an attachment head for two direct position measuring systems	197
6.9.5	Example: Measuring system switchover when encoders are missing in certain parts of the range.....	201
6.10	Switching over encoder data sets	204
7	Contour tunnel monitoring (option).....	209
7.1	Function	209
7.2	Commissioning	210
7.2.1	Parameterization.....	210
7.3	Restrictions	212
8	Compensations	213
8.1	Introduction.....	213
8.2	Temperature compensation.....	214
8.2.1	Function	214
8.2.2	Commissioning	217
8.2.3	Example	218
8.2.3.1	Commissioning the temperature compensation for the Z axis of a lathe	218
8.3	Backlash compensation.....	221
8.3.1	Mechanical backlash compensation.....	221
8.3.1.1	Function	221
8.3.1.2	Commissioning: Axis-specific machine data	222
8.3.2	Dynamic backlash compensation.....	224
8.3.2.1	Function	224
8.3.2.2	Commissioning: Axis-specific machine data	225
8.3.3	Dual position feedback.....	226
8.3.3.1	Commissioning: Axis-specific machine data	227
8.3.3.2	Supplementary conditions	227
8.4	Nodding compensation.....	228
8.4.1	Description of the function.....	228

8.4.1.1	Options	228
8.4.1.2	Features	228
8.4.1.3	Position-dependent compliance factor.....	230
8.4.1.4	Compensation relationships	231
8.4.2	Commissioning: Machine data	231
8.4.2.1	Overview	231
8.4.2.2	Time constant for smoothing nodding compensation values.....	232
8.4.2.3	Accelerating axis.....	233
8.4.2.4	Adaptation axis.....	234
8.4.2.5	Number of positions of the adaptation characteristic	235
8.4.2.6	Positions of the adaptation characteristic.....	236
8.4.2.7	Compliance factors	237
8.4.2.8	Function diagram.....	239
8.5	Interpolatory compensation (IC) General properties.....	240
8.6	IC: Leadscrew error and measuring system error compensation.....	242
8.6.1	Function	242
8.6.2	Commissioning	243
8.6.3	Example	246
8.7	IC: Sag and angularity error compensation	247
8.7.1	General information.....	247
8.7.2	Commissioning: Machine data	249
8.7.3	Commissioning: Setting data.....	251
8.7.4	Commissioning: System variables.....	251
8.7.5	Commissioning: Basic procedure	255
8.7.6	Commissioning: Overview diagram	257
8.7.7	Example 1: Sag compensation.....	258
8.7.8	Example 2: Compensation with table multiplication.....	259
8.7.9	Example 3: 2-dimensional array of compensation values	260
8.8	IC: Direction-dependent leadscrew error compensation	266
8.8.1	Description of functions	266
8.8.2	Commissioning	266
8.8.3	Example	269
8.9	IC: Multi Error Compensation (MEC)	272
8.9.1	Preconditions.....	272
8.9.2	Function description	273
8.9.3	Commissioning: Configuring the correction table.....	274
8.9.4	Commissioning: Axis-specific configuring.....	275
8.9.5	Commissioning: Activating the correction table	276
8.9.6	Commissioning: Defining the correction tables	276
8.9.7	Example	278
8.9.8	Constraints	279
8.10	IC: Cylinder error compensation	280
8.10.1	Function	280
8.10.2	Commissioning	280
8.10.3	Examples	285
8.11	IC: Constraints	288
8.12	Dynamic feedforward control (following error compensation)	290
8.12.1	General properties	290

8.12.2	Speed feedforward control	292
8.12.3	Torque feedforward control.....	294
8.12.4	Dynamic response adaptation	295
8.12.5	Forward feed control for command- and PLC axes.....	296
8.12.6	Secondary conditions.....	298
8.13	Friction compensation overview	299
8.14	Friction compensation with a constant compensation value	299
8.14.1	Description of functions	299
8.14.2	Commissioning	300
8.14.2.1	Circularity test.....	301
8.14.3	Supplementary conditions	306
8.15	Friction compensation with adaptive characteristic	306
8.15.1	Description of functions	306
8.15.2	commissioning	307
8.15.3	Supplementary conditions	310
8.16	Friction compensation with adaptive characteristics	310
8.16.1	Description of the function.....	310
8.16.2	Commissioning	312
8.16.2.1	Activating the function	312
8.16.2.2	Commissioning functions of the SINUMERIK Operate user interface	313
8.16.2.3	Parameterization of the acceleration at the characteristic interpolation points.....	315
8.16.2.4	Velocity setpoint pulse	317
8.16.2.5	Torque setpoint pulse.....	319
8.16.2.6	Velocity threshold for triggering the injection pulses.....	321
8.17	Compensation functions for suspended axes	322
8.17.1	Electronic counterweight	322
8.17.2	Special function: Reboot delay	323
A	Appendix.....	327
A.1	List of abbreviations.....	327
	Index	333

Introduction

1.1 About SINUMERIK

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

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

1.2 About this documentation

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

SINUMERIK Function Manuals

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

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

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

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

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

Overview of contents

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

Validity

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

System data

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

Basic Program Plus and Basic Program

In the TIA Portal, two different types of basic program are available for programming the PLC functionality:

- Basic Program Plus works only with symbolic addressing. The functionality is implemented via data types, PLC blocks, functions and instructions. Basic Program Plus is available only for SINUMERIK ONE.
- Basic Program works with both symbolic and absolute addressing. The programs are thus largely compatible with already existing programs from the SIMATIC Manager. Basic Program is available for ONE and MC controllers.

Basic Program Plus and Basic Program cannot be used together.

PLC blocks and data types

The descriptions of functions often refer to the PLC signals and functions. In the text, they are primarily the blocks and data types of Basic Program Plus, which is only available for SINUMERIK ONE.

You can find detailed information in the PLC Function Manual for the relevant controller.

Interface signals

The symbolic addressing of Basic Program Plus is the first choice for addressing interface signals. To derive signals for Basic Program, there are tables at the end of the relevant section with the PLC signals used for the different signal directions, as shown in the following example:

NC → PLC

Basic Program Plus	Basic Program	
<Axis>.basic.in.enc1Synchronized	LBP_Axis*.E_RefSyn1	DB31,DBX60.4
...		

PLC → NC

Basic Program Plus	Basic Program	
<Axis>.basic.out.refPointApproachCam	LBP_Axis*.A_DelayRef	DB31,DBX12.7
...		

Note**Quantity structure**

In Basic Program Plus, axes, channels etc. are represented via instances of the corresponding objects.

The absolute addressing in Basic Program is based on the absolute maximum number of the following components: Mode groups (DB11), channels (DB21, ...), axes/spindles (DB31, ...)

Standard scope

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

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

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

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

Websites of third-party companies

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

1.3 Documentation on the internet

1.3.1 Documentation overview SINUMERIK ONE

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



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

The documentation is divided into the following categories:

- User: Operating
- User: Programming
- Manufacturer/Service: Functions
- Manufacturer/Service: Hardware
- Manufacturer/Service: Configuration/Setup
- Manufacturer/Service: Safety Integrated
- Information and training
- Manufacturer/Service: SINAMICS

1.3.2 Documentation overview SINUMERIK operator components

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

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

The documentation is divided into the following categories:

- Operator Panels
- Machine control panels
- Machine Pushbutton Panel
- Handheld Unit/Mini handheld devices
- Further operator components

An overview of the most important documents, entries and links to SINUMERIK is provided at SINUMERIK Overview - Topic

Page (<https://support.industry.siemens.com/cs/document/109766201/sinumerik-an-overview-of-the-most-important-documents-and-links?dti=0&lc=en-WW>).

1.4 Feedback on the technical documentation

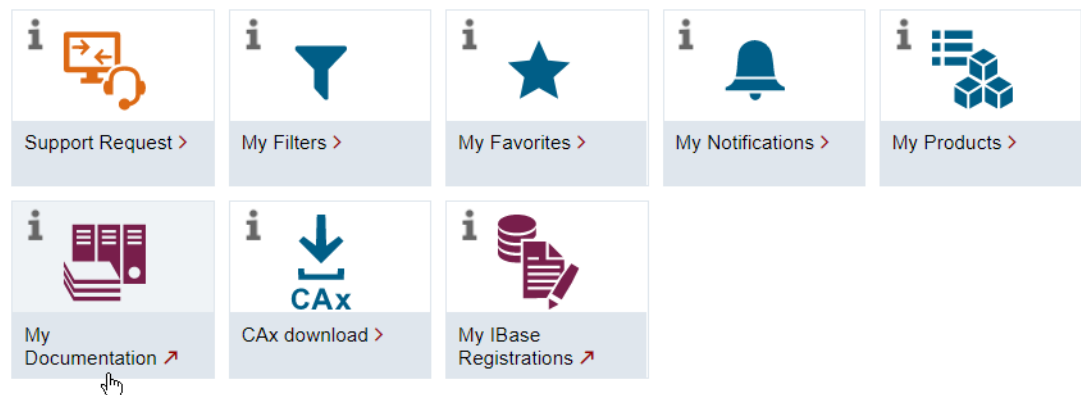
If you have any questions, suggestions or corrections regarding the technical documentation which is published in the Siemens Industry Online Support, use the link "Send feedback" link which appears at the end of the entry.

1.5 mySupport documentation

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

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

mySupport Links and Tools



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

Note

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

1.6 Service and Support

Product support

You can find more information about products on the internet:

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

The following is provided at this address:

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

Technical support

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

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

Training

You can find information on SITRAIN at the following address (<https://www.siemens.com/sitrain>).

SITRAIN offers training courses for automation and drives products, systems and solutions from Siemens.

Siemens support on the go



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

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

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

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

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

1.7 Using OpenSSL

This product can contain the following software:

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

You can find more information on the internet:

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

1.8 Compliance with the General Data Protection Regulation

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

1.8 Compliance with the General Data Protection Regulation

For this product, this means:

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

Fundamental safety instructions

2.1 General safety instructions

 WARNING
Danger to life if the safety instructions and residual risks are not observed If the safety instructions and residual risks in the associated hardware documentation are not observed, accidents involving severe injuries or death can occur. • Observe the safety instructions given in the hardware documentation. • Consider the residual risks for the risk evaluation.

 WARNING
Malfunctions of the machine as a result of incorrect or changed parameter settings As a result of incorrect or changed parameterization, machines can malfunction, which in turn can lead to injuries or death. • Protect the parameterization against unauthorized access. • Handle possible malfunctions by taking suitable measures, e.g. emergency stop or emergency off.

2.2 Warranty and liability for application examples

Application examples are not binding and do not claim to be complete regarding configuration, equipment or any eventuality which may arise. Application examples do not represent 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.

2.3 Security information

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. Siemens' products and solutions constitute one element of such a concept.

Customers are responsible for preventing unauthorized access to their plants, systems, machines and networks. Such systems, machines and components should only be connected

2.3 Security information

to an enterprise network or the internet if and to the extent such a connection is necessary and only when appropriate security measures (e.g. firewalls and/or network segmentation) are in place.

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

<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 are available and that the latest product versions are used. Use of product versions that are no longer supported, and failure to apply the latest updates may increase customer's exposure to cyber threats.

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

<https://www.siemens.com/cert>.

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.

Geometric machine modeling

3.1 Function description

3.1.1 Features

This section describes how the geometry of machine parts can be mapped based on protection areas and parameterized in the control using system variables for NC functions such as "collision avoidance".

The system variables are retentively saved in the NC and can be archived and/or read as "NC data" via SINUMERIK Operate using the commissioning archive.

By assigning a protective area to an element of the kinematic chain described in the previous section, the position and motion of the machine part can be clearly described within the machine space.

Note

Graphical editor

As an alternative to writing the system variables in a part program, the machine can be modeled from the SINUMERIK Operate user interface:

Operating area: "Commissioning" > "NC" > "Machine model"

Changes to the machine model

The direct changes to the machine model at the system variables only become visible at the user interface after explicitly requesting that the machine model is recalculated by calling the PROTA() (Page 88) or PROTS() (Page 89) function.

Changes to the machine model made at the user interface are immediately accepted in the system variables of the NC. The changes only become active after explicitly requesting that the machine model is recalculated by calling the PROTA() (Page 88) or PROTS() (Page 89) function.

Protection area

The protection areas are the central element when geometrically modeling a machine. The geometric dimensions of a machine part, its reference to the kinematic chain and additional general features are described by a protection area.

A protection area has the following parameters:

- Name of the protection area
- Name of the kinematic element to which the protection area is assigned
- Type of the protection area
- Name of the first protection area element

3.1 Function description

- Color and transparency of the protection area
- Detail level for the protection area
- Number of NC/PLC interface bits of the protection area
- Initialization status of the protection area
- Address of the geometric data of the machine element to the protected (only relevant for automatic protection areas)

Each parameter is mapped by a system variable. The individual parameters and/or system variables are described in detail in Chapter "System variables: Protection areas (Page 30)".

Kinematic chain

The corresponding protection area is assigned to an element of the kinematic chain in order to map the position and motion of a machine part. The geometric data of the protection area relates then to the local coordinate system of this kinematic element.

Type of the protection area

The following types of protection areas are available:

- Machine protection areas (type: "MACHINE")
Machine protection areas are used for general machine modeling. They are used to map stationary and moving machine parts whose geometry is defined once upon commissioning and no longer changes when the machine is in operation.
- Automatic tool protection areas (type: "TOOL")
Automatic tool protection areas are used to map tools. The geometry of the tool is not provided directly with this but is produced automatically by the control when the tool is activated.
See Section "Automatic tool protection areas (Page 22)".

Protection area element

Using a protection area element, the geometrical element used is described regarding its geometrical and general properties.

A protection area element has the following parameters:

- Name of the protection area element
- Name of the following protection area element
- Name of the following protection area element parallel to \$NP_NEXT
- Color and transparency of the protection area element
- Detail level for the protection area element
- Usage of the protection area element
- Type of the protection area element
- File name of the STL/NPP file that contains the geometric data of the protection area element (only relevant for type "FILE")
- Geometrical parameters of the protection area body (only relevant for types "BOX", "SPHERE", "CYLINDER", "CONE" or "TORUS")

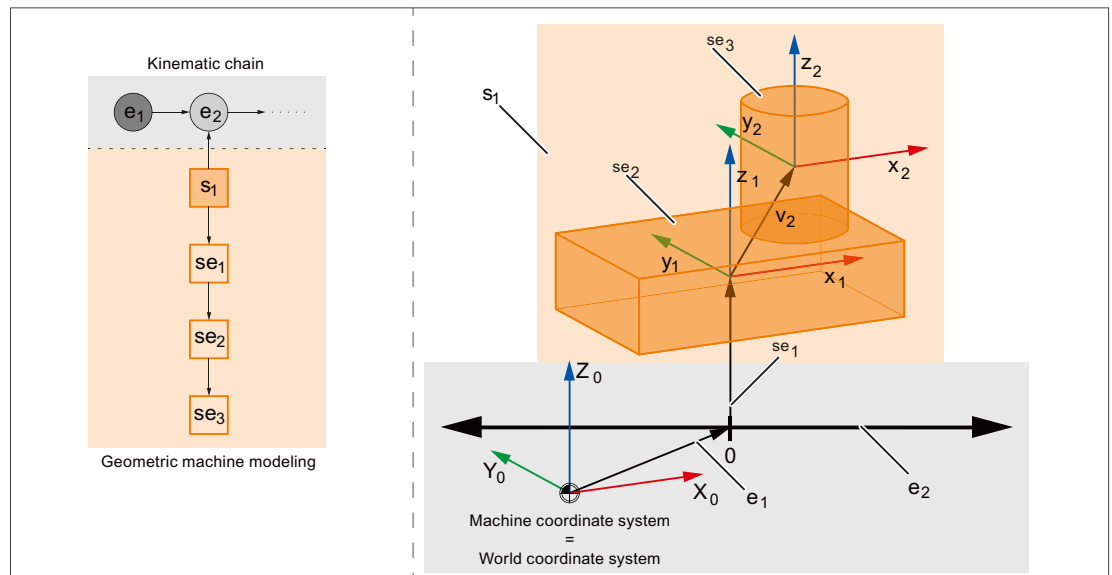
- Offset vector of the protection area element local coordinate system
- Direction vector for the rotation of the protection area element local coordinate system
- Angle for the rotation of the protection area element local coordinate system

Each parameter is mapped by a system variable. The individual parameters or system variables are described in detail in:

- Chapter "System variables: Protection area elements for machine protection areas (Page 41)"
- Chapter "System variables: Protection area elements for automatic tool protection areas (Page 64)"

Protection area, protection area elements and kinematic chain

Using an example of a protection area with two protection area elements, the following diagram shows the interrelationship between the protection area, its protection area elements and the assignment to an element of the kinematic chain.



- e_1 Kinematic element 1, type "OFFSET", continuous offset
- e_2 Kinematic element 2, type "AXIS_LIN", machine axis AX1
- s_1 Protection area
- se_1 Protection area element 1, type "FRAME", offset
- se_2 Protection area element 2, type "BOX"
- se_3 Protection area element 3, type "CYLINDER"

Figure 3-1 Protection area, protection area elements and kinematic chain

3.1.2 Automatic tool protection areas

Unlike machine protection areas whose geometry is defined once during the machine modeling and then no longer changes, the geometry of a tool protection area can change with every tool change. As a result the geometry of an automatic tool protection area is not described directly when the machine model is created, and instead the address (magazine number, magazine location, etc.) is provided under which the tool data is stored. The following actions are then carried out automatically by the control.

1. The tool modeling (see "Tool modeling" section below) produces an STL file and an NPP file.
2. If the tool can be modeled with geometric primitives, then one or more protection area elements of the relevant type ("BOX", "SPHERE", "CYLINDER", "CONE" or "TORUS") are created and stored in the NPP file. Otherwise, the tool is modeled from triangles and the data is stored in the STL file.
Depending on the tool type, it is also possible that both modeling variants are used. In these cases, one part of the modeling is stored in the NPP file and the other part in the STL file.
3. The protection area element is assigned to the protection area type "TOOL".

Tool definition independent of the tool mounting position

Normally the parameter "\$NP_1ST_PROT (Page 33)" remains empty for the definition of a tool protection area. The name of the protection area element is only entered when the tool is activated by the control (see above).

In order for there to be a tool definition independently of the tool mounting position, a protection area element of the "FRAME (Page 49)" type can be assigned via the parameter "\$NP_1ST_PROT" (transformation element). The transformations to align the tool can be carried out via this additional element. When a tool is activated the name of the internal protection area element is entered in the parameter "\$NP_NEXT (Page 43)" of the transformation element by the control.

The following rules must be observed:

- The transformation element may only be of the "FRAME" type.
- Only one transformation element may be used for each tool protection area.
- The parameter "\$NP_NEXTP" of the transformation element is not evaluated.

Tool reference point

The position of the tool reference point in the machine model is determined by the kinematic element to which the tool protection area is assigned. In addition the tool reference point can be offset within the tool protection area by an optional transformation element.

Kinematic transformations

The tool reference point may only be determined via the kinematic chain when defining a kinematic transformation. Offsets by the transformation element of the tool protection area are not taken into account.

NOTICE**Determination of the tool reference point for kinematic transformations**

Offsets of the tool reference point by the transformation element of the tool protection area are not taken into account by kinematic transformations.

Tool modeling

The model of a tool is created by the control heuristically from the tool data. The tool data used for this (L1, L2, L3, R) is always the overall dimensions resulting from the individual components, e.g. length plus wear, as is also entered in the program processing for tool offset.

Programmable tool offsets

Programmable tool offsets such as OFFN (offset to the programmed contour) are not taken into account as they may change in any block, even without a tool change.

Tool type-dependent model generation

The following tool types are distinguished for the model generation:

- Milling tool and all other tools that are neither turning tools or grinding tools
 - Modeling
The tool is modeled by a cylinder with the height L1 and radius R. If the length L1 is negative the absolute value of L1 is used for the cylinder height. The sign L1 is taken into account when positioning the cylinder in the machine model. The cylinder axis is parallel to L1.
Tool type 110 (ball end mill cutter) and 111 (face cutter) are modeled using half a sphere or sphere segments.
If the radius is negative the absolute value of the radius is used. If the value for the radius is less than 1/3 mm, a radius of 1/3 mm is used.
 - Positioning
The cylinder is positioned in the machine model using the tool length components L2 and L3.
For milling tools (tool type 100 ... 199) and drilling tools (tool type 200 ... 299) on turning machines, the cylinder is positioned using the cutting edge position.
Requirement: Cutting edge position == 5 ... 8
- Grinding tools
 - Modeling
Grinding tools (grinding wheel, tool type 400 ... 499) are modeled by a cylinder with the tool length as the radius and double the tool radius as the height.
 - Positioning
The cylinder is positioned in the machine model using the tool lengths L1, L2 and L3.
- Turning tools
In the case of turning tools only the cutting tips are taken into account in the machine model but not their connection to the tool reference point.
The following data is taken into account when modeling a cutting tip:
 - Tool type
 - Cutting edge position
 - Cutting edge radius
 - Clearance angle
 - Holder angle
 - Cutting tip length
 - Cutting tip width
 - Tip thickness (assumption: tip thickness = 10% tip length)

Tool model

A tool is modeled as standard with an accuracy of one third of the collision tolerance (Page 80). The geometry data of the modeled tool is stored in an internal file in STL and/or NPP format:

- Directory: `_N_PROT3D_DIR/_N_TOOL_DIR`
- Identifiers: Name of the associated protection area with prefix "`_N_`" and ending "`_STL`" or "`_NPP`"

The coordinate system of the geometric data always has its origin at the point from which the tool length offsets point to the tip of the tool.

System variable

All parameters for an automatic tool protection area can be read via System variable (Page 41).

3.2 Commissioning

3.2.1 General

3.2.1.1 Overview

The commissioning of an NC function such as "Collision avoidance" is done by means of:

- Machine data
 - Specifications for the quantity structure of protection areas, protection area elements, NC/PLC interface signals, triangles for the geometry modelling
 - Creation mode of the machine model
 - Creation mode for automatic tool protection areas
- System variables
 - Parameterization of the protection areas
 - Parameterization of the protection area elements of a protection area

3.2.1.2 Structure of the system variables

The system variables are structured according to the following scheme:

- `$NP_<name>[<index_1>]`
- `$NP_<name>[<index_1>, <index_2>]`

General

The system variables to describe the protection areas or protection area elements have the following properties:

- Prefix: **\$NP_**, (N for NC, P for protection).
- They can be read and written via NC programs.
- They can be stored in archives and loaded to the NC again.

Data type

STRING

All system variables of the STRING data type have the following properties:

- Maximum string length: 31 characters
- No distinction is made between upper and lower case
Example: "Axis1" is identical to "AXIS1"
- Spaces and special characters are permitted
Example: "Axis1" is not identical to "Axis 1"
- Names that **start** with **two** underscores " __ " are reserved for system purposes and must **not** be used for user-defined names.

Note

Leading space

Since spaces are valid and distinct characters, names that **start** with a **space** followed by **two** underscores " __ " can, in principle, be used for user-defined names. However, because they can be easily mistaken for system names, this procedure is **not** recommended.

Index_1

System variables for protection areas

The individual protection areas are addressed via index_1. Index 0 → 1st protection area, index 1 → 2nd protection area, ... m → (m+1) protection area, where m = (\$MN_MM_MAXNUM_3D_PROT_AREAS - 1)

All system variables of a protection area have the same index.

System variables for protection areas elements

The individual protection area elements are addressed via index_1. Index 0 → 1st protection area element, index 1 → 2nd protection area element, ... n → (n+1)th protection area element, where n = (\$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - 1)

All system variables of a protection area element have the same index.

Index_2

Depending on the respective system variables, index_2 has different meanings.

3.2.1.3 Color chart

The following color chart provides an overview of the RGB color values and the associated colors. A RGB color value comprises three bytes. One byte per color:

3rd byte	2nd byte	1st byte
Color value for red	Color value for green	Color value for blue
0 - 255 _D or 0 - FF _H	0 - 255 _D or 0 - FF _H	0 - 255 _D or 0 - FF _H

```

330000 333300 336600 339900 33CC00 33FF00 66FF00 66CC00 669900 666600 663300 660000 FF0000 FF3300 FF6600 FF9900 FFCC00 FFFF00

330033 333333 336633 339933 33CC33 33FF33 66FF33 66CC33 669933 666633 663333 660033 FF0033 FF3333 FF6633 FF9933 FFCC33 FFFF33

330066 333366 336666 339966 33CC66 33FF66 66FF66 66CC66 669966 666666 663366 660066 FF0066 FF3366 FF6666 FF9966 FFCC66 FFFF66

330099 333399 336699 339999 33CC99 33FF99 66FF99 66CC99 669999 666699 663399 660099 FF0099 FF3399 FF6699 FF9999 FFCC99 FFFF99

3300CC 3333CC 3366CC 3399CC 33CCC
C 33FFCC 66FFCC 66CCC
C 6699CC 6666CC 6633CC 6600CC FF00CC FF33CC FF66CC FF99CCFFCCCC FFFFCC

3300FF 3333FF 3366FF 3399FF 33CCFF 33FFFF 66FFFF 66CCFF 6699FF 6666FF 6633FF 6600FF FF00FF FF33FF FF66FF FF99FF FFCCFF FFFFFF

0000FF 0033FF 0066FF 0099FF 00CCFF 00FFFF 99FFFF 99CCFF 9999FF 9966FF 9933FF 9900FF CC00FF CC33FF CC66FF CC99FF CCCCFF CCFFFF

0000CC 0033CC 0066CC 0099CC 00CCC
C 00FFCC 99FFCC 99CCCC 9999CC 9966CC 9933CC 9900CC CC00CC CC33CC CC66CC CC99CC CCCCC
C CCFFCC

000099 003399 006699 009999 00CC99 00FF99 99FF99 99CC99 999999 996699 993399 990099 CC0099 CC3399 CC6699 CC9999 CCCC99 CCFF99

000066 003366 006666 009966 00CC66 00FF66 99FF66 99CC66 999966 996666 993366 990066 CC0066 CC3366 CC6666 CC9966 CCCC66 CCFF66

000033 003333 006633 009933 00CC33 00FF33 99FF33 99CC33 999933 996633 993333 990033 CC0033 CC3333 CC6633 CC9933 CCCC33 CCFF33

000000 003300 006600 009900 00CC00 00FF00 99FF00 99CC00 999900 996600 993300 990000 CC0000 CC3300 CC6600 CC9900 CCCC00 CCFF00

```

3.2.2 Machine data

3.2.2.1 Maximum number of protection areas

The maximum number of all types of parameterizable protection areas (Page 32) is specified with the machine data.

MD18890 \$MN_MM_MAXNUM_3D_PROT_AREAS = <number>

3.2.2.2 Maximum number of protection area elements for machine protection areas

The parameterizable protection area elements for machine protection areas (\$NP_PROT_TYPE == "MACHINE" (Page 32)) include both geometric primitives and frames within a protection area. Their maximum number is set with the machine data:

MD18892 \$MN_MM_MAXNUM_3D_PROT_AREA_ELEM = <number of geometric primitives> + <number of frames>

3.2.2.3 Maximum number of protection area elements for automatic tool protection areas

The maximum number of protection area elements for automatic tool protection areas is specified with the following machine data. For each automatic tool protection area the control precisely creates one protection area element. As a consequence, using the value parameterized here, the maximum possible number of parameterizable automatic tool protection areas (\$NP_PROT_TYPE == "TOOL" (Page 32)) is also simultaneously defined.

MD18893 \$MN_MM_MAXNUM_3D_T_PROT_ELEM = <number>

3.2.2.4 Maximum number of NC/PLC interface signals for the preactivation of protection areas

The machine data is used to inform the control of the number of NC/PLC interface signals actually used by the interface <Nc>.collisionProtAreas (signals for protection areas).

The number of interface signals used increases the memory space required for each part program block. Counting the number of NC/PLC interface signals used starts at the signal with the lowest address.

MD18897 \$MN_MM_MAXNUM_3D_INTERFACE_IN = <number>

More information

A detailed description of the interface signals can be found in the Function Manual PLC.

PLC signals

Basic Program Plus	Basic Program	
<Nc>.collisionProtAreas.out.activate	LBP_NC.A_CollCheck	DB10.DBX234.0..241.7

3.2.2.5 Maximum number of triangles for machine protection areas

With the machine data, the maximum number of triangles for protection area bodies (\$NP_TYPE == "FILE" (Page 49)) of machine protection areas (\$NP_PROT_TYPE == "MACHINE" (Page 32)) to be provided by the control is defined.

MD18895 \$MN_MM_MAXNUM_3D_FACETS = <number>

3.2.2.6 Maximum number of triangles for automatic tool protection areas

With the machine data, the maximum number of triangles for protection area bodies of automatic tool protection areas to be provided by the control is defined.

MD18894 \$MN_MM_MAXNUM_3D_FACETS_INTERN = <number>

The control system automatically models the protection area bodies based on the geometric data of the tool active at the time of creation. The number of triangles correspondingly increases:

- as the geometric complexity of the tool increases.
- as the parameterized collision tolerance decreases.

Note

Protection area bodies and collision tolerance

The protection area bodies of automatic tool protection areas are, as standard, created by the control with a precision of 1/3 of the collision tolerance (Page 80).

3.2.2.7 Creation mode for automatic tool protection areas

The machine data defines how the control creates protection area bodies of automatic tool protection areas.

MD18899 \$MN_PROT_AREA_TOOL_MASK = <mode>

<Mode>		
Bit	Value	Meaning
0	0	Do not apply heuristic model generation
	1	Heuristic model generation using tool data

3.2.3 System variables: Protection areas

3.2.3.1 Overview

A protection area is parameterized with the following system variables:

Name	Meaning
\$NP_PROT_NAME	Name of the protection area
\$NP_CHAIN_ELEM	Name of the kinematic element to which the protection area is assigned
\$NP_PROT_TYPE	Type of the protection area
\$NP_1ST_PROT	Name of the first protection area element
\$NP_PROT_COLOR	Color and transparency of the protection area.
\$NP_PROT_D_LEVEL	Detail level for the protection area
\$NP_BIT_NO	Number of NC/PLC interface bits of the protection area
\$NP_INIT_STAT	Initialization status of the protection area
\$NP_INDEX	Address of the geometric data of the machine element to be protected (only relevant for automatic protection areas)

The system variables are described in detail in the following sections.

Note

Establish a defined initial state

It is recommended that a defined initial state be generated before parameterizing the protection areas. To do this, set the system variables of the protection areas to their default values with the DELOBJ() function.

Change system variable values

If the value of one of the system variables listed above is changed, the change becomes immediately visible at the user interface, e.g. SINUMERIK Operate. The machine model of the NC is only updated after explicitly requesting that the machine model is recalculated by calling the PROTA() (Page 88) or PROTS() (Page 89) function.

3.2.3.2 \$NP_PROT_NAME

Function

Enter the NC-wide unique protection area name in the system variable. The protection area is referenced via this name, e.g. by a protection area element. The name is also displayed in the graphical editor of SINUMERIK Operate.

Syntax

```
$NP_PROT_NAME [<m>] = "<name>"
```

Meaning

\$NP_PROT_NAME:	Name of the protection area	
	Data type:	STRING
	Default value:	"" (empty string)
<m>:	System variable or protection area index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (\$MN_MM_3D_MAXNUM_PROT_AREAS - 1)
<name>:	Name of the protection area	
	Data type:	STRING

Example

The 6th protection area is assigned the name "Spindle":

Program code	Comment
N100 \$NP_PROT_NAME[5] = "Spindle"	; 6. Protection area, ; name = "Spindle"

3.2.3.3 \$NP_CHAIN_ELEM

Function

Enter the name of the kinematic element to which the protection area will be connected in the system variable.

Note

Reference coordinate system

The geometric data of the protection area, starting from the first protection area element (\$NP_1ST_PROT (Page 33)), refer to the local coordinate system of the kinematic element, with which the protection area is connected.

Syntax

```
$NP_CHAIN_ELEM[<m>] = "<Name>"
```

Meaning

\$NP_CHAIN_ELEM:	Name of the kinematic element to which the protection area will be connected	
	Data type:	STRING
	Default value:	"" (empty string)
<m>:	System variable or protection area index	
	Data type:	INT
	Value range:	0, 1, 2, ... (\$MN_MM_3D_MAXNUM_PROT_AREAS - 1)

<name>:	Name of the kinematic element	
	Data type:	STRING
	Value range:	For names parameterized in \$NK_NAME, refer to the Function Manual Basic Functions, Section "Kinematic chain".

Example

The 6th protection area is connected to the kinematic element with the name "Z axis":

Program code	Comment
N100 \$NP_CHAIN_ELEM[5] = "Z ax- is"	; 6. Protection area, ; name of the kin. element: "Z axis"

3.2.3.4 \$NP_PROT_TYPE

Function

The protection area type should be entered in the system variable:

- Machine protection area: "MACHINE"
The protection area body is defined using one or several protection area elements.
\$NP_1ST_PROT (Page 33) refers to the first protection area element.
- Automatic tool protection area: "TOOL"
The control calculates the dimensions of the protection area body from the tool data.
\$NP_INDEX (Page 39) refers to the tool.
- Workholder protection area: "FIXTURE"
- Workpiece protection area: "WORKPIECE"

Syntax

`$NP_PROT_TYPE[<m>] = "<type>"`

Meaning

\$NP_PROT_TYPE:	Type of the protection area	
	Data type:	STRING
	Range of values:	"MACHINE", "TOOL", "FIXTURE", "WORKPIECE"
	Default value:	"" (empty string)
<m>:	System variable or protection area index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (\$MN_MM_3D_MAXNUM_PROT_AREAS - 1)
<Type>:	Type	
	Data type:	STRING

Example

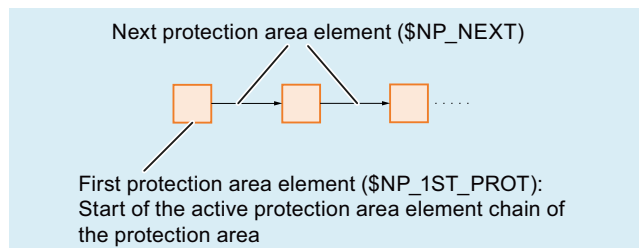
The 6th protection area is a machine protection area:

Program code	Comment
N100 \$NP_PROT_TYPE[5] = "MACHINE"	; 6. Protection area, ; type = "MACHINE"

3.2.3.5 \$NP_1ST_PROT

Function

Enter the name of the first protection area element (Page 42) in the system variable.



Syntax

```
$NP_1ST_PROT [<m>] = "<name>"
```

Meaning

\$NP_1ST_PROT:	Name of the first protection area element of the protection area	
	Data type:	STRING
	Range of values:	Names parameterized in \$NP_NAME (Page 42)
	Default value:	"" (empty string)
<m>:	System variable or protection area index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (\$MN_MM_3D_MAXNUM_PROT_AREAS - 1)
<name>:	Protection area name	
	Data type:	STRING

Automatic tool protection areas, \$NP_PROT_TYPE == "TOOL"

For automatic tool protection areas, only the following values are permissible for \$NP_1ST_PROT:

- "" (empty string)
- Name of a protection area element, type "FRAME"

Behavior for value == "" (empty string)

When activating the associated tool, the control creates a protection area element for the tool with a unique, internal name and protection area body generated from the geometric data of the tool. The name is assigned system variable \$NP_1ST_PROT.

- Protection area "TOOL" : \$NP_1ST_PROT = "<internal name>"

The coordinates of the tool protection area refer to the local coordinate system of the kinematic element to which it is assigned.

Behavior for value == name of a protection area element, type "FRAME"

When activating the associated tool, the control creates a protection area element for the tool with a unique, internal name and protection area body generated from the geometric data of the tool. The name is assigned a system variable \$NP_NEXT of the protection area element, type "FRAME", to which \$NP_1ST_PROT refers.

- Protection area "TOOL" : \$NP_1ST_PROT = "WKZ_Frame" →
 – Protection area element "WKZ_Frame" : \$NP_NEXT = "<internal name>"

The coordinates of the tool protection area refer to the local coordinate system of the protection area element, type "FRAME".

Possible applications: Tool definition independent of the position where it is mounted on the machine.

Example

The 1st protection area element which makes up the 6th protection area has the name "Spindle head":

Program code	Comment
N100 \$NP_1ST_PROT[5] = "Spindle head"	; 6. Protection area, ; 1. Protection area element = "spindle box"

3.2.3.6 \$NP_PROT_COLOR**Function**

Enter the protection area element-specific value for alpha/transparency and color (ARGB) in the system variable. This value is used for the display of the protection area or protection area element on the user interface. If a separate value is entered for a protection area element in \$NP_COLOR (Page 46), this is used for the display of the protection area element.

Structure

Alpha/transparency and color are specified as a double word in hexadecimal format:

AARRGGBB_H

- 1. - 3. Byte: RGB color value. See Chapter "Color chart (Page 27)".
- 4. Byte: Alpha channel or transparency value

	Byte	Meaning	Range of values
OC	1	Blue	0 - 255 _D or 0 - FF _H
GG	2	Green	
RR	3	Red	
AA	4	Alpha channel or transparency ¹⁾	
1) 0 = transparent or not visible, 255 _D = FF _H = not transparent or filled			

Syntax

\$NP_PROT_COLOR[<m>] = <value>

Meaning

\$NP_PROT_COLOR:	Alpha/transparency and color value of the protection area	
	Data type:	DWORD
	Range of values:	00000000 _H - FFFFFFFF _H
	Default value:	00000000 _H (black, not visible)
<m>:	System variable or protection area index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (\$MN_MM_3D_MAXNUM_PROT_AREAS - 1)
<value>:	Transparency and color value	
	Data type:	DWORD

Example

The 6th protection area is to be displayed half transparent and in a green-blue color on the user interface:

- AA = 7F_H = 127_D $\hat{=}$ 50% transparency
- RR (red) = 00 $\hat{=}$ no red component
- GG (green) = FF_H = 255_D $\hat{=}$ 100% green color component
- BB (blue) = 33_H = 51_D $\hat{=}$ 20% blue color component

Program code	Comment
N100 \$NP_PROT_COLOR[5] = 'H7F00FF33'	; 6. Protection area, ; alpha/transparency and color value = H7F00FF33

3.2.3.7 \$NP_PROT_D_LEVEL

Function

The detail level as of which the protection area or the protection area elements are displayed on the user interface is specified via the system variable. If a separate value is entered for a protection area element in \$NP_D_LEVEL (Page 47), this is used for the display of the protection area element.

Detail level

- Lowest detail level: 0
- Highest detail level: 3

If the detail level x is selected for the display on the HMI, all protection areas and protection area elements are displayed for which the following applies for the detail level D : $D \leq x$

Syntax

`$NP_PROT_D_LEVEL[<m>] = <value>`

Meaning

\$NP_PROT_D_LEVEL:	Detail level for the protection area	
	Data type:	INT
	Range of values:	$0 \leq D \leq 3$
	Default value:	0
<m>:	System variable or protection area index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (\$MN_MM_3D_MAXNUM_PROT_AREAS - 1)
<value>:	Detail level	
	Data type:	INT

Example

The 6th protection area is to be displayed as of detail level 3:

Program code	Comment
N100 \$NP_PROT_D_LEVEL[5] = 3	; 6. Protection area, ; detail level = 3

3.2.3.8 \$NP_BIT_NO

Function

Enter the bit number (0, 1, 2, 63) of the NC/PLC interface signal with which the protection area is to be connected in system variable \$NP_BIT_NO. If the protection area is not to be connected to an NC/PLC interface signal, enter the value -1.

NC/PLC interface

Activation / deactivation of the protection area can be requested via NC/PLC interface signals from the PLC user program or this can be used for feedback on the current status to the PLC user program:

- Request interface: <Nc>.collisionProtAreas.out
- Request interface: <Nc>.collisionProtAreas.in

Requirement

The status of the protection area must be "preactivated" or "PLC-controlled" in order that the assigned NC/PLC interface signal of the protection area is taken into account:

\$NP_INIT_STAT (Page 38) == "P" (preactivated or PLC-controlled)

Syntax

\$NP_BIT_NO[<m>] = <bit number>

Meaning

\$NP_BIT_NO:	Bit number of the interface signal to activate/deactivate the protection area	
	Data type:	INT
	Value range:	-1, 0, 1, 2, ... (\$MN_MM_MAXNUM_3D_INTERFACE_IN - 1)
	Default value:	-1 (no interface signal selected)
<m>:	System variable or protection area index	
	Data type:	INT
	Value range:	0, 1, 2, ... (\$MN_MM_3D_MAXNUM_PROT_AREAS - 1)
<Bit number>:	Bit number (0, 1, 2, ... 63) of the 64-bit interface	
	Data type:	INT

Example

The 6th protection area is assigned the 18th bit of the request and feedback interface:

Program code	Comment
N100 \$NP_BIT_NO[5] = 17	; 6. Protection area, ; IN: <Nc>.collisionProtAreas.in, BIT17 ; OUT: <Nc>.collisionProtAreas.out, BIT17

More information

A detailed description of the interface signals can be found in the Function Manual PLC.

PLC signals

Basic Program Plus	Basic Program	
<Nc>.collisionProtAreas.in	---	DB10.DBX226.0..233.7
<Nc>.collisionProtAreas.out	---	DB10.DBX234.0..241.7
<Nc>.collisionProtAreas.in.active	LBP_NC.E_CollCheck	DB10.DBX226.0..233.7
<Nc>.collisionProtAreas.out.activate	LBP_NC.A_CollCheck	DB10.DBX234.0..241.7

3.2.3.9 \$NP_INIT_STAT

Function

Enter the protection area initialization status in the system variable.

The status of a protection area is set to the parameterized initialization status in the following situations:

- When the control ramps up
- When calling the PROTA (Page 88) function after the protection area has been recreated during operation by writing the protection-area-specific system variables
- When calling the PROTA (Page 88) function with parameter "R"
- When calling the PROTS (Page 89) function with parameter "R"

Syntax

```
$NP_INIT_STAT[<m>] = "<status>"
```

Meaning

\$NP_INIT_STAT:	Initialization status of the protection area	
	Data type:	STRING
	Value range:	"A", "a", "I", "i", "P", "p"
	Value	Protection area status
	"A" or "a"	Activated
	"I" or "i"	Inactive
	"P" or "p"	Preactivated or PLC-controlled ¹⁾
	Default value:	"I" (inactive)

<m>:	System variable or protection area index	
	Data type:	INT
	Value range:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREAS - 1)
<Status>:	Initialization status	
	Data type:	STRING
1) The activation/deactivation is performed via: <Nc>.collisionProtAreas.out.activate		

Example

The initialization status of the 6th protection area is set to "P" (preactivated or PLC-controlled):

Program code	Comment
N100 \$NP_INIT_STAT[5] = "P"	; 6. Protection area, ; initialization status = "P"

The current status depends on the state of the interface signal parameterized in \$NP_BIT_NO (Page 37).

PLC signals

Basic Program Plus	Basic Program	
<Nc>.collisionProtAreas.in	---	DB10, DBX226.0..233.7
<Nc>.collisionProtAreas.out	---	DB10, DBX234.0..241.7
<Nc>.collisionProtAreas.in.active	LBP_NC.E_CollCheck	DB10, DBX226.0..233.7
<Nc>.collisionProtAreas.out.activate	LBP_NC.A_CollCheck	DB10, DBX234.0..241.7

3.2.3.10 \$NP_INDEX

Function

For automatic protection areas (\$NP_PROT_TYPE (Page 32)), the address, under which the geometric data of the machine part, tool, etc. to be protected is saved, should be entered in the system variable. The control then automatically creates the geometrical dimensions of the protection area from the geometric data.

Example

For example, for an automatic tool protection area (\$NP_PROT_TYPE == "TOOL") the geometrical dimensions of the protection area are created based on the tool data.

Syntax

```
$NP_INDEX[<m>,<i>] = <value>
```

Meaning

\$NP_INDEX:	Address of the geometric data for the automatic protection areas	
	Data type:	INT[3]
<m>:	System variable or protection area index	
	Data type:	INT
	Value range:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREAS - 1)
<i>:	Index The meaning of the system variables \$NP_INDEX[<m>, <i>], with i = 0, 1, 2, ... is dependent on the type (\$NP_PROT_TYPE) of the automatic protection area. See type-specific tables.	
	Data type:	INT
	Value range:	0, 1, 2
<value>:	Address	
	Data type:	INT

Type: Automatic tool protection area (\$NP_PROT_TYPE == "TOOL")

<i>	<value>	
	Tool management active	Tool management not active
0	Tool location number ¹⁾	Spindle number
1	Magazine number ²⁾	---
2	TOA area ³⁾	

^{1), 2)} If the tool is located in a magazine location of a turret magazine, the tool location number in the turret magazine and the magazine number of the turret magazine are indicated here. For tools located in the spindle, the tool location and magazine location number for the tool spindle is specified here. See examples!

³⁾ TOA area "1" can be addressed with 0 as well as also with 1.

Note

With active tool management, the control can also automatically model tools located at the magazine locations of a turret magazine and monitor them for collision.

Automatic tool modeling is not possible for chain and plane magazines.

Examples

The following applies for the following examples:

- The specified protection area is an automatic tool protection area (\$NP_PROT_TYPE == "TOOL").
The geometrical dimensions of the protection area should be generated from the geometry data of the tool.
- Tool management is active.

Example 1

The tool is located at a magazine location of a turret magazine:

- Tool location number: 3
- Magazine number: 1 (turret magazine)
- TOA area: 1

Program code	Comment
; The dimensions of the 5th protection area are based on the tool data	
; of the tool located at the following position:	
N100 \$NP_INDEX[4,0] = 3	; Tool location number = 3
N110 \$NP_INDEX[4,1] = 1	; Magazine number = 1 (turret magazine)
N120 \$NP_INDEX[4,2] = 1	; TOA area = 1

Example 2

The tool is located on the active spindle:

- Tool location number: 1
- Magazine number: 9998 (spindle 1)
- TOA area: 1

Program code	Comment
; The dimensions of the 6th protection area are based on the tool data	
; of the tool located at the following position:	
N100 \$NP_INDEX[5,0] = 1	; Tool location number = 1
N110 \$NP_INDEX[5,1] = 9998	; magazine number = 9998 (spindle 1)
N120 \$NP_INDEX[5,2] = 1	; TOA area = 1

See also

\$NP_PROT_TYPE (Page 32)

3.2.4 System variables: Protection area elements for machine protection areas

3.2.4.1 Overview

A protection area element of a machine protection area is parameterized with the following system variables:

Name	Meaning
\$NP_NAME	Name of the protection area element
\$NP_NEXT	Name of the following protection area element

Name	Meaning
\$NP_NEXTP	Name of the following protection area element parallel to \$NP_NEXT
\$NP_COLOR	Color and transparency of the protection area element.
\$NP_D_LEVEL	Detail level for the protection area element
\$NP_USAGE	Usage of the protection area element
\$NP_TYPE	Type of the protection area element
\$NP_FILENAME	File name of the STL/NPP file that contains the geometric data of the protection area element (only relevant for \$NP_TYPE == "FILE")
\$NP_PARA	Geometric parameters of the protection area body (only relevant for \$NP_TYPE == "BOX", "SPHERE", "CYLINDER", "CONE" or "TORUS")
\$NP_OFF	Offset vector of the protection area element local coordinate system
\$NP_DIR	Direction vector for the rotation of the protection area element local coordinate system
\$NP_ANG	Angle for the rotation of the protection area element local coordinate system

The system variables are described in detail in the following sections.

Note

Establish a defined initial state

It is recommended that a defined initial state be generated before parameterizing the protection area elements. To do this, set the system variables of the protection area elements to their default values with the DELOBJ() function.

Change system variable values

If the value of one of the system variables listed above is changed, the change becomes immediately visible at the user interface, e.g. SINUMERIK Operate. The machine model of the NC is only updated after explicitly requesting that the machine model is recalculated by calling the PROTA() (Page 88) or PROTS() (Page 89) function.

3.2.4.2 \$NP_NAME

Function

Enter the NC-wide unique protection area element name in the system variable. The protection area element is referenced via this name. The name is also displayed in the graphical editor of SINUMERIK Operate.

Syntax

```
$NK_NAME [<n>] = "<name>"
```

Meaning

\$NP_NAME:	Name of the protection area element	
	Data type:	STRING
	Default value:	"" (empty string)
<n>:	System variable or protection area element index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - 1)
<name>:	Name of the protection area element	
	Data type:	STRING

Example

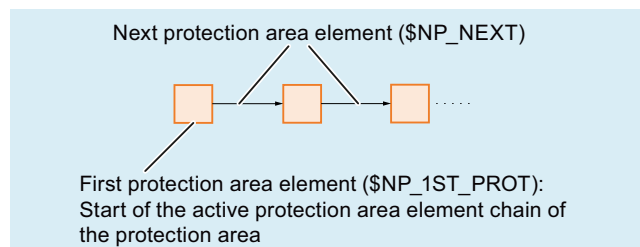
The 19th protection area element is assigned the name "Spindle head":

Program code	Comment
N100 \$NP_NAME[18] = "Spindle head"	; 19th protection area element, ; name = "Spindle head"

3.2.4.3 \$NP_NEXT

Function

If a protection area is made up of several protection area elements, they must be linked. To do this, the name of the following protection area element must be entered in the system variable \$NP_NEXT in every protection area element.



If there is no following protection area element, the name must be entered as an empty string "".

Offset and rotation

An offset and/or rotation in the current protection area element (\$NP_OFF (Page 60), \$NP_DIR (Page 61) and \$NP_ANG (Page 63)) affects the following protection area element specified in \$NP_NEXT. This means that the specification of the spatial position and orientation of the following protection area element is relative to the current protection area element.

Syntax

```
$NK_NEXT [<n>] = "<name>"
```

Meaning

\$NP_NEXT:	Name of the following protection area element	
	Data type:	STRING
	Range of values:	All names contained in \$NP_NAME (Page 42)
	Default value:	"" (empty string)
<n>:	System variable or protection area element index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - 1)
<name>:	Protection area name	
	Data type:	STRING

Example

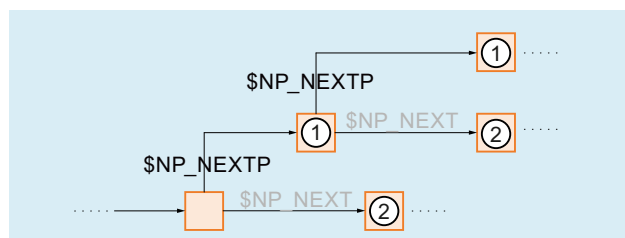
The following protection area element with the name "Coolant nozzle 1" is attached to the 19th protection area element:

Program code	Comment
N100 \$NP_NAME[18] = "Coolant nozzle 1"	; 19th protection area element, ; name of the following element: "Coolant nozzle 1"

3.2.4.4 \$NP_NEXTP

Function

The protection area element chain can be branched via the system variable \$NP_NEXTP. To do this, the following protection area elements must be specified in the system variables \$NP_NEXT and \$NP_NEXTP in a protection area element. These protection area elements are then parallel to one another in two separate subchains.



- ① Following parallel protection area element
- ② Following protection area element of the same subchain

Figure 3-2 Protection area elements in parallel subchains

Application example

The separate subchains can be used, for example, to model different machine parts of a protection area for visualization or collision avoidance. Typically "C" (collision avoidance) is specified for the protection area element referred to by \$NP_NEXT for use in \$NP_USAGE (Page 48), and for the protection area element referred to in \$NP_NEXTP, the value "V" (visualization).

Offset and rotation

An offset and/or rotation in the current protection area element (\$NP_OFF (Page 60), \$NP_DIR (Page 61) and \$NP_ANG (Page 63)) affects the following protection area element specified in \$NP_NEXTP. This means that the specification of the spatial position and orientation of the following protection area element is relative to the current protection area element.

Syntax

```
$NP_NEXTP [<n>] = "<name>"
```

Meaning

\$NP_NEXTP:	Name of the branching protection area element	
	Data type:	STRING
	Range of values:	All names contained in \$NP_NAME (Page 42)
	Default value:	"" (empty string)
<n>:	System variable or protection area element index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - 1)
<name>:	Protection area name	
	Data type:	STRING

Example

The following parallel protection area element with the name "Coolant nozzle 2" is attached to the 19th protection area element:

Program code	Comment
N100 \$NP_NEXTP[18] = "Coolant nozzle 2"	; 19. Protection area element, ; name of the parallel following element: "Coolant nozzle 2"

3.2.4.5 \$NP_COLOR

Function

Enter the protection area element-specific value for alpha/transparency and color (ARGB) in the system variable. This value is used for the display of the protection area element on the user interface. If a specific value is not parameterized for a protection area element, the protection area-specific value from \$NP_PROT_COLOR (Page 34) applies.

Structure

Alpha/transparency and color are specified as a double word in hexadecimal format:

AARRGGBB_H

- 1st - 3rd Byte: RGB color value. See Chapter "Color chart (Page 27)".
- 4th byte: Alpha channel or transparency value

	Byte	Meaning	Value range
OC	1	Blue	0 - 255 _D or 0 - FF _H
GG	2	Green	
RR	3	Red	
AA	4	Alpha channel or transparency ¹⁾	
1) 0 = transparent or not visible, 255 _D = FF _H = not transparent or filled			

Syntax

\$NP_COLOR[<n>] = <Value>

Meaning

\$NP_COLOR:	Alpha/transparency and color value of the protection area element	
	Data type:	DWORD
	Value range:	00000000 _H - FFFFFFFF _H
	Default value:	00000000 _H (black, not visible)
<n>:	System variable or protection area element index	
	Data type:	INT
	Value range:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - 1)
<value>:	Transparency and color value	
	Data type:	DWORD

Example

The 19th protection area is to be displayed half transparently and in a green-blue color on the user interface:

- AA = 7F_H = 127_D ≙ 50% transparency
- RR (red) = 00 ≙ no red component

- GG (green) = $FF_H = 255_D \triangleq 100\%$ green color component
- BB (blue) = $33_H = 51_D \triangleq 20\%$ blue color component

Program code	Comment
N100 \$NP_COLOR[18] = 'H7F00FF33'	; 19. Protection area, ; Alpha/transparency and ; Color value = 'H7F00FF33'

3.2.4.6 \$NP_D_LEVEL

Function

The detail level as of which the protection area element is displayed on the user interface is specified via the system variable. If no value different from the default value is assigned for a protection area element the protection area-specific value takes effect \$NP_PROT_D_LEVEL (Page 36).

Detail level

- Lowest detail level: 0
- Highest detail level: 3

If the detail level x is selected for the visualization of the machine model, all protection areas and protection area elements are displayed for which the following applies for the detail level D : $D \leq x$

Syntax

`$NP_D_LEVEL[<n>] = <value>`

Meaning

\$NP_PROT_D_LEVEL:	Detail level for the protection area element	
	Data type:	INT
	Range of values:	$0 \leq D \leq 3$
	Default value:	0
<m>:	System variable or protection area element index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - 1)
<value>:	Detail level	
	Data type:	INT

Example

The 19th protection area is always to be displayed ⇒ detail level 0:

Program code	Comment
N100 \$NP_PROT_D_LEVEL[18] = 0	; 19. Protection area, ; detail level = 0

3.2.4.7 \$NP_USAGE**Function**

Enter the protection area element usage in the system variable. The usage specifies how the protection area element is to be considered for the collision avoidance.

- Only visualization, no collision calculation
- Only collision calculation, no visualization
- Visualization and collision calculation

Usage	Meaning
Visualization	The protection area element is displayed in the machine model on the SINUMERIK Operate user interface
Collision calculation	The protection area element is also taken into account for the collision calculation

Syntax

`$NP_USAGE [<n>] = "<value>"`

Meaning

\$NP_USAGE:	Use of the protection area element	
	Data type:	CHAR
	Range of values:	"V", "v", "C", "c", "A", "a"
	Value	Meaning
	"V" or "v"	Only visualization, no collision calculation
	"C" or "c"	Only collision calculation, no visualization
	"A" or "a"	Visualization and collision calculation
	Default value:	"A"
<n>:	System variable or protection area element index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - 1)
<value>:	Usage	
	Data type:	CHAR

Example

The 19th protection area element is to be displayed at the user interface and included in the collision calculation:

Program code	Comment
N100 \$NP_USAGE[18] = "A"	; 19. Protection area, ; usage = "A"

3.2.4.8 \$NP_TYPE

Function

Enter the protection area element type in the system variable.

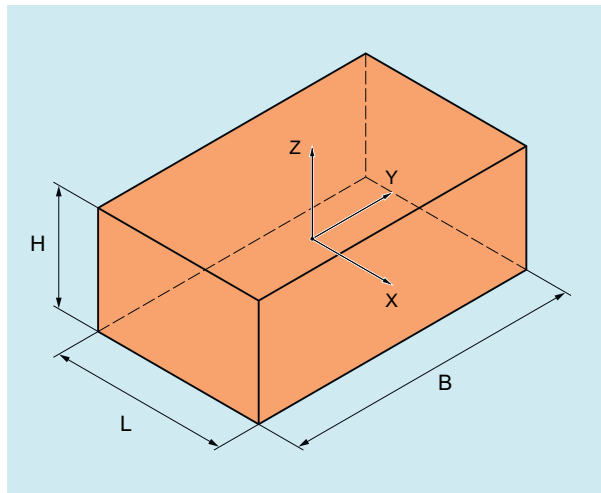
Type: "FRAME"

A protection area element of the "FRAME" type does not contain any bodies, but defines a coordinate transformation of the local coordinate system. The coordinate transformation affects all the following (\$NP_NEXT (Page 43)) and/or parallel (\$NP_NEXTP (Page 44)) protection area elements. The values of the coordinate transformation are set via:

- Offset: \$NP_OFF (Page 60)
- Rotation direction vector: \$NP_DIR (Page 61)
- Angle of rotation: \$NP_ANG (Page 63)

No parameters are specified in \$NP_PARA (Page 59) for the "FRAME" type.

Type: "BOX"



- L Length in the X direction
- B Width in the Y direction
- H Height in the Z direction

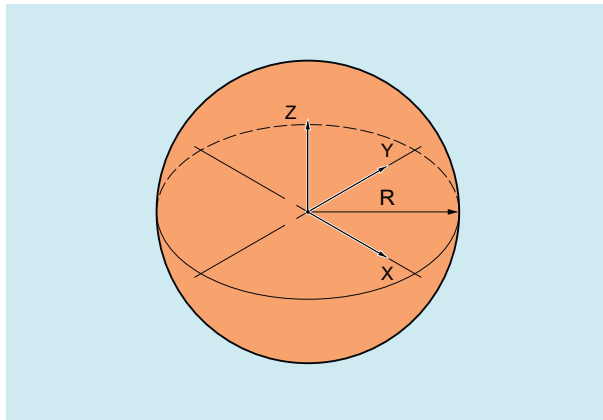
A protection area element of the "BOX" type defines a paraxial box in the local coordinate system of the protection area element. The mid-point of the box is at the origin of the

local coordinate system. At the same time as the definition of the body, the local coordinate system can be transformed via the following system variables:

- Offset: \$NP_OFF (Page 60)
- Rotation direction vector: \$NP_DIR (Page 61)
- Angle of rotation: \$NP_ANG (Page 63)

The parameters length, width and height are to be entered in \$NP_PARA (Page 59)

Type: "SPHERE"



R Radius of the sphere:

A protection area element of the "SPHERE" type defines a sphere in the local coordinate system of the protection area element. The mid-point of the sphere is at the origin of the local coordinate system. At the same time as the definition of the body, the local coordinate system can be transformed via the following system variables:

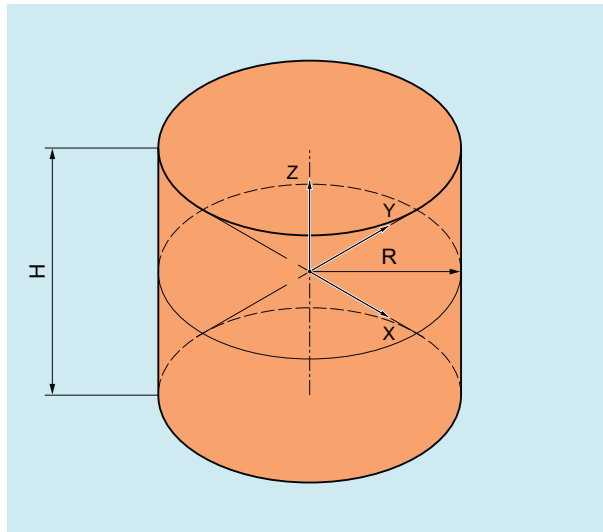
- Offset: \$NP_OFF (Page 60)
- Rotation direction vector: \$NP_DIR (Page 61)
- Angle of rotation: \$NP_ANG (Page 63)

The parameter radius is to be entered in \$NP_PARA (Page 59)

Note

Rotation

As the center point of the sphere and the starting point of the direction vector are in the coordinate origin of the local coordinate system of the protection area element, a rotation using the direction vector \$NP_DIR (Page 61) and angle of rotation \$NP_ANG (Page 63) has no effect on the position of the sphere.

Type: "CYLINDER"

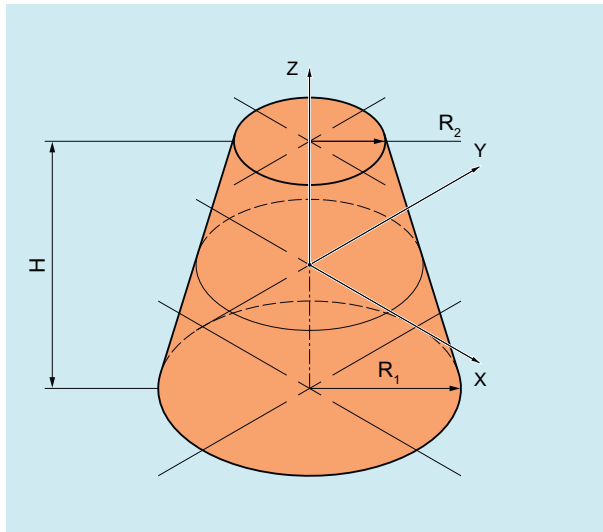
H Height in the Z direction

R Radius in the X/Y plane

A protection area element of the "CYLINDER" type defines a cylinder in the local coordinate system of the protection area element. The mid-point of the cylinder is at the origin of the local coordinate system. At the same time as the definition of the body, the local coordinate system can be transformed via the following system variables:

- Offset: \$NP_OFF (Page 60)
- Rotation direction vector: \$NP_DIR (Page 61)
- Angle of rotation: \$NP_ANG (Page 63)

The parameters height and radius are to be entered in \$NP_PARA (Page 59)

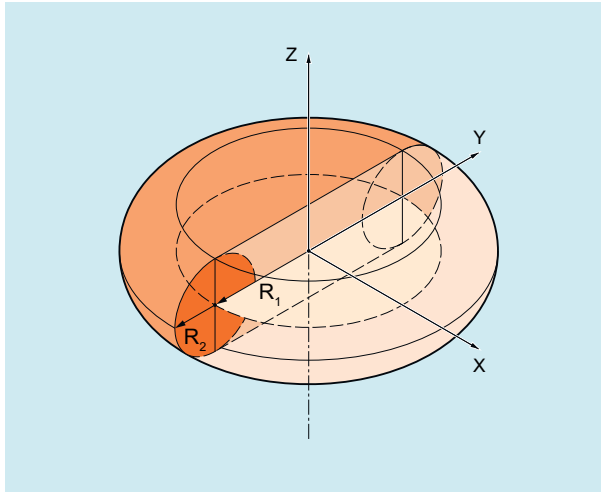
Type: "CONE"

- H Height in the Z direction
R1 Radius 1 in the X/Y plane
R2 Radius 2 in the X/Y plane

A protection area element of the "CONE" type defines a cone in the local coordinate system of the protection area element. The mid-point of the cone (half cone height on the line of symmetry of the cone) is at the origin of the local coordinate system. At the same time as the definition of the body, the local coordinate system can be transformed via the following system variables:

- Offset: \$NP_OFF (Page 60)
- Rotation direction vector: \$NP_DIR (Page 61)
- Angle of rotation: \$NP_ANG (Page 63)

The parameters height, radius 1, and radius 2 must be entered in \$NP_PARA (Page 59)

Type: "TORUS"

- R1 Major radius (= distance from the center of the circle to the center of the torus in the X/Y plane)
 R2 Minor radius (= radius of the circle)

A protection area element of the "TORUS" type defines a filled torus in the local coordinate system of the protection area element. The shape of a torus can most easily be described by a circle revolved around an axis that lies in its plane. In contrast to a normal torus, however, the hole in the middle of a filled torus is filled in. The center of the filled torus lies at the origin of the local coordinate system. At the same time as the definition of the body, the local coordinate system can be transformed via the following system variables:

- Offset: \$NP_OFF (Page 60)
- Rotation direction vector: \$NP_DIR (Page 61)
- Angle of rotation: \$NP_ANG (Page 63)

The parameters radius 1 and radius 2 must be entered in \$NP_PARA (Page 59)

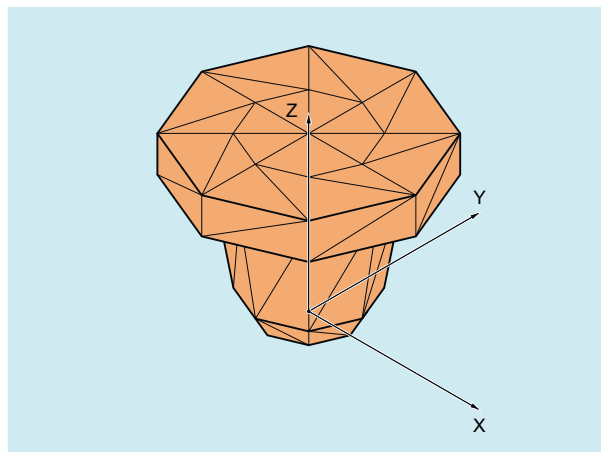
Type: "FILE"

Figure 3-3 Example bodies in STL format

A protection area element of the "FILE" type defines a body whose geometry data is contained in the specified file in STL format (triangular areas), in the local coordinate system of the protection area element. The zero point of the body is at the origin of the local coordinate system. At the same time as the definition of the body, the local coordinate system can be transformed via the following system variables:

- Offset: \$NP_OFF (Page 60)
- Rotation direction vector: \$NP_DIR (Page 61)
- Angle of rotation: \$NP_ANG (Page 63)

The parameter is to be entered in \$NP_FILENAME (Page 54):

Syntax

```
$NP_TYPE[<n>] = "<type>"
```

Meaning

\$NP_TYPE:	Type of the protection area element	
	Data type:	STRING
	Range of values:	"FRAME", "BOX", "SPHERE", "CYLINDER", "CONE", "TORUS", "FILE"
	Default value:	"" (empty string)
<n>:	System variable or protection area element index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - 1)
<Type>:	Type designation	
	Data type:	STRING

Example

The 19th protection area element is a cube:

Program code	Comment
N100 \$NP_TYPE[18] = "BOX"	; 19. Protection area element, ; type = "Box"

3.2.4.9 \$NP_FILENAME

Function

For protection area elements of the "FILE" type (\$NP_TYPE (Page 49)), enter the name of the file that describes the geometry data of the protection area element in the system variable.

The following file types are currently possible:

- STL files
- NPP files

STL files

An STL file (file extension .STL) must contain a description of the geometry data of a 3D body using triangles in the STL format (**S**tandard **T**essellation **L**anguage).

Search path

A search is made in the directories predefined in the static NC memory for the file entered in the system variables in the following sequence:

1. /oem/sinumerik/nck/prot_data/machine/3d_data/mm
2. /oem/sinumerik/nck/prot_data/machine/3d_data/inch

Interpretation of the length specifications

Depending on the storage directory, the length specifications contained in the STL file are interpreted in mm or inches:

- <path>/**mm**: Interpretation of the length specifications in millimeters
- <path>/**inch**: Interpretation of the length specifications in inches

Note

Maximum length of the file name

The maximum file name length, including point and the file extension is 49 characters. For more than 49 characters, an alarm is displayed when generating an archive.

NPP files

An NPP file (file extension .NPP) must contain a description of the geometry data of one or several protection area elements using NPP system variables (**NC** Protection Area **P**rimitives). All geometrical primitives available in the NC can be defined as protection area elements using an NPP file (see \$NP_TYPE (Page 49)).

Search path

A search is made in the directories predefined in the static NC memory for the file entered in the system variables in the following sequence:

1. /oem/sinumerik/nck/prot_data/machine/3d_data/mm
2. /oem/sinumerik/nck/prot_data/machine/3d_data/inch

Interpretation of the length specifications

Depending on the storage directory, the length specifications contained in the NPP file are interpreted in mm or inches:

- `<path>/mm`: Interpretation of the length specifications in millimeters
- `<path>/inch`: Interpretation of the length specifications in inches

NPP file properties

- An NPP file must start with the following comments lines:

```
;COLLISION AVOIDANCE DATA
;LOC_NP_ROOT_NAME="<root_name>"
```
- As `<root_name>`, the name of the first protection area element contained in the NPP file, specified there under `$NP_NAME`, must be entered.
- Comment line are lines that start with the `;` character
- NPP files are allowed to contain empty lines

Properties of NPP system variables

NPP system variables contained in the NPP file have the following properties:

- Same name, significance and syntax as the corresponding system variables used in NC programs
- The system variables of the NC are **not** overwritten by the NPP system variables.
- **Within** an NPP file, the indices of NPP system variables must be **unique**.
- The indices of NPP system variables can be the **same** in **various** NPP files.
- The indices and the values assigned in the NPP system variables must be constants.

Constraints

- The values for `$NP_COLOR` (Page 46), `$NP_D_LEVEL` (Page 47), `$NP_USAGE` (Page 48) are, for the protection area elements defined in the NPP file, inherited from the protection area element in which they are integrated. As a consequence, all protection area elements of an NPP file have the same values for these properties.
- For positioning the protection area elements of an NPP file, the same conditions apply as for positioning the protection area elements with the system variables of the NC (`$NP_TYPE` (Page 49): "BOX", "SPHERE" and "CYLINDER").
- No additional STL or NPP files are embedded in an NPP file.

Note**Maximum length of the file name**

The maximum file name length, including point and the file extension is 49 characters. For more than 49 characters, an alarm is displayed when generating an archive.

Syntax

```
$NP_FILENAME[<n>] = "<name>"
```

Meaning

\$NP_FILENAME:	Name of the STL or NPP file	
	Data type:	STRING
	Default value:	"" (empty string)
<n>:	System variable or protection area element index	
	Data type:	INT
	Value range:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - 1)
<name>:	Name of the STL or NPP file	
	Data type:	STRING

Examples

Using an STL file

The geometry data for the 19th protection area element are stored in file KUEHLDUESE_1.STL:

Program code	Comment
N100 \$NP_FILENAME[18] = "KUEHLDUESE_1.STL"	; 19th protection area element, ; file name = "KUEHLDUESE_1.STL"

Using an NPP file

The NPP file "Kopf_A.NPP" file is loaded into the system variables of the NC for the 19th protection area element. This contains the following three protection area elements "Box-1", "Sphere-1" and "Cylinder-1".

Program code	Comment
\$NP_NAME[18] = "Head"	; 19th protection area element
\$NP_NEXT[18] = ""	
\$NP_NEXTP[18] = ""	
\$NP_TYPE[18] = "FILE"	
\$NP_OFF[18,0] = 80.0	
\$NP_OFF[18,1] = 100.0	
\$NP_OFF[18,2] = -50.0	
\$NP_DIR[18,0] = 0.0	
\$NP_DIR[18,1] = 0.0	
\$NP_DIR[18,2] = 0.0	
\$NP_ANG[18] = 0.0	
\$NP_COLOR[18] = 0	; 1)
\$NP_D_LEVEL[18] = 0	; 1)
\$NP_USAGE[18] = "A"	; 1)
\$NP_FILENAME[18] = "Kopf_A.npp"	
; 1) Active for all protection area elements loaded from file Kopf_A.NPP	

Content of file "Kopf_A.NPP"

Program code	Comment
;COLLISION AVOIDANCE DATA	; 1st header
;LOC_NP_ROOT_NAME = "Box-1"	; 2nd header
 \$NP_NAME[0] = "Box-1"	 ; 1st protection area element
\$NP_NEXT[0] = "Sphere-1"	
\$NP_NEXTP[0] = "Cylinder-1"	
\$NP_TYPE[0] = "BOX"	
\$NP_PARA[0,0] = 340	
\$NP_PARA[0,1] = 340	
\$NP_PARA[0,2] = 340	
\$NP_OFF[0,0] = 42	
\$NP_OFF[0,1] = 73	
\$NP_OFF[0,2] = -100	
\$NP_DIR[0,0] = 0	
\$NP_DIR[0,1] = 0	
\$NP_DIR[0,2] = 1	
\$NP_ANG[0] = 30	
 \$NP_NAME[1] = "Sphere-1"	 ; 2nd protection area element
\$NP_NEXT[1] = ""	
\$NP_NEXTP[1] = ""	
\$NP_TYPE[1] = "SPHERE"	
\$NP_PARA[1,0] = 20	
\$NP_PARA[1,1] = 0	
\$NP_PARA[1,2] = 0	
\$NP_OFF[1,0] = 170	
\$NP_OFF[1,1] = 170	
\$NP_OFF[1,2] = 170	
\$NP_DIR[1,0] = 0	
\$NP_DIR[1,1] = 0	
\$NP_DIR[1,2] = 0	
\$NP_ANG[1] = 0	
 \$NP_NAME[2] = "Cylinder-1"	 ; 3rd protection area element
\$NP_NEXT[2] = ""	
\$NP_NEXTP[2] = ""	
\$NP_TYPE[2] = "CYLINDER"	
\$NP_PARA[2,0] = 20	
\$NP_PARA[2,1] = 20	
\$NP_PARA[2,2] = 0	
\$NP_OFF[2,0] = 170	
\$NP_OFF[2,1] = 170	
\$NP_OFF[2,2] = 170	

Program code	Comment
\$NP_DIR[2,0] = 1	
\$NP_DIR[2,1] = 2	
\$NP_DIR[2,2] = 3	
\$NP_ANG[2] = 73	

3.2.4.10 \$NP_PARA

Function

Enter the dimensions of the protection area body according to the type of the protection area element (\$NP_TYPE (Page 49)) in the system variable.

Coordinate system

The local coordinate system in which the position of the protection area body is specified, is defined by the system variables \$NP_OFF (Page 60), \$NP_DIR (Page 61), \$NP_ANG (Page 63).

Syntax

\$NP_PARA[<n>,<i>] = <value>

Meaning

\$NP_PARA:	Parameter values corresponding to the type of the protection area element					
	Data type:	REAL				
	Default value:	0.0				
<n>:	System variable or protection area element index					
	Data type:	INT				
	Value range:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - 1)				
<i>:	Parameter index					
	Data type:	INT				
	Value range:	0, 1, 2				
	Parameter index	Type of the protection area element				
		BOX	SPHERE	CYLINDER	CONE	TORUS
	0	Length in X	Radius	Height in Z	Height in Z	Radius 1 ¹⁾
	1	Width in Y	---	Radius ¹⁾	Radius 1 ¹⁾	Radius 2
	2	Height in Z	---	---	Radius 2 ^{1) 2)}	---

3.2 Commissioning

<value>:	Parameter value	
	Data type:	REAL
	Value range:	$0.0 < x \leq \text{max. REAL value}$

- 1) Radius in the X/Y plane
 2) Parameter value 0.0 is permissible.
 ---: Parameters that are not evaluated

Example

The 19th protection area element is a cube with the dimensions:

- Length: 50.0 in the X axis
- Width: 100.0 in the Y axis
- Height: 75.5 in the Z axis

Program code	Comment
; 19. protection area element	
N100 \$NP_TYPE[18] = "BOX"	; type = "BOX"
N120 \$NP_PARA[18,0] = 50.0	; length in X = 50.0
N130 \$NP_PARA[18,1] = 100.0	; width in Y = 100.0
N140 \$NP_PARA[18,2] = 75.5	; height in Z = 75.5

3.2.4.11 \$NP_OFF

Function

The offset vector by which the local coordinate system of the protection area element is moved compared to the coordinate system of the previous protection area element should be entered in the system variable.

Syntax

`$NP_OFF[<n>,<i>] = <value>`

Meaning

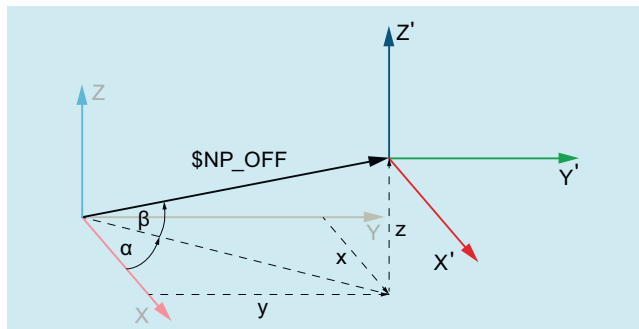
\$NP_OFF:	Offset vector	
	Data type:	REAL
	Value range:	$-\text{max. REAL value} \leq x \leq +\text{max. REAL value}$
	Default value:	(0.0, 0.0, 0.0)
<m>:	System variable or protection area element index	
	Data type:	INT
	Value range:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - 1)

<i>:	Coordinate index	
	Data type:	INT
	Value range:	0: X coordinate (abscissa) 1: Y coordinate (ordinate) 2: Z coordinate (applicata)
<value>:	Coordinate value	
	Data type:	REAL
	Value range:	- max. REAL value $\leq x \leq$ + max. REAL value

Example

The local coordinate system of the 19th The local coordinate system of the protection area element is moved by the following vector compared to the coordinate system of the previous protection area element:

- X direction: 25.0
- Y direction: 50.0
- Z direction: 37.25



X, Y, Z Coordinate system of the previous protection area element

X', Y', Z' Coordinate system of the current protection area element

Program code	Comment
; 19. protection area element, offset vector	
N100 \$NP_OFF[18,0] = 25.0	X = 25.0
N110 \$NP_OFF[18,1] = 50.0	Y = 50.0
N120 \$NP_OFF[18,2] = 37.25	Z = 37.25

3.2.4.12 \$NP_DIR

Function

The direction vector through which the local coordinate system of the protection area element is rotated compared to the coordinate system of the previous protection area element should be entered in the system variable. The angle of rotation should be entered in \$NP_ANG (Page 63).

Boundary conditions

- The absolute value of the direction vector must be greater than: $1 \cdot 10^{-6}$
- A work offset parameterized in \$NP_OFF (Page 60) is performed **before** the rotation.

Syntax

```
$NP_DIR[<n>,<i>] = <value>
```

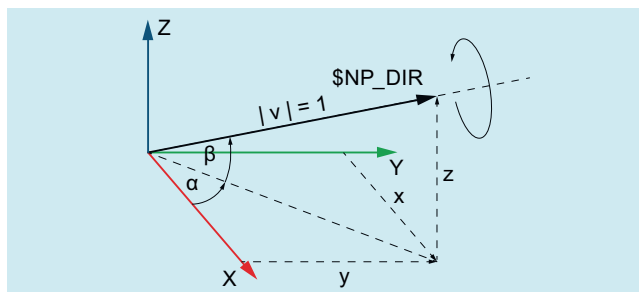
Meaning

\$NP_DIR:	Direction vector	
	Data type:	REAL
	Range of values:	- max. REAL value $\leq x \leq \pm$ max. REAL value
	Default value:	(0.0, 0.0, 0.0)
<n>:	System variable or protection area element index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - 1)
<i>:	Coordinate index	
	Data type:	INT
	Range of values:	0 $\rightarrow$ X; 1 $\rightarrow$ Y; 2 $\rightarrow$ Z
<value>:	Coordinate value	
	Data type:	REAL
	Range of values:	- max. REAL value $\leq x \leq \pm$ max. REAL value

Example

The local coordinate system of the 19th protection area element is rotated around the direction vector compared to the coordinate system of the previous protection area element. The direction vector is the unit vector (1; 0; 0), rotated through $\alpha=90^\circ$ in the X/Y plane and $\beta=10^\circ$ in the Y/Z plane, in relation to the world coordinate system. The following values result from this for the individual components of the direction vector:

- X component = $\cos(\alpha) \cdot \cos(\beta) = \cos(90) \cdot \cos(10) = 0.0$
- Y component = $\sin(\alpha) \cdot \cos(\beta) = \sin(90) \cdot \cos(10) \approx 0.985$
- Z component = $\sin(\beta) = \sin(10) \approx 0.174$



Program code	Comment
; 19. protection area element, direction vector	
N100 \$NP_DIR[18.0] = COS(90) * COS(10)	; 0 = X component
N110 \$NP_DIR[18.1] = SIN(90) * COS(10)	; 1 = Y component
N120 \$NP_DIR[18.2] = SIN(10)	; 2 = Z component

3.2.4.13 \$NP_ANG

Function

The angle through which the local coordinate system of the protection area element is rotated around the direction vector (\$NP_DIR (Page 61)) compared to the coordinate system of the previous protection area element should be entered in the system variable.

Syntax

\$NP_ANG[<n>] = <value>

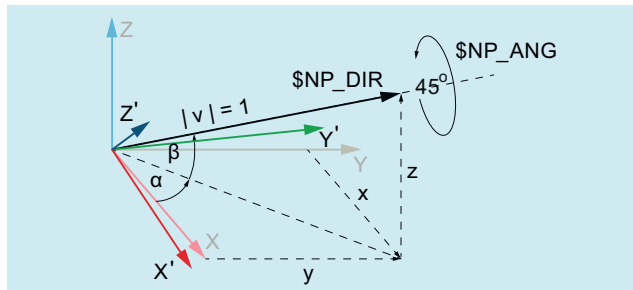
Meaning

\$NP_ANG:	Angle of rotation	
	Data type:	REAL
	Range of values:	$-360^{\circ} < x \leq 360^{\circ}$
	Default value:	0.0
<n>:	System variable or protection area element index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (\$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - 1)
<value>:	Angle	
	Data type:	REAL

Example

The local coordinate system of the 19th protection area element is rotated by the angle by $\delta=45.0^{\circ}$ around the direction vector compared to the coordinate system of the previous protection area element. The direction vector is the unit vector (1; 0; 0), rotated through $\alpha=90^{\circ}$ in the X/Y plane and $\beta=10^{\circ}$ in the Y/Z plane, in relation to the world coordinate system. The following values result from this for the individual components of the direction vector:

- X component = $\cos(\alpha) * \cos(\beta) = \cos(90) * \cos(10) = 0.0$
- Y component = $\sin(\alpha) * \cos(\beta) = \sin(90) * \cos(10) \approx 0.985$
- Z component = $\sin(\beta) = \sin(10) \approx 0.174$
- Angle $\delta = 45.0^{\circ}$



Program code	Comment
; 19. Protection area element, direction vector and angle of rotation	
N100 \$NP_DIR[18.0] = COS(90)*COS(10)	; 0 = X component
N110 \$NP_DIR[18.1] = SIN(90)*COS(10)	; 1 = Y component
N120 \$NP_DIR[18.2] = SIN(10)	; 2 = Z component
N130 \$NP_ANG[18] = 45.0	; Angle of rotation $\delta = 45^\circ$

3.2.5 System variables: Protection area elements for automatic tool protection areas

The protection area element of an automatic tool protection area is defined using the subsequent system variables. The control automatically generates the values of the system variables from the geometric data of the associated tool, and these can only be read.

Name ¹⁾	Meaning	Analog to ²⁾
\$NP_T_NAME[<n>]	Name of the protection area element	\$NP_NAME (Page 42)
\$NP_T_TYPE[<n>]	Type of the protection area element	\$NP_TYPE (Page 49)
\$NP_T_FILENAME[<n>]	File name of the STL/NPP file that contains the geometric data of the protection area element (only relevant for \$NP_T_TYPE == "FILE")	\$NP_FILENAME (Page 54)
\$NP_T_PARA[<n>,<i>]	Geometric parameters of the protection area body (only relevant for \$NP_T_TYPE == "BOX", "SPHERE", "CYLINDER", "CONE" or "TORUS")	\$NP_PARA (Page 59)
\$NP_T_OFF[<n>,<i>]	Offset vector of the protection area element local coordinate system	\$NP_OFF (Page 60)
\$NP_T_DIR[<n>,<i>]	Direction vector for the rotation of the protection area element local coordinate system	\$NP_DIR (Page 61)
\$NP_T_ANG[<n>]	Angle for the rotation of the protection area element local coordinate system	\$NP_ANG (Page 63)
1) n = 0, 1, ... (\$MN_MM_MAXNUM_3D_T_PROT_ELEM - 1)		
2) The system variables of the automatic tool protection areas correspond to those of the machine protection areas.		

3.2.6 Boundary conditions

Protection area bodies for spindles

For spindles that are not in position-controlled operation, the associated protection area bodies are only modeled statically. As a result the following boundary conditions must be met when modeling protection area bodies connected with a spindle as a kinematic element:

- The protection area body **must be symmetrical around its axis of rotation**.
- The symmetrical axis of the protection area body must lie on the axis of rotation of the spindle (**collinear**)

This should be taken into account for all types of protection area bodies:

- The protection area body is a basic geometrical body (sphere, cylinder, cone or torus).
- The protection area body comprises several, geometrical basic bodies.
- The protection area body comprises triangles (STL file).
- The protection area body for an automatic tool protection area is created from the geometric tool data.

Note

Automatic tool protection areas

Only using tools that are symmetrical around the axis of rotation is recommended for automatic tool protection areas in connection with spindles.

Following protection areas

The rotational symmetry and collinearity of the protection area body in respect of the axis of rotation of the spindle must also be complied with for all protection areas that are connected with elements of the kinematic chain (\$NP_NEXT, \$NP_NEXTP) following on from the spindle.

Schematic example of such a kinematic chain: ... → (rotary axis/spindle) → (offset) → (linear axis) → (offset) → ...

Rotary axes

The boundary conditions stated above must also be observed in connection with rotary axes where these are also operated as spindles.

Tool reference point and kinematic transformation

The position of the associated tool reference point is determined in principle through the assignment of an automatic tool protection area to an element in the kinematic chain. However, the position of the tool reference point can be changed through offsets within the automatic tool protection area via the system variables \$NP_T_OFF, \$NP_T_DIR and \$NP_T_ANG (Page 64). A change in position for the tool reference point of this type is not captured by the kinematic transformations. In the context of kinematic transformation it is therefore highly recommended that no offsets are used for the geometric machine modeling which move the tool reference point.

Collision avoidance, internal

4.1 Function description

4.1.1 Options

The function "Collision avoidance" is an option that requires a license. The following versions are available:

- **Protect MyMachine /3D Primitives**
Properties:
 - Protection: Machine - Machine
 - HMI visualization
 - Multi-channel control configurations
 - Protection area elements: geometric primitives, files in the NPP format
- **Protect MyMachine /3D STL**
Properties:
 - such as Protect MyMachine /3D Primitives
 - Protection area elements: Files in STL and NPP format

4.1.2 Characteristics

The "Collision avoidance" function is used to prevent collisions of machine parts and tool cutting edges while the machine axes are being traversed. To do this, the function cyclically calculates the clearance to the protection areas enveloping the bodies to be protected. If two protection areas approach one another to within a configurable safety clearance, an alarm is displayed and the NC program stopped before the relevant traversing block (AUTOMATIC, MDI mode) or the traversing motion is stopped (JOG mode).

Sequence

The collision avoidance is set up in the following sequence:

1. Enabling of the "Collision avoidance" function by setting the corresponding option.
2. Setting the machine data for the basic parameterization of the functions:
 - Kinematic chain, Function Manual "Basic Functions", chapter "Kinematic chain"
 - Geometric machine modeling, chapter "Machine data (Page 28)"
 - Collision avoidance, chapter "Machine data (Page 80)"

4.1 Function description

3. Description of the kinematic structure of the machine with kinematic elements.
Refer to Function Manual "Basic Functions", chapter "Kinematic chain".
4. Writing of the protection areas and protection area elements as enveloping geometry of the machine parts, tools and workpieces to be protected. Assignment of the protection area to elements of the kinematic chain.
See Section "Geometric machine modeling (Page 19)".
5. Definition of collision pairs, i.e. of two protection areas that are to be monitored for collision.
See chapter "\$NP_COLL_PAIR (Page 83)".
6. Triggering a recalculation of the kinematic and geometric model.
See chapter "Request recalculation of the machine model of the collision avoidance (PROTA) (Page 88)".
7. Activation of the protection areas to be monitored.
See chapter "Setting the protection zone status (PROTS) (Page 89)".
8. Optional: Use of the extended functions and system variables

Limits of the collision avoidance

The function cannot guarantee complete protection against a collision when traversing machine parts, tools or workpieces. On the one hand, the collision protection can only be as good as the parameterized kinematic and geometric model or the machine and the protection areas. On the other hand, bodies that have not been modeled cannot be monitored at all. For this reason, it remains the responsibility of the machine operator to ensure that a traversing motion is executed collision-free even when the collision protection is activated.

States of protection areas

Whether a protection area is taken into account for collisions depends on the state of the protection area:

State	Meaning
Active	The protection area is taken into account for collisions.
Inactive	The protection area is not taken into account for collisions.
Preactivated	The protection area is taken into account for collisions. A collision alarm is only issued when it has also been activated by a protection area-specific NC/PLC interface signal.

State after control ramp-up

After the control has powered up, all protection areas are in the state corresponding to their respective setting in \$NP_INIT_STAT. See Section "\$NP_INIT_STAT (Page 38)".

State change

The state of a protection areas can be changed by:

- The PROTS() (Page 89) procedure
- Change of the initialization state to \$NP_INIT_STAT followed by recalculation of the machine model through the PROTA() (Page 88) procedure.

Requirements

The following requirements must be satisfied so that the protection areas of a collision pair can be monitored:

- **Axes or spindles: referenced/synchronized**
The position measurement system for the axes or spindles which move a protection area must be referenced or synchronized. If this has not been done the corresponding protection area is in an "inactive" state.
- **External motion**
With traversing motions that are not performed by the NC, e.g. PLC axis or manually moved axis, the current axis position must be known in the NC.
- **How protection area elements are used**
When forming a collision pair, it should be observed that every protection area of this pair has at least one protection area element that is marked with \$NP_USAGE = "C" or "A" (Page 48). Otherwise, it is not permissible to perform a collision or clearance calculation (Page 90) using the protection area.
- **Protection area state**
The protection areas of a collision pair are only checked for collision when both protection areas are in the "Protection area monitored" state. See "Status diagram protection area (Page 72)".

If the preconditions are satisfied for at least one collision pair, then the following signal is set at the NC/PLC interface:

`<Nc>.basic.in.collisionMonitoringActive` (collision avoidance: protection area monitoring active)

PLC signals

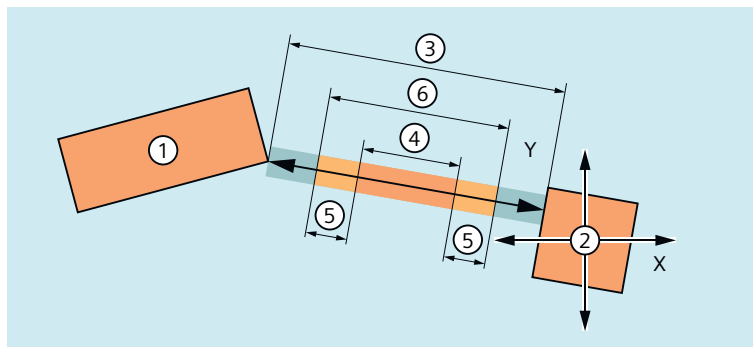
Basic Program Plus	Basic Program	
<code><Nc>.basic.in.collisionMonitoringActive</code>	<code>LBP_NC.E_CollCheckOff</code>	<code>DB10.DBX106.0</code>

4.1.3 Reaction of the control to a risk of collision

The collision avoidance takes the following parameterizable limit values into account for the collision detection:

- Collision tolerance
- Safety clearance

4.1 Function description



- ① Protection area 1 (static)
- ② Protection area 2 (can be moved in X and Y direction)
- ③ Current clearance
- ④ Safety clearance
- ⑤ Collision tolerance / 2
- ⑥ Collision clearance = safety clearance + collision tolerance

Figure 4-1 Current clearance, collision tolerance and safety clearance

Collision tolerance and safety clearance

Safety clearance

The safety clearance defines a clearance up to which two active protection areas, monitored for collision, can approach one another. The collision avoidance ensures that this clearance is not violated and that the collision is displayed.

The safety clearance can be set specifically for each collision pair via a system variable (Page 85).

For all collision pairs for which no specific safety clearance is set via a system variable, the general value that can be set via MD10622 \$MN_COLLISION_SAFETY_DIST (Page 80) applies.

Collision tolerance

The collision tolerance defines an NC-wide additional clearance valid to the safety clearance. Two active protection areas monitored for collision may therefore approach each other up to the collision clearance (safety clearance + collision tolerance). Ideally, the collision avoidance stops the traversing motion of the protection areas exactly at the collision clearance and displays the collision. However, it is permissible that the collision tolerance is not complied with exactly or that a brief violation does not result in collision detection and the traversing motion is not stopped.

The collision tolerance is set the same for all collision pairs via MD10619 \$MN_COLLISION_TOLERANCE (Page 80).

Note**Difference between collision tolerance and safety clearance**

The collision tolerance can be violated and is permissible. The safety clearance is always maintained.

Responses if there is a risk of collision

The following responses occur if the control identifies a pending collision:

- All machine axes, with the exception of the spindles, are stopped along the original path planned by the control before the collision clearance was violated.
- The following channel-specific signal is set at the NC/PLC interface:
`<Chan>.basic.in.collisionStopActive == 1` (collision avoidance: stop)
- Depending on which operating mode is currently active, one of the two following alarms is output:

Operating mode	Alarm		Delete criterion
AUTOMATIC	26265	"Channel <channel number> block <block number>: Collision of the two protection areas <name of the 1st protection area> and <name of the 2nd protection area>"	Must be acknowledged by the user with a reset.
JOG	26275	"Channel <channel number>: Collision of the two protection areas <name of the 1st protection area> and <name of the 2nd protection area>"	Self-clearing.

- Program execution is canceled.

Risk of collision cannot be predicted

Situations exist in which a collision cannot be predicted. One example involves realtime motion such as positioning axis motion from synchronized actions or the PLC, which can result in a collision immediately after being started. In situations such as this, the control identifies the collision based on a check of the actual setpoint positions, and the following responses are initiated:

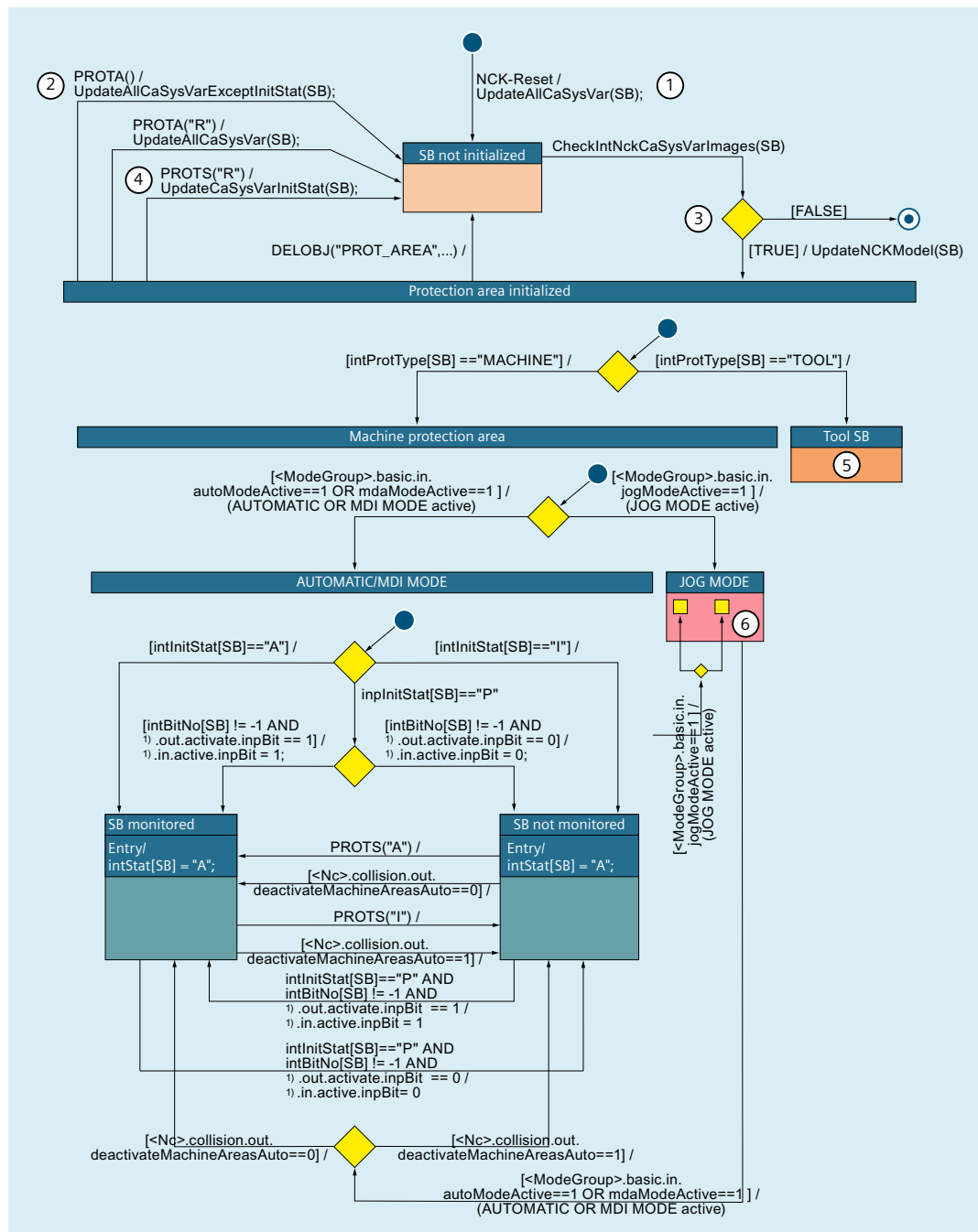
- Fast stop for all axes, with the exception of the spindles.
- Alarm 26285 is output:
"Channel <channel number> block <block number>: Emergency Stop due to a collision
<1st protection area> <2nd protection area>
The user must acknowledge alarm 26285 with a reset.

Depending on how long the braking distances of the axes are, in cases such as these, it is possible that a collision cannot be prevented. If this occurs, or the safety clearance is violated, then the user must deactivate collision avoidance and move the axes from the collision position.

PLC signals

Basic Program Plus	Basic Program	
<code><Chan>.basic.in.collisionStopActive</code>	<code>LBP_Chan*.E_CollCheckStop</code>	<code>DB21,DBX377.0</code>

4.1.4 Status diagram protection area



SB Protection area

BA Operating mode

① UpdateAllCaSysVar(SB)

All system variables of the collision avoidance are read into NC-internal variables:
int... = \$N...

- ② UpdateAllCaSysVarExeptlInitStat(SB) function
As with the UpdateAllCaSysVar(SB) function, but the \$NP_INIT_STAT system variable is not read in. Internally in the NC, the last value of the intlInitStat initialization status is therefore retained.
- ③ CheckIntNckCaSysVarlImages(SB) function
The NC-internal variables read in from the system variables are checked for consistency.
Return value when an error is detected: FALSE; if error-free: TRUE.
- ④ UpdateCaSysVarlInitStat(SB) function
Only the system variable \$NP_INIT_STAT is read into the NC-internal variable intlInitStat.
- ⑤ The internal structure of the "Tool protection area" state is the same as the "Machine protection area" state.
- ⑥ The internal structure of the "JOG mode" state is the same as the "AUTO/MDI mode" state.
- 1) <Nc>.collisionProtAreas

PLC signals

Basic Program Plus	Basic Program	
<ModeGroup>.basic.in.autoModeActive	LBP_ModeGroup.E_AUTO	DB11.DBX6.0
<ModeGroup>.basic.in.mdaModeActive	LBP_ModeGroup.E_MDA	DB11.DBX6.1
<ModeGroup>.basic.in.jogModeActive	LBP_ModeGroup.E_JOG	DB11.DBX6.2
<Nc>.collisionProtAreas.out.activate	LBP_NC.A_CollCheck	DB10.DBX234.0..241.7
<Nc>.collisionProtAreas.in.active	LBP_NC.E_CollCheck	DB10.DBX226.0..233.7
<Nc>.collision.out.deactivateMachineAreasAuto	LBP_NC.A_CA_DisableAutoMach	DB10.DBX58.0

4.1.5 Tools

Modeling

Protection areas for tools can be modeled automatically from the collision avoidance and be updated automatically following a tool change with some limitations. The following conditions must be fulfilled for this purpose:

- The protection area for the tool is modeled as an automatic tool protection area (type: TOOL). See system variable \$NP_PROT_TYPE (Page 32)
- The tool is managed by the control tool management.
- The tool data stored in the tool management matches the actual geometric dimensions of the tool.
- The collision avoidance recognizes the completion of the tool change. Normally, by programming the corresponding tool offset number D_x , with $x = 0, 1, 2, 3, \dots$

4.1 Function description

- The collision avoidance recognizes the automatic tool protection area for which the tool was changed.
- If the tool path refers to a normalized tool, the tool radius of the actual tool is specified as either positive or negative deviations referred to the normalized tool. In this case, collision avoidance calculates with the following values:
 - Positive value: Tool radius = specified value, however, as a minimum, the value of the parameterized collision tolerance
 - Negative value: Tool radius = value of the parameterized collision tolerance

Change to the machine model

If a tool which is located in a magazine or tool adapter modeled in the active machine model of the collision avoidance is changed in the machine then the machine model must be updated. This is the case where one of the following actions is carried out, for example:

- A tool is loaded / unloaded in a tool magazine modeled in the machine model.
Example: Tool change in a revolver (circular) magazine.
Update: The machine model update must be explicitly requested by the user using `PROTA` once the tool change has completed.
- The tool that is in a tool holder is changed.
Example: Tool change in the tool holder of the main spindle.
Update: The machine model is automatically updated following the tool change (standard: M6) with output of the programmed tool offset number `Dx` in the main run.

Tool changes

The collision avoidance only updates the active machine model after a tool change without an explicit request by `PROTA` if a tool offset is selected (output of the programmed tool offset number `Dx` at the NC/PLC interface).

However, at the control, tool changes can be made, which are not associated with a tool offset selection. However, for these changes the active machine model is not updated. These types of tool changes include, for example:

- Loading/unloading a tool at the user interface.
`SINUMERIK Operate: Operating area "Parameter" > "Tool list" > Vertical softkey: "Loading" or "Unloading"`
- Carrying out a tool change via the PLC user program
- Directly writing to the tool buffer memory using the system variable `$TC_MPP6[9998, <location>]`
- `SETMS(<spindle number>):` Changing the master spindle in the channel
- `TMMVTL:` PI service "Prepare magazine location for loading, unload tool"
Further information: Function Manual PLC; PI services > PI service: TMMVTL
- `MVTOOL:` Command for movement of a tool
Further information: Function Manual Tool Management; NC programming > NC language commands > MVTOOL - Language command for movement of a tool

If such a change is carried out, then the machine manufacturer must be requested to update the machine module via the PLC user program. Examples of options include:

- A new channel reset is requested if the channel is in the "reset" state. When the reset response is appropriately set (MD20110 \$MC_RESET_MODE_MASK), then the actual tool offset number D_x is output again.
- Starting an ASUB or manufacturer's cycle, which includes the output of the tool offset number D_x and the request to update the machine model (PROTA).

Tool wear

Minimal tool changes do not have to be taken into account in the machine model, as generally they are far less than the collision clearance.

If a tool change occurs, which is relevant for collision avoidance, e.g. diameter changes for grinding tools, then these must be taken into account by explicitly requesting that the machine model is updated (PROTA).

No change to the machine model

The active machine model does **not** change if a fully modeled machine part with tools, e.g. a tool magazine, is moved in the machine.

Example: Revolver magazine on a lathe

The revolver magazine on a lathe is fully modeled in the machine model of the collision avoidance:

- The geometry of the magazine and the tools located inside it
- The motions of the magazine through the machine axes

A rotation of the revolver magazine then does not represent a change to the machine model:

- since no tools are changed inside the machine model the geometries of all the protection areas remain unchanged.
- As the motions of the protection areas by the machine axes are fully captured by the collision avoidance via the kinematic chain.

Constraints

Several spindles in the channel

For configurations with several spindles in the channel, the collision avoidance function assumes that a tool change takes place in the master spindle of the channel (S1). As a result only the automatic tool protection area of the master spindle is updated through the collision avoidance once the tool change has taken place.

Unsupported tool configurations

Tool configurations in accordance with ISO modes 4 and 5 (H numbers) along with "flat D numbers" are not supported by the collision avoidance.

4.1.6 Boundary conditions

Allowance for the following error

The collision avoidance uses the position setpoints of the relevant machine axes for the clearance calculation of the protection areas. However, the actual positions of the machine axes differ from the position setpoint by the following error. For this reason, there is also a difference between the position setpoint and the actual position for the protection areas. This difference must be taken into account by the user through configuration of a suitably large safety clearance or enlargement of the protection area.

Compensations

The various compensation functions of the NC – for instance, temperature, spindle and pitch error and sag compensation – ensure that positions programmed in the workpiece coordinate system are actually assumed in the machine coordinate system. The collision avoidance takes into account the position corrections made by the compensation functions.

**WARNING****Risk of collision**

If compensations are used for other purposes, e.g. in order to implement axis couplings in the machine coordinate system, the collision avoidance working with position setpoints can no longer be performed reliably. There is a risk of collision.

Actual value offset in the machine coordinate system (PRESETON)

With collision avoidance active and if an actual value offset is applied in the machine coordinate system (PRESETON), then the user is solely responsible in ensuring that the geometric model of the collision avoidance remains consistent.

**WARNING****Risk of collision**

If an actual value offset is made in the machine coordinate system with PRESETON and the geometric model of the collision avoidance not adapted accordingly, the collision avoidance working with position setpoints can no longer be performed reliably. There is a risk of collision.

Block search

For the following types of block search there are **no** collision calculations carried out:

- Type 1: Block search without calculation
- Type 2: Block search with calculation at contour
- Type 4: Block search with calculation at block end point

With the following type of block search collision calculations are carried out (in the preprocessing):

- Type 5: Block search with calculation in "Program test" (SERUPRO) mode

AUTOMATIC modes: Incomplete protection area data for collision

When a large number of protection areas have been configured, in exceptional cases, the following behavior can occur:

- Several protection areas have approached one another, down to the collision tolerance.
- Only two protection areas are specified in Alarm 26260 "Collision **two** protection areas" that is displayed.
- The collision of the other protection areas is only displayed when manually traversing the axes after changing into the JOG mode.

Incomplete collision check

Collision avoidance is a function that takes up a lot of CPU time. Depending on the hardware being used, the interpolation clock cycle set, the size of the collision model and other NC functions that involve a lot of CPU time that are simultaneously running, it is possible that the collision checks cannot be completed in the required time. In this case, collision protection is restricted and Alarm 26303 "Collision check incomplete" is output. The machine manufacturer can suppress this alarm (MD11415 \$MN_SUPPRESS_ALARM_MASK_2).

Unsupported tool configurations

Collision avoidance does not support multitools. Only the selected tool of the multitool is monitored.

4.2 Commissioning

4.2.1 General

4.2.1.1 Overview

The commissioning of the "Collision avoidance" function is performed using:

- Option data
 - Enabling the function
- Machine data
 - Specification of the quantity structure
 - Specification of general properties of the collision pairs
- System variables
 - Parameterization of the collision pairs and their properties

4.2.1.2 Structure of the system variables

The system variables are structured according to the following scheme:

\$NP_<name>[<index_1>,<index_2>]

Note

Index_2 is not available for all system variables.

General

The system variables to describe the protection areas have the following properties:

- Prefix: **\$NP_**, (N for NC, P for protection).
- They can be read and written via NC programs.
- They can be stored in archives and loaded to the NC again.

Data type

STRING

All system variables of the STRING data type have the following properties:

- Maximum string length: 31 characters
- No distinction is made between upper and lower case
Example: "Axis1" is identical to "AXIS1"

- Spaces and special characters are permitted
Example: "Axis1" is not identical to "Axis 1"
- Names that **start** with **two** underscores "__" are reserved for system purposes and must **not** be used for user-defined names.

Note**Leading space**

Since spaces are valid and distinct characters, names that **start** with a **space** followed by **two** underscores "__" can, in principle, be used for user-defined names. However, because they can be easily mistaken for system names, this procedure is **not** recommended.

Index_1

The individual protection areas are addressed via index_1. Index 0 → 1st protection area, index 1 → 2nd protection area, ... n → (n+1) protection area, where n = (\$MN_MM_MAXNUM_3D_PROT_AREAS - 1)

All system variables of a protection area have the same index.

Index_2

For system variables that define a collision pair, the protection areas of the collision pair are addressed via index_2.

- 0 → 1. Protection area
- 1 → 2. Protection area

4.2.2 Option data

The following option data is relevant when enabling collision avoidance with the required functionality:

- MD19830 \$ON_COLLISION_MASK (collision avoidance functional scope)

Note

Alarm 8032 is output if options are set in MD19830 that cannot be active simultaneously.

- MD19840 \$ON_NUM_CHANNELS_CA_MULTICHAN (maximum number of channels for collision avoidance, multi channel)

Note

Alarm 8029 is output when an attempt is made to use more channels than are enabled in MD19840.

4.2.3 Machine data

4.2.3.1 Collision tolerance

The collision tolerance (accuracy of the collision check) for all protection areas of the NC monitored for collision is set with the machine data. If the clearance of two safety areas is less than the collision clearance, i.e. the sum of safety clearance (Page 80) and collision tolerance, then there is a collision.

MD10619 \$MN_COLLISION_TOLERANCE = <collision tolerance>

Accuracy of automatic generated protection areas

The collision tolerance also determines the accuracy of the protection area bodies of protection areas produced automatically, e.g. automatic tool protection areas. The accuracy of the protection area bodies approached using triangle areas is 1/3 of the collision tolerance.

Effects

The smaller the collision tolerance is set, the greater the number of triangular areas required for the modelling of the automatically generated protection areas and the computation time required for the collision detection.

Recommended setting

Collision tolerance $\approx$ 1 mm

4.2.3.2 Safety clearance

The safety clearance for all protection areas of the NC monitored for collision is set with the machine data. The collision avoidance ensures that the safety clearance is not violated.

MD10622 \$MN_COLLISION_SAFETY_DIST = <safety clearance>

Note

Collision pair-specific safety clearance

If a specific safety clearance has been set for a collision pair via the system variable \$NP_SAFETY_DIST (Page 85), this has priority over the NC-specific safety clearance set in the machine data.

4.2.3.3 Maximum memory space

The maximum value of the memory space in KB that can be allocated to the collision avoidance is set with the machine data.

MD18896 \$MN_MM_MAXNUM_3D_COLLISION = <Value>

Value	Meaning
0	The maximum value of the memory space is determined automatically by the control using the following machine data: - MD18890 \$MN_MM_MAXNUM_3D_PROT_AREAS - MD18892 \$MN_MM_MAXNUM_3D_PROT_AREA_ELEM - MD18894 \$MN_MM_MAXNUM_3D_FACETS_INTERN - MD18895 \$MN_MM_MAXNUM_3D_FACETS
> 0	Maximum value = parameterized value [KB]

Note

Only one > 0 value must be entered in the machine data when one of the following alarms is displayed:

- Alarm 26262 "Insufficient memory space for the collision check of two protection areas"
- Alarm 26263 "Insufficient memory space for determining the clearance of two protection areas"

4.2.3.4 Maximum number of collision pairs

The maximum number of possible collision pairs has an impact on:

- Length m of the system variable fields (e.g. \$NP_COLL_PAIR[m, ...])
- The user memory required for collision avoidance
- The size of the commissioning archive

The maximum number of collision pairs can be restricted with machine data:

MD18898 \$MN_MM_MAXNUM_3D_COLL_PAIRS = <value>

<Value>	Meaning
0	The following applies to the maximum number of possible collision pairs MCP: MCP = maximum value of the machine data
x > 0	The following applies to the maximum number of possible collision pairs MCP: MCP = x, with $0 < x \leq$ maximum value of the machine data
A value greater than the maximum permissible value of the machine data is limited internally to the maximum value. No feedback regarding this is sent to the user.	

4.2.3.5 Length of the preview

The buffer length for the collision avoidance preview is defined using machine data:

MD18951 \$MN_MM_COLLISION_PREVIEW_LEN = <number of IPO cycles>

Default value: 100

4.2 Commissioning

For machines that require a long deceleration distance due to low set dynamic values, this value should be increased as follows:

MD18951 = 200

If alarm 26285 "Emergency stop due to collision" still occurs with this setting instead of alarm 26265 "Collision of the two protection areas ...", the machine data should be set to the maximum value:

MD18951 = 300

4.2.3.6 Length of the braking the ramp

The buffer length for the collision avoidance braking ramp is defined using machine data:

MD18952 \$MN_MM_COLLISION_BRAKEPATH_LEN = <number of IPO cycles>

Default value: 200

For machines that require a long deceleration distance due to low set dynamic values, this value should be increased as follows:

MD18951 = 400

If alarm 26285 "Emergency stop due to collision" still occurs with this setting instead of alarm 26265 "Collision of the two protection areas ...", the machine data should be set to the maximum value:

MD18951 = 600

4.2.3.7 Protection levels for collision avoidance On/Off

The protection level for switching the collision avoidance On/Off is set with the machine data via the user interface. The protection level can be specified according to operating mode and protection area type.

Machine data = <protection level>

Number	Identifier: \$MN_	Meaning: Protection level for switching the collision avoidance On/Off
MD51160	ACCESS_WRITE_CA_MACH_JOG	Machine protection areas, JOG/MDI operating mode
MD51161	ACCESS_WRITE_CA_MACH_AUTO	Machine protection areas, AUTOMATIC mode,
MD51162	ACCESS_WRITE_CA_TOOL	Tool protection areas

Further information

A detailed description of protection levels can be found in:

Function Manual Basic Functions; Various NC/PLC interface signals and functions > Functions
> Access protection via password and keyswitch

4.2.4 System variables

4.2.4.1 Overview

A collision pair is parameterized with the following system variables:

Name	Meaning
\$NP_COLL_PAIR	Name of a collision pair protection area
\$NP_SAFETY_DIST	Safety clearance of the protection area pair

The system variables are described in detail in the following sections.

Note

Establish a defined initial state

It is recommended that a defined initial state be generated before parameterizing the collision avoidance. To do this, set the system variables of the collision avoidance to their default values with the DELOBJ() function.

Change system variable values

If the value of one of the system variables listed above is changed, the change becomes immediately visible at the user interface, e.g. SINUMERIK Operate. The machine model of the NC is only updated after explicitly requesting that the machine model is recalculated by calling the PROTA() (Page 88) or PROTS() (Page 89) function.

4.2.4.2 \$NP_COLL_PAIR

Function

The names of the two protection areas, which form a collision pair, is entered in a system variable. The two protection areas can be in any sequence.

Collision pairs

As the collision check requires a lot of computation time, it does not make sense to monitor all parameterized protection areas for collision with one another through the collision avoidance. Examples in which a collision check does not make sense:

- Protection areas that cannot collide because of the construction
- Protection areas that have been defined without anchoring to the kinematic chain

The user must define which parameterized protection areas can actually collide on the machine and define them as collision pairs. Only these protection areas are monitored by the collision avoidance.

To define a collision pair, the names of the two protection areas must be entered in two system variables with the same collision pair index. One protection area under the protection area index 0, the other under the protection area index 1.

4.2 Commissioning

Part of a collision pair

Using the COLLPAIR() (Page 88) function, a check can be made as to whether two protection areas are parameterized as collision pair.

Syntax

```
$NP_COLL_PAIR[<m>,<i>] = "<name>"
```

Meaning

\$NP_COLL_PAIR:	Name of the first or second protection area of a collision pair	
	Data type:	STRING
	Default value:	"" (empty string)
<m>:	System variable or collision pair index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (M - 1) ¹⁾
<i>:	Protection area index	
	Data type:	INT
	Range of values:	0 (first protection area), 1 (second protection area)
<name>:	Protection area name	
	Data type:	STRING
¹⁾ $M = n * (n - 1) / 2$ where $n = \$MN_MM_MAXNUM_3D_PROT_AREAS$		

Example

Two protection areas defined with the names "Rotary table" and "Tool in spindle" are to be checked for collision. The two protection areas are to be mutually monitored for collision. The definition for this is in the seventh collision pair:

Program code	Comment
N100 \$NP_COLL_PAIR[6,0] = "Rotary table"	; 7th collision pair, ; 1st Protection area
N110 \$NP_COLL_PAIR[6,1] = "Tool in spindle"	; 7th collision pair, ; 2nd Protection area

Constraints

- When forming a collision pair, it should be observed that every protection area of this pair has at least one protection area element that is marked with \$NP_USAGE = "C" or "A" (Page 48). Otherwise, it is not permissible to perform a collision or clearance calculation (Page 90) using the protection area.
- The protection areas of a collision pair are only checked for collision when both protection areas are in the "Protection area monitored" state. See "Status diagram protection area (Page 72)".

4.2.4.3 \$NP_SAFETY_DIST

Function

The collision pair-specific safety clearance is entered in the system variable. The collision avoidance ensures that this safety clearance is not violated.

If a value not equal to 0.0 is entered in the system variable, the general safety clearance from MD10622 \$MN_COLLISION_SAFETY_DIST (Page 80) is not taken into account for this collision pair.

If the value 0.0 is entered in the system variable, the safety clearance set in the machine data applies.

Syntax

`$NP_SAFETY_DIST[<m>] = <value>`

Meaning

\$NP_SAFETY_DIST:	Safety clearance of the collision pair	
	Data type:	REAL
	Default value:	0.0
<m>:	System variable or protection area index	
	Data type:	INT
	Range of values:	0, 1, 2, ... (M - 1) ¹⁾
<value>:	Safety clearance	
	Data type:	REAL
	Range of values:	$0.0 \leq x \leq +\text{max. REAL value}$
	Unit:	mm or inch depending on the current dimensions setting
1) $M = n * (n - 1) / 2$ where $n = \$MN_MM_MAXNUM_3D_PROT_AREAS$		

Example

The safety clearance for the protection areas of the seventh collision pair should be 1.0 mm (input system: metric).

Program code	Comment
N100 \$NP_SAFETY_DIST[6] = 1.0	; 7. collision pair, ; safety clearance = 1.0

4.2.5 Extend system variables

4.2.5.1 Overview

More information on the internal states and values of the collision avoidance can be read via the following system variables:

- Status data (Page 86)
- Braking distance estimations (Page 87)

4.2.5.2 Status data

The state data of the collision avoidance can be read via the following system variables (OPI variables)

System variable	OPI variable	Meaning
\$AN_COLL_STATE[<m>]	anCollState[<m>]	Current state of a protection area (active/inactive) with regard to the collision avoidance
\$AN_COLL_STATE_COND[<m>]	anCollStateCond[<m>]	Monitoring state (bit-coded) of a protection area
\$AN_COLL_IPO_ACTIVE	anCollIpoActive	Activation state of the collision avoidance in the main run (active/inactive)
\$AN_COLL_IPO_LIMIT	anCollIpoLimit	Velocity reduction through collision avoidance in the main run (active/inactive)
\$AN_COLL_LOAD[<i>] ¹⁾	anCollLoad[<i>] ¹⁾	Computation time requirement for the collision avoidance function <i>
\$AN_COLL_TIMEOUT_CNTR	anCollTimeoutCntr	Counter for incomplete collision avoidance
\$AN_ACTIVATE_COLL_CHECK[<j>]	anActivateColl-Check[<j>]	Current state of the NC/PLC interface with index <j> for each 8 bytes: <Nc>.collisionProtAreas.out.activate (activate protection areas)
\$AN_COLL_CHECK_OFF	anCollCheckOff	Current state of the NC/PLC interface: <Nc>.collision.out.deactivate... (deactivate protection area group 1 - 8) See "PLC signals" table below
\$AA_COLLPOS[<a>]	aaCollPos	Position of axis <a> in the machine coordinate system (MCS) when the last collision alarm occurred
\$AC_COLLPOS[<k>]	acCollPos	Vector <k> for the collision position in the world coordinate system when the last collision alarm occurred

a: Axis name

i: 0 = function 1, 1 = function 2, 2 = function 3, ...

j: Index 0, 1, 2, ... for one bit field each of 8 bytes.

k: Coordinate index k = 1, 2, 3 for X, Y, Z coordinates

m: System variable or protection area index 0, 1, 2, ... (MD18890 \$MN_MM_MAXNUM_3D_PROT_AREAS - 1)

1) The system variable can be reset by writing the value 0. Every other value is rejected with an error message.

More information

- For a detailed description of the system variables, refer to:
Parameter Manual System Variable
- A detailed description of the interface signals can be found in:
Function Manual PLC

PLC signals

Basic Program Plus	Basic Program	
<Nc>.collisionProtAreas.out.activate	LBP_NC.A_CollCheck	DB10.DBX234.0..241.7
<Nc>.collision.out.deactivateMachineAreasAuto	LBP_NC.A_CA_DisableAutoMach	DB10.DBX58.0
<Nc>.collision.out.deactivateToolAreasAuto	LBP_NC.A_CA_DisableAutoTools	DB10.DBX58.1
<Nc>.collision.out.deactivateFixtureAreasAuto	LBP_NC.A_CA_DisableAutoWorkh	DB10.DBX58.2
<Nc>.collision.out.deactivateWorkpieceAreasAuto	LBP_NC.A_CA_DisableAutoWorkp	DB10.DBX58.3
<Nc>.collision.out.deactivateMachineAreasJog	LBP_NC.A_CA_DisableJogMach	DB10.DBX58.4
<Nc>.collision.out.deactivateToolAreasJog	LBP_NC.A_CA_DisableJogTools	DB10.DBX58.5
<Nc>.collision.out.deactivateFixtureAreasJog	LBP_NC.A_CA_DisableJogWorkh	DB10.DBX58.6
<Nc>.collision.out.deactivateWorkpieceAreasJog	LBP_NC.A_CA_DisableJogWorkp	DB10.DBX58.7

4.2.5.3 Braking distance estimations

The estimated total braking distance (linear approximation) and the proportional braking distances of superimposed motions for an axis can be read via the following system variables (OPI variables). The estimation only takes the current state of the axis into account. It returns, e.g. the braking distance 0.0 for an axis that is just at the reversal point as part of a circular path.

Note

The system variables are used for support in the development of user-specific functions as part of the collision avoidance, and similar functions.

Table 4-1 Basic coordinate system (BCS)

System variable	OPI variable	Meaning
Total braking distance		
\$AA_DTBREB[<a>]	aaDtbreb	Estimated, linearly approximated total braking distance
Proportional braking distances for superimposed motions		
\$AA_DTBREB_CMD[<a>]	aaDtbrebCmd	Command component
\$AA_DTBREB_CORR[<a>]	aaDtbrebCorr	Correction component
\$AA_DTBREB_DEP[<a>]	aaDtbrebDep	Coupling component
<a>: Axis name		

4.3 Programming

Table 4-2 Machine coordinate system (MCS)

System variable	OPI variable	Meaning
Total braking distance		
\$AA_DTBREM[<a>]	aaDtbrem	Estimated, linearly approximated total braking distance
Proportional braking distances for superimposed motions		
\$AA_DTBREM_CMD[<a>]	aaDtbremCmd	Command component
\$AA_DTBREM_CORR[<a>]	aaDtbremCorr	Correction component
\$AA_DTBREM_DEP[<a>]	aaDtbremDep	Coupling component
<a>: Axis name		

Further information

For a detailed description of the system variables, refer to:

List Manual, System Variables

4.3 Programming

4.3.1 Check for collision pair (COLLPAIR)

See also

Status diagram protection area (Page 72)

4.3.2 Request recalculation of the machine model of the collision avoidance (PROTA)

If system variables of the kinematic chain \$NK_..., the geometric machine modeling or the collision avoidance \$NP_... are written in the part program, the PROTA procedure must subsequently be called so that the change becomes effective in the NC-internal machine model of the collision avoidance.

Syntax

```
PROTA [ (<Par> ) ]
```

Meaning

PROTA:	Request recalculation of the machine model of the collision avoidance
	- Triggers a preprocessing stop. - Must be alone in the block.

<Par>:	Parameter (optional)		
	Data type:	STRING	
	Value:	---	No parameters. The machine model is recalculated. The states of the protection areas are retained.
		"R"	The machine model is recalculated. The protection areas are set to their initialization status corresponding to \$NP_INIT_STAT (Page 38).

Supplementary conditions

Simulation

The `PROTA` procedure must not be used in part programs in conjunction with the simulation (`simNC`).

Example: Avoiding the `PROTA` call while the simulation is active.

Program code	Comment
...	
IF \$P_SIM == FALSE	; IF simulation not active
PROTA	THEN recalculate collision model
ENDIF	; ENDIF
...	

See also

Setting the protection zone status (`PROTS`) (Page 89)

4.3.3 Setting the protection zone status (`PROTS`)

The `PROTS ()` procedure sets the state of protection areas to the specified value.

Syntax

```
PROTS (<State>[, <Name_1>, ..., <Name_n>])
```

Meaning

PROTS:	Sets the state of protection areas
	Must be alone in the block.

4.3 Programming

<State>:	Status to which the specified protection areas are to be set		
	Data type:	CHAR	
	Value:	"A" or "a"	Status: Active
		"I" or "i"	Status: Inactive
		"P" or "p"	Status: Preactivated or PLC-controlled ¹⁾
"R" or "r"		Status: NC-internal value of the initialization status ²⁾	
<Name_1> ... <Name_n>:	Name of one or more protection areas that are to be set to the specified status (optional) If no name is specified, the specified status is set for all defined protection areas.		
	Data type:	STRING	
	Value range:	Parameterized protection area names	
	Note The maximum number of protection areas that can be specified as parameters depends only on the maximum possible number of characters per program line.		
	¹⁾ The activation/deactivation is performed via: <Nc>.collisionProtAreas.out.activate ²⁾ The status is set to the NC-internal value of the initialization status, i.e. to the value that the system variable \$NP_INIT_STAT (Page 38) had at the time of the last PROTA(...) (Page 88) call.		

PLC signals

Basic Program Plus	Basic Program	
<Nc>.collisionProtAreas.out.activate	LBP_NC.A_CollCheck	DB10.DBX234.0..241.7

4.3.4 Determining the clearance of two protection zones (PROTD)

The `PROTD()` function calculates the clearance of two protection areas.

Function properties:

- The clearance calculation is performed independent of the protection area status (activated, deactivated, preactivated).
- To calculate the clearance of two protection areas, only protection area elements are used, which are marked with `$NP_USAGE = "C"` or `"A"` (Page 48). Protection area elements of the protection area, which are marked with `$NP_USAGE = "V"`, are not taken into consideration.
- Protection areas, where all protection area elements of the protection area are marked with `$NP_USAGE = "V"`, cannot be used for the clearance calculation.

- The clearance calculation is performed with the positions valid at the end of the previous block.
- Overlays that are included in the main run calculation (e.g. DRF offset or external work offset) are included in the clearance calculation with the values valid at the function **interpretation time**.

Note

Synchronization

When using the `PROTD()` function, it is the sole responsibility of the user to synchronize the main run and preprocessing, if required, with the `STOPRE` preprocessing stop.

Collision

If there is a collision between the specified protection areas, the function returns a clearance of 0.0. There is a collision if both the protection areas touch or intersect each other.

The safety clearance for the collision check (MD10622 \$MN_COLLISION_SAFETY_DIST) is not taken into account in the clearance calculation.

Syntax

```
[<RetVal> =] PROTD([<Name_1>], [<Name_2>], VAR <Vector>[, <System>])
```

Meaning

PROTD:	Calculates the clearance of the two specified protection areas.	
	• Must be alone in the block.	
<RetVal>:	Function return value: Absolute clearance value of the two protection areas or 0.0 with collision (see above: Collision paragraph)	
	Data type:	REAL
	Range of values:	$0.0 \leq x \leq +\text{max. REAL value}$
<Name_1>, <Name_2>:	Names of the two protection areas whose clearance is to be calculated (optional)	
	Data type:	STRING
	Range of values:	Parameterized protection area names
	Default value:	"" (empty string) If no protection areas have been specified, the function calculates the current smallest clearance from all the activated and preactivated protection areas in the collision model.
<Vector>:	Return value: 3-dimensional clearance vector from protection area <Name_2> to protection area <Name_1> with:	
	• <Vector>[0]: X coordinate in the world coordinate system • <Vector>[1]: Y coordinate in the world coordinate system • <Vector>[2]: Z coordinate in the world coordinate system	
	For collision: <Vector> == zero vector	
	Data type:	VAR REAL [3]
	Range of values:	<Vector> [n]: $0.0 \leq x \leq \pm \text{max. REAL value}$

4.4 Example

<System>:	System of units (inch/metric) for clearance and clearance vector (optional)		
	Data type:	BOOL	
	Value:	FALSE (Default)	System of units corresponding to the currently active G command from G Group 13 (G70, G71, G700, G710).
		TRUE	System of units corresponding to the set basic system: MD52806 \$MN_ISO_SCALING_SYSTEM

4.4 Example

4.4.1 Specifications

General Information

The basic procedure when parameterizing collision avoidance using a part program is shown using a simplified 3-axis milling machine as example. All of the system variables relevant for collision avoidance are written to the part program:

- Kinematic chain \$NK_...
- Geometric machine modeling \$NP_...
- Collision avoidance \$NP_...

The machine model is subsequently activated in the part program, so that after executing the part program, the collision avoidance has been completely parameterized in the 3-axis milling machine and is active.

Option and machine data

The following option and machine data must be set for the example:

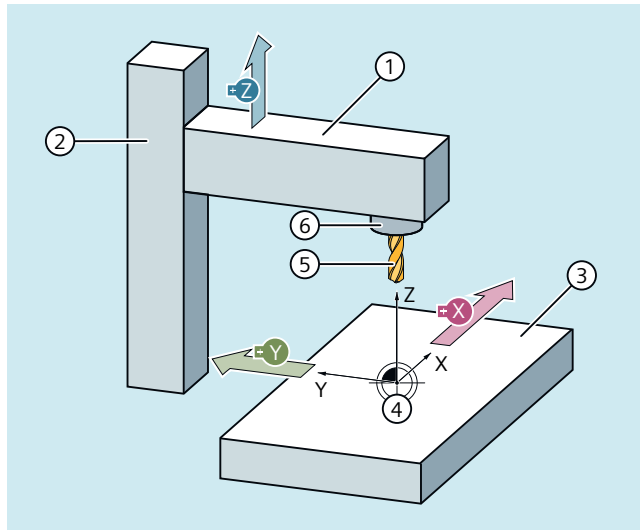
No.	Option data: \$ON_	Value
MD19830	COLLISION_MASK	2
MD19840	NUM_CHANNELS_CA_MULTICHAN	1

No.	Machine data: \$MN_	Value
MD10619	COLLISION_TOLERANCE	1
MD18880	MM_MAXNUM_KIN_CHAIN_ELEM	10
MD18890	MM_MAXNUM_3D_PROT_AREAS	10
MD18892	MM_MAXNUM_3D_PROT_AREA_ELEM	10
MD18893	MM_MAXNUM_3D_T_PROT_ELEM	1
MD18894	MM_MAXNUM_3D_FACETS_INTERN	1000
MD18895	MM_MAXNUM_3D_FACETS	3000

No.	Machine data: \$MN_	Value
MD18896	MM_MAXNUM_3D_COLLISION	0
MD18897	MM_MAXNUM_3D_INTERFACE_IN	16
MD18899	PROT_AREA_TOOL_MASK	1

Principle design of the 3-axis milling machine

The principle machine design is shown in the following diagram.



- ① Z axis
- ② Column
- ③ Table
- ④ Machine zero = reference point
- ⑤ Tool
- ⑥ Tool adapter

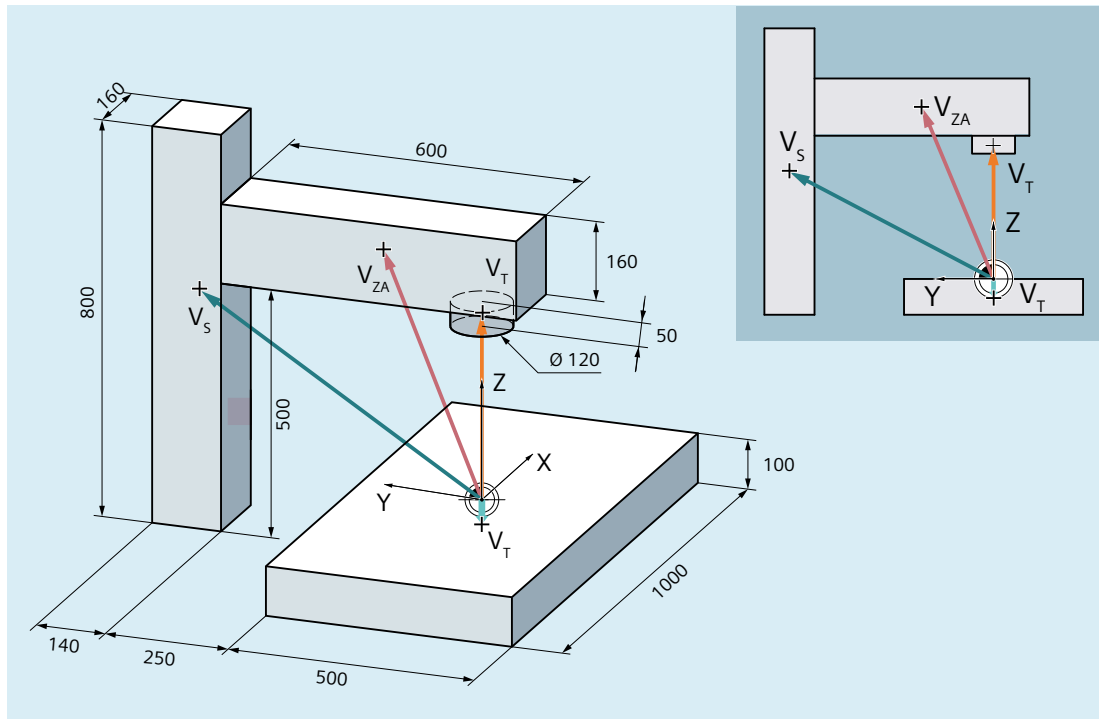
The machine parts and/or protection areas are assigned to the following machine axes.

Machine parts and/or protection areas	Machine axis
Table	X1, Y1
Z axis	Z1
Column	---
Tool adapter	Z1
Tool	Z1

4.4 Example

Dimension drawing

The dimensions of the protection area elements as well as their position (vectors to the center point of the protection area element) referred to the machine zero point are specified in the following dimension drawing.



Vectors to the center point of the protection area elements

v_{tool}	Tool adapter (0;0;25)
v_{ZA}	Z axis (0;200;130)
v_S	Column (0;570;350)
v_T	Table (0;0;-50)

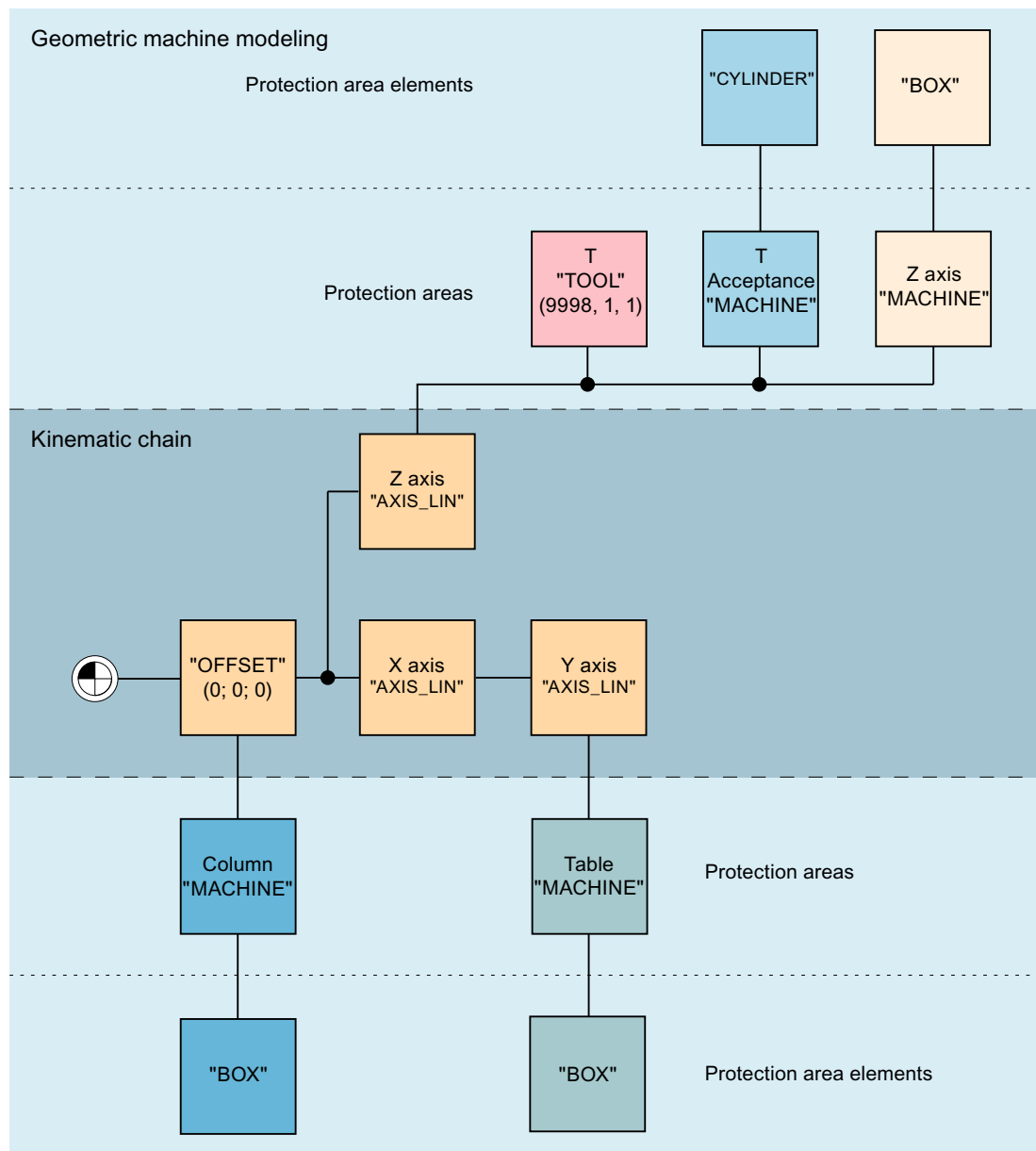
Kinematic chain

The kinematic chain (see next diagram) starts with an "Offset" type element. These are assigned to all static protection areas of the machine. In the example, this is only the "Column" protection area.

The kinematic elements of the machine axes follow the offset element:

- Z axis and X axis traverse independently of one another $\Rightarrow$ \$NK_PARALLEL
- The Y axis traverses dependent on the X axis $\Rightarrow$ \$NK_NEXT

The various protection areas of the geometric machine modeling are assigned the kinematic elements of the machine axes



Collision pairs

For the example, it is assumed that only the following collision pairs are to be taken into account:

- Tool adapter - table
- Tool - table

4.4 Example

4.4.2 Part program of the machine model

Program code

```

;*****
;***** Example *****
; milling machine: 3 linear axes, 1 spindle
;   table   => X1, Y1
;   Z axis, tool adapter, tool => Z1
;*****
; status: February 11, 2013, 15:34
;
;=====
; Collision machine data used
;=====
; MD10619 $MN_COLLISION_TOLERANCE = 1
;
; MD18880 $MN_MM_MAXNUM_KIN_CHAIN_ELEM   = 100
; MD18890 $MN_MM_MAXNUM_3D_PROT_AREAS    = 10
; MD18892 $MN_MM_MAXNUM_3D_PROT_AREA_ELEM = 10
; MD18893 $MN_MM_MAXNUM_3D_T_PROT_ELEM   = 100
; MD18894 $MN_MM_MAXNUM_3D_FACETS_INTERN = 1000
; MD18895 $MN_MM_MAXNUM_3D_FACETS        = 3000
; MD18896 $MN_MM_MAXNUM_3D_COLLISION     = 0
; MD18897 $MN_MM_MAXNUM_3D_INTERFACE_IN  = 16
; MD18899 $MN_PROT_AREA_TOOL_MASK        = 1
;
; MD19830 $ON_COLLISION_MASK = 2 ; option
; MD19840 $ON_NUM_CHANNELS_CA_MULTICHAN = 1 ; option
;
;
;=====
; Definitions
;=====
DEF INT RETVAL = 0 ; return value of the delete function
;
DEF INT C_NKE = 0 ; index for kinematic elements
DEF INT C_NPC = 0 ; index for protection areas
DEF INT C_NPE = 0 ; index for protection area elements
DEF INT C_NPP = 0 ; index for collision pairs
;
;

```

Program code

```

;=====
; initialization of the collision data
;=====
MSG("protection areas")
G4 F3
; reset all parameters to their initial setting
;
RETVAl = DELOBJ("KIN_CHAIN_ELEM")
IF (RETVAl <> 0)
    MSG("error: DELOBJ KIN_CHAIN_ELEM")
    G4 F5
ENDIF
;
RETVAl = DELOBJ("PROT_AREA_ALL")
IF RETVAl <> 0
    MSG("error: DELOBJ PROT_AREA_ALL")
    G4 F5
ENDIF
;
RETVAl = DELOBJ("PROT_AREA_COLL_PAIRS")
IF RETVAl <> 0
    MSG("error: DELOBJ PROT_AREA_COLL_PAIRS")
    G4 F5
ENDIF
;
;
;=====
; kinematic chain
;=====
; KE1: ROOT
; -----
$NK_NAME[C_NKE]      = "ROOT"
$NK_NEXT[C_NKE]      = "X axis"
$NK_PARALLEL[C_NKE]  = ""
$NK_TYPE[C_NKE]      = "OFFSET"
;
$NK_OFF_DIR[C_NKE, 0] = 0.0      ; X
$NK_OFF_DIR[C_NKE, 1] = 0.0      ; Y
$NK_OFF_DIR[C_NKE, 2] = 0.0      ; Z
;
$NK_AXIS[C_NKE]      = ""
$NK_A_OFF[C_NKE]     = 0.0
;
C_NKE = C_NKE + 1          ; next kinematic element
;

```

4.4 Example

Program code

```

; -----
; kinematic element: X axis
; -----
$NK_NAME[C_NKE]      = "X axis"
$NK_NEXT[C_NKE]      = "Y axis"
$NK_PARALLEL[C_NKE]  = "Z axis"
$NK_TYPE[C_NKE]      = "AXIS_LIN"
;
$NK_OFF_DIR[C_NKE, 0] = 1.0      ; X
$NK_OFF_DIR[C_NKE, 1] = 0.0      ; Y
$NK_OFF_DIR[C_NKE, 2] = 0.0      ; Z
;
$NK_AXIS[C_NKE]       = "X1"
$NK_A_OFF[C_NKE]      = 0.0
;
C_NKE = C_NKE + 1      ; next kinematic element
;
; -----
; kinematic element: Y axis
; -----
$NK_NAME[C_NKE]      = "Y axis"
$NK_NEXT[C_NKE]      = ""
$NK_PARALLEL[C_NKE]  = ""
$NK_TYPE[C_NKE]      = "AXIS_LIN"
;
$NK_OFF_DIR[C_NKE, 0] = 0.0      ; X
$NK_OFF_DIR[C_NKE, 1] = 1.0      ; Y
$NK_OFF_DIR[C_NKE, 2] = 0.0      ; Z
;
$NK_AXIS[C_NKE]       = "Y1"
$NK_A_OFF[C_NKE]      = 0.0
;
C_NKE = C_NKE + 1      ; next kinematic element
;
; -----
; kinematic element: Z axis
; -----
$NK_NAME[C_NKE]      = "Z axis"
$NK_NEXT[C_NKE]      = ""
$NK_PARALLEL[C_NKE]  = ""
$NK_TYPE[C_NKE]      = "AXIS_LIN"
;
$NK_OFF_DIR[C_NKE, 0] = 0.0      ; X
$NK_OFF_DIR[C_NKE, 1] = 0.0      ; Y
$NK_OFF_DIR[C_NKE, 2] = 1.0      ; Z
;
$NK_AXIS[C_NKE]       = "Z1"
$NK_A_OFF[C_NKE]      = 0.0
;
C_NKE = C_NKE + 1      ; next kinematic element
;
;

```

Program code

```

;=====
; protection areas with protection area elements
;=====
; protection area 1: Column
; -----
$NP_PROT_NAME[C_NPC] = "Column"
$NP_PROT_TYPE[C_NPC] = "MACHINE"
$NP_CHAIN_ELEM[C_NPC] = "ROOT"
$NP_1ST_PROT[C_NPC] = "SBE column"
$NP_PROT_COLOR[C_NPC] = 'HFFA0A0A4' ; AARRGGBB
$NP_BIT_NO[C_NPC] = -1
$NP_INIT_STAT[C_NPC] = "A"
;
C_NPC = C_NPC + 1 ; next protection area
;
; -----
; protection area element 1.1: SBE column
; -----
$NP_NAME[C_NPE] = "SBE column"
$NP_NEXT[C_NPE] = ""
$NP_NEXTP[C_NPE] = ""
$NP_TYPE[C_NPE] = "BOX"
;
$NP_PARA[C_NPE,0] = 160.0 ; cube dimensions
                        ; length
$NP_PARA[C_NPE,1] = 140.0 ; width
$NP_PARA[C_NPE,2] = 800.0 ; height
;
$NP_OFF[C_NPE,0] = 0.0 ; center point
                        ; X
$NP_OFF[C_NPE,1] = 570.0 ; Y: xxx rear table edge
$NP_OFF[C_NPE,2] = 350.0 ; Z: Lower edge equal to lower edge table
;
$NP_DIR[C_NPE,0] = 0.0
$NP_DIR[C_NPE,1] = 0.0
$NP_DIR[C_NPE,2] = 0.0
;
$NP_ANG[C_NPE] = 0.0
;
$NP_COLOR[C_NPE] = 0
$NP_D_LEVEL[C_NPE] = 0
$NP_USAGE[C_NPE] = "V" ; V = visualize only
$NP_FILENAME[C_NPE] = ""
;
C_NPE = C_NPE + 1 ; next protection area element
;

```

4.4 Example

Program code

```

;+++++
; protection area 2: Tool adapter
; -----
$NP_PROT_NAME[C_NPC] = "Tool adapter"
$NP_PROT_TYPE[C_NPC] = "MACHINE"
$NP_CHAIN_ELEM[C_NPC] = "Z axis"
$NP_1ST_PROT[C_NPC] = "SBE tool adapter"
$NP_PROT_COLOR[C_NPC] = 'HFF0000FF' ; AARRGGBB
$NP_BIT_NO[C_NPC] = -1
$NP_INIT_STAT[C_NPC] = "A"
;
C_NPC = C_NPC + 1 ; next protection area
;
; -----
; protection area element 2.1: SBE tool adapter
; -----
$NP_NAME[C_NPE] = "SBE tool adapter"
$NP_NEXT[C_NPE] = ""
$NP_NEXTP[C_NPE] = ""
$NP_TYPE[C_NPE] = "CYLINDER"
; cylinder dimensions
$NP_PARA[C_NPE,0] = 50.0 ; height
$NP_PARA[C_NPE,1] = 60.0 ; radius
$NP_PARA[C_NPE,2] = 0.0
; center point
$NP_OFF[C_NPE,0] = 0.0 ; X
$NP_OFF[C_NPE,1] = 0.0 ; Y
$NP_OFF[C_NPE,2] = 25.0 ; Z: Half height
;
$NP_DIR[C_NPE,0] = 0.0
$NP_DIR[C_NPE,1] = 0.0
$NP_DIR[C_NPE,2] = 0.0
;
$NP_ANG[C_NPE] = 0.0
;
$NP_COLOR[C_NPE] = 0
$NP_D_LEVEL[C_NPE] = 0
$NP_USAGE[C_NPE] = "A"
$NP_FILENAME[C_NPE] = ""
;
C_NPE = C_NPE + 1 ; next protection area element
;

```

Program code

```

; ++++++
; protection area 3: Tool
; -----
$NP_PROT_NAME[C_NPC] = "Tool"
$NP_PROT_TYPE[C_NPC] = "TOOL"
$NP_CHAIN_ELEM[C_NPC] = "Z axis"
$NP_1ST_PROT[C_NPC] = ""
$NP_PROT_COLOR[C_NPC] = 'FFFFFF0000' ; AARRGGBB
$NP_BIT_NO[C_NPC] = -1
$NP_INIT_STAT[C_NPC] = "A"
;
$NP_INDEX[C_NPC,0] = 1 ; only relevant for type "TOOL"
$NP_INDEX[C_NPC,1] = 9998 ; tool location No. / spindle No.
$NP_INDEX[C_NPC,2] = 1 ; magazine No. / -
;
$NP_INDEX[C_NPC,2] = 1 ; TOA area
;
C_NPC = C_NPC + 1 ; next protection area
;

```

4.4 Example

Program code

```

; ++++++
; protection area 4: Z axis
; -----
$NP_PROT_NAME[C_NPC] = "Z axis"
$NP_PROT_TYPE[C_NPC] = "MACHINE"
$NP_CHAIN_ELEM[C_NPC] = "Z axis"
$NP_1ST_PROT[C_NPC] = "SBE Z axis"
$NP_PROT_COLOR[C_NPC] = 'HFFA0A0A4' ; AARRGGBB
$NP_BIT_NO[C_NPC] = -1
$NP_INIT_STAT[C_NPC] = "A"
;
C_NPC = C_NPC + 1 ; next protection area
;
; -----
; protection area element 4.1: SBE horizontal column
; -----
$NP_NAME[C_NPE] = "SBE Z axis"
$NP_NEXT[C_NPE] = ""
$NP_NEXTP[C_NPE] = ""

$NP_TYPE[C_NPE] = "BOX"
;
$NP_PARA[C_NPE,0] = 160.0
$NP_PARA[C_NPE,1] = 600.0
$NP_PARA[C_NPE,2] = 160.0
;
$NP_OFF[C_NPE,0] = 0.0
$NP_OFF[C_NPE,1] = 200.0
$NP_OFF[C_NPE,2] = 130.0
;
$NP_DIR[C_NPE,0] = 0.0
$NP_DIR[C_NPE,1] = 0.0
$NP_DIR[C_NPE,2] = 0.0
;
$NP_ANG[C_NPE] = 0.0
;
$NP_COLOR[C_NPE] = 0
$NP_D_LEVEL[C_NPE] = 0
$NP_USAGE[C_NPE] = "A"
$NP_FILENAME[C_NPE] = ""
;
C_NPE = C_NPE + 1 ; next protection area element
;

```

Program code

```

; ++++++
; protection area 5: Table
; -----
$NP_PROT_NAME[C_NPC] = "Table"
$NP_PROT_TYPE[C_NPC] = "MACHINE"
$NP_CHAIN_ELEM[C_NPC] = "Y axis"
$NP_1ST_PROT[C_NPC] = "SBE table"
$NP_PROT_COLOR[C_NPC] = 'HFF00FF00' ; AARRGGBB
$NP_BIT_NO[C_NPC] = -1
$NP_INIT_STAT[C_NPC] = "A"
;
C_NPC = C_NPC + 1 ; next protection area
;
; -----
; protection area element 5.1: SBE table
; -----
$NP_NAME[C_NPE] = "SBE table"
$NP_NEXT[C_NPE] = ""
$NP_NEXTP[C_NPE] = ""
$NP_TYPE[C_NPE] = "BOX"
;
$NP_PARA[C_NPE,0] = 1000.0
$NP_PARA[C_NPE,1] = 500.0
$NP_PARA[C_NPE,2] = 100.0
;
$NP_OFF[C_NPE,0] = 0.0
$NP_OFF[C_NPE,1] = 0.0
$NP_OFF[C_NPE,2] = -50.0
;
$NP_DIR[C_NPE,0] = 0.0
$NP_DIR[C_NPE,1] = 0.0
$NP_DIR[C_NPE,2] = 0.0
;
$NP_ANG[C_NPE] = 0.0
;
$NP_COLOR[C_NPE] = 0
$NP_D_LEVEL[C_NPE] = 0
$NP_USAGE[C_NPE] = "A"
$NP_FILENAME[C_NPE] = ""
;
C_NPE = C_NPE + 1 ; next protection area element
;
;

```

4.4 Example

Program code

```
;=====
; collision pairs
;=====
$NP_COLL_PAIR[C_NPP, 0] = "Tool adapter"
$NP_COLL_PAIR[C_NPP, 1] = "Table"
;
C_NPP = C_NPP + 1                ; next collision pair
;
$NP_COLL_PAIR[C_NPP, 0] = "Tool"
$NP_COLL_PAIR[C_NPP, 1] = "Table"
;
C_NPP = C_NPP + 1                ; next collision pair
;
;
;=====
; activating the machine model
;=====
PROTA
PROTS("A")
;
M2
;===== ENDE =====
```

Protection areas

5.1 Function

Protection zones are static or moveable in 2- or 3-dimensional ranges within a machine to protect machine elements against collisions - and must be defined by the user.

The following elements can be protected:

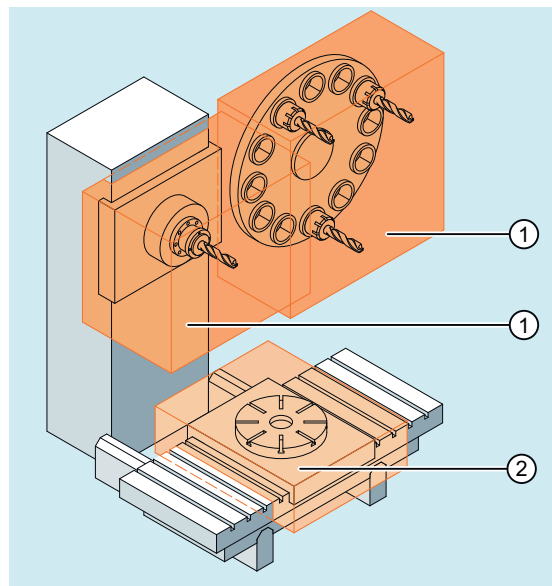
- Fixed machine elements (e.g. tool magazine, probe that can be swiveled).
- Moving machine elements that belong to the tool (e.g. tool, tool holder)
- Moving machine elements that belong to the workpiece (e.g. parts of the workpiece, clamping table, clamping shoe, spindle chuck, tailstock).

In order that it is guaranteed that the machine element is protected, the protection zones must be defined so that they completely envelope the element to be protected.

Protection zone monitoring is channel-specific, i.e. all the active protection zones of a channel monitor one another for collisions.

During automatic execution of part programs in the AUTOMATIC or MDA mode, the NC checks at the start of every part program block whether a collision between protection zones can occur upon moving along the programmed path.

After manual deactivation of an active protection zone, traversing can be performed in this zone. After leaving the protection zone, the protection zone automatically becomes active again.



- ① **Tool-related** protection areas
- ② **Workpiece-related** protection area

Figure 5-1 Example of protection zones on a milling machine

Defining protection zones

A protection zone can be 2 or 3 dimensional, and defined using polylines with a maximum of 10 corner points and arcs as contour elements. The definition can be made using commands in the part program (see "Defining protection zones (CPROTDEF, NPROTDEF) (Page 113)") or using system variables. All of the contour elements lie in the plane that can be selected with G17, G18 or G19.

3rd dimension

The expansion in the protection area in the 3rd dimension can be limited between $-\infty$ to $+\infty$:

- $-\infty$ to $+\infty$
- $-\infty$ up to the upper limit
- Lower limit to $+\infty$
- Lower limit to upper limit

Coordinate system

The definition of a protection zone takes place with reference to the geometry axis of a channel and therefore in the basic coordinate system (BCS).

Reference

- Tool-related protection areas
Coordinates for **tool**-related protection zones must be specified as absolute values referred to the **tool holder reference point F**.
- Workpiece-related protection areas
Coordinates for **workpiece**-related protection zones must be specified as absolute values referred to the zero point of the **basic coordinate system (BCS)**.

Note

If no tool-related protection zone is active, the tool path is checked against the workpiece-related protection zones.

If no workpiece-oriented protection area is active, then there is no protection zone monitoring.

Orientation

The orientation of the protection zones is determined by the plane definition (abscissa/ordinate), in which the contour is described, and the axis perpendicular to the contour (vertical axis).

The orientation of the protection zones must be the same for the tool- and workpiece-related protection zones.

Machine/channel-specific protection zones

- Machine-specific protection zones
Data for machine specific protection zones are defined once in the control. These protection zones can be activated by all channels.
- Channel-specific protection zones
Data for channel-specific protection zones are defined in a channel. These protection zones can be activated only by this channel.

System variable

When the protection zones are defined using commands in the part program, the protection zone data is stored in system variables. The system variables can also be written directly so that the definition of protection areas can also be performed directly in the system variables. The same supplementary conditions apply for the definition of the contour of a protection zone as for a protection zone definition using commands in the part program (see "Defining protection zones (CPROTDEF, NPROTDEF) (Page 113)").

The protection zone definitions cover following system variables:

System variable	Type	Meaning
\$SN_PA_ACTIV_IMMED[<n>] \$SC_PA_ACTIV_IMMED[<n>]	BOOL	Activation type The protection zone is active / not active immediately after the control system has been powered up and the axes referenced.
		FALSE Not immediately active
		TRUE Immediately active
\$SN_PA_T_W[<n>] \$SC_PA_T_W[<n>]	INT	Protection zone type
		0 Workpiece-related protection area
		1 Reserved
		2 Reserved
\$SN_PA_ORI[<n>] \$SC_PA_ORI[<n>]	INT	Orientation of the protection zone, i.e. polygon definition in the plane of:
		0 1st and 2nd geometry axis
		1 3rd and 1st geometry axis
		2 2nd and 3rd geometry axis
\$SN_PA_LIM_3DIM[<n>] \$SC_PA_LIM_3DIM[<n>]	INT	Type of limitation in the 3rd dimension
		0 No limitation
		1 Limitation in plus direction
		2 Limitation in minus direction
\$SN_PA_PLUS_LIM[<n>] \$SC_PA_PLUS_LIM[<n>]	REAL	Value of the limit in the positive direction in the 3rd dimension
\$SN_PA_MINUS_LIM[<n>] \$SC_PA_MINUS_LIM[<n>]	REAL	Value of the limit in the negative direction in the 3rd dimension
\$SN_PA_CONT_NUM[<n>] \$SC_PA_CONT_NUM[<n>]	INT	Number of valid contour elements

5.1 Function

System variable	Type	Meaning
\$SN_PA_CONT_TYP[<n>, <i>] \$SC_PA_CONT_TYP[<n>, <i>]	INT	Contour type[<i>], contour type (G1, G2, G3) of the <i>th contour element
\$SN_PA_CONT_ABS[<n>, <i>] \$SC_PA_CONT_ABS[<n>, <i>]	REAL	End point of the contour[<i>], abscissa value
\$SN_PA_CONT_ORD[<n>, <i>] \$SC_PA_CONT_ORD[<n>, <i>]	REAL	End point of the contour[<i>], ordinate value
\$SN_PA_CENT_ABS[<n>, <i>] \$SC_PA_CENT_ABS[<n>, <i>]	REAL	Center point of the circular contour[<i>], absolute abscissa value
\$SN_PA_CENT_ORD[<n>, <i>] \$SC_PA_CENT_ORD[<n>, <i>]	REAL	Center point of the circular contour[<i>], absolute ordinate value
\$SN_... are system variables for NC and machine-specific protection zones. \$SC_... are system variables for channel-specific protection zones. The index "<n>" corresponds to the number of the protection zone: 0 = 1st protection zone The index "<i>" corresponds to the number of the contour element: 0 = 1st contour element The contour elements must be defined in ascending order.		

Note

The system variables of the protection zone definitions are not restored with REORG!

Data of the protection zone definitions**Data storage**

The protection zone definitions are stored in the following files:

File	Blocks
_N_NCK_PRO	Data block for machine-specific protection zones
_N_CHAN1_PRO	Data block for channel-specific protection zones in channel 1
_N_CHAN2_PRO	Data block for channel-specific protection zones in channel 2

Data backup

The protection zone definitions are saved in the following files:

File	Blocks
_N_INITIAL_INI	All data blocks of the protection zones
_N_COMPLETE_PRO	All data blocks of the protection zones
_N_CHAN_PRO	All data blocks of the channel-specific protection zones

Activating, preactivating and deactivating protection zones**Activation state**

The activation state of a protection zone can have the following values:

- Activated
- Preactivated

- Preactivated with conditional stop
- Deactivated

The activation state is always channel-specific even for machine specific protection zones.

Activating, preactivating and deactivating in the part program

The activation state of a protection zone can be changed in the part program at any time using commands (see "Activating/deactivating protection zones (CPROT, NPROT) (Page 116)").

Note

A protection zone is only taken into account after the referencing of all geometry axes of the channel in which it has been activated.

Protection zones that are to be activated at a later time from the PLC user program (see below) must be preactivated in the part program:

Preactivated protection zones are displayed via the following NC/PLC interface signals:

- `<Chan>.protAreas[1].in.ncPreactivated` (machine-specific protection area 1 - 10 preactivated) == 1
- `<Chan>.protAreas[1].in.chanPreactivated` (channel-specific protection area 1 - 10 preactivated) == 1

Preactivated with conditional stop

NOTICE

Protection zone violation possible

If a preactivated protection zone with conditional stop is not activated in good time, the NC may no longer be able to stop before the protection zone in good time because the braking distance after the activation point has not been taken into account.

In the case of a preactivated protection zone with conditional stop, a traversing motion is **not** stopped before this if the traversing motion goes into the protection zone. A stop is only performed when the protection zone has been activated. This behavior is to enable uninterrupted machining controlled by the user when the protection zone is only required temporarily.

Activating using NC/PLC interface signals

Only protection zones that have been **preactivated** via the part program can be activated in the PLC user program via the NC/PLC interface signals:

- `<Chan>.protAreas[1].out.nc` (activate machine-specific protection area 1 - 10) = 1
- `<Chan>.protAreas[1].out.chan` (activate channel-specific protection area 1 - 10) = 1

The activation of preactivated protection zones must be performed prior to the traversing motion of the geometry axes! If the activation is performed during the traversing motion, these protection zones are not taken into account in the current traversing motion. Response:

- Alarm 10704 "Protection area monitoring is not guaranteed"
- `<Chan>.basic.in.protAreasNotGuaranteed` (protection area monitoring not guaranteed) = 1

Note

The activation of preactivated protection zones must be performed prior to the traversing motion of the geometry axes!

Deactivating using NC/PLC interface signals

Only protection zones that have been **preactivated** via a part program and activated via the NC/PLC interface signals, can be deactivated again via the NC/PLC interface signals:

- `<Chan>.protAreas[1].out.nc` (activate machine-specific protection area 1 - 10) = 0
- `<Chan>.protAreas[1].out.chan` (activate channel-specific protection area 1 - 10) = 0

Protection zones that have been **activated** directly via a part program **cannot** be deactivated from the PLC user program.

Automatic deactivation for transformation change/geometry axis interchange

In the default setting, when changing a transformation, or for a geometry axis interchange, the active protection zones are automatically deactivated. On the other hand, if active protection zones should remain active, then the bit-coded machine data MD10618 \$MN_PROTAREA_GEOAX_CHANGE_MODE must be correspondingly adapted (see "Machine data (Page 112)").

Activation state in special system states**Block search with calculation**

For block search with calculation, the last programmed activation state of a protection zone is always taken into account.

Program test

In the AUTOMATIC and MDI modes, activated and preactivated protection zones are also monitored in the "Program test" state.

NC RESET and end of program

The activation state of a protection zone is retained even after NC-RESET and program end.

Display of protection zone violations

Violations of activated protection zones or possible violations of preactivated protection zones, if they would be activated, are displayed using the following NC/PLC interface signals:

- `<Chan>.protAreas[1].in.ncViolated` (machine-specific protection area 1 - 10 violated) == 1
- `<Chan>.protAreas[1].in.chanViolated` (channel-specific protection area 1 - 10 violated) == 1

Checking for protection zone violation

Function CALCPOSI can be used to check whether geometry axes can traverse a specified path without violating a protection zone (see "Checking for protection zone violation, working area limitation and software limit switches (CALCPOSI) (Page 120)").

PLC signals

Basic Program Plus	Basic Program	
<code><Chan>.protAreas[1].in.ncPreactivated</code>	<code>LBP_Chan*.E_MRPactivated[1] - [10]</code>	DB21,DBX272.0..273.1
<code><Chan>.protAreas[1].in.chanPreactivated</code>	<code>LBP_Chan*.E_CSPactivated[1] - [10]</code>	DB21,DBX274.0..275.1
<code><Chan>.basic.in.protAreasNotGuaranteed</code>	<code>LBP_Chan*.E_ProtZoneNOK</code>	DB21,DBX39.0
<code><Chan>.protAreas[1].out.nc</code>	<code>LBP_Chan*.A_MRP[1] - [10]</code>	DB21,DBX8.0..9.1
<code><Chan>.protAreas[1].out.chan</code>	<code>LBP_Chan*.A_CSP[1] - [10]</code>	DB21,DBX10.0..11.1
<code><Chan>.protAreas[1].in.ncViolated</code>	<code>LBP_Chan*.E_MRPIviolated[1] - [10]</code>	DB21,DBX276.0..277.1
<code><Chan>.protAreas[1].in.chanViolated</code>	<code>LBP_Chan*.E_CSPIviolated[1] - [10]</code>	DB21,DBX278.0..279.1

5.2 Commissioning

5.2.1 Machine data

Memory requirements

The memory required for protection zones is parameterized via the following machine data:

- Persistent memory
 - MD18190 \$MN_MM_NUM_PROTECT_AREA_NCK (number of available machine-specific protection zones)
 - MD28200 \$MC_MM_NUM_PROTECT_AREA_CHAN (number of available channel-specific protection zones)
- Dynamic memory
 - MD28210 \$MC_MM_NUM_PROTECT_AREA_ACTIVE (maximum number of protection zones that can be activated simultaneously in the channel)
 - MD28212 \$MC_MM_NUM_PROTECT_AREA_CONTUR (maximum number of definable contour elements per protection zone)

Response for a transformation change/geometry axis interchange

The following machine data can be used to define as to whether, when changing a transformation or for a geometry axis interchange, active protection zones should be kept or deactivated:

MD10618 \$MN_PROTAREA_GEOAX_CHANGE_MODE

Bit	Value	Meaning
0	0	The active protection zones are deactivated during the transformation change.
	1	The active protection zones remain active during the transformation change.
1	0	The active protection zones are deactivated during the geometry axis change.
	1	The active protection zones remain active during the geometry axis change.

5.3 Programming

5.3.1 Defining protection zones (CPROTDEF, NPROTDEF)

Protection zones, which protect machine elements against collisions, are defined in the part program in blocks. These contain the following elements:

1. Definition of the machining plane
Before the actual protection zone definition, the machining plane must be selected, to which the contour description of the protection zone refers.
2. Start of the definition
Depending on the particular NC command, either a channel-specific or machine-specific protection zone is created.
3. Contour description of the protection zone
The contour of a protection zone is defined with traversing motion. These are not executed and have no connection to previous or subsequent geometry descriptions. They only define the protection zone.
4. End of definition

Syntax

```
DEF INT <Var>
G17/G18/G19
CPROTDEF/NPROTDEF (<n>, <t>, <AppLim>, <AppPlus>, <AppMinus>)
G0/G1/... X/Y/Z...
...
EXECUTE (<Var>)
```

Meaning

DEF INT <Var>:	Definition of a local help variable, of the INTEGER data type	
<Var>:	Name of the Help variable	
G17/G18/G19:	Machining plane Note: It is not permissible to change the machining plane before the end of the definition. Programming the applicator is not permitted between start and end of the definition.	
CPROTDEF () :	Predefined procedure to define a channel -specific protection zone	
NPROTDEF () :	Predefined procedure to define a machine -specific protection zone	
<n>:	Number of defined protection zone	
	Data type:	INT

<t>:	Type of protection zone	
	Data type:	BOOL
	Value:	TRUE Tool -related protection zone
		FALSE Workpiece -related protection zone
<AppLim>:	Type of limitation in the third dimension	
	Data type:	INT
	Value:	0 No limitation
		1 Limit in plus direction
		2 Limit in minus direction
		3 Limit in positive and negative direction
<AppPlus>:	Value of the limit in the positive direction in the 3rd dimension	
	Data type:	REAL
<AppMinus>:	Value of the limit in the negative direction in the 3rd dimension	
	Data type:	REAL
G0/G1/... X/Y/Z... :	The contour of a protection zone is specified with up to 11 traversing movements in the selected machining plane. The first traversing movement is the movement to the contour. The last point in the contour description must always coincide with the first point of the contour description. The valid protection zone is the zone left of the contour: - Internal protection zone The contour of an internal protection zone must be described in the counterclockwise direction. - External protection zones (permitted only for workpiece-related protection zones) The contour for an external protection zone must be described in the clockwise direction. The following contour elements are permissible: - G0, G1 for straight contour elements - G2 for circle segments in the clockwise direction Permissible only for workpiece-related protection zones. Not permissible for tool-related protection zones because they must be convex. - G3 for circular segments in the counter-clockwise direction Note: A protection zone cannot be described by a complete circle. A complete circle must be divided into two semicircles. Note: The sequence G2 → G3 or G3 → G2 is not permissible! A short G1 block must be inserted between the two circular blocks.	
EXECUTE(<Var>):	Predefined procedure that marks the end of the definition A switch is made back to normal program processing with EXECUTE.	

Example

See example under "Activating/deactivating protection zones (CPROT, NPROT) (Page 116)".

Additional information

Machine-specific protection zones

A machine-specific protection zone or its contour is defined using the geometry axis, i.e. referenced to the basic coordinate system (BCS) of a channel. In order that correct protection-zone monitoring can take place in all channels in which the machine-specific protection zone is active, the basic coordinate system (BCS) of all of the channels involved must be identical:

- position of the coordinate origin referred to the machine zero
- Orientation of the coordinate axes

Reference point for contour description

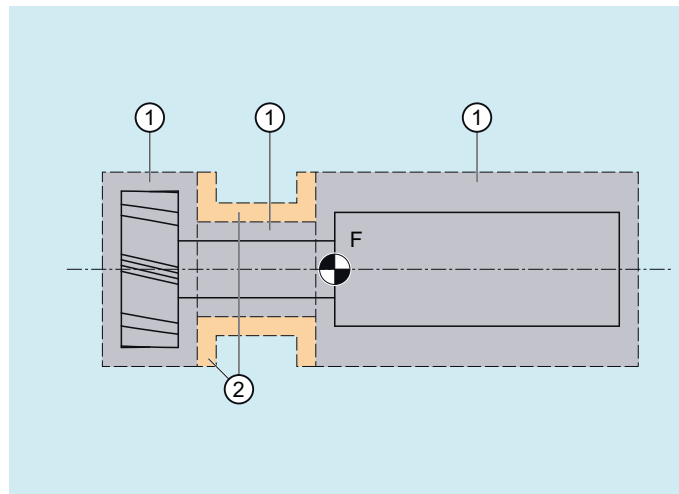
- Tool-related protection zones
Coordinates for **tool**-related protection zones must be specified as absolute values referred to the **tool holder reference point F**.
- Workpiece-related protection zones
Coordinates for **workpiece**-related protection zones must be specified as absolute values referred to the zero point of the **basic coordinate system (BCS)**.

Protection zones symmetrical around the center of rotation

For protection zones symmetrical around the axis or rotation (e.g. spindle chuck), you must describe the complete contour and not only the contour up to the center of rotation.

Tool-related protection zones

Tool-related protection zones must always be convex. If a concave protected zone is desired, this should be subdivided into several convex protection zones.



- ① Convex protection zones
- ② Concave protection zones (**not permissible!**)
- F Toolholder reference point

General conditions

During the definition of a protection zone, the following functions must not be active or used:

- Tool radius compensation (cutter radius compensation, tool nose radius compensation)
- Transformation
- Reference point approach (G74)
- Fixed point approach (G75)
- Dwell time (G4)
- Block search stop (STOPRE)
- End of program (M17, M30)
- M functions: M0, M1, M2

5.3.2 Activating/deactivating protection zones (CPROT, NPROT)

Protection zones previously defined in the part program can be activated at any time – or can be preactivated for subsequent activation by the PLC user program. Active protection zones can be deactivated at any time.

When activating or preactivating, it is also possible to relatively shift the reference point of the protection zone.

Note

A protection zone is only taken into account after the referencing of all geometry axes of the channel in which it has been activated.

Note**Monitoring protection zones**

If a tool-related protection area is not active, the tool path is checked against the workpiece-related protection zones.

If no workpiece-oriented protection zone is active, then there is no protection zone monitoring.

Syntax

```
CPROT (<n>, <Status>, <XMov>, <YMov>, <ZMov>)
NPROT (<n>, <Status>, <XMov>, <YMov>, <ZMov>)
```

Meaning

CPROT:	Predefined procedure to activate a channel -specific protection zone	
NPROT:	Predefined procedure to activate a machine -specific protection zone	
<n>:	Number of the protection zone	
	Data type:	INT

<Status>:	The channel-specific activation status is set using this parameter		
	Data type:	INT	
	Value:	0	Deactivate protection zone
		1	Preactivate protection zone
		2	Activate protection zone
		3	Preactivate protection zone with conditional stop
<XMov>, <YMov>, <ZMov>:	Additive offset values in the X/Y/Z direction The offset can take place in 1, 2, or 3 dimensions. The offset values refer to: • The machine zero for a workpiece-related protection zone • The tool carrier reference point F for a tool-specific protection zone		
	Data type:	REAL	

Example

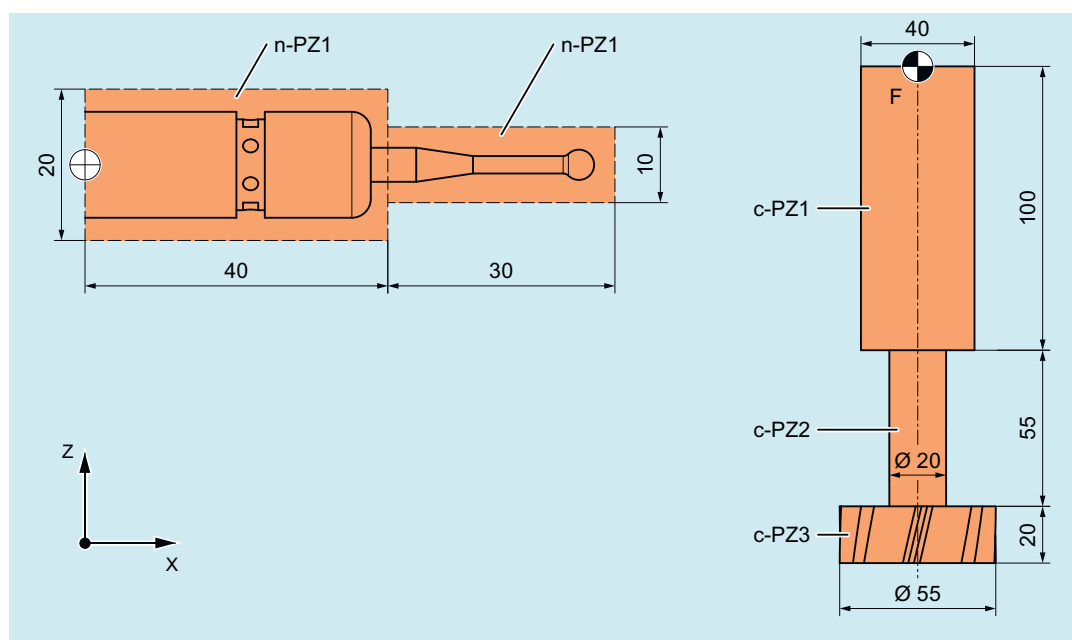
Possible collision of a milling cutter with the measuring probe is to be monitored on a milling machine. The position of the measuring probe is to be defined by an offset when the function is activated.

The following protection zones are defined for this:

- A machine-specific and a workpiece-related protection zone for both the measuring probe holder (n-PZ1) and the measuring probe itself (n-PZ2).
- A channel-specific and a tool-related protection zone for the milling cutter holder (c-PZ1), the cutter shank (c-PZ2) and the milling cutter itself (c-PZ3).

The orientation of all protection zones is in the Z direction.

The position of the reference point of the measuring probe on activation of the function must be X = -120, Y = 60 and Z = 80.



① Name for the protection zone of the probe

F Toolholder reference point

Program code	Comment
DEF INT PROTZONE	; Definition of a Help variable
G17	; machining plane XY
; defining protection zones:	
NPROTDEF(1,FALSE,3,10,-10)	; protection zone n-PZ1
G01 X0 Y-10	
X40	
Y10	
X0	
Y-10	
EXECUTE (PROTZONE)	
NPROTDEF(2,FALSE,3,5,-5)	; protection zone n-PZ2
G01 X40 Y-5	
X70	
Y5	
X40	
Y-5	
EXECUTE (PROTZONE)	
CPROTDEF(1,TRUE,3,0,-100)	; protection zone c-PZ1
G01 X-20 Y-20	
X20	
Y20	
X-20	
Y-20	

Program code	Comment
EXECUTE (PROTZONE)	
CPROTDEF (2,TRUE,3,-100,-150)	; protection zone c-PZ2
G01 X0 Y-10	
G03 X0 Y10 J10	
X0 Y-10 J-10	
EXECUTE (PROTZONE)	
CPROTDEF (3,TRUE,3,-150,-170)	; protection zone c-PZ3
G01 X0 Y-27.5	
G03 X0 Y27.5 J27.5	
X0 Y27.5 J-27.5	
EXECUTE (PROTZONE)	
; activating protection zones:	
NPROT (1,2,-120,60,80)	; activate protection zone n-PZ1 with offset
NPROT (2,2,-120,60,80)	; activate protection zone n-PZ2 with offset
CPROT (1,2,0,0,0)	; activate protection zone c-PZ1
CPROT (2,2,0,0,0)	; activate protection zone c-PZ2
CPROT (3,2,0,0,0)	; activate protection zone c-PZ3

Further information

Activation status after the control powers up

A protection zone can already be active after the control system powers up and the axes have been referenced. This is the case if, for the protection zone, the following system variable is set to TRUE:

- \$SN_PA_ACTIV_IMMED[<n>] (for machine-specific protection zone) or
- \$SC_PA_ACTIV_IMMED[<n>] (for channel-specific protection zone)
Index "<n>" corresponds to the number of the protection zone: 0 = 1. Protection zone

The protection zone is activated with status = 2 – and without offset.

Multiple activation of a protection zone

A machine-specific protection zone can be active simultaneously in several channels (e.g. protection zone of a tailstock where there are two opposite sides). The protection zones are only monitored if all geometry axes have been referenced.

A protection zone cannot be activated simultaneously with different offsets in a single channel.

Protection zone monitoring for active tool radius compensation

For active tool radius compensation, a functioning protection zone monitoring is only possible if the plane of the tool radius compensation is identical to the plane of the protection zone definitions.

5.3.3 Checking for protection zone violation, working area limitation and software limit switches (CALCPOSI)

Function

In the workpiece coordinate system (WCS), the CALCPOSI function checks whether, starting from the starting position, the **geometry axes** can be traversed a specified distance without violating active limits. For the case that the distance cannot be fully traversed because of limits, a positive, decimal-coded status value and the maximum possible traversing distance are returned.

Definition

```
INT CALCPOSI (VAR REAL[3] <Start>, VAR REAL[3] <Dist>, VAR REAL[5]  
<Limit>, VAR REAL[3] <MaxDist>, BOOL <MeasSys>, INT <TestLim>)
```

Syntax

```
<Status> = CALCPOSI (VAR <Start>, VAR <Dist>, VAR <Limit>, VAR  
<MaxDist>, <MeasSys>, <TestLim>)
```

Meaning

CALCPOSI (...):	Predefined function for testing limit violations regarding the geometry axes	
	Preprocessing stop:	No
	Alone in the block:	Yes

<status>: (Part 1)	Function return value. Negative values indicate error states.	
	Data type:	INT
	Value range:	$-8 \leq x \leq 100000$
	Value	Meaning
	0	The distance can be traversed completely.
	-1	At least one component is negative in <Limit>.
	-2	Error in a transformation calculation. Example: The traversing distance passes through a singularity so that the axis positions cannot be defined.
	-3	The specified traversing distance <Dist> and the maximum possible traversing distance <MaxDist> are linearly dependent. Note Can only occur in conjunction with <TestLim>, bit 4 == 1.
	-4	The projection of the traversing direction contained in <Dist> on to the limitation surface is the zero vector, or the traversing direction is perpendicular to the violated limitation surface. Note Can only occur in conjunction with <TestLim>, bit 5 == 1.
	-5	In <TestLim>, bit 4 == 1 AND bit 5 == 1
	-6	At least one machine axis that has to be considered for checking the traversing limits has not been referenced.
	-7	Collision avoidance function: Invalid definition of the kinematic chain or the protection zones.
	-8	Collision avoidance function: This command cannot be executed because of insufficient memory.
<status>: (Part 2)	Units digit	
	Note If several limits are violated simultaneously, the limit with the greatest restriction on the specified traversing distance is signaled.	
	Value	Meaning
	1	Software limit switches are limiting the traversing distance
	2	Working area limits are limiting the traversing distance
	3	Protection areas are limiting the traversing path
	4	Collision avoidance function: Protection areas are limiting the traversing path
	Tens digit	
	Value	Meaning
	1x	The initial value violates the limit
	2x	The specified straight line violates the limit. This value is returned even if the end point does not violate any limit itself, but the path from the starting point to the end point would cause a limit value to be violated (e.g. by passing through a protection zone, curved software limit switches in the WCS for non-linear transformations, e.g. transmit).

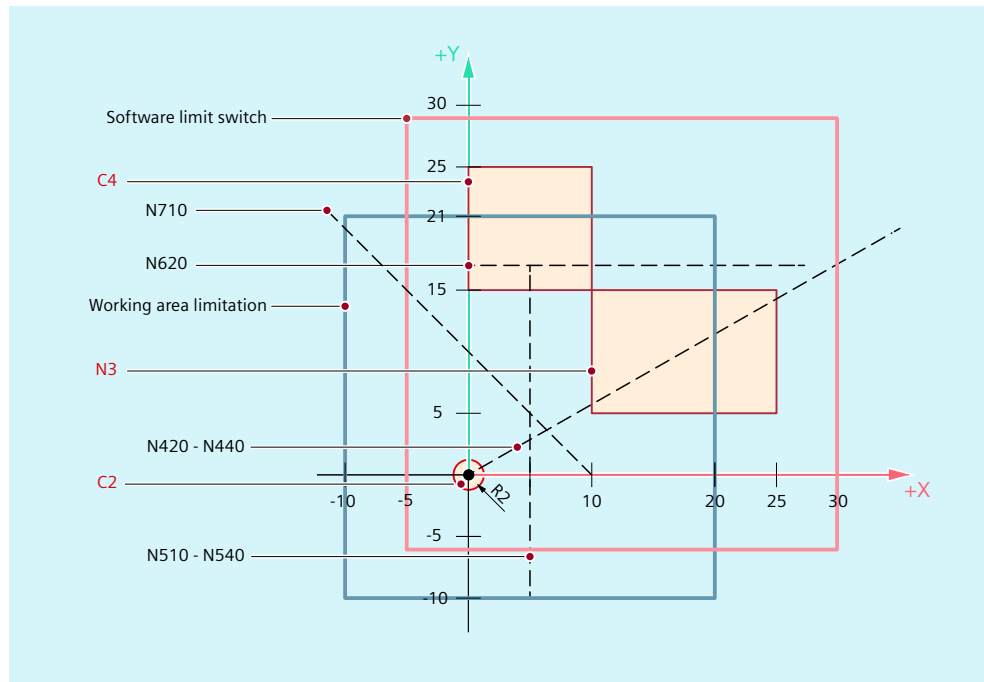
<status>: (Part 3)	Hundreds digit	
	Value	Meaning
	1xx	AND units digit == 1 or 2: The positive limit value has been violated.
		AND units digit == 3 ¹⁾ : An NC-specific protection area has been violated.
	2xx	AND units digit == 1 or 2: The negative limit value has been violated.
		AND units digit == 3 ¹⁾ : A channel-specific protection zone is violated.
<status>: (Part 4)	Thousands digit	
	Value	Meaning
	1xxx	AND units digit == 1 or 2: Factor with which the axis number is multiplied that violates the limit. Numbering of the axes begins with 1. Reference: • Software limit switches: Machine axes • Working area limitation: Geometry axes AND units digit == 3 ¹⁾ : Factor with which the number of the violated protection zone is multiplied.
<status>: (Part 5)	Hundred thousands digit	
	Value	Meaning
	0xxxxx	Hundred thousands digit == 0: <Dist> remains unchanged
	1xxxxx	A direction vector is returned in <Dist>, which defines the further motion direction on the limitation surface. Can only occur with the following supplementary conditions: • Software limit switch or working area limit violated (not in the starting point) • A transformation is not active • <TestID>, bit 4 or bit 5 == 1
<Start>:	Reference to a vector with the start positions: • <Start> [0]: 1st geometry axis • <Start> [1]: 2nd geometry axis • <Start> [2]: 3rd geometry axis	
	Parameter type:	Input
	Data type:	VAR REAL [3]
	Value range:	-max. REAL value ≤ x[<n>] ≤ +max. REAL value

<Dist>:	Reference to a vector.	
	Input: Incremental traversing distance	
	• <Dist> [0]: 1st geometry axis • <Dist> [1]: 2nd geometry axis • <Dist> [2]: 3rd geometry axis	
	Output (only for set hundred thousands digit in <Status>):	
	<Dist> contains a unit vector v as output value which defines the further traversing direction in the WCS.	
	Case 1: Formation of vector v for <TestID>, bit 4 == 1 The input vectors <Dist> and <MaxDist> span the motion plane. This plane is cut by the violated limitation surface. The intersecting line of the two planes defines the direction of vector v . The orientation (sign) is selected so that the angle between the input vector <MaxDist> and v is not greater than 90 degrees.	
	Case 2: Formation of vector v for <TestID>, bit 5 == 1 Vector v is the unit vector in the projection direction of the traversing vector contained in <Dist> on the limitation surface. If the projection of the traversing vector on the limitation surface is the zero vector, an error is returned.	
	Parameter type:	Input/output
<Limit>:	Data type:	VAR REAL [3]
	Value range:	-max. REAL value ≤ x[<n>] ≤ +max. REAL value
	Reference to an array of length 5.	
	• <Limit> [0 - 2]: Minimum clearance of the geometry axes to the limits: – <Limit> [0]: 1st geometry axis – <Limit> [1]: 2nd geometry axis – <Limit> [2]: 3rd geometry axis	
	The minimum clearances are observed with: – Working area limitation: No restrictions – Software limit switches: If no transformation is active, or a transformation is active in which a clear assignment of the geometry axes to the linear machine axes is possible, e.g. 5-axis transformations.	
<Limit>:	• <Limit> [3]: Contains the minimum clearance for linear machine axes which, for example, cannot be assigned a geometry axis because of a non-linear transformation. This value is also used as limit value for the monitoring of the conventional protection zones and the collision avoidance protection zones. • <Limit> [4]: Contains the minimum clearance for rotary machine axes which, for example, cannot be assigned a geometry axis because of a non-linear transformation.	
	Note This value is only active for the monitoring of the software limit switches for special transformations.	
	Parameter type:	Input
	Data type:	VAR REAL [5]
	Value range:	-max. REAL value ≤ x[n] ≤ +max. REAL value

<MaxDist>:	Reference to a vector with the incremental traversing distance in which the specified minimum clearance of an axis limit is not violated by any of the relevant machine axes:	
	• <Dist> [0]: 1st geometry axis • <Dist> [1]: 2nd geometry axis • <Dist> [2]: 3rd geometry axis	
	If the traversing distance is not restricted, the contents of this return parameter are the same as the contents of <Dist>.	
	For <TestID>, bit 4 == 1: <Dist> and <MaxDist>	
	<MaxDist> and <Dist> must contain vectors as input values that span a motion plane. The two vectors must be mutually linearly independent. The absolute value of <MaxDist> is arbitrary. For the calculation of the motion direction, see the description for <Dist>.	
<MeasSys>:	Parameter type:	Output
	Data type:	VAR REAL [3]
	Value range:	-max. REAL value ≤ x[<n>] ≤ +max. REAL value
	Measuring system (inch/metric) for position and distance specifications (optional)	
	Data type:	BOOL
<TestLim>:	Value	Meaning
	FALSE (Default)	System of units corresponding to the currently active G command from the G group 13 (G70, G71, G700, G710). Note If G70 is active and the basic system is metric (or G71 is active and the basic system is inch), the system variables \$AA_IW and \$AA_MW are provided in the basic system and, if used, must be converted for CALC-POS1.
	TRUE	System of units according to the set basic system: MD52806 \$MN_ISO_SCALING_SYSTEM
	Bit-coded selection of the limits to be monitored (optional)	
	Data type:	INT
<TestLim>:	Default:	Bit 0, 1, 2, 3, 6, 7 = 1 (207)
	Value	Meaning
	1	Software limit switch
	2	Working area limitation
	4	Activated conventional protection zones
	8	Preactivated conventional protection zones
	16	With violated software limit switches or working area limits in <Dist>, return the traversing direction as in Case 1 (see above).
	32	With violated software limit switches or working area limits in <Dist>, return the traversing direction as in Case 2 (see above).
	64	Activated collision avoidance protection zones
	128	Preactivated collision avoidance protection zones
	256	Pairs of activated and preactivated collision avoidance protection zones
¹⁾ If several protection zones are violated, the protection zone with the greatest restriction on the specified traversing distance is returned.		

Example

Limitations



In the example, the active software limit switches and working area limits in the X-Y plane and the following three protection zones are displayed:

- C2: Tool-related, channel-specific protection zone, active, circular, radius = 2 mm
- C4: Workpiece-related, channel-specific protection zone, preactivated, square, side length = 10 mm
- N3: Machine-specific protection zone, active, rectangular, side length = 10 mm x 15 mm

NC program

The protection zones and working area limits are defined first in the NC program. The `CALCPOS1()` function is then called with different parameter assignments.

Program code

```

N10 DEF REAL _START[3]
N20 DEF REAL _DIST[3]
N30 DEF REAL _LIMIT[5]
N40 DEF REAL _MAXDIST[3]
N50 DEF INT _PA
N60 DEF INT _STATUS

: toolrelated protection zone C2
N70 CPROTDEF(2, TRUE, 0)
N80 G17 G1 X-2 Y0
N90 G3 I2 X2
N100 I-2 X-2
N110 EXECUTE( PA)

```

Program code

```
; workpiece-related protection zone C4
N120 CPROTDEF(4, FALSE, 0)
N130 G17 G1 X0 Y15
N140 X10
N150 Y25
N160 X0
N170 Y15
N180 EXECUTE(_PA)

; machine-specific protection zone N3
N190 NPROTDEF(3, FALSE, 0)
N200 G17 G1 X10 Y5
N210 X25
N220 Y15
N230 X10
N240 Y5
N250 EXECUTE(_PA)

; activate or preactivate protection zones
N260 CPROT(2, 2, 0, 0, 0)
N270 CPROT(4, 1, 0, 0, 0)
N280 NPROT(3, 2, 0, 0, 0)

; define working area limits
N290 G25 XX=-10 YY=-10
N300 G26 XX=20 YY=21
N310 _START[0] = 0.
N320 _START[1] = 0.
N330 _START[2] = 0.
N340 _DIST[0] = 35.
N350 _DIST[1] = 20.
N360 _DIST[2] = 0.
N370 _LIMIT[0] = 0.
N380 _LIMIT[1] = 0.
N390 _LIMIT[2] = 0.
N400 _LIMIT[3] = 0.
N410 _LIMIT[4] = 0.
N420 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST)
N430 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,,3)
N440 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,,1)
N450 _START[0] = 5.
N460 _START[1] = 17.
N470 _START[2] = 0.
N480 _DIST[0] = 0.
N490 _DIST[1] = -27.
N500 _DIST[2] = 0.
N510 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,,14)
N520 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,, 6)
N530 _LIMIT[1] = 2.
N540 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,, 6)
N550 _START[0] = 27.
N560 _START[1] = 17.1
N570 _START[2] = 0.
```

Program code

```

N580 _DIST[0] = -27.
N590 _DIST[1] = 0.
N600 _DIST[2] = 0.
N610 _LIMIT[3] = 2.
N620 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,,12)
N630 _START[0] = 0.
N640 _START[1] = 0.
N650 _START[2] = 0.
N660 _DIST[0] = 0.
N670 _DIST[1] = 30.
N680 _DIST[2] = 0.
N690 TRANS X10
N700 AROT Z45
N710 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST)
; delete frames from N690 and N700 again
N720 TRANS
N730 _START[0] = 0.
N740 _START[1] = 10.
N750 _START[2] = 0.
; vectors _DIST and _MAXDIST define the motion plane
N760 _DIST[0] = 30.
N770 _DIST[1] = 30.
N780 _DIST[2] = 0.
N790 _MAXDIST[0] = 1.
N800 _MAXDIST[1] = 0.
N810 _MAXDIST[2] = 1.
N820 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,,17)
N830 M30

```

Results of CALCPOSI()

N...	<status>	<MaxDist>[0] $\triangleq$ X	<MaxDist>[1] $\triangleq$ Y	Remarks
420	3123	8.040	4.594	N3 is violated.
430	1122	20.000	11.429	No protection zone monitoring, working area limitation is violated.
440	1121	30.000	17.143	Only software limit monitoring is still active.
510	4213	0.000	0.000	Starting point violates C4
520	0000	0.000	-27.000	Preactivated C4 is not monitored. The specified distance can be traversed completely.
540	2222	0.000	-25.000	Because _LIMIT[1] = 2, the traversing distance is restricted by the working area limitation.
620	4223	-13.000	0.000	Clearance to C4 is a total of 4 mm due to C2 and _LIMIT[3]. Clearance C2 $\rightarrow$ N3 of 0.1 mm does not result in limitation of the traversing distance.

N...	<status>	<MaxDist>[0] $\triangleq$ X	<MaxDist>[1] $\triangleq$ Y	Remarks
710	1221	0.000	21.213	Frame with translation and rotation active. The permissible traversing distance in <code>_DIST</code> applies in the shifted and rotated WCS.
820	102121	18.000	18.000	The software limit switch of the Y axis is violated. The calculation of a further traversing direction is requested with <code><_TESTLIM> = 17</code> . This direction is in <code>_DIST</code> (0.707, 0.0, 0.707). It is valid because the hundred thousands digit is set in <code><_STATUS></code> .

More information

"Referenced" axis status

All machine axes considered by `CALCPOSI()` must be homed.

Circle-related distance specifications

All circle-related distance specifications are **always** interpreted as radius specifications. This must be taken into account particularly for transverse axes with activated diameter programming (`DIAMON/DIAM90`).

Traversing distance reduction

If the specified traversing distance of an axis is limited, the traversing distance of the other axes is also reduced proportionally in the `<MaxDist>` return value. The resulting end point is therefore still on the specified path.

Rotary axes

Rotary axes are only monitored when they are not modulo rotary axes.

It is permissible that no software limit switches, working area limitations or protection areas are defined for one or more of the relevant axes.

Software limit switch and working area limitation status

Software limit switches and working area limitations are only taken into account if they are active during the execution of `CALCPOSI()`. The status can be influenced, for example, via:

- Machine data: `MD21020 $MC_WORKAREA_WITH_TOOL_RADIUS`
- Setting data: `$AC_WORKAREA_CS_...`
- NC/PLC interface signals: `<Axis>.basic.out.swLimitSwitch2Minus / ...Plus`
- Commands: `WALIMON / WALIMOF`

Software limit switches and transformations

With `CALCPOSI()`, the positions of the machine axes (MCS) cannot always be uniquely determined from the positions of the geometry axes (WCS) during various kinematic transformations (e.g. `TRANSMIT`) because of ambiguities at certain positions of the traversing distance. In normal traversing operation, the uniqueness generally results from the history and the condition that a continuous motion in the WCS must correspond to a continuous motion in the MCS. Therefore, when monitoring the software limit switches, the

machine position at the time when `CALCPOSI ()` is executed is used to resolve the ambiguity in such cases.

Note

Preprocessing stop

When using `CALCPOSI ()` in conjunction with transformations, it is the sole responsibility of the user to program a preprocessing stop (`STOPRE`) with the preprocessing before `CALCPOSI ()` for the synchronization of the machine axis positions.

Protection area clearance and conventional protection areas

With conventional protection areas, there is **no** guarantee that the safety clearance set in parameter `<Limit>[3]` is maintained for all protection zones during a traversing movement on the specified path. It is only guaranteed that no protection area will be violated when the end point returned in `<Dist>` is extended by the safety clearance in the traversing direction. However, the straight line can pass very close to a protection area.

Protection zone clearance and collision avoidance protection areas

With collision avoidance protection areas, there is a guarantee that the safety clearance set in parameter `<Limit>[3]` is maintained for all protection zones during a traversing movement on the specified traversing path.

The safety clearance specified in parameter `<Limit>[3]` only takes effect when the following applies:

`<Limit>[3] > (MD10619 $MN_COLLISION_TOLERANCE)`

If bit 4 is set in parameter `<TestLim>` (calculation of the ongoing traversing direction), then the direction vector received in `<DIST>` is only valid when the hundred thousands digit is set in the function return value (`<status>`). If a direction such as this cannot be determined, either because protection areas were violated, or because a transformation is active, then the input value in `<DIST>` remains unchanged. An additional error message is not output.

PLC signals

Basic Program Plus	Basic Program	
<code><Axis>.basic.out.swLimitSwitch2Minus</code>	<code>LBP_Axis*.A_SWLimit2Minus</code>	<code>DB31,DBX12.2</code>
<code><Axis>.basic.out.swLimitSwitch2Plus</code>	<code>LBP_Axis*. A_SWLimit2Plus</code>	<code>DB31,DBX12.3</code>

5.4 Special situations

5.4.1 Temporary enabling of protection zones

If a protection zone violation occurs when starting or during a traversing motion, under certain circumstances, the protection zone can be enabled, i.e. for temporary traversing. In AUTOMATIC and MDA mode as well as in JOG mode, the temporary enabling of protection zones is performed via operator actions.

See also:

- Behavior in the AUTOMATIC and MDI modes (Page 130)
- Behavior in JOG mode (Page 131)

Note

Temporarily enabling protection zones is **only** possible for **workpiece**-related protection zones. **Tool**-related protection zones must either be deactivated in the part program or via the NC/PLC interface in the "Preactivated" state.

Temporary enabling of a protection zone is terminated after the following events:

- When the control system powers up
- AUTOMATIC or MDA mode: The end of the block is outside the protection zone
- JOG mode: The end of the traversing motion is outside the protection zone
- The protection zone is activated

5.4.2 Behavior in the AUTOMATIC and MDI modes

In AUTOMATIC and MDA mode, no traversing motion is enabled into or through active protection zones:

- A traversing motion that would lead from outside into an active protection zone, is stopped at the end of the last block located outside the protection zone.
- A traversing motion that begins within an active protection zone is not started.

Temporary enabling of protection zones

If a traversing motion is stopped in AUTOMATIC or MDA mode because of a protection zone violation, this is indicated to the operator by an alarm. If the operator decides that the traversing motion can be continued, the crossing of protection zones can be enabled.

The enabling is only temporarily and is performed by triggering NC start:

```
<Chan>.basic.out.ncStart (NC start) = 1
```

An alarm is displayed for each violated protection zone. An NC start signal must be triggered by the operator for each protection zone to be enabled.

The traversing motion is continued when all protection zones that have resulted in the stopping of the traversing motion, have been enabled.

Continuation of a traversing motion without temporary enabling

A traversing motion was stopped with an alarm because of a protection zone violation. If the relevant protection zone is set to the "Preactivated" state via the NC/PLC interface, the traversing motion can be continued with NC start, without the protection zone being temporarily enabled.

Increased protection against the enabling of protection zones

If the enabling of a protection zone is to be protected better than just by using an NC start, then this must be realized by the machine manufacturer or user in the PLC user program.

PLC signals

Basic Program Plus	Basic Program	
<Chan>.basic.out.ncStart	LBP_Chan*.A_NCStart	DB21,DBX7.1

5.4.3 Behavior in JOG mode

Simultaneous traversing of several geometry axes

In JOG mode, traversing motions can be performed simultaneously in several geometry axes. The traversing range of every participating geometry axis is limited axis-specifically at the start of the traversing motion with regard to the traversing range limits (software limit switches, working area limitation, etc.) and active protection zones. However, safe monitoring of all active protection zones cannot be guaranteed. The following is provided as feedback to the user:

- Alarm 10704 "Protection area monitoring is not guaranteed"
- <Chan>.basic.in.protAreasNotGuaranteed (protection area monitoring not guaranteed) = 1

After the end of the traversing motion, the alarm is cleared automatically.

If the current position is within an activated or preactivated protection zone, then the following actions are triggered:

- Alarm message 10702 "NC protection zone violated during manual mode" or 10703 "Channel-specific protection zone violated during manual mode" is output – specifying the violated protection zone and the traversing axis.
- Further traversing motion is disabled.
- The following NC/PLC interface signal is set for the protection zone involved:
 <Chan>.protAreas[1].in.ncViolated (machine-specific protection area 1 - 10 violated) == 1
 or
 <Chan>.protAreas[1].in.chanViolated (channel-specific protection area 1 - 10 violated) == 1

To continue, see paragraph "Temporary enabling of protection zones".

Example:

Three activated protection zones and simultaneous traversing of two geometry axes:

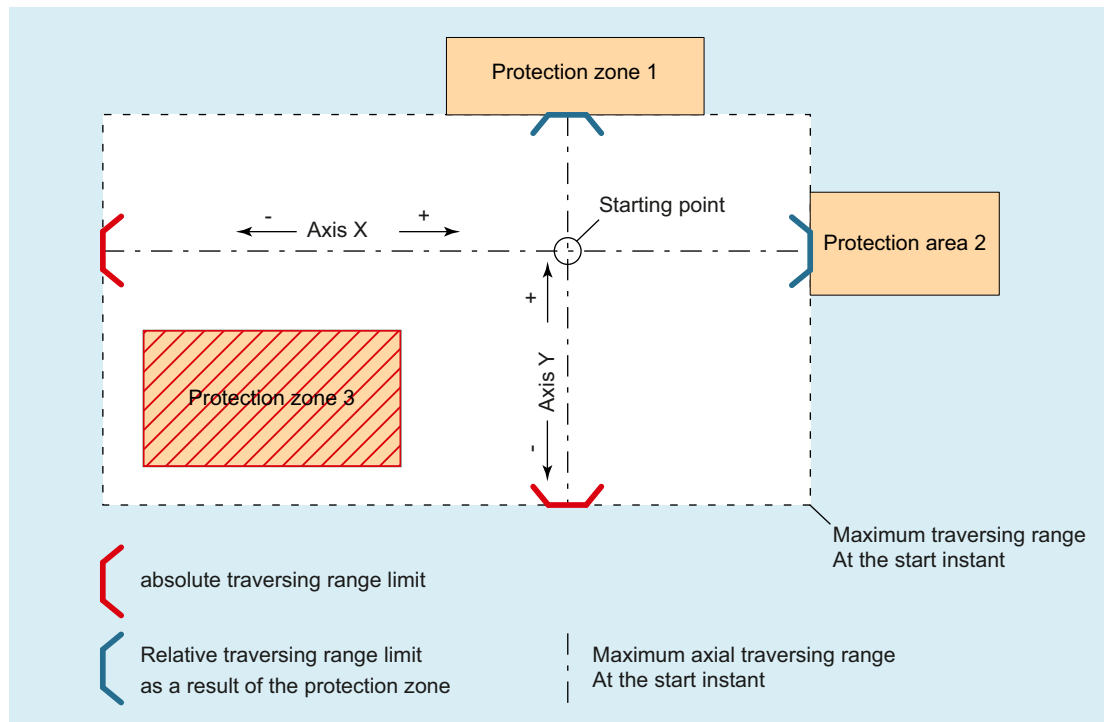


Figure 5-2 Motion range of the geometry axes at the start time

At the start time of traversing motion of axes X and Y, the axis-specific traversing range limits are determined from the start time:

- X axis
 - Positive traversing direction: Protection area 2
 - Negative traversing direction: Absolute traversing range limit (e.g. software limit switches)
- Y axis
 - Positive traversing direction: Protection area 1
 - Negative traversing direction: Absolute traversing range limit (e.g. working area limitation)

The resulting maximum traversing range at the start time does not take protection zone 3 into account. As a consequence, protection zone 3 could be violated.

Note

Activated and preactivated protection zones are also monitored in the manual operating modes JOG, INC and DRF.

Limiting the traversing motion of an axis

If the traversing motion of an axis is limited as a protection zone has been reached, then alarm 10706 "NC protection zone reached during manual mode" or 10707 "Channel-specific protection zone reached during manual mode" is output, specifying the protection zone reached and the traversing axis. It is assured that no protection zone will be violated when an axis is traversing in JOG. (This response is analogous to approaching software limit switches or a working area limitation.)

The alarm is reset:

- When an axis is traversed that does not lead into the protection zone.
- When the protection zone is enabled.
- When the control powers up.

If an axis starts to move towards a protection zone when it is at a protection zone limit, then alarm 10706 or 10707 is output, and motion is not started.

Temporary enabling of protection zones

If traversing motion is started within an activated protection zone, or at the limit of an activated protection zone, then alarm 10702 "NC protection zone violated during manual mode" or 10703 "Channel-specific protection zone violated during manual mode" is output – specifying the violated protection zone and the traversing axis – and motion is not started. The traversing motion can be performed when the relevant protection zone is temporarily enabled. The following actions must be performed for this:

- Generate a positive edge at the NC/PLC interface signal:
`<Chan>.basic.out.enableProtAreas` (enable protection area)
- Start the same traversing motion again

Note

The NC/PLC interface signal "Protection zone violated" is set when the temporarily enabled protection zone is traversed:

```
<Chan>.protAreas[1].in.ncViolated (machine-specific protection area 1 - 10  
violated) == 1
```

or

```
<Chan>.protAreas[1].in.chanViolated (channel-specific protection area 1 - 10  
violated) == 1
```

The enable signal is reset if a motion is started that does not lead into the enabled protection zone.

If further protection zones are affected by the traversing motion, additional alarms 10702 or 10703 are displayed for each protection zone. The protection areas displayed in the alarms can be enabled by creating further positive edges on the NC/PLC interface signal:

```
<Chan>.basic.out.enableProtAreas
```

Behavior at change of operating mode

The protection zones temporarily enabled in JOG mode are retained after a change to AUTOMATIC or MDI mode. The temporary enable signals set in the AUTOMATIC or MDI mode are also retained after a change to JOG mode.

Reset of an enable

The enable signal is reset internally and on the NC/PLC interface at the next standstill of a geometry axis for which the temporarily enabled protection zone has been completely exited:

```
<Chan>.protAreas[1].in.ncViolated (machine-specific protection area 1 - 10 violated) == 0
```

or

```
<Chan>.protAreas[1].in.chanViolated (channel-specific protection area 1 - 10 violated) == 0
```

PLC signals

Basic Program Plus	Basic Program	
<Chan>.basic.in.protAreasNotGuaranteed	LBP_Chan*.E_ProtZoneNOK	DB21,DBX39.0
<Chan>.basic.out.enableProtAreas	LBP_Chan*.A_ProtZone	DB21,DBX1.1
<Chan>.protAreas[1].in.ncViolated	LBP_Chan*.E_MRPViolated[1] - [10]	DB21,DBX276.0..277.1
<Chan>.protAreas[1].in.chanViolated	LBP_Chan*.E_CSPViolated[1] - [10]	DB21,DBX278.0..279.1

5.5 Boundary conditions

Restrictions in protection-zone monitoring

In the following cases, protection area monitoring is **not** possible:

- Traversing motion of orientation axes
- **Fixed machine**-specific protection zones for face transformation (TRANSMIT) or cylindrical surface transformation (TRACYL).
Exception: Protection areas that are symmetrical around the axis of rotation of the spindle axis. Here, no DRF offset must be active.
- Mutual monitoring of **tool**-related protection areas

Positioning axes

For positioning axes, only the programmed block end point is monitored. Alarm 10704 "Protection zone monitoring is not guaranteed" is output while a positioning axis traverses:

Axis interchange

With regard to the protection zones after an axis replacement, the starting position is the last position approached in the relinquishing channel. Traversing motion in the channel taking over is not taken into account. For this reason, you must ensure that an axis replacement is not performed from a position with protection zone violation.

If an axis intended for the axis interchange is not active in a channel, the position of the axis last approached in the channel is taken as the current position. If this axis has not yet been traversed in the channel, 0.0 is taken as the position.

Behavior for superimposed motions

Superimposed motions that are included in the main run, cannot be taken into account by the block preparation with regard to the active protection zones.

This results in the following reactions:

- Alarm 10704 "Protection area monitoring is not guaranteed"
- `<Chan>.basic.in.protAreasNotGuaranteed = 1` (protection area monitoring not guaranteed)

PLC signals

Basic Program Plus	Basic Program	
<code><Chan>.basic.in.protAreasNotGuaranteed</code>	<code>LBP_Chan*.E_ProtZoneNOK</code>	<code>DB21,DBX39.0</code>

5.6 Example

5.6.1 Protection zone on a lathe

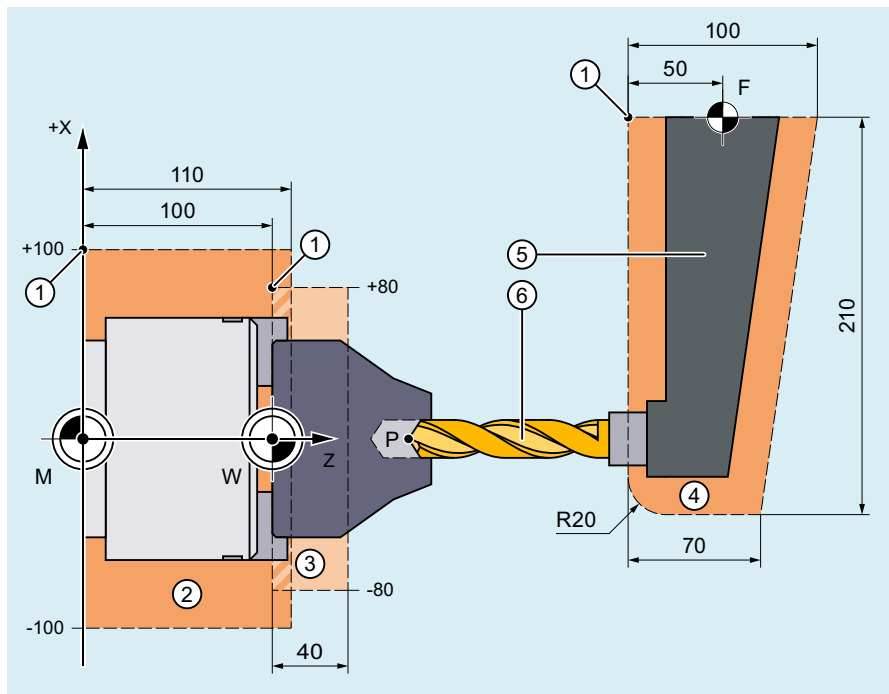
The following internal protection zones are to be defined for a turning machine:

- one machine-specific and workpiece-related protection zone for the spindle chuck, without limitation in the 3rd dimension
- one channel-specific protection zone for the workpiece, without limitation in the 3rd dimension
- one channel-specific, tool-related protection zone for the tool carrier, without limitation in the 3rd dimension

The workpiece zero is placed at machine zero to define the protection zone for the workpiece.

When activated, the protection zone is then offset by 100 mm in the Z axis in the positive direction.

5.6 Example



- ① Starting point
- ② Protection zone for spindle chuck
- ③ Protection zone for workpiece
- ④ Protection area for tool carrier
- ⑤ Tool carrier
- ⑥ Drill

5.6.2 Protection zone definition in the part program

Part program excerpt for protection zone definition:

Program code	Comment
DEF INT AB	
G18	; Working plane ZX
NPROTDEF(1,FALSE,0,0,0)	; Start of definition: Protection zone for spindle chuck
G01 X100 Z0	; Contour description: Traversing motion along the contour
G01 X-100 Z0	; Contour description: 1st Contour element
G01 X-100 Z110	; Contour description: 2nd Contour element
G01 X100 Z110	; Contour description: 3rd Contour element
G01 X100 Z0	; Contour description: 4th Contour element
EXECUTE (AB)	; End of definition: Protection zone for spindle chuck
CPROTDEF(1,FALSE,0,0,0)	; Start of definition: Protection zone for work-piece

Program code	Comment
G01 X80 Z0	; Contour description: Traversing motion along the contour
G01 X-80 Z0	; Contour description: 1st Contour element
G01 X-80 Z40	; Contour description: 2nd Contour element
G01 X80 Z40	; Contour description: 3rd Contour element
G01 X80 Z0	; Contour description: 4th Contour element
EXECUTE (AB)	; End of definition: Protection zone for workpiece
CPROTDEF (2, TRUE, 0, 0, 0)	; Start of definition: Protection zone for toolholder
G01 X0 Z-50	; Contour description: Traversing motion along the contour
G01 X-190 Z-50	; Contour description: 1st Contour element
G03 X-210 Z-30 I-20	; Contour description: 2nd Contour element
G01 X-210 Z20	; Contour description: 3rd Contour element
G01 X0 Z50	; Contour description: 4th Contour element
G01 X0 Z-50	; Contour description: 5th Contour element
EXECUTE (AB)	; End of definition: Protection zone for toolholder

5.6.3 Protection zone definition with system variables

Machine-specific protection zone for the spindle chuck

System variable	Value	Description
\$SN_PA_ACTIV_IMMED[0]	0	Protection zone for spindle chuck not immediately active
\$SN_PA_T_W[0]	0	The protection zone for the spindle chuck is workpiece-related
\$SN_PA_ORI[0]	1	Orientation of the protection zone: 1 = 3rd and 1st geometry axis
\$SN_PA_LIM_3DIM[0]	0	Type of limitation in the 3rd dimension: 0 = No limit
\$SN_PA_PLUS_LIM[0]	0	Value of the limit in the positive direction in the 3rd dimension
\$SN_PA_MINUS_LIM[0]	0	Value of the limit in the minus direction in the 3rd dimension
\$SN_PA_CONT_NUM[0]	4	Number of valid contour elements
\$SN_PA_CONT_TYP[0,0]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone for spindle chuck, contour element 1
\$SN_PA_CONT_TYP[0,1]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone for spindle chuck, contour element 2
\$SN_PA_CONT_TYP[0,2]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone for spindle chuck, contour element 3
\$SN_PA_CONT_TYP[0,3]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone for spindle chuck, contour element 4
\$SN_PA_CONT_TYP[0,4]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone for spindle chuck, contour element 5
\$SN_PA_CONT_TYP[0,5]	0	Contour type[<i>] : 0 = not defined, protection zone for spindle chuck, contour element 6
\$SN_PA_CONT_TYP[0,6]	0	Contour type[<i>] : 0 = not defined, protection zone for spindle chuck, contour element 7

5.6 Example

System variable	Value	Description
\$SN_PA_CONT_TYP[0,7]	0	Contour type[<i>] : 0 = not defined, protection zone for spindle chuck, contour element 8
\$SN_PA_CONT_TYP[0,8]	0	Contour type[<i>] : 0 = not defined, protection zone for spindle chuck, contour element 9
\$SN_PA_CONT_TYP[0,9]	0	Contour type[<i>] : 0 = not defined, protection zone for spindle chuck, contour element 10
\$SN_PA_CONT_ORD[0,0]	-100	End point of contour[<i>], ordinate value protection zone for spindle chuck, contour element 1
\$SN_PA_CONT_ORD[0,1]	-100	End point of contour[<i>], ordinate value protection zone for spindle chuck, contour element 2
\$SN_PA_CONT_ORD[0,2]	100	End point of contour[<i>], ordinate value protection zone for spindle chuck, contour element 3
\$SN_PA_CONT_ORD[0,3]	100	End point of contour[<i>], ordinate value protection zone for spindle chuck, contour element 4
\$SN_PA_CONT_ORD[0,4]	0	End point of contour[<i>], ordinate value protection zone for spindle chuck, contour element 5
\$SN_PA_CONT_ORD[0,5]	0	End point of contour[<i>], ordinate value protection zone for spindle chuck, contour element 6
\$SN_PA_CONT_ORD[0,6]	0	End point of contour[<i>], ordinate value protection zone for spindle chuck, contour element 7
\$SN_PA_CONT_ORD[0,7]	0	End point of contour[<i>], ordinate value protection zone for spindle chuck, contour element 8
\$SN_PA_CONT_ORD[0,8]	0	End point of contour[<i>], ordinate value protection zone for spindle chuck, contour element 9
\$SN_PA_CONT_ORD[0,9]	0	End point of contour[<i>], ordinate value protection zone for spindle chuck, contour element 10
\$SN_PA_CONT_ABS[0,0]	0	End point of contour[<i>], abscissa value protection zone for spindle chuck, contour element 1
\$SN_PA_CONT_ABS[0,1]	110	End point of contour[<i>], abscissa value protection zone for spindle chuck, contour element 2
\$SN_PA_CONT_ABS[0,2]	110	End point of contour[<i>], abscissa value protection zone for spindle chuck, contour element 3
\$SN_PA_CONT_ABS[0,3]	0	End point of contour[<i>], abscissa value protection zone for spindle chuck, contour element 4
\$SN_PA_CONT_ABS[0,4]	0	End point of contour[<i>], abscissa value protection zone for spindle chuck, contour element 5
\$SN_PA_CONT_ABS[0,5]	0	End point of contour[<i>], abscissa value protection zone for spindle chuck, contour element 6
\$SN_PA_CONT_ABS[0,6]	0	End point of contour[<i>], abscissa value protection zone for spindle chuck, contour element 7
\$SN_PA_CONT_ABS[0,7]	0	End point of contour[<i>], abscissa value protection zone for spindle chuck, contour element 8
\$SN_PA_CONT_ABS[0,8]	0	End point of contour[<i>], abscissa value protection zone for spindle chuck, contour element 9
\$SN_PA_CONT_ABS[0,9]	0	End point of contour[<i>], abscissa value protection zone for spindle chuck, contour element 10

System variable	Value	Description
\$SN_PA_CENT_ORD[0,0]	0	Center point of circular contour[<i>], ordinate value protection zone for spindle chuck, contour element 1
\$SN_PA_CENT_ORD[0,1]	0	Center point of circular contour[<i>], ordinate value protection zone for spindle chuck, contour element 2
\$SN_PA_CENT_ORD[0,2]	0	Center point of circular contour[<i>], ordinate value protection zone for spindle chuck, contour element 3
\$SN_PA_CENT_ORD[0,3]	0	Center point of circular contour[<i>], ordinate value protection zone for spindle chuck, contour element 4
\$SN_PA_CENT_ORD[0,4]	0	Center point of circular contour[<i>], ordinate value protection zone for spindle chuck, contour element 5
\$SN_PA_CENT_ORD[0,5]	0	Center point of circular contour[<i>], ordinate value protection zone for spindle chuck, contour element 6
\$SN_PA_CENT_ORD[0,6]	0	Center point of circular contour[<i>], ordinate value protection zone for spindle chuck, contour element 7
\$SN_PA_CENT_ORD[0,7]	0	Center point of circular contour[<i>], ordinate value protection zone for spindle chuck, contour element 8
\$SN_PA_CENT_ORD[0,8]	0	Center point of circular contour[<i>], ordinate value protection zone for spindle chuck, contour element 9
\$SN_PA_CENT_ORD[0,9]	0	Center point of circular contour[<i>], ordinate value protection zone for spindle chuck, contour element 10
\$SN_PA_CENT_ABS[0,0]	0	Center point of circular contour[<i>], abscissa value protection zone for spindle chuck, contour element 1
\$SN_PA_CENT_ABS[0,1]	0	Center point of circular contour[<i>], abscissa value protection zone for spindle chuck, contour element 2
\$SN_PA_CENT_ABS[0,2]	0	Center point of circular contour[<i>], abscissa value protection zone for spindle chuck, contour element 3
\$SN_PA_CENT_ABS[0,3]	0	Center point of circular contour[<i>], abscissa value protection zone for spindle chuck, contour element 4
\$SN_PA_CENT_ABS[0,4]	0	Center point of circular contour[<i>], abscissa value protection zone for spindle chuck, contour element 5
\$SN_PA_CENT_ABS[0,5]	0	Center point of circular contour[<i>], abscissa value protection zone for spindle chuck, contour element 6
\$SN_PA_CENT_ABS[0,6]	0	Center point of circular contour[<i>], abscissa value protection zone for spindle chuck, contour element 7
\$SN_PA_CENT_ABS[0,7]	0	Center point of circular contour[<i>], abscissa value protection zone for spindle chuck, contour element 8
\$SN_PA_CENT_ABS[0,8]	0	Center point of circular contour[<i>], abscissa value protection zone for spindle chuck, contour element 9
\$SN_PA_CENT_ABS[0,9]	0	Center point of circular contour[<i>], abscissa value protection zone for spindle chuck, contour element 10

Channel-specific protection zone for the workpiece

System variable	Value	Remark
\$SC_PA_ACTIV_IMMED[0]	0	Protection zone for workpiece not immediately active
\$SC_PA_TW[0]	0	Protection zone for workpiece is workpiece-related

5.6 Example

System variable	Value	Remark
\$SC_PA_ORI[0]	1	Orientation of the protection zone: 1 = 3rd and 1st geometry axis
\$SC_PA_LIM_3DIM[0]	0	Type of limitation in the 3rd dimension: 0 = no limitation
\$SC_PA_PLUS_LIM[0]	0	Value of the limit in the positive direction in the 3rd dimension
\$SC_PA_MINUS_LIM[0]	0	Value of the limit in the minus direction in the 3rd dimension
\$SC_PA_CONT_NUM[0]	4	Number of valid contour elements
\$SC_PA_CONT_TYP[0,0]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone workpiece, contour element 1
\$SC_PA_CONT_TYP[0,1]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone workpiece, contour element 2
\$SC_PA_CONT_TYP[0,2]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone workpiece, contour element 3
\$SC_PA_CONT_TYP[0,3]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone workpiece, contour element 4
\$SC_PA_CONT_TYP[0,4]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone workpiece, contour element 5
\$SC_PA_CONT_TYP[0,5]	0	Contour type[<i>] : 0 = not defined, protection zone for workpiece, contour element 6
\$SC_PA_CONT_TYP[0,6]	0	Contour type[<i>] : 0 = not defined, protection zone for workpiece, contour element 7
\$SC_PA_CONT_TYP[0,7]	0	Contour type[<i>] : 0 = not defined, protection zone for workpiece, contour element 8
\$SC_PA_CONT_TYP[0,8]	0	Contour type[<i>] : 0 = not defined, protection zone for workpiece, contour element 9
\$SC_PA_CONT_TYP[0,9]	0	Contour type[<i>] : 0 = not defined, protection zone for workpiece, contour element 10
\$SC_PA_CONT_ORD[0,0]	-80	End point of contour[<i>], ordinate value protection zone for workpiece, contour element 1
\$SC_PA_CONT_ORD[0,1]	-80	End point of contour[<i>], ordinate value protection zone for workpiece, contour element 2
\$SC_PA_CONT_ORD[0,2]	80	End point of contour[<i>], ordinate value protection zone for workpiece, contour element 3
\$SC_PA_CONT_ORD[0,3]	80	End point of contour[<i>], ordinate value protection zone for workpiece, contour element 4
\$SC_PA_CONT_ORD[0,4]	0	End point of contour[<i>], ordinate value protection zone for workpiece, contour element 5
\$SC_PA_CONT_ORD[0,5]	0	End point of contour[<i>], ordinate value protection zone for workpiece, contour element 6
\$SC_PA_CONT_ORD[0,6]	0	End point of contour[<i>], ordinate value protection zone for workpiece, contour element 7
\$SC_PA_CONT_ORD[0,7]	0	End point of contour[<i>], ordinate value protection zone for workpiece, contour element 8
\$SC_PA_CONT_ORD[0,8]	0	End point of contour[<i>], ordinate value protection zone for workpiece, contour element 9
\$SC_PA_CONT_ORD[0,9]	0	End point of contour[<i>], ordinate value protection zone for workpiece, contour element 10
\$SC_PA_CONT_ABS[0,0]	0	End point of contour[<i>], abscissa value protection zone for workpiece, contour element 1

System variable	Value	Remark
\$SC_PA_CONT_ABS[0,1]	40	End point of contour[<i>], abscissa value protection zone for workpiece, contour element 2
\$SC_PA_CONT_ABS[0,2]	40	End point of contour[<i>], abscissa value protection zone for workpiece, contour element 3
\$SC_PA_CONT_ABS[0,3]	0	End point of contour[<i>], abscissa value protection zone for workpiece, contour element 4
\$SC_PA_CONT_ABS[0,4]	0	End point of contour[<i>], abscissa value protection zone for workpiece, contour element 5
\$SC_PA_CONT_ABS[0,5]	0	End point of contour[<i>], abscissa value protection zone for workpiece, contour element 6
\$SC_PA_CONT_ABS[0,6]	0	End point of contour[<i>], abscissa value protection zone for workpiece, contour element 7
\$SC_PA_CONT_ABS[0,7]	0	End point of contour[<i>], abscissa value protection zone for workpiece, contour element 8
\$SC_PA_CONT_ABS[0,8]	0	End point of contour[<i>], abscissa value protection zone for workpiece, contour element 9
\$SC_PA_CONT_ABS[0,9]	0	End point of contour[<i>], abscissa value protection zone for workpiece, contour element 10
\$SC_PA_CENT_ORD[0,0]	0	Center point of circular contour[<i>], ordinate value protection zone for workpiece, contour element 1
\$SC_PA_CENT_ORD[0,1]	0	Center point of circular contour[<i>], ordinate value protection zone for workpiece, contour element 2
\$SC_PA_CENT_ORD[0,2]	0	Center point of circular contour[<i>], ordinate value protection zone for workpiece, contour element 3
\$SC_PA_CENT_ORD[0,3]	0	Center point of circular contour[<i>], ordinate value protection zone for workpiece, contour element 4
\$SC_PA_CENT_ORD[0,4]	0	Center point of circular contour[<i>], ordinate value protection zone for workpiece, contour element 5
\$SC_PA_CENT_ORD[0,5]	0	Center point of circular contour[<i>], ordinate value protection zone for workpiece, contour element 6
\$SC_PA_CENT_ORD[0,6]	0	Center point of circular contour[<i>], ordinate value protection zone for workpiece, contour element 7
\$SC_PA_CENT_ORD[0,7]	0	Center point of circular contour[<i>], ordinate value protection zone for workpiece, contour element 8
\$SC_PA_CENT_ORD[0,8]	0	Center point of circular contour[<i>], ordinate value protection zone for workpiece, contour element 9
\$SC_PA_CENT_ORD[0,9]	0	Center point of circular contour[<i>], ordinate value protection zone for tool holder, contour element 10
\$SC_PA_CENT_ABS[0,0]	0	Center point of circular contour[<i>], abscissa value protection zone for workpiece, contour element 1
\$SC_PA_CENT_ABS[0,1]	0	Center point of circular contour[<i>], abscissa value protection zone for workpiece, contour element 2
\$SC_PA_CENT_ABS[0,2]	0	Center point of circular contour[<i>], abscissa value protection zone for workpiece, contour element 3
\$SC_PA_CENT_ABS[0,3]	0	Center point of circular contour[<i>], abscissa value protection zone for workpiece, contour element 4

5.6 Example

System variable	Value	Remark
\$SC_PA_CENT_ABS[0,4]	0	Center point of circular contour[<i>], abscissa value protection zone for workpiece, contour element 5
\$SC_PA_CENT_ABS[0,5]	0	Center point of circular contour[<i>], abscissa value protection zone for workpiece, contour element 6
\$SC_PA_CENT_ABS[0,6]	0	Center point of circular contour[<i>], abscissa value protection zone for workpiece, contour element 7
\$SC_PA_CENT_ABS[0,7]	0	Center point of circular contour[<i>], abscissa value protection zone for workpiece, contour element 8
\$SC_PA_CENT_ABS[0,8]	0	Center point of circular contour[<i>], abscissa value protection zone for workpiece, contour element 9
\$SC_PA_CENT_ABS[0,9]	0	Center point of circular contour[<i>], abscissa value protection zone for workpiece, contour element 10

Channel-specific protection zone for the tool holder

System variable	Value	Remark
\$SC_PA_ACTIV_IMMED[1]	0	Protection zone for tool holder not immediately active
\$SC_PA_TW[1]	3	Protection zone for the tool holder is tool-related
\$SC_PA_ORI[1]	1	Orientation of the protection zone: 1 = 3rd and 1st geometry axis
\$SC_PA_LIM_3DIM[1]	0	Type of limitation in the 3rd dimension: 0 = no limitation
\$SC_PA_PLUS_LIM[1]	0	Value of the limit in the positive direction in the 3rd dimension
\$SC_PA_MINUS_LIM[1]	0	Value of the limit in the minus direction in the 3rd dimension
\$SC_PA_CONT_NUM[1]	5	Number of valid contour elements
\$SC_PA_CONT_TYP[1,0]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone tool holder, contour element 1
\$SC_PA_CONT_TYP[1,1]	3	Contour type[<i>] : 3 = G3 for circle element, counter clockwise, protection zone for tool holder, contour element 2
\$SC_PA_CONT_TYP[1,2]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone for tool holder, contour element 3
\$SC_PA_CONT_TYP[1,3]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone for tool holder, contour element 4
\$SC_PA_CONT_TYP[1,4]	1	Contour type[<i>] : 1 = G1 for straight line, protection zone for tool holder, contour element 5
\$SC_PA_CONT_TYP[1,5]	0	Contour type[<i>] : 0 = not defined, protection zone for tool holder, contour element 6
\$SC_PA_CONT_TYP[1,6]	0	Contour type[<i>] : 0 = not defined, protection zone for tool holder, contour element 7
\$SC_PA_CONT_TYP[1,7]	0	Contour type[<i>] : 0 = not defined, protection zone for tool holder, contour element 8
\$SC_PA_CONT_TYP[1,8]	0	Contour type[<i>] : 0 = not defined, protection zone for tool holder, contour element 9
\$SC_PA_CONT_TYP[1,9]	0	Contour type[<i>] : 0 = not defined, protection zone for tool holder, contour element 10
\$SC_PA_CONT_ORD[1,0]	-190	End point of contour[<i>], ordinate value protection zone for tool holder, contour element 1

System variable	Value	Remark
\$SC_PA_CONT_ORD[1,1]	-210	End point of contour[<i>], ordinate value protection zone for tool holder, contour element 2
\$SC_PA_CONT_ORD[1,2]	-210	End point of contour[<i>], ordinate value protection zone for tool holder, contour element 3
\$SC_PA_CONT_ORD[1,3]	0	End point of contour[<i>], ordinate value protection zone for tool holder, contour element 4
\$SC_PA_CONT_ORD[1,4]	0	End point of contour[<i>], ordinate value protection zone for tool holder, contour element 5
\$SC_PA_CONT_ORD[1,5]	0	End point of contour[<i>], ordinate value protection zone for tool holder, contour element 6
\$SC_PA_CONT_ORD[1,6]	0	End point of contour[<i>], ordinate value protection zone for tool holder, contour element 7
\$SC_PA_CONT_ORD[1,7]	0	End point of contour[<i>], ordinate value protection zone for tool holder, contour element 8
\$SC_PA_CONT_ORD[1,8]	0	End point of contour[<i>], ordinate value protection zone for tool holder, contour element 9
\$SC_PA_CONT_ORD[1,9]	0	End point of contour[<i>], ordinate value protection zone for tool holder, contour element 10
\$SC_PA_CONT_ABS[1,0]	-50	End point of contour[<i>], abscissa value protection zone for tool holder, contour element 1
\$SC_PA_CONT_ABS[1,1]	-30	End point of contour[<i>], abscissa value protection zone for tool holder, contour element 2
\$SC_PA_CONT_ABS[1,2]	20	End point of contour[<i>], abscissa value protection zone for tool holder, contour element 3
\$SC_PA_CONT_ABS[1,3]	50	End point of contour[<i>], abscissa value protection zone for tool holder, contour element 4
\$SC_PA_CONT_ABS[1,4]	-50	End point of contour[<i>], abscissa value protection zone for tool holder, contour element 5
\$SC_PA_CONT_ABS[1,5]	0	End point of contour[<i>], abscissa value protection zone for tool holder, contour element 6
\$SC_PA_CONT_ABS[1,6]	0	End point of contour[<i>], abscissa value protection zone for tool holder, contour element 7
\$SC_PA_CONT_ABS[1,7]	0	End point of contour[<i>], abscissa value protection zone for tool holder, contour element 8
\$SC_PA_CONT_ABS[1,8]	0	End point of contour[<i>], abscissa value protection zone for tool holder, contour element 9
\$SC_PA_CONT_ABS[1,9]	0	End point of contour[<i>], abscissa value protection zone for tool holder, contour element 10
\$SC_PA_CENT_ORD[1,0]	0	Center point of circular contour[<i>], ordinate value protection zone for tool holder, contour element 1
\$SC_PA_CENT_ORD[1,1]	-190	Center point of circular contour[<i>], ordinate value protection zone for tool holder, contour element 2
\$SC_PA_CENT_ORD[1,2]	0	Center point of circular contour[<i>], ordinate value protection zone for tool holder, contour element 3
\$SC_PA_CENT_ORD[1,3]	0	Center point of circular contour[<i>], ordinate value protection zone for tool holder, contour element 4

5.6 Example

System variable	Value	Remark
\$SC_PA_CENT_ORD[1,4]	0	Center point of circular contour[<i>], ordinate value protection zone for tool holder, contour element 5
\$SC_PA_CENT_ORD[1,5]	0	Center point of circular contour[<i>], ordinate value protection zone for tool holder, contour element 6
\$SC_PA_CENT_ORD[1,6]	0	Center point of circular contour[<i>], ordinate value protection zone for tool holder, contour element 7
\$SC_PA_CENT_ORD[1,7]	0	Center point of circular contour[<i>], ordinate value protection zone for tool holder, contour element 8
\$SC_PA_CENT_ORD[1,8]	0	Center point of circular contour[<i>], ordinate value protection zone for tool holder, contour element 9
\$SC_PA_CENT_ORD[1,9]	0	Center point of circular contour[<i>], ordinate value protection zone for tool holder, contour element 10
\$SC_PA_CENT_ABS[1,0]	0	Center point of circular contour[<i>], abscissa value protection zone for tool holder, contour element 1
\$SC_PA_CENT_ABS[1,1]	-30	Center point of circular contour[<i>], abscissa value protection zone for tool holder, contour element 2
\$SC_PA_CENT_ABS[1,2]	0	Center point of circular contour[<i>], abscissa value protection zone for tool holder, contour element 3
\$SC_PA_CENT_ABS[1,3]	0	Center point of circular contour[<i>], abscissa value protection zone for tool holder, contour element 4
\$SC_PA_CENT_ABS[1,4]	0	Center point of circular contour[<i>], abscissa value protection zone for tool holder, contour element 5
\$SC_PA_CENT_ABS[1,5]	0	Center point of circular contour[<i>], abscissa value protection zone for tool holder, contour element 6
\$SC_PA_CENT_ABS[1,6]	0	Center point of circular contour[<i>], abscissa value protection zone for tool holder, contour element 7
\$SC_PA_CENT_ABS[1,7]	0	Center point of circular contour[<i>], abscissa value protection zone for tool holder, contour element 8
\$SC_PA_CENT_ABS[1,8]	0	Center point of circular contour[<i>], abscissa value protection zone for tool holder, contour element 9
\$SC_PA_CENT_ABS[1,9]	0	Center point of circular contour[<i>], abscissa value protection zone for tool holder, contour element 10

5.6.4 Activating protection zones

Part program excerpt for activating protection zones for spindle chuck, workpiece, and tool holder:

Program code	Comment
NPROT(1,2,0,0,0)	; Protection zone: Spindle chuck
CPROT(1,2,0,0,100)	; Protection zone: Workpiece with 100 mm offset in the Z axis
CPROT(2,2,0,0,0)	; Protection zone: Toolholder

Axis monitoring functions

6.1 Contour monitoring

6.1.1 Contour error

Contour errors are caused by signal distortions in the position control loop.

Signal distortions can be linear or non-linear.

Linear signal distortions

Linear signal distortions are caused by:

- Speed and position controller not being set optimally
- Unequal following errors of the axes involved in the path
In this regard, the feedforward control types, DSC setting and equivalent time constants of the feedforward control should usually be set the same. Additional checks should include: Position setpoint filter (e.g. AX_JERK_MODE and AX_JERK_TIME), as well as Kv (in particular for FFWOF, FFW_MODE=0 or FFW_MODE=3).
- Unequal dynamic response of the feed drives
Unequal drive dynamic responses lead to path deviations especially on contour changes. Circles are distorted into ellipses by unequal dynamic responses of the two feed drives.

Non-linear signal distortions

Non-linear signal distortions are caused by:

- Activation of the current limitation within the machining area
- Activation of the limitation of the speed setpoint
- Range of inversion within and/or outside of the position control loop
When traversing a circular path, contour errors occur primarily due to the reversal error and friction.
During motion along straight lines, a contour error arises due to a reversal error outside the position control loop, e.g. due to a tilting milling spindle. This causes a parallel offset between the actual and the set contour. The shallower the gradient of the straight line, the larger the offset.
- Nonlinear friction behavior of slide guides

6.1.2 Following-error monitoring

Function

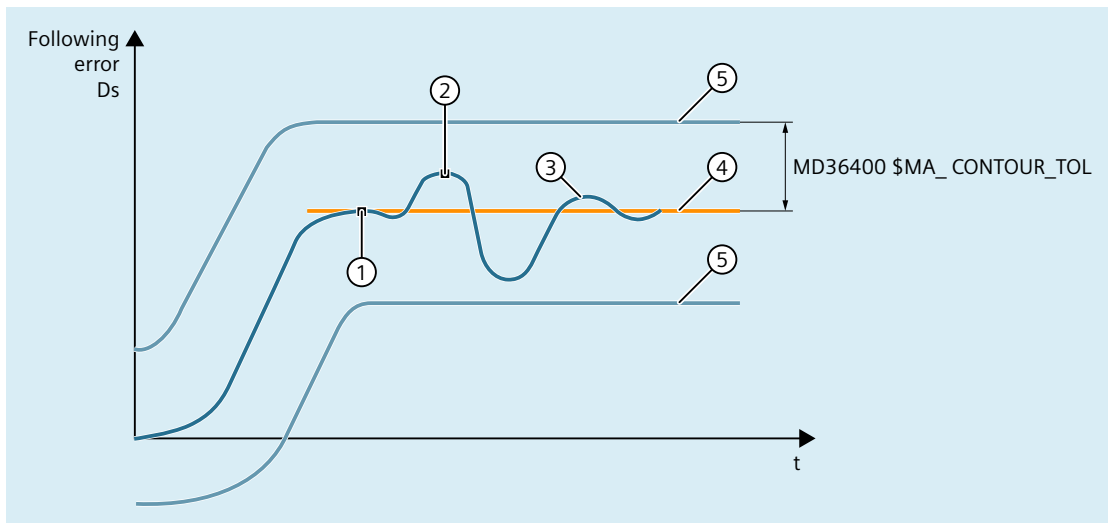
In control engineering terms, traversing along a machine axis always produces a certain following error, i.e. a difference between the set and actual position.

The following error that arises depends on:

- Position control loop gain
MD32200 \$MA_POSCTRL_GAIN (servo gain factor)
- Maximum acceleration
MD32300 \$MA_MAX_AX_ACCEL (maximum axis acceleration)
- Maximum velocity
MD32000 \$MA_MAX_AX_VELO (maximum axis velocity)
- With activated feedforward control:
Precision of the path model and the parameters:
MD32610 \$MA_VELO_FFW_WEIGHT (factor for the velocity feedforward control)
MD32620 \$MA_FFW_MODE (feedforward control mode)
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)
- Position setpoint filter
MD32402 \$MA_AX_JERK_MODE (filter type for axial jerk limitation)
MD32410 \$MA_AX_JERK_TIME (time constant for the axial jerk filter)
MD32910 4ma_dyn_match_TIME (time constant for dynamic response adaptation)

In the acceleration phase, the following error initially increases when traversing along a machine axis. After a time depending on the parameterization of the position control loop, the following error then remains constant in the ideal case. Due to external influences, more or less large fluctuations in the following error always arise during a machining process. To prevent these fluctuations in the following error from triggering an alarm, a tolerance range within which the following error may change must be defined for the following-error monitoring:

MD36400 \$MA_CONTOUR_TOL (Contour monitoring tolerance range)



- ① Start machining
- ② Section
- ③ Measured distance
- ④ Calculated following error (model)
- ⑤ Tolerance band

Figure 6-1 Following-error monitoring

Effectiveness

The following-error monitoring only operates with active position control and the following axis types:

- Linear axes with and without feedforward control
- Rotary axes with and without feedforward control
- Position-controlled spindles

Fault

If the configured tolerance limit is exceeded, the following alarm appears:

25050 "Axis <Axis name> Contour monitoring"

The affected axis/spindle is stopped via the configured braking ramp in follow-up mode:

MD36610 \$MA_AX_EMERGENCY_STOP_TIME

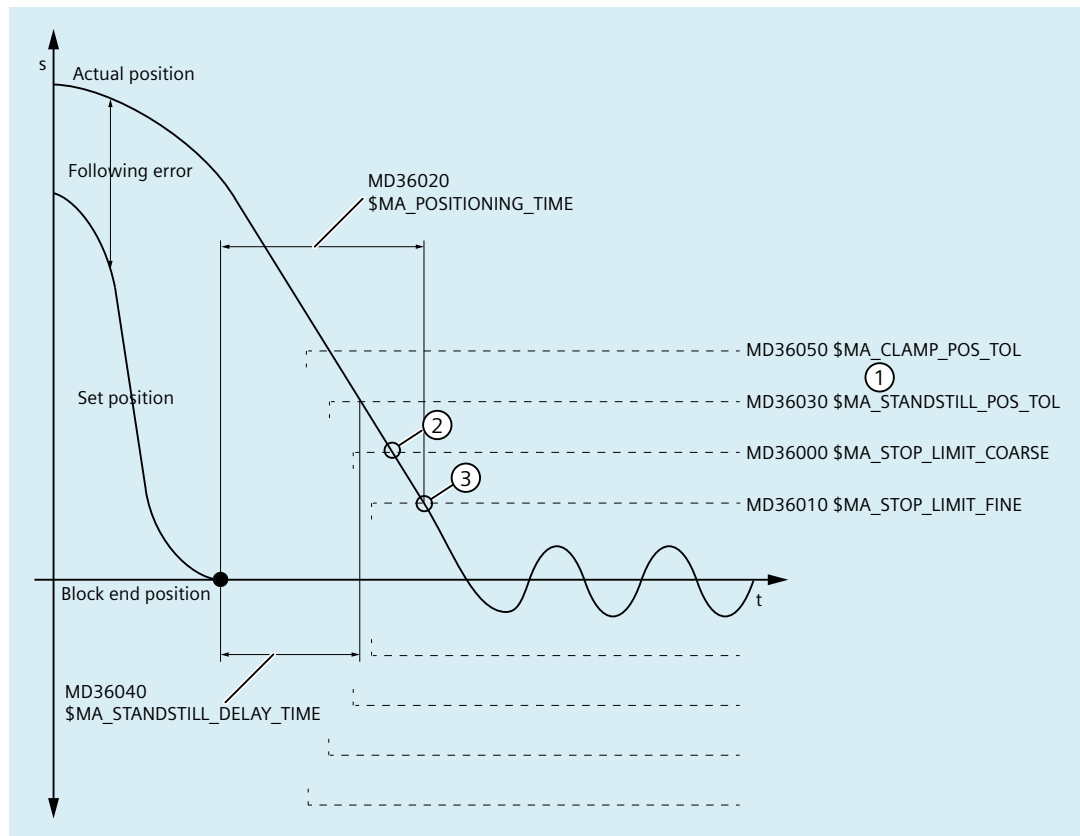
(maximum time for braking ramp when an error occurs)

6.2 Positioning, zero speed and clamping monitoring

6.2.1 Correlation between positioning, zero-speed and clamping monitoring

Overview

The following overview shows the correlation between the positioning, zero speed and clamping monitoring functions:



- ① Monitored limit value depending on NC/PLC interface signal:
`<Axis>.basic.out.clampingInProgress`
 - 0: MD36030 \$MA_STANDSTILL_POS_TOL
 - 1: MD36050 \$MA_CLAMP_POS_TOL
- ② Exact stop coarse reached: `<Axis>.basic.in.posCoarseReached = 1`
- ③ Exact stop fine reached: `<Axis>.basic.in.posFineReached = 1`

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.out.clampingInProgress	LBP_Axis*.A_Clamp	DB31,DBX2.3
<Axis>.basic.in.posCoarseReached	LBP_Axis*.E_ExactCoarse	DB31,DBX60.6
<Axis>.basic.in.posFineReached	LBP_Axis*.E_ExactFine	DB31,DBX60.7

6.2.2 Positioning monitoring

Function

At the end of a positioning operation:

- Set velocity = 0 **AND**
- <Axis>.basic.in.traversMinusCommandActiv /
 <Axis>.basic.in.traversPlusCommandActive (traversing command minus / plus)
 = 0

Checks the position monitoring to ensure that the following error of every participating machine axis is smaller than the "Exact stop tolerance fine" during the delay time:

- MD36010 \$MA_STOP_LIMIT_FINE (exact stop fine)
- MD36020 \$MA_POSITIONING_TIME (delay time exact stop fine)

After reaching "Exact stop fine", the position monitoring is deactivated.

Note

The smaller the "Exact stop tolerance fine" is, the longer the positioning operation takes and the longer the time until block change.

Rules for MD setting

Table 6-1 "Exact stop fine" with regard to positioning time

MD36010 \$MA_STOP_LIMIT_FINE	MD36020 \$MA_POSITIONING_TIME
Large	Short possible
Small	Long required

Table 6-2 Kv factor with respect to positioning time

MD32200 \$MA_POSCTRL_GAIN (servo gain factor)	MD36020 \$MA_POSITIONING_TIME
Small	Long required
Large	Short possible

Effectiveness

The position monitoring only operates with active position control and the following axis types:

- Linear axes
- Rotary axes
- Position-controlled spindles

Fault

If the configured position-monitoring time is exceeded, the following alarm appears:

25080 "Axis <Axis name> Position monitoring"

The affected axis is stopped via the configured braking ramp in follow-up mode:

MD36610 \$MA_AX_EMERGENCY_STOP_TIME (maximum time for braking ramp when a fault occurs)

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.in.traversMinusCommandActiv	LBP_Axis*.E_TCMinus	DB31,DBX64.6
<Axis>.basic.in.traversPlusCommandActive	LBP_Axis*.E_TCPlus	DB31,DBX64.7

6.2.3 Zero-speed monitoring

Function

At the end of a positioning operation:

- Set velocity = 0 **AND**
- <Axis>.basic.in.traversMinusCommandActiv /
<Axis>.basic.in.traversPlusCommandActive (traversing command minus / plus)
= 0

checks the zero-speed monitoring to ensure that the following error of every participating machine axis is smaller than the standstill tolerance during the delay time.

- MD36040 \$MA_STANDSTILL_DELAY_TIME (zero-speed monitoring delay time)
- MD36030 \$MA_STANDSTILL_POS_TOL (standstill tolerance)

After reaching the required exact-stop state, the positioning operation is completed:

<Axis>.basic.in.posCoarseReached /
<Axis>.basic.in.posFineReached (position reached with exact stop coarse/fine) = 1

The position-monitoring function is deactivated and is replaced by the zero-speed monitoring.

Zero-speed monitoring monitors the adherence to the standstill tolerance. If no new travel request is received, the machine axis must not depart from the standstill tolerance.

Effectiveness

The zero-speed monitoring only operates with active position control and the following axis types:

- Linear axes
- Rotary axes
- Position-controlled spindles

Fault

If the delay time and/or the standstill tolerance is exceeded, the following alarm appears:

25040 "Axis <Axis name> Zero-speed monitoring"

The affected axis is stopped via the configured braking ramp in follow-up mode:

MD36610 \$MA_AX_EMERGENCY_STOP_TIME

(maximum time for braking ramp when an error occurs)

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.in.traversMinusCommandActiv	LBP_Axis*.E_TCMinus	DB31,DBX64.6
<Axis>.basic.in.traversPlusCommandActive	LBP_Axis*.E_TCPlus	DB31,DBX64.7
<Axis>.basic.in.posCoarseReached	LBP_Axis*.E_ExactCoarse	DB31,DBX60.6
<Axis>.basic.in.posFineReached	LBP_Axis*.E_ExactFine	DB31,DBX60.7

6.2.4 Parameter set-dependent exact stop and standstill tolerance

For adaptation to different machining situations and/or axis dynamics, e.g.:

- Operating state A: High precision, long machining time
- Operating state B: Lower precision, shorter machining time
- Changing of the mass relationships after gear change

the positioning tolerances:

- MD36000 \$MA_STOP_LIMIT_COARSE (exact stop coarse)
- MD36010 \$MA_STOP_LIMIT_FINE (exact stop fine)
- MD36030 \$MA_STANDSTILL_POS_TOL (standstill tolerance)

can be weighted with a common factor depending on the parameter set:

MD36012 \$MA_STOP_LIMIT_FACTOR (exact stop coarse/fine and standstill factor)

Because the factor applies in common for all three position tolerances, the relationship between the values remains constant.

6.2.5 Clamping monitoring

6.2.5.1 Function

For machine axes that are mechanically clamped upon completion of a positioning operation, a displacement of the axis from the position setpoint can result from the clamping process. Setting the following NC/PLC interface signal causes the clamping tolerance (MD36050 \$MA_CLAMP_POS_TOL) rather than the standstill tolerance (MD36030 \$MA_STANDSTILL_POS_TOL) to be monitored for the duration of the clamping process:

`<Axis>.basic.out.clampingInProgress` (clamping in progress)

Alarm 26000 "clamping monitoring" is issued if the clamping tolerance is overshoot.

Alarm delay time

If a time-limited overshooting of the clamping tolerance is permitted, an alarm delay time must be specified via machine data MD36051 \$MA_CLAMP_POS_TOL_TIME. The alarm is then output only after expiration of the parameterized time when the clamping tolerance is overshoot. No alarm is output when the clamping tolerance is undershot again after expiration of the time. The time is restarted when the clamping tolerance is overshoot again.

To permit a response to overshooting the clamping tolerance prior to expiration of the alarm delay time, the following axis-specific NC/PLC interface signal is set:

`<Axis>.basic.in.clampingToleranceExceeded` (clamping tolerance exceeded)

The signal is reset again when the clamping tolerance is undershot.

PLC signals

Basic Program Plus	Basic Program	
<code><Axis>.basic.out.clampingInProgress</code>	<code>LBP_Axis*.A_Clamp</code>	DB31,DBX2.3
<code><Axis>.basic.in.clampingToleranceExceeded</code>	<code>LBP_Axis*.E_ClampTolerance</code>	DB31,DBX102.3

6.2.5.2 Machine data

Clamping tolerance

The clamping tolerance, which is higher than the standstill tolerance, is entered in machine data:

MD36050 \$MA_CLAMP_POS_TOL[<axis>] = <clamping tolerance>

Alarm delay time

If a time-limited overshooting of the clamping tolerance is tolerated, the maximum permissible alarm delay time must be specified in the machine data.

MD36051 \$MA_CLAMP_POS_TOL_TIME[<axis>] = <alarm delay time>

When the clamping tolerance is exceeded, alarm 26000 "Clamping monitoring" is only displayed after the alarm delay time has elapsed.

An alarm is not output, if the clamping tolerance is fallen below again before the alarm delay time lapses.

The alarm delay time is restarted when the clamping tolerance is exceeded again.

Special clamping functions

The special clamping functions that automate the release and set of the part program execution sequence are activated bit-by-bit using machine data:

MD36052 \$MA_STOP_ON_CLAMPING[<Achse>], <bit> = <value>

<Bit>	<Value>	Meaning
0		Automatic stop to release the clamping
	0	Not active
	1	Active
1		Optimized clamping release
	0	Not active
	1	Active, precondition: Bit 0 == 1
2		Automatic stop to set the clamping
	0	Not active
	1	Active

6.2.5.3 NC/PLC interface signals

Activating the clamping monitoring

The clamping monitoring is activated by setting the NC/PLC interface signal:

<Axis>.basic.out.clampingInProgress (clamping in progress)

Overshooting the clamping tolerance

Overshooting the clamping tolerance is indicated with the NC/PLC interface signal:

<Axis>.basic.in.clampingToleranceExceeded (clamping tolerance exceeded)

The signal is **set** when the clamping tolerance is overshoot within the alarm delay time.

The signal is **reset** when the clamping tolerance is undershot within the alarm delay time or the follow-up mode is activated for the axis.

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.out.clampingInProgress	LBP_Axis*.A_Clamp	DB31,DBX2.3
<Axis>.basic.in.clampingToleranceExceeded	LBP_Axis*.E_ClampTolerance	DB31,DBX102.3

6.2.5.4 Fault responses

Fault responses when the clamping tolerance is exceeded:

- Alarm 26000 "Clamping monitoring" is indicated
- The axis is brought to a standstill with the parameterized maximum acceleration:
MD32300 \$MA_MAX_AX_ACCEL
Whereby, the maximum duration of the braking ramp for error states is monitored:
MD36610 \$MA_AX_EMERGENCY_STOP_TIME
- Follow-up mode is activated for the axis:
`<Axis>.basic.in.followUpModeActive == 1`
- The "clamping tolerance exceeded" signal is reset:
`<Axis>.basic.in.clampingToleranceExceeded == 0`

PLC signals

Basic Program Plus	Basic Program	
<code><Axis>.basic.in.followUpModeActive</code>	<code>LBP_Axis*.E_Follow_upMode</code>	DB31,DBX61.3
<code><Axis>.basic.in.clampingToleranceExceeded</code>	<code>LBP_Axis*.E_ClampTolerance</code>	DB31,DBX102.3

6.2.5.5 "Automatic stop to release the clamping" clamping function

The "Automatic stop to release the clamping" clamping function adds an NC-internal stop before the traversing block of the clamping axis for continuous-path mode.

The stop is not effective or the continuous path mode is not interrupted if the controller enable signal (`<Axis>.basic.out.enableCtrl`) of the clamping axis is set prior to the block change.

If the controller enable signal of the clamping axis is **not** set prior to the block change, the stop acts.

Parameterization

MD36052 \$MA_STOP_ON_CLAMPING[<clamping axis>] = 'H01'

Requirements/assumptions

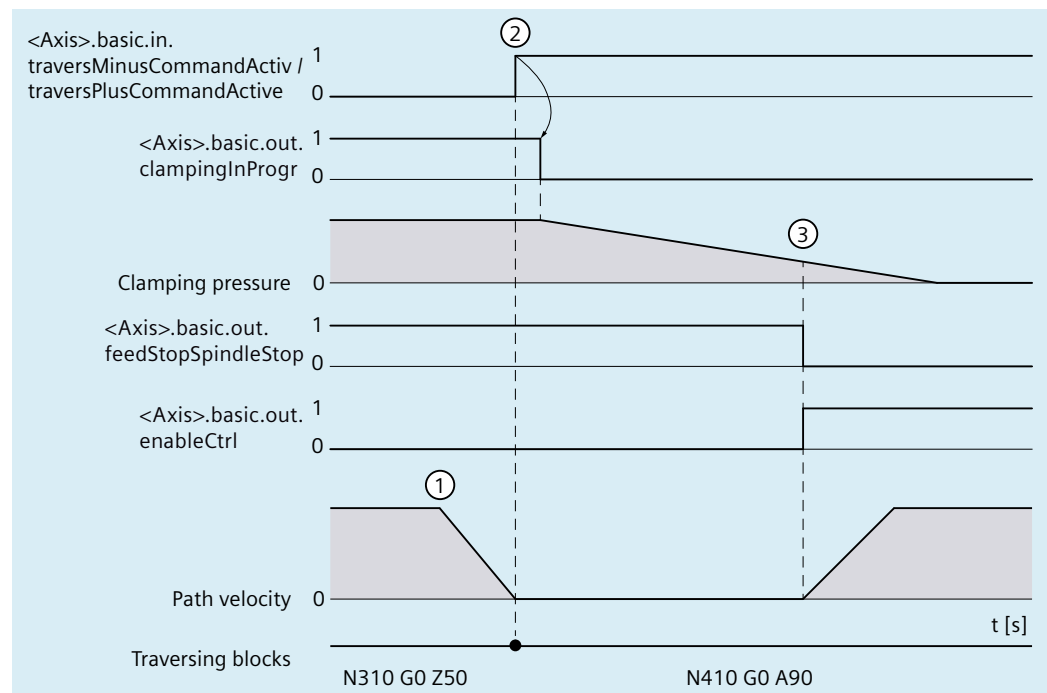
- If for the clamping axis there is a pending **traversing command** (`<Axis>.basic.in.traversMinusCommandActive` / `.traversPlusCommandActive`), then the clamping is **released** by the PLC user program.
- The following relationship must exist between the controller enable signal (`<Axis>.basic.out.enableCtrl`) and the clamping of the clamping axis:

Controller enable	⇒	Clamping axis
not set	⇒	Clamped
Set	⇒	Not clamped

Example

Program code	Comment
N100 G0 X0 Y0 Z0 A0 G90 G54 F500	; Approach initial state
N101 G641 ADIS=.1 ADISPOS=5	; Activate continuous-path mode
N210 G1 X10	; Traversing block
N220 G1 X5 Y20	; "
N310 G0 Z50	; Positioning block
N410 G0 A90	; " (clamping axis)
N510 G0 X100	; "
N520 G0 Z2	; "
N610 G1 Z-4	; Traversing block
N620 G1 X0 Y-20	; "

Schematic change of the NC/PLC interface signals and states for the N310 and N410 blocks:



- ① NC: The automatically inserted stop causes a stop at the N310 block end.
- ② NC → PLC: After the block change, the travel command for the clamping axis is set.
PLC: The clamping is released based on the travel command.
- ③ PLC → NC: The clamping pressure is removed appropriately. The clamping axis is enabled for traversal.

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.out.enableCtrl	LBP_Axis*.A_ContrEnable	DB31,DBX2.1
<Axis>.basic.in.traversMinusCommandActiv	LBP_Axis*.E_TCMinus	DB31,DBX64.6

Basic Program Plus	Basic Program	
<Axis>.basic.in.traversPlusCommandActive	LBP_Axis*.E_TCPlus	DB31,DBX64.7
<Axis>.basic.out.feedStopSpindleStop	LBP_Axis*.A_FDSpStop	DB31,DBX4.3

6.2.5.6 "Time-optimized release of the clamping" clamping function

The "Time-optimized release of the clamping" clamping function in conjunction with the "Automatic stop to release the clamping" clamping function for continuous-path mode requests the release of the clamping NC-internal by the Look Ahead setting of the travel command for the clamping axis. The travel command is set only when until the traversal of the clamping axis, positioning (G0 blocks) but no processing (G1 blocks) is performed.

To obtain the reference to the traversing block of the clamping axis, the travel command prefixes a maximum of two rapid traverse blocks (G0), including any internally generated intermediate blocks, to the traversing block.

Activation

MD36052 \$MA_STOP_ON_CLAMPING[<clamping axis>] = 'H03'

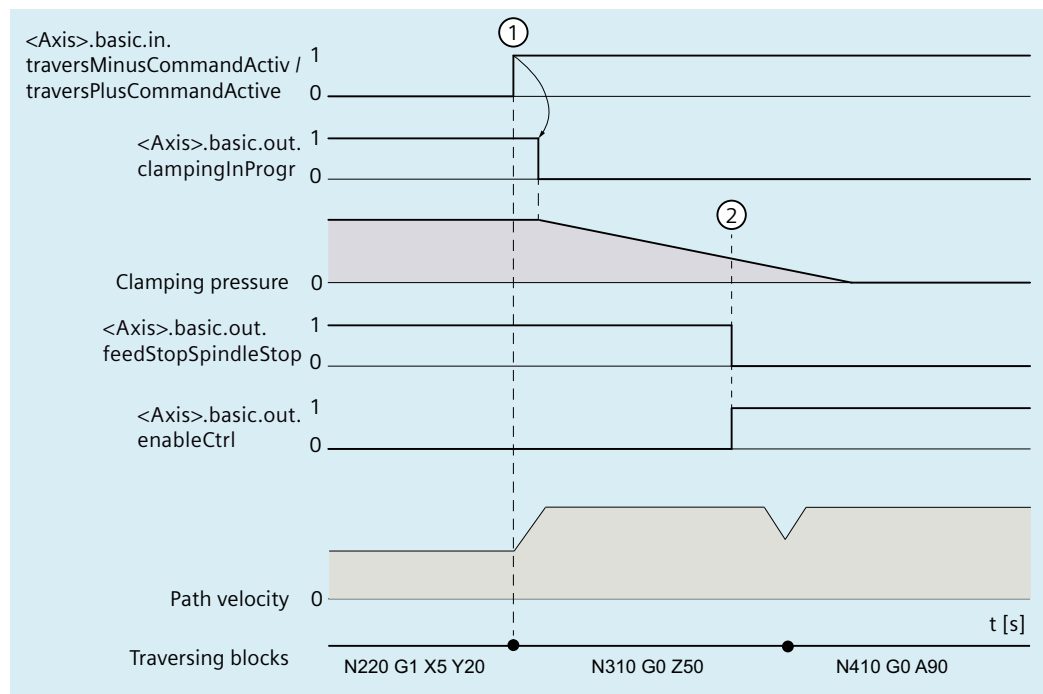
Requirements/assumptions

- If for the clamping axis there is a pending **traversing command** (<Axis>.basic.in.traversMinusCommandActive / .traversPlusCommandActive), then the clamping is **released** by the PLC user program.
- While other axes are traversing with rapid traverse (G0), the clamping axis must **not** be clamped.

Example

Program code	Comment
N100 G0 X0 Y0 Z0 A0 G90 G54 F500	; Approach initial state
N101 G641 ADIS=.1 ADISPOS=5	; Activate continuous-path mode
N210 G1 X10	; Machining block
N220 G1 X5 Y20	; "
N310 G0 Z50	; Positioning block
N410 G0 A90	; " (clamping axis)
N510 G0 X100	; "
N520 G0 Z2	; "
N610 G1 Z-4	; Machining block
N620 G1 X0 Y-20	; "

Schematic change of the NC/PLC interface signals and states for the N220 to N410 blocks:



- ① NC → PLC: The travel command for the clamping axis is set because of the block change. PLC: The clamping is released based on the travel command.
- ② PLC → NC: The clamping pressure is removed appropriately. The clamping axis is enabled for traversal.

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.out.enableCtrl	LBP_Axis*.A_ContrEnable	DB31,DBX2.1
<Axis>.basic.in.traversMinusCommandActiv	LBP_Axis*.E_TCMinus	DB31,DBX64.6
<Axis>.basic.in.traversPlusCommandActive	LBP_Axis*.E_TCPlus	DB31,DBX64.7
<Axis>.basic.out.feedStopSpindleStop	LBP_Axis*.A_FDSpStop	DB31,DBX4.3
<Chan>.basic.out.pathFeedrateOvrFactor	LBP_Chan*.A_FD_OR	DB21,DBB4

6.2.5.7 "Automatic stop to set the clamping" clamping function

The clamping process takes a while. In continuous-path mode, an explicit stop of the traversing must be provided by programming, e.g. G09, G60 or auxiliary function output, so that the clamping is reliably active before machining started.

The "Automatic stop to set the clamping" clamping function stops the traversing automatically in continuous-path mode. Clamping motion is stopped **before** or **in the** next machining block (traversing block without rapid traverse G0), if the clamping axis has not clamped up until then. The criterion that clamping took place and additional traversing motions are enabled is the setting of the channel-specific feedrate override by the PLC user program not equal to 0% (<Chan>.basic.out.pathFeedrateOvrFactor)

Activation

MD36052 \$MA_STOP_ON_CLAMPING[<clamping axis>] = 'H04'

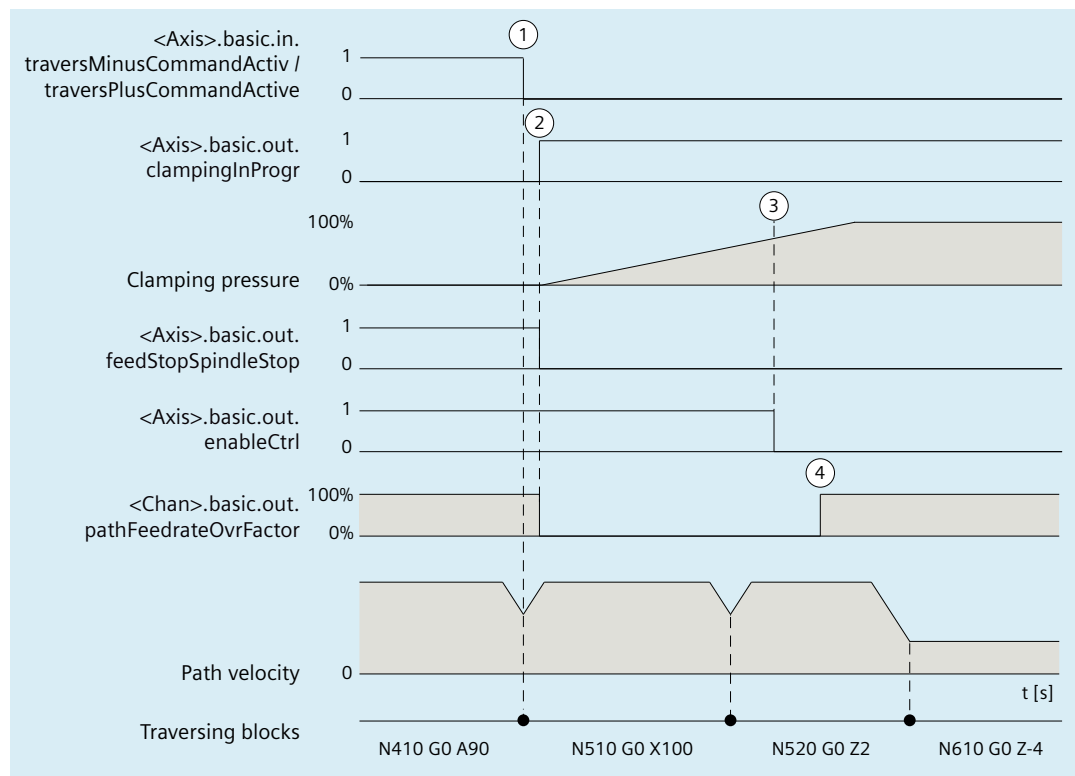
Requirements/assumptions

- If, for the clamping axis there is **no** pending traversing command (`<Axis>.basic.in.traversMinusCommandActive / .traversPlusCommandActive`), then the clamping is **closed** by the PLC user program.
- While other axes are traversing with rapid traverse (G0), the clamping axis must **not** be clamped.
- If the channel-specific feedrate override is **not equal to 0%** (`<Chan>.basic.out.pathFeedrateOvrFactor`), the clamping axis is **clamped**.

Example

Program code	Comment
N100 G0 X0 Y0 Z0 A0 G90 G54 F500	; Approach initial state
N101 G641 ADIS=.1 ADISPOS=5	; Activate continuous-path mode
N210 G1 X10	; Machining block
N220 G1 X5 Y20	; "
N310 G0 Z50	; Positioning block
N410 G0 A90	; " (clamping axis)
N510 G0 X100	; "
N520 G0 Z2	; "
N610 G1 Z-4	; Machining block
N620 G1 X0 Y-20	; "

Schematic change of the NC/PLC interface signals and states for the N410 to N610 blocks:



- ① NC → PLC: The travel command for the clamping axis is reset because of the block change.
- ② PLC: The clamping is initiated
- ③ PLC → NC: The clamping pressure is sufficiently large to reset the controller enable
- ④ PLC → NC: Enable the N610 machining by setting the channel-specific feedrate override not 0%.

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.out.enableCtrl	LBP_Axis*.A_ContrEnable	DB31,DBX2.1
<Axis>.basic.in.traversMinusCommandActiv	LBP_Axis*.E_TCMinus	DB31,DBX64.6
<Axis>.basic.in.traversPlusCommandActive	LBP_Axis*.E_TCPlus	DB31,DBX64.7
<Axis>.basic.out.feedStopSpindleStop	LBP_Axis*.A_FDSpStop	DB31,DBX4.3
<Chan>.basic.out.pathFeedrateOvrFactor	LBP_Chan*.A_FD_OR	DB21,DBB4

6.2.5.8 Supplementary conditions

Interrupted continuous-path mode

If during the above-described clamping functions, the continuous-path mode and thus also the "LookAhead" function is interrupted by blocks without traversing (e.g. output of an M function M82/M83), the functions behave as follows:

- Clamping function: "Time-optimized release of the axis clamping"**
 (MD36052 \$MA_STOP_ON_CLAMPING[<axis>] = 'B011')
 The function no longer acts because the travel command is set in Look Ahead mode only for blocks with active continuous-path mode. The output of the M function M82 in block N320 of the sample program below stops the traversing and so interrupts the continuous-path mode.
 The Look Ahead stopping on N410 by the function is not necessary because stopping occurs anyway by N320.
- Clamping function: "Automatic stop to set the clamping":**
 (MD36052 \$MA_STOP_ON_CLAMPING[<axis>] = 'B100')
 The function generates a stop irrespective of M83 that is executed as a function of "feedrate override 0%". The axis is thus stopped before the first machining block.

Note

Using clamping functions without clamping

The following clamping functions can also be used independent of clamping the axis:

- "Automatic stop to release the clamping":**
 MD36052 \$MA_STOP_ON_CLAMPING[<axis>] = 'B001'
Behavior: A stop is made in the current block on the path when the controller enable (<Axis>.basic.out.enableCtrl) for the parameterized <axis> is **not** set, but it is traversed in one of the following blocks.
- "Automatic stop to set the clamping":**
 MD36052 \$MA_STOP_ON_CLAMPING[<axis>] = 'B100'
Behavior: A stop is made in the current block on the path when at the transition from rapid traverse blocks (G0) to traversing blocks (G1), the channel-specific feedrate override (<Chan>.basic.out.pathFeedrateOvrFactor) == 0%.

In both cases it is ensured that the path motion in continuous-path mode is already stopped before the start of the relevant part program block and not just within the block.

Table 6-3 Sample program: Interrupted continuous-path mode

Program code	Comment
N100 G0 X0 Y0 Z0 A0 G90 G54 F500	; Approach initial state
N101 G641 ADIS=.1 ADISPOS=5	; Activate continuous-path mode
N210 G1 X10	; Traversing block
N220 G1 X5 Y20	; "
N310 G0 Z50	; Rapid traverse block
N320 M82	; Interrupt continuous-path mode
N410 G0 A90	; Rapid traverse block
N420 M83	; Interrupt continuous-path mode
N510 G0 X100	; Rapid traverse block
N520 G0 Z2	; "

Program code	Comment
N610 G1 Z-4	; Traversing block
N620 G1 X0 Y-20	; "

Block change criterion: Clamping tolerance

After activating clamping monitoring (`<Axis>.basic.out.clampingInProgress`), the clamping tolerance rather than the exact stop tolerance is used as the block change criterion for the clamping axis in traversing blocks that stop at the end of the block:

MD36050 \$MA_CLAMP_POS_TOL (clamping tolerance with interface signal "Clamping active")

Behavior for releasing the clamping

If the clamping axis was moved by the clamping process from the position setpoint, it is returned by the NC to the position setpoint after releasing the clamping and setting the (`<Axis>.basic.out.enableCtrl`) controller enable signal. Repositioning depends on whether "Follow-up mode" was activated for the axis during the clamping process:

- `<Axis>.basic.out.followUpMode == 0` (follow-up mode not active) ⇒ stepped by the position controller
- `<Axis>.basic.out.followUpMode == 1` (follow-up mode active) ⇒ interpolatory method

Note

The following data can be evaluated by the PLC user program as the criterion for activating the follow-up mode (`<Axis>.basic.out.followUpMode`):

- `<Axis>.basic.in.posCoarseReached / .posFineReached` (position reached with exact stop coarse/fine)
- Actual position of the clamping axis

Follow-up mode

Clamping monitoring is not active in follow-up mode (`<Axis>.basic.out.followUpMode`).

PLC signals

Basic Program Plus	Basic Program	
<code><Axis>.basic.out.enableCtrl</code>	<code>LBP_Axis*.A_ContrEnable</code>	DB31,DBX2.1
<code><Axis>.basic.in.posCoarseReached</code>	<code>LBP_Axis*.E_ExactCoarse</code>	DB31,DBX60.6
<code><Axis>.basic.in.posFineReached</code>	<code>LBP_Axis*.E_ExactFine</code>	DB31,DBX60.7
<code><Axis>.basic.out.followUpMode</code>	<code>LBP_Axis*.A_Follow_upMode</code>	DB31,DBX1.4
<code><Axis>.basic.out.clampingInProgress</code>	<code>LBP_Axis*.A_Clamp</code>	DB31,DBX2.3
<code><Chan>.basic.out.pathFeedrateOvrFactor</code>	<code>LBP_Chan*.A_FD_OR</code>	DB21,DBB4

6.3 Speed-setpoint monitoring

Function

The speed setpoint comprises:

- Speed setpoint of the position controller
- Speed setpoint portion of the feedforward control (with active feedforward control only)
- Drift compensation (only for drives with analog setpoint interface)

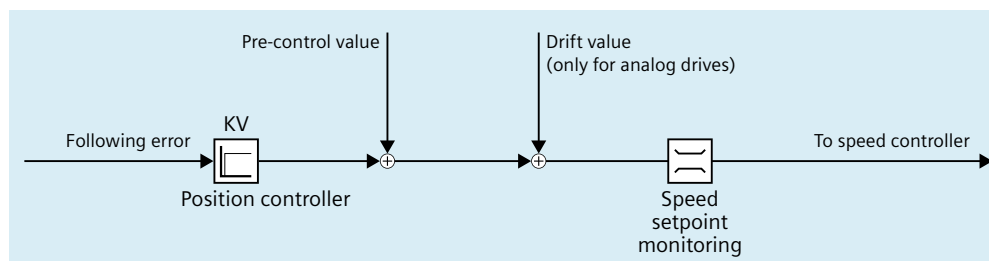


Figure 6-2 Speed setpoint calculation

The speed-setpoint monitoring ensures by limiting the control or output signal (10 V for analog setpoint interface or rated speed for digital drives) that the physical limitations of the drives are not exceeded:

MD36210 \$MA_CTRLOUT_LIMIT (maximum speed setpoint)

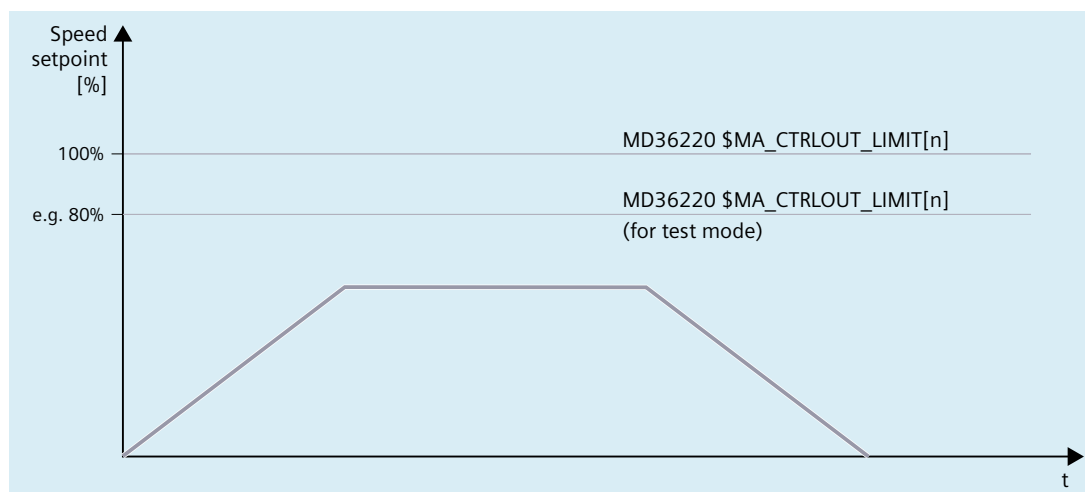


Figure 6-3 Speed setpoint limitation

Speed-setpoint monitoring delay

To prevent an error reaction from occurring in every speed-limitation instance, a delay time can be configured:

MD36220 \$MA_CTRLOUT_LIMIT_TIME (speed-setpoint monitoring delay)

Only if the speed limitation is required for longer than the configured time does the corresponding error reaction occur.

Effectiveness

The speed-setpoint monitoring is only active for closed-loop position-controlled axes and cannot be deactivated.

Fault

If the configured delay time is exceeded, the following alarm appears:

25060 "Axis <Axis name> Speed-setpoint limitation"

The affected axis is stopped via the configured braking ramp in follow-up mode:

MD36610 \$MA_AX_EMERGENCY_STOP_TIME
(maximum time for braking ramp when an error occurs)

Note

Upon reaching the speed-setpoint monitoring, the position feedback loop of the axis becomes non-linear due to the limitation. Contour errors result if the axis is involved in generating the contour.

6.4 Actual-velocity monitoring

Function

The actual-velocity monitoring checks that the actual velocity of a machine axis/spindle does not exceed the configured threshold:

MD36200 \$MA_AX_VELO_LIMIT (velocity-monitoring threshold)

The threshold should be 10-15% above the configured maximum velocity.

- For axes:
MD32000 \$MA_MAX_AX_VELO (maximum axis velocity)
- For spindles:
MD35110 \$MA_GEAR_STEP_MAX_VELO_LIMIT[n] (maximum speed of gear stage)

If you use this setting the speed will not normally exceed the velocity-monitoring threshold (exception: Drive error).

Activation

The actual-velocity monitoring is activated as soon as the active measuring system returns valid actual values (encoder limit frequency not exceeded).

Effectiveness

The actual-velocity monitoring only operates with active position control and the following axis types:

- Linear axes
- Rotary axes
- Open-loop-controlled and position-controlled spindles

Fault

If the threshold is exceeded, the following alarm is displayed:

25030 "Axis <Axis name> Actual-velocity alarm limit"

The affected axis is stopped via the configured braking ramp in follow-up mode:

MD36610 \$MA_AX_EMERGENCY_STOP_TIME

(maximum time for braking ramp when an error occurs)

6.5 Measuring system monitoring

The NC has no direct access to the measuring system hardware, therefore measuring system monitoring is mainly performed by the drive software.

Monitoring functions in the drive

- Monitoring of hardware faults (e.g. measuring system failure, wire breakage)
- Zero mark monitoring

Further information:

Function Manual Drive Functions SINAMICS S120

Measuring system monitoring functions carried out in the drive are mapped on the NC alarms (alarm 25000 and following) or NC reactions (e.g. abort of referencing or on-the-fly measuring). The exact behavior of the NC depends on the setting in the machine data:

MD36310 \$MA_ENC_ZERO_MONITORING

Value	Meaning	
0	Monitoring of HW faults:	ON If a hardware fault is detected in the active measuring system, POWER ON alarm 25000 is displayed: "Axis <Axis name> Hardware fault active encoder" The affected axis is stopped via the configured braking ramp in follow-up mode: MD36610 \$MA_AX_EMERGENCY_STOP_TIME (maximum time for braking ramp when a fault occurs) If a hardware fault is detected in the passive measuring system, alarm 25001 is displayed: "Axis <Axis name> Hardware fault passive encoder" There is no further alarm response.
	Zero-mark monitoring:	OFF Alarms 25020 and 25021 (see below) are suppressed.
100	No zero-mark monitoring as well as hiding of all encoder monitoring functions, i.e. in addition to alarm 25020 (25021), alarms 25000 (25001) and 25010 (25011) are suppressed.	
0 < x < 100	Monitoring of HW faults:	ON (see above)
	Zero-mark monitoring:	ON If zero-mark monitoring is tripped in the active measuring system, alarm 25020 is displayed: "Axis <Axis name> Zero-mark monitoring active encoder" The affected axis is stopped via the configured braking ramp in follow-up mode: MD36610 \$MA_AX_EMERGENCY_STOP_TIME (maximum time for braking ramp when a fault occurs) If zero-mark monitoring is tripped in the passive measuring system, alarm 25021 is displayed: "Axis <Axis name> Zero-mark monitoring passive encoder" There is no further alarm response.
> 100	Monitoring of HW faults:	ON with attenuated error message: The POWER ON alarm 25000 is replaced by the reset alarm 25010 and the reset alarm 25001 replaced by the cancel alarm 25011.
	Zero-mark monitoring:	ON (see above)

For details on the alarms, see:

More information

Diagnostics Manual

Note

For hardware faults, the referencing status of the machine axis is reset:

`<Axis>.basic.in.enc1Synchronized / .enc2Synchronized (referenced/
synchronized 1/2) = 0`

Monitoring functions in the NC

- Encoder-limit-frequency monitoring
- Plausibility check for absolute encoders

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.in.enc1Synchronized	LBP_Axis*.E_RefSyn1	DB31,DBX60.4
<Axis>.basic.in.enc2Synchronized	LBP_Axis*.E_RefSyn2	DB31,DBX60.5

6.5.1 Encoder-limit-frequency monitoring**Function**

The NC encoder-limit-frequency monitoring is based on the configuration and telegram information of the drive. It monitors that the encoder frequency does not exceed the configured encoder limit frequency:

MD36300 \$MA_ENC_FREQ_LIMIT (encoder limit frequency)

Encoder-limit-frequency monitoring always refers to the active measuring system selected in the NC/PLC interface:

`<Axis>.basic.out.enclActivation / .enclActivation` (position measuring system 1/2)

Effectiveness

The encoder limit frequency is operative for:

- Linear axes
- Rotary axes
- Open-loop-controlled and position-controlled spindles

Fault

Upon exceeding of the encoder limit frequency, the following occurs:

- Message to the PLC:
`<Axis>.basic.in.enc1FreqLimitExceeded / .enc1FreqLimitExceeded = 1`
(encoder limit frequency exceeded 1 or 2)
- Spindles
Spindles are not stopped but continue to turn with speed control.
If the spindle speed is reduced so much that the encoder frequency passes below the encoder limit frequency, the actual value system of the spindle is automatically resynchronized.
- Axes
The following alarm is displayed:
21610 "Channel <Channel number> Axis <Axis name> Encoder <Encoder number >
Frequency exceeded"
The affected axis is stopped via the configured braking ramp in follow-up mode:
MD36610 \$MA_AX_EMERGENCY_STOP_TIME
(maximum time for braking ramp when an error occurs)

Note

If the encoder limit frequency is exceeded, a position-controlled machine axis must be re-referenced (see Function Manual "Axes and Spindles", Section "R1: Referencing").

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.out.enc1Activation	LBP_Axis*.A_PosMeas1	DB31,DBX1.5
<Axis>.basic.out.enc2Activation	LBP_Axis*.A_PosMeas2	DB31,DBX1.6
<Axis>.basic.in.enc1FreqLimitExceeded	LBP_Axis*.E_EncodFreq1	DB31,DBX60.2
<Axis>.basic.in.enc2FreqLimitExceeded	LBP_Axis*.E_EncodFreq2	DB31,DBX60.3

6.5.2 Plausibility check for absolute encoders

Function

With absolute encoders (MD30240 \$MA_ENC_TYPE = 4), absolute values supplied by the measuring system are used to check the plausibility of the actual value.

During the check, the NC compares the cyclic position value held in the position control cycle clock based on the incremental information from the encoder with a new position value generated directly from the absolute and incremental information and checks that the calculated position difference does not exceed the permissible deviation.

MD36310 \$MA_ENC_ZERO_MONITORING (permissible deviation in 1/2 coarse increments between the absolute and the incremental encoder track)

Note

The plausibility check of absolute encoders specifically detects all deviations caused by dirt on the absolute track or by faults when transferring the absolute value. However, small errors in the incremental track (burst interference, impulse errors) are not detected. In such instances the plausibility check only responds to deviations in the millimeter range. This form of monitoring should therefore serve as additional monitoring to assist the diagnosis of absolute-position faults.

Note

Rotary absolute encoders

If the plausibility check is to be used for a rotary absolute encoder, the SINAMICS parameter p0979 must be taken into account when setting the modulo range (MD34220 \$MA_ENC_ABS_TURNS_MODULO).

Note

Upgrading the NC software

If the plausibility check is activated in absolute encoders (MD36310 > 0), the existing MD36310 settings must be checked and, if necessary, increased during an upgrade of the NC software.

Zero mark diagnostics

With absolute encoders, the permissible deviation must be determined for the plausibility check during commissioning. This can be performed via the machine data:

MD36312 \$MA_ENC_ABS_ZEROMON_WARNING (zero-mark monitoring warning threshold)

Value	Meaning
0	No zero mark diagnostics
> 0	Permissible deviation in 1/2 coarse increments between the absolute and the incremental encoder track

Procedure when commissioning the system:

1. Deactivate zero-mark monitoring:
MD36310 \$MA_ENC_ZERO_MONITORING = 0
2. Activate zero-mark diagnostics:
MD36312 \$MA_ENC_ABS_ZEROMON_WARNING = 1
3. Move axis and monitor system variable \$VA_ENC_ZERO_MON_ERR_CNT (number of detected limit value violations).

4. If \$VA_ENC_ZERO_MON_ERR_CNT $\neq$ 0:
Increase MD36312 value and repeat step 3.
5. If \$VA_ENC_ZERO_MON_ERR_CNT = 0 (over a longer period of time!):
The correct value for MD36310 is located! Apply the value from MD36312 to MD36310 and then set MD36312 to "0".

Note

Depending on the rigidity of the machine (minimal load masses / moments of inertia are optimum) and the controller settings, the control play "oscillates" with varying degrees of intensity. Account must be taken of this by entering machine-specific limit values in MD36310.

Fault

Alarm 25020

If the plausibility check is tripped in the **active** measuring system, alarm 25020 is displayed:

"Axis <Axis name> Zero-mark monitoring active encoder"

The affected axis is stopped via the configured braking ramp in follow-up mode:

MD36610 \$MA_AX_EMERGENCY_STOP_TIME
(maximum time for braking ramp when an error occurs)

Alarm 25021

If the plausibility check is tripped in the **passive** measuring system, alarm 25021 is displayed:

"Axis <Axis name> Zero-mark monitoring passive encoder"

There is no further alarm response.

Note

In the event of a fault, the adjustment of the absolute encoder is lost and the axis is no longer referenced. The absolute encoder must be readjusted (see Function Manual "Axes and Spindles, Section "Referencing for absolute encoders").

Note

Errors in the incremental track that cannot be detected with amplitude monitoring can cause position deviations in the millimeter range. The deviation depends on the lattice pitch/line count and the traversing velocity of the axis when the error occurs.

Complete position monitoring is only possible through redundancy, i.e. through comparison with an independent second measuring system.

6.5.3 Customized error reactions

Customized zero-mark monitoring

The default alarm and reaction behavior of the zero-mark monitoring can be adapted in absolute measuring systems (MD30240 \$MA_ENC_TYPE = 4) with the aid of system variables. This allows you to perform your own monitoring using a synchronized action or OEM application and to use all of the reaction options available in this application, e.g.:

- Transmit alarm
- Use cycles (e.g. approach tool-change position)
- ...

Example:

Users can adjust the alarm and reaction behavior so that when machining an expensive workpiece, which could be damaged if the axis is stopped as a result of an alarm, machining stops before the machining quality of the workpiece is assessed using appropriate synchronized action commands.

Effectiveness

Customized monitoring can be activated in parallel to or as an alternative to standard zero-mark monitoring, depending on the setting in machine data:

MD36310 \$MA_ENC_ZERO_MONITORING

Value	Meaning
0	If only user-specific monitoring is to be implemented, the default zero-mark monitoring must be deactivated: MD36310 = 0 and MD36312 = 0
> 0	Customized monitoring and standard zero-mark monitoring operate in parallel.
100	All encoder monitoring functions are deactivated.

If both monitoring functions are active (MD36310 > 0), you can perform **cascaded monitoring**.

Example:

If a value falls below the threshold specified in MD36310, customized monitoring triggers a prewarning; standard zero-marking monitoring will only detect a fault if the threshold is exceeded and will then deactivate automatically.

System variables

You can implement customized error reactions using the following system variables:

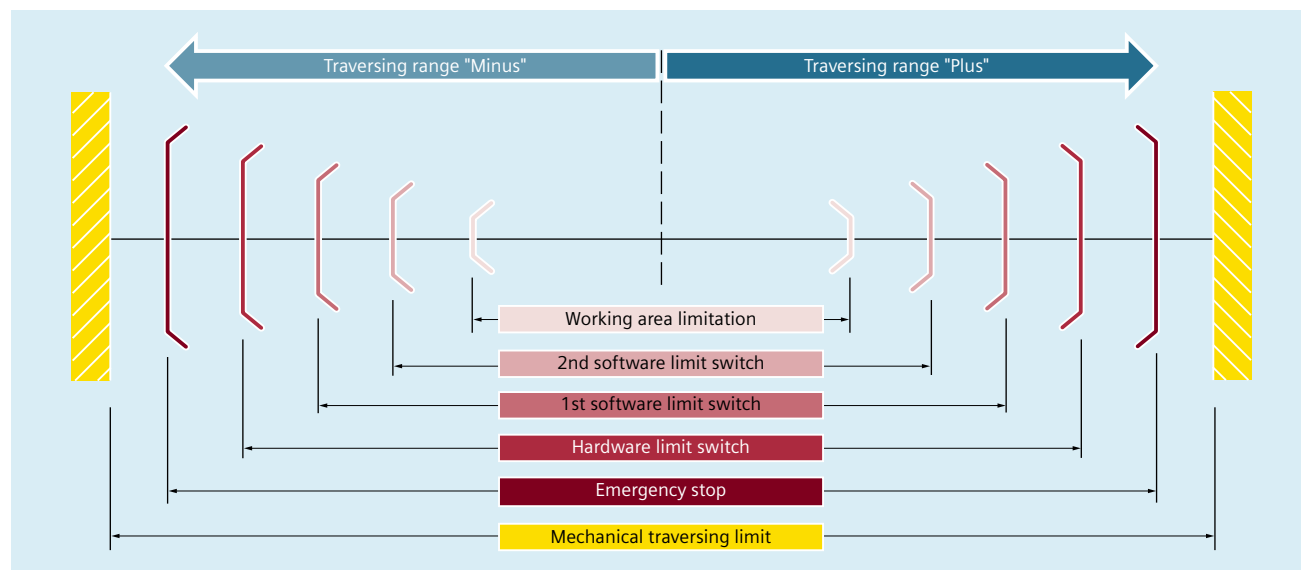
System variable	Meaning
\$VA_ENC_ZERO_MON_ERR_CNT[<n>,<axis>]	Number of detected limit value violations. Contains the current number of detected limit value violations when comparing the absolute and the incremental encoder tracks. The value is reset to 0 at: • POWER ON • Selection/deselection of parking Reset does not cause a reset.
\$VA_ABSOLUTE_ENC_DELTA_INIT[<n>,<axis>]	Initial difference for absolute encoders. Contains the initial difference between the last buffered absolute position in the static NC memory and the current absolute position. Format of the difference value: Number of internal increments (see MD10200 \$MN_INT_INCR_PER_MM or MD10210 \$MN_INT_INCR_PER_DEG) The value is updated at: • POWER ON • Warm restart • Deselection of parking • Return below the encoder limit frequency There is no reset at reset.

<n>: Encoder number

<axis>: Axis name

6.6 Limit switch monitoring

Overview of the end stops and possible limit-switch monitoring:



6.6.1 Hardware limit switch

Function

A hardware limit switch is normally installed at the end of the traversing range of a machine axis. It serves to protect against accidental overtravelling of the maximum traversing range of the machine axis while the machine axis is not yet referenced.

If the hardware limit switch is triggered, the PLC user program created by the machine manufacturer sets the corresponding interface signal:

```
<Axis>.basic.out.hwLimitSwitchMinus / .hwLimitSwitchPlus = 1 (hardware limit switch minus/plus)
```

Parameterization

The braking behavior of the machine axis upon reaching the hardware limit switch is configurable via the machine data:

MD36600 \$MA_BRAKE_MODE_CHOICE (braking behavior on hardware limit switch)

Value	Meaning
0	Braking with the configured axial acceleration
1	Rapid stop (set velocity = 0)

Effectiveness

The hardware limit-switch monitoring is active after the controller has ramped up in all modes.

Reactions of hardware limit switch monitoring

Upon reaching the hardware limit switch, the following occurs:

- Alarm 21614 "Channel <channel number> axis <axis name> hardware limit switch <direction>"
- The machine axis is braked according to the configured braking behavior.
- If the axis/spindle is involved in interpolation with other axes/spindles, these are also braked according to their configured braking behavior.
- The traversing keys of the affected machine axis are blocked based on the direction.

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.out.hwLimitSwitchMinus	LBP_Axis*.A_HWLLimitMinus	DB31,DBX12.0
<Axis>.basic.out.hwLimitSwitchPlus	LBP_Axis*.A_HWLLimitPlus	DB31,DBX12.1

6.6.2 Software limit switch

Function

Software limit switches serve to limit the traversing range of an axis. The positions of the software limit switches must be inside the range of the hardware limit switches.

Two (1st and 2nd) axis-specific software limit switches referenced to the machine coordinate system (MCS) can be set per machine axis for each traversing direction:

MD36100 \$MA_POS_LIMIT_MINUS (1st software limit switch minus)

MD36110 \$MA_POS_LIMIT_PLUS (1st software limit switch plus)

MD36120 \$MA_POS_LIMIT_MINUS2 (2nd software limit switch minus)

MD36130 \$MA_POS_LIMIT_PLUS2 (2nd software limit switch plus)

By default, the 1st software limit switch is active. The 2nd software limit switch can be activated for a specific direction with the PLC user program:

<Axis>.basic.out.swLimitSwitch2Minus / swLimitSwitch2Plus (2nd software limit switch minus)

In addition to the axis-specific software limit switches, channel-specific software limit switches related to the basic coordinate system (BCS) can also be set for the geometry axes:

MD21600 \$MC_POS_LIMIT_GEO_PLUS[<n>] (geometry axis limit plus)

MD21601 \$MC_POS_LIMIT_GEO_MINUS[<n>] (geometry axis limit minus)

MD21602 \$MC_POS_LIMIT_GEO_ACTIVATION[<n>] (activating the geometry axis limit)

With <n> = 0, 1, 2 ($\hat{=}$ geometry axis 1, 2, 3)

Their difference from axis-specific software limit switches is particularly evident in the case of active transformations, since they enable limiting in a Cartesian traverse space even when the machine axes are not perpendicular to each other, e.g. in grinding machines with persistent oblique angle transformation (TRAANG).

If axis-specific **and** channel-specific software limit switches are active, the software limit switch that limits the traversing range more strongly takes effect.

Traversing the axis to the position of the active software limit switch is permitted.

Effectiveness

Software limit switches are effective:

- Immediately after the successful referencing of the machine axis (axes)
- In all operating modes

Reactions of software limit switch monitoring

Automatic operating modes (AUTOMATIC, MDI)

- Without transformation, without overlaid motion, unchanged software limit switch:
A part program block with a programmed traversing motion that would lead to overrunning of the software limit switch is not started.
- With transformation:
Different reactions occur depending on the transformation type:
 - Behavior as above.
or
 - The part program block with a programmed traversing motion that would lead to overrunning of the software limit switch is started. The affected machine axis stops at the active software limit switch. The other machine axes participating in the traversing motion are braked. The programmed contour is left during this process.
- With overlaid motion
The part program block with a programmed traversing motion that would lead to overrunning of the software limit switch is started. Machine axes that are traveling with overlaid motion or have traveled with overlaid motion stop at the active software limit switch in question. The other machine axes participating in the traversing motion are braked. The programmed contour is left during this process.

Manual operating modes

- JOG without transformation
The machine axis stops at the software limit switch position.
- JOG with transformation
Depending on which software limit switch comes into effect, different reactions result:
 - Axis-specific software limit switch effective:
The machine axis stops at the software limit switch position. Other machine axes participating in the traversing motion are braked. The preset path is left during this process.
 - Geometry axis limitation effective:
The axis stops at the software limit switch position. Other axes participating in the traversing motion are braked. The preset path is **not** left during this process.

Response in the event of an error

Changing of the software limit switch (1st ↔ 2nd software limit switch)

If the actual position of the machine axis after changing lies behind the software limit switch, it is stopped with the maximum permissible acceleration.

Overrunning the software limit switch in JOG mode

If the position of the software limit switch is reached and renewed pressing of the traversing button should cause further travel in this direction, an alarm is displayed and the axis is not traversed farther:

Alarm 10621 "Channel <channel number> axis <axis name> is at software limit switch <direction>"

or

Alarm 10626 "Channel <channel number> Axis <axis name> reached Limitation of geometry axis <direction>"

Constraints

PRESET

After use of the function PRESET, the software limit-switch monitoring is no longer active. The machine must first be re-referenced.

Endlessly rotating rotary axes

No software limit-switch monitoring takes place for endlessly rotating rotary axes (MD30310 \$MA_ROT_IS_MODULO == 1).

Exception: Setup-rotary axes

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.out.swLimitSwitch2Minus	LBP_Axis*.A_SWLimit2Minus	DB31,DBX12.2
<Axis>.basic.out.swLimitSwitch2Plus	LBP_Axis*.A_SWLimit2Plus	DB31,DBX12.3

6.7 Working area limitation monitoring**6.7.1 General****Function**

The "working area limitation" function can be used to limit the traversing range of a channel's geometry and special axes to a permissible operating range. The function monitors compliance with working area limits both in AUTOMATIC mode and in JOG mode.

The following versions are available:

- Working area limitation in the Basic Coordinate System (BCS)
The traversing range limits are specified relative to the Basic Coordinate System.
- Working area limitation in the workpiece coordinate system (WCS) or adjustable zero system (AZS)
The traversing range limits are specified relative to the workpiece coordinate system or to the adjustable zero system.

The two types of monitoring are independent of each other. If they are both active at the same time, the traversing range limit which most restricts the access will take effect, depending on the direction of travel.

Reference point at the tool

Taking into account the tool data (tool length and tool radius) and therefore the reference point at the tool when monitoring the working area limitation depends on the status of the transformation in the channel:

- **Transformation inactive**

Without transformations during traversing motion with an active tool the position of the tool tip P is monitored, i.e. during the monitoring the tool length is considered automatically. Consideration of the tool radius must be activated separately:

MD21020 \$MC_WORKAREA_WITH_TOOL_RADIUS (Consideration of the tool radius in the working area limitation)

- **Transformation active**

In the case of certain transformations the monitoring of the working area limitation may differ from the behavior without transformation:

- The tool length is a component of the transformation (\$MC_TRAFO_INCLUDES_TOOL_X = TRUE):
In this case the tool length is not considered, i.e. the monitoring refers to the tool carrier reference point.
- Transformation with change in orientation:
In the case of transformations with changes in orientation, monitoring is always based on the tool center point. MD21020 has no influence.

Note

The machine data \$MC_TRAFO_INCLUDES_TOOL_... is analyzed only in certain transformations. Condition for a possible evaluation is that the orientation of the tool with respect to the base coordinate system cannot be changed by the transformation. With standard transformations, the condition is only fulfilled for the "inclined axis" type of transformation.

Response

Automatic operating modes

- With / without transformation
The parts program block with a programmed traversing motion that would lead to overrunning of the working area limits is not executed.
- With superimposed motion
The axis, which would violate the working area limitation due to a superimposed motion, is braked with maximum acceleration and without jerk limits (BRISK), and will come to a stop in the position of the working area limitation. Other axes involved in the movement are braked according to current acceleration behavior (e.g. SOFT). The path correlation may be lost due to different braking accelerations (contour violation).

Manual operating modes

- JOG with / without transformation
The axis is positioned at the working area limitation and then stopped.

Powerup response

If an axis moves outside the permissible working area when activating the working area limits, it will be immediately stopped with the maximum permissible acceleration.

Overrunning of the working area limitation in JOG mode

In JOG mode, an axis is moved to no further than its working area limit by the control system. When the traverse button is pressed again, an alarm is displayed and the axis does not traverse any further.

Geo-axis replacement

Through the following machine data it is adjustable, whether during geometry axis change the active working area limitation is retained or deactivated:

MD10604 \$MN_WALIM_GEOAX_CHANGE_MODE = <value>

<value>	Meaning
0	The working area limitation is deactivated during the geometry axis change.
1	The working area limitation remains activated during the geometry axis change.

6.7.2 Working area limitation in BCS**Application**

Using the "working area limitation in BCS", the working area of a machine tool is limited so that the surrounding devices (e.g. tool revolver, measuring stations) are protected against damage.

Working area limits

The lower and upper working area limits of each axes are adjusted through setting data or programmed through part program commands:

Working area limitation through setting data

The adjustments are done through the immediately effective axis-specific setting data:

SD43420 \$SA_WORKAREA_LIMIT_PLUS (working area limitation plus)

SD43430 \$SA_WORKAREA_LIMIT_MINUS (working area limitation minus)

Programmed working area limitation

The programming is done using the G commands:

G25 X... Y... Z... lower working area limitation

G26 X... Y... Z... upper working area limitation

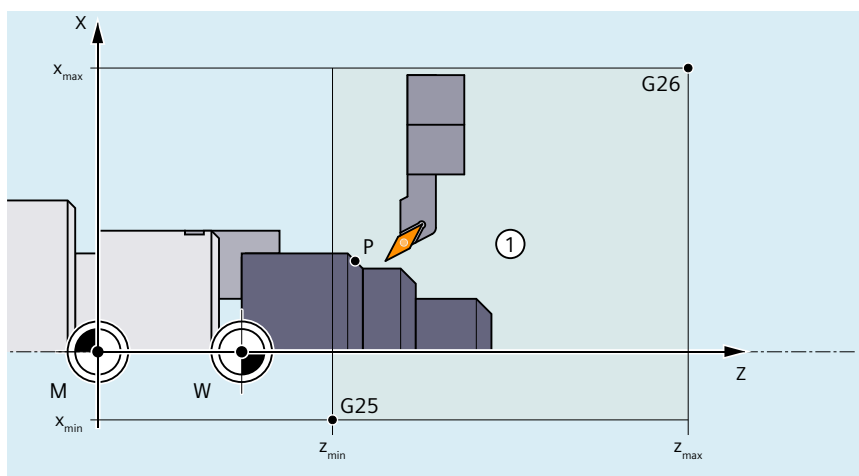


Figure 6-4 Programmed working area limitation

The programmed working area limitation has priority and overwrites the values entered in SD43420 and SD43430.

Activation/Deactivation

Working area limitation through setting data

The activation/deactivation of the working area limitation for each axis takes place in a direction-specific manner via the immediately effective setting data:

SD43400 \$SA_WORKAREA_PLUS_ENABLE (working area limitation active in the positive direction)

SD43410 \$SA_WORKAREA_MINUS_ENABLE (working area limitation active in the negative direction)

Value	Meaning
0	The working area limitation in positive or negative direction is switched off .
1	The working area limitation in positive or negative direction is active .

Programmed working area limitation

Activation or deactivation of the overall "working area limitation in the BCS" is arranged via part program commands:

WALIMON Working area limitation ON

or

WALIMOF Working area limitation OFF

Changing the working area limitation

Working area limitation through setting data

6.7 Working area limitation monitoring

HMI user interface: Operating area "Parameter"

- Automatic modes:
 - Changes: Possible only in the RESET state
 - Effectiveness: Immediately
- Manual operating modes:
 - Changes: Always possible
 - Effectiveness: At the start of the next traversing motion

Programmed working area limitation

The working area limitation can be changed in the part program via G25 or G26 <Axis name> <value>. The change takes effect immediately.

The new working area limitation value is retained after NC RESET and POWER ON if the back-up process has been activated in the NC's retentive data storage for SD43420 and SD43430:

MD10710 \$MN_PROG_SD_RESET_SAVE_TAB[0] = 43420

MD10710 \$MN_PROG_SD_RESET_SAVE_TAB[1] = 43430

Reset position

The reset position for the working area limitation (WALIMON or WALIMOF) is configurable via:

MD20150 \$MC_GCODE_RESET_VALUES (reset setting of the G groups)

6.7.3 Working area limitation in WCS/SZS

Application

The "working area limitation" in the WCS/SZS enables a flexible workpiece-specific limitation of the traversing range of the channel axes in the workpiece coordinate system (WCS) or settable zero system (SZS). It is intended mainly for use in conventional lathes.

Requirement

The channel axes must be referenced.

Working area limitation group

In order that the axis-specific working area limits do not have to be rewritten for all channel axes when switching axis assignments, e.g. when switching transformations or the active frame on/off, working area limitation groups are available.

A working area limitation group comprises the following data:

- Working area limits for all channel axes
- Reference system of the working area limitation

The number of the working area limitation groups is set channel-specific in the machine data:

MD28600 \$MC_MM_NUM_WORKAREA_CS_GROUPS

Maximum 10 working area limitation groups are possible per channel.

Set working area limits

The working area limits within a channel are set for each channel axis via the following system variables:

- \$P_WORKAREA_CS_LIMIT_PLUS[<WALimNo>, <Ax>]
- \$P_WORKAREA_CS_LIMIT_MINUS[<WALimNo>, <Ax>]

wit <WALimNo> = Working area limitation group
h:

Value range: 0 (group 1) ... 9 (group 10)
<Ax> = Channel axis name

Enable working area limits

The working area limits within a channel are enabled for each channel axis via the following system variables:

- \$P_WORKAREA_CS_PLUS_ENABLE[<WALimNo>, <Ax>]
- \$P_WORKAREA_CS_MINUS_ENABLE[<WALimNo>, <Ax>]

wit <WALimNo> = Working area limitation group
h:

Value range: 0 (group 1) ... 9 (group 10)
<Ax> = Channel axis name

Using the direction-specific enable, it is possible to limit the working range for an axis in just one direction.

There is no activation through the enable.

Select reference system

The reference system for a working area limitation group within a channel is set via the following system variable:

\$P_WORKAREA_CS_COORD_SYSTEM[<WALimNo>] = <value>

wit <WALimNo> = Working area limitation group

h:

Value range: 0 (group 1) ... 9 (group 10)

<value>	Meaning
1	The reference system is the WCS.
3	The reference system is the SZS.

Activation of working area limits

The working area limits of a working area limitation group are activated in the part program with the G command `WALCS<n>`.

wit <n> = Number of the working area limitation group

h:

Value range: 1 ... 10

Deactivation of working area limits

The working area limits of a working area limitation group active in the channel are deactivated in the part program with the G command `WALCS0`.

Change working area limits

The working area limits can be changed at any time via the system variables mentioned above. Changes take effect with the next activation of the working area limitation group (`WALCSn`).

Data storage

The system variables of the working area limits are stored retentively in the static memory of the NC.

Note

For the storage of the limiting values for the linear axes, the default setting is considered for the system of units (MD10240 `$MN_SCALING_SYSTEM_IS_METRIC`).

Data backup

The system variables of the working area limits can be backed up in separate files:

- `_N_CHx_WAL`
To save the system variable values for channel x.
- `_N_COMPLETE_WAL`
To save the system variable values for all channels.

Note

The system variables of the working area limits are part of the "_N_INITIAL_INI" file.

Behavior in JOG mode

Initial situation:

- In JOG mode, **several** geometry axes traverse simultaneously (e.g. using several handwheels).
- A **rotating** frame is active between the basic coordinate system (BCS) and the reference coordinate system of the working area limitation (WCS or SZS).

Behavior when a working area limitation responds:

- The traversing motions of the geometry axes that are not affected are continued.
- The affected geometry axis is stopped at the working area limit.

Setting the initial setting

The working area limitation group that is to take effect at ramp up, reset or part program end and part program start is predefined channel-specifically via the machine data:

MD20150 \$MC_GCODE_RESET_VALUE[59] = <n>

wit <n> = Number of the working area limitation group
h:

Value range: 1 ... 10

The following setting determines whether the pre-selected working area limitation acts for ramp up and reset or part program end:

MD20152 \$MC_GCODE_RESET_MODE[59] = <value>

<value>	Meaning
0	The working area group takes effect in accordance with MD20150 (default setting).
1	The last active working area group remains active.

6.7.4 Example: Working area limitation in WCS/SZS**Assumption****Channel axes**

Four axes are defined in the channel:

- Linear axes: X, Y, Z
- Rotary axis: A (not modulo)

Requirements

Channel axes

Four axes are defined in the channel:

- Linear axes: X, Y, Z
- Rotary axis: A (not modulo)

Working area limitation groups

Three working area limitation groups should be available in the channel:

MD28600 \$MC_MM_NUM_WORKAREA_CS_GROUP = 3

From these three working area limitation groups, two groups are defined in the following.

Coordinate systems

- Working area limitation group 1: Working area limitation in the adjustable zero system (**AZS**).
- Working area limitation group 2: Working area limitation in the workpiece coordinate system (**WCS**).

Working area limitation group 1

- X axis in the plus direction: 10 mm
- X axis in the minus direction: No limitation
- Y axis in the plus direction: No limitation
- Y axis in the minus direction: 25 mm
- Z axis in the plus direction: No limitation
- Z axis in the minus direction: No limitation
- A axis in the plus direction: 10 degrees
- A axis in the minus direction: -40 degrees

Definitions via system variables in the NC program.

Program code

```

; Working area limitation group 1
$P_WORKAREA_CS_COORD_SYSTEM[1] = 3          ; working area limitation in
the SZS
$P_WORKAREA_CS_PLUS_ENABLE[1,X] = TRUE
$P_WORKAREA_CS_LIMIT_PLUS[1,X] = 10
$P_WORKAREA_CS_MINUS_ENABLE[1,X] = FALSE
$P_WORKAREA_CS_PLUS_ENABLE[1,Y] = FALSE
$P_WORKAREA_CS_MINUS_ENABLE[1,Y] = TRUE
$P_WORKAREA_CS_LIMIT_MINUS[1,Y] = 25
$P_WORKAREA_CS_PLUS_ENABLE[1,Z] = FALSE
$P_WORKAREA_CS_MINUS_ENABLE[1,Z] = FALSE
$P_WORKAREA_CS_PLUS_ENABLE[1,A] = TRUE

```

Program code

```
$P_WORKAREA_CS_LIMIT_PLUS[1,A] = 10
$P_WORKAREA_CS_MINUS_ENABLE[1,A] = TRUE
$P_WORKAREA_CS_LIMIT_MINUS[1,A] = -40
```

Working area limitation group 2

- X axis in the plus direction: 10 mm
- X axis in the minus direction: No limitation
- Y axis in the plus direction: 34 mm
- Y axis in the minus direction: -25 mm
- Z axis in the plus direction: No limitation
- Z axis in the minus direction: -600 mm
- A axis in the plus direction: No limitation
- A axis in the minus direction: No limitation

Definitions via system variables in the NC program.**Program code**

```
; Working area limitation group 2
$P_WORKAREA_CS_COORD_SYSTEM[2] = 1          ; working area limitation in
the WCS
$P_WORKAREA_CS_PLUS_ENABLE[2,X] = TRUE
$P_WORKAREA_CS_LIMIT_PLUS[2,X] = 10
$P_WORKAREA_CS_MINUS_ENABLE[2,X] = FALSE
$P_WORKAREA_CS_PLUS_ENABLE[2,Y] = TRUE
$P_WORKAREA_CS_LIMIT_PLUS[2,Y] = 34
$P_WORKAREA_CS_MINUS_ENABLE[2,Y] = TRUE
$P_WORKAREA_CS_LIMIT_MINUS[2,Y] = -25
$P_WORKAREA_CS_PLUS_ENABLE[2,Z] = FALSE
$P_WORKAREA_CS_MINUS_ENABLE[2,Z] = TRUE
$P_WORKAREA_CS_LIMIT_PLUS[2,Z] = -600
$P_WORKAREA_CS_PLUS_ENABLE[2,A] = FALSE
$P_WORKAREA_CS_MINUS_ENABLE[2,A] = FALSE
```

Activation

The working area limitation groups are activated in the NC program using command `WALCS<x>`, with x: Number of the working area limitation group

6.8 Parking a machine axis

If a machine axis is brought into the "Parking" state, then for this particular axis, no encoder actual values are acquired, and all of the monitoring functions described in the preceding sections (measuring system, standstill, clamping monitoring, etc.) are deactivated.

Activation / deactivation

Activate parking

The "Parking" function is **activated** for a machine axis by **resetting** the axis-specific NC/PLC interface signals for the position measuring systems and the controller enable:

- `<Axis>.basic.out.enc1Activation = 0` (position measuring system 1)
- `<Axis>.basic.out.enc2Activation = 0` (position measuring system 2)
- `<Axis>.basic.out.enableCtrl = 0` (controller enable)

The encoder status of the position measuring system of the axis is then set to "Not referenced":

- `<Axis>.basic.in.enc1Synchronized = 0` (referenced/synchronized, position measuring system 1)
- `<Axis>.basic.in.enc2Synchronized = 0` (referenced/synchronized, position measuring system 2)

The following further NC/PLC interface signals are also reset:

- `<Axis>.basic.in.posCtrlActive = 0` (position controller active)
- `<Axis>.basic.in.speedCtrlActive = 0` (speed controller active)
- `<Axis>.drive.in.currentCtrlActive = 0` (current controller active)
- `<Axis>.drive.in.pulsesEnabled = 0` (enable pulses)
- `<Axis>.basic.in.enc1Active = 0` (position measuring system 1 activated)
- `<Axis>.basic.in.enc2Active = 0` (position measuring system 2 activated)

Deactivate parking

The "Parking" function is **deactivated** for a machine axis by **setting** the axis-specific NC/PLC interface signals for the position measuring system to be activated and the controller enable:

- `<Axis>.basic.out.enc1Activation = 1` (position measuring system 1)
or
`<Axis>.basic.out.enc2Activation = 1` (position measuring system 2) = 1
- `<Axis>.basic.out.enableCtrl = 1` (controller enable) = 1

The position control for the machine axis becomes active again at the current position.

The encoder state of the position measuring systems depends on the measuring system type:

- Incremental position measuring system ⇒ "not referenced" state
 - `<Axis>.basic.in.enc1 Synchronized = 0` (referenced/synchronized, position measuring system 1)
 - `<Axis>.basic.in.enc2 Synchronized = 0` (referenced/synchronized, position measuring system 2)
- Absolute position measuring system ⇒ "referenced/synchronized" state
 - `<Axis>.basic.in.enc1 Synchronized = 1` (referenced/synchronized, position measuring system 1)
 - `<Axis>.basic.in.enc2 Synchronized = 1` (referenced/synchronized, position measuring system 2)

The following NC/PLC interface signals are also set again:

- `<Axis>.basic.in.posCtrlActive = 1` (position controller active)
- `<Axis>.basic.in.speedCtrlActive = 1` (speed controller active)
- `<Axis>.drive.in.currentCtrlActive = 1` (current controller active)
- `<Axis>.drive.in.pulsesEnabled = 1` (enable pulses)
- `<Axis>.basic.in.enc1Active = 1` (position measuring system 1 activated)
- `<Axis>.basic.in.enc2Active = 1` (position measuring system 2 activated)

Incremental position measuring systems

After deactivation of the "parking" state, incremental position measuring systems have to be referenced before they have "referenced" encoder status.



WARNING

Incorrect synchronization of the position measuring system caused by offset of the actual machine axis position

If changes have been made to the position measuring system during "parking" that require a change to the parameterized machine data, for example, another encoder has been mounted, the position measuring system must be completely remeasured and referenced. See Function Manual "Axes and Spindles", Section "R1: Referencing".

Machine axis without position measuring system

For a machine axis without a position measuring system (speed-controlled spindle), a status equivalent to "parking" is activated by canceling the controller enable:

- `<Axis>.basic.out.enableCtrl = 0` (controller enable)

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.out.enc1Activation	LBP_Axis*.A_PosMeas1	DB31,DBX1.5
<Axis>.basic.out.enc2Activation	LBP_Axis*.A_PosMeas2	DB31,DBX1.6
<Axis>.basic.out.enableCtrl	LBP_Axis*.A_ContrEnable	DB31,DBX2.1
<Axis>.basic.in.enc1Synchronized	LBP_Axis*.E_RefSyn1	DB31,DBX60.4
<Axis>.basic.in.enc2Synchronized	LBP_Axis*.E_RefSyn2	DB31,DBX60.5
<Axis>.basic.in.posCtrlActive	LBP_Axis*.E_PositContr	DB31,DBX61.5
<Axis>.basic.in.speedCtrlActive	LBP_Axis*.E_SpeedContr	DB31,DBX61.6
<Axis>.drive.in.currentCtrlActive	LBP_Axis*.E_CurrentContr	DB31,DBX61.7
<Axis>.drive.in.pulsesEnabled	LBP_Axis*.E_PulseEnable	DB31,DBX93.7
<Axis>.basic.in.enc1Active	LBP_Axis*.E_PosMeas1	DB31,DBX102.5
<Axis>.basic.in.enc2Active	LBP_Axis*.E_PosMeas2	DB31,DBX102.6

6.9 Parking the passive position measuring system**6.9.1 Function**

Contrary to the "Parking a machine axis (Page 186)" function, for which all position measuring systems of a machine axis are deactivated, using the "Park the passive position measuring system" function, users have the option, of only "parking" the passive position measuring system of a machine axis (i.e. to deactivate the encoder evaluation and monitoring in the drive and in the control), while the active position measuring system can remain operational.

Note

For explanations regarding active/passive measuring system, see Function Manual "Axes and Spindles", Section "Setpoint/actual value system".

Application

The "Park passive measuring system" function can, for example, be used in the following cases:

- Changing attachment heads with and without integrated encoder
Using the "Park passive position measuring system" function, it is possible to mount attachment heads alternating with and without integrated encoder for different machining tasks on the main spindle, without the missing encoder signals initiating drive and control faults.
See also:
 - Example: Changing an attachment head for a direct position measuring system (Page 193)
 - Example: Changing an attachment head for two direct position measuring systems (Page 197)
- Using linear position measuring systems, which are not available over the complete traversing range of a machine axis
Using the "Park passive position measuring system" function, it is possible to pass through the range outside the linear position measuring system, without the missing encoder signals initiating drive and control faults.
See also:
 - "Example: Measuring system switchover when encoders are missing in certain parts of the range (Page 201)".

Activation / deactivation

Activation

The passive position measuring system of a machine axis is parked under the following conditions:

- "Park passive position measuring system" function is active for the measuring system:
MD31046 \$MA_ENC_PASSIVE_PARKING[<n>] = 1
with <n> = 0 (position measuring system 1) or 1 (position measuring system 2)

Note

MD31046 is **not active**:

- for axes with fewer than two encoders:
MD30200 \$MA_NUM_ENCS < 2
 - for simulated encoders:
MD30240 \$MA_ENC_TYPE = 0
-

Note

For position measuring systems, which are used as motor measuring systems, the "Park passive position measuring system" function should be deactivated (MD31046 = 0)!

and

- The user sets the following NC/PLC interface signal to "0":
<Axis>.basic.out.enc1Activation (position measuring system 1) = 0
or
<Axis>.basic.out.enc2Activation (position measuring system 2) = 0

The controller then sets the status for the activation state of the position measuring system to "0":

```
<Axis>.basic.in.enc1Active (position measuring system 1 activated) == 0
```

or

```
<Axis>.basic.in.enc2Active (position measuring system 2 activated) == 0
```

The position measuring system is now no longer monitored and updated.

Deactivation

"Park" is deactivated if the user activates the position measuring system:

```
<Axis>.basic.out.enc1Activation (position measuring system 1) = 1
```

or

```
<Axis>.basic.out.enc2Activation (position measuring system 2) = 1
```

The controller then sets the status for the activation state of the position measuring system back to "1":

```
<Axis>.basic.in.enc1Active (position measuring system 1 activated) == 1
```

or

```
<Axis>.basic.in.enc2Active (position measuring system 2 activated) == 1
```

Note

Switching over to a parked position measuring system takes longer than to a non-parked position measuring system. Because of the time taken, we recommend switching over while the axes are stationary.

Position of the position measuring system

Absolute position measuring systems

For absolute position measuring systems, the position after deactivating "park" corresponds to the actual absolute encoder position.

The position measuring system is referenced:

```
<Axis>.basic.in.enc1Synchronized (referenced/synchronized, position measuring system 1) == 1
```

or

```
<Axis>.basic.in.enc2Synchronized (referenced/synchronized, position measuring system 2) == 1
```

Incremental position measuring systems

For incremental position measuring systems, the position after deactivating "park" always corresponds to the last deactivation position of the position measuring system.

Switching over to the parked position measuring system is only realized if the parameterized permissible deviation between the actual values of the two position measuring systems (see

MD36500 \$MA_ENC_CHANGE_TOL) is not exceeded. Otherwise, users must apply the "Park a machine axis" function, in which case such a check is not made.

The position measuring system is **not** referenced:

`<Axis>.basic.in.enc1Synchronized (referenced/synchronized, position measuring system 1) == 0`

or

`<Axis>.basic.in.enc2Synchronized (referenced/synchronized, position measuring system 2) == 0`

Incremental position measuring systems with position transfer

Alternatively, for incremental position measuring systems, when the "Park passive position measuring system" is active (MD31046 \$MA_ENC_PASSIVE_PARKING[<n>] = 1) it is possible, after deactivating "Park" to transfer the position, and where relevant, also the "Referenced" status, from the previously active position measuring system.

For every position measuring system of a machine axis, this function can be activated using machine data:

MD34210 \$MA_ENC_REFP_STATE[<n>]

with <n> = 0 (position measuring system 1) or 1 (position measuring system 2)

Value	Meaning
1	Only the position from the previous active position measuring system is transferred. The position measuring system is not referenced: <code><Axis>.basic.in.enc1Synchronized (referenced/synchronized, position measuring system 1) == 0</code> or <code><Axis>.basic.in.enc2Synchronized (referenced/synchronized, position measuring system 2) == 0</code>
2	Position and "Referenced" status are transferred from the previously active position measuring system. The position measuring system is referenced: <code><Axis>.basic.in.enc1Synchronized (referenced/synchronized, position measuring system 1) == 1</code> or <code><Axis>.basic.in.enc2Synchronized (referenced/synchronized, position measuring system 2) == 1</code>

Note

Position and "Referenced" status of the previously active position measuring system are only transferred for incremental position measuring systems, depending on MD34210 \$MA_ENC_REFP_STATE[<n>] – and only when the "Park passive position measuring system" function is active (MD31046 \$MA_ENC_PASSIVE_PARKING[<n>] = 1).

The transferred position has the accuracy of the previous active position measuring system. The position measuring system should be referenced if this accuracy is not adequate.

**WARNING****Incorrect synchronization of the position measuring system caused by offset of the actual machine axis position**

If changes have been made to the position measuring system during "parking" that require a change to the parameterized machine data, for example, another encoder has been mounted, the position measuring system must be completely remeasured and referenced. See Function Manual "Axes and Spindles", Section "R1: Referencing".

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.out.enc1Activation	LBP_Axis*.A_PosMeas1	DB31,DBX1.5
<Axis>.basic.out.enc2Activation	LBP_Axis*.A_PosMeas2	DB31,DBX1.6
<Axis>.basic.in.enc1Synchronized	LBP_Axis*.E_RefSyn1	DB31,DBX60.4
<Axis>.basic.in.enc2Synchronized	LBP_Axis*.E_RefSyn2	DB31,DBX60.5
<Axis>.basic.in.enc1Active	LBP_Axis*.E_PosMeas1	DB31,DBX102.5
<Axis>.basic.in.enc2Active	LBP_Axis*.E_PosMeas2	DB31,DBX102.6

6.9.2 Supplementary conditions**Interaction with "dual position feedback"**

The "Park passive position measuring system" function **cannot** be used in conjunction with the "dual position feedback" function (MD32960 \$MA_POSCTRL_DUAL_FEEDBACK_TIME > 0).

Interaction with "position difference input"

The "Park passive position measuring system" function **cannot** be used in conjunction with the "position difference input" function (MD32950 \$MA_POSCTRL_DAMPING > 0).

Interaction with APC (optional)

The "Park passive measuring system" function **cannot** be used in conjunction with the "Advanced positioning control (APC)" drive function.

Interaction with encoder safety protection concept

Only the 1 encoder safety protection concept can be used in conjunction with the "Park passive position measuring system" function.

Interaction when connecting and disconnecting at the DRIVE-CLiQ

If, instead of the encoder cable, the DRIVE-CLiQ cables, e.g. between the SMC and Motor Module are unplugged and plugged, such encoders can only be unparked without fault via the "Parking a machine axis (Page 186)" function.

6.9.3 Example: Changing an attachment head for a direct position measuring system**Initial situation**

- Attachment head "A" has an encoder E2.
- Attachment head "B" does not have an encoder.
- Spindle "SP" has an encoder E1.
- One of the following two telegram types is configured in MD13060 \$MN_DRIVE_TELEGRAM_TYPE (standard telegram type for PROFIdrive):
 - Telegram 116 (motor encoder + an external encoder)
 - or
 - Telegram 136 (motor encoder + an external encoder, and torque precontrol)
- The following position measuring systems are configured in the spindle "SP":
 - Motor encoder E1 as position measuring system 1
 - Direct encoder E2 as position measuring system 2
- Attachment head "A" with encoder E2 is currently mounted on the spindle.
- Position measuring system 2 is the active measuring system:


```
<Axis>.basic.out.enc2Activation = 1
```

 Position measuring system 1 is passive.
- The "Park passive position measuring system" function is:
 - Not active for position measuring system 1:

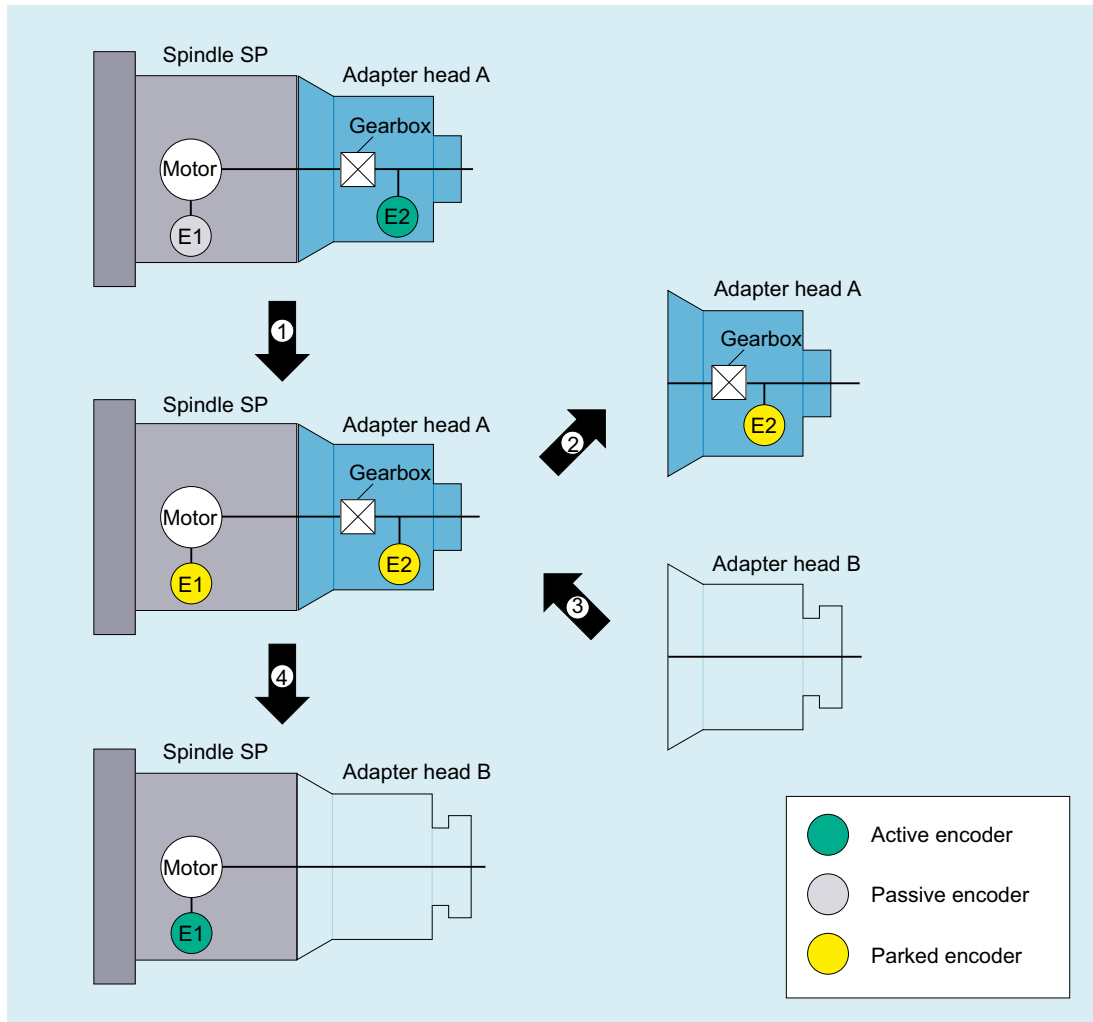

```
MD31046 $MA_ENC_PASSIVE_PARKING [ 0 ] = 0
```
 - Active for position measuring system 2:


```
MD31046 $MA_ENC_PASSIVE_PARKING [ 1 ] = 1
```

Objective

The user would like to change from attachment head "A" to attachment head "B."

Execution



- Before changing an attachment head, the user must deactivate all position measuring systems of the machine axis using the "Parking a machine axis (Page 186)" function:

```
<Axis>.basic.out.enc1Activation (position measuring system 1) = 0
```

```
<Axis>.basic.out.enc2Activation (position measuring system 2) = 0
```

The controller then resets the status signals for the position measuring systems:

```
<Axis>.basic.in.enc1Active (position measuring system 1 activated) == 0
```

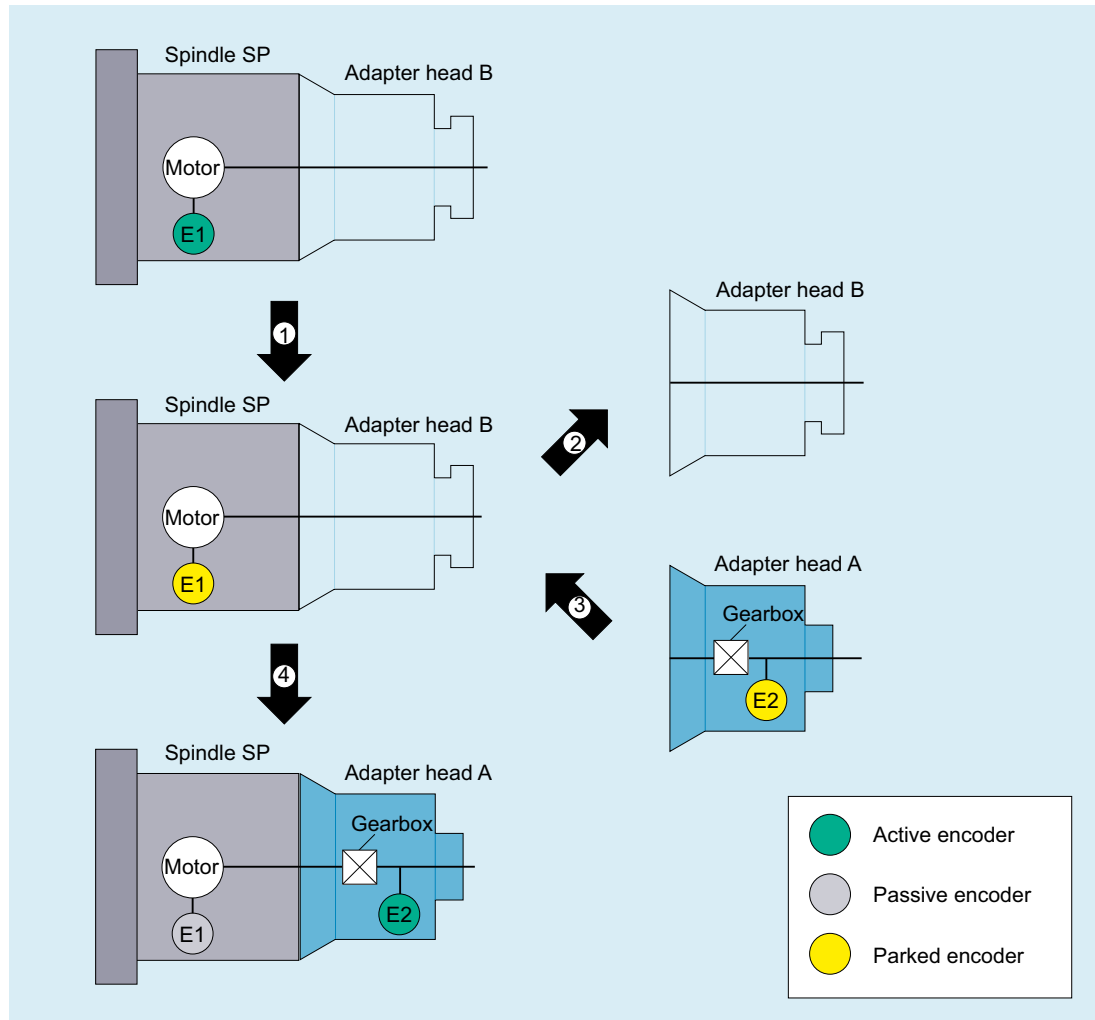
```
<Axis>.basic.in.enc2Active (position measuring system 2 activated) == 0
```
- The user waits for the status signals and only now removes the attachment head "A" from the spindle. This electrically disconnects the encoder cable between the attachment head "A" and coupling. The absence of encoder E2 does not generate any NC or drive faults.

- ③ Attachment head "B" is now mounted on the spindle.
- ④ The user only activates position measuring system 1:
`<Axis>.basic.out.enclActivation (position measuring system 1) = 1`
The control then sets the status signal:
`<Axis>.basic.in.enclActive (position measuring system 1 activated) == 1`
The "Park passive position measuring system" function is active for position measuring system 2.
This means that position measuring system 2 does not become passive, but remains in the "Park" state.

Objective

The user now wishes to remount attachment head "A" again.

Execution



- ① Using the "Park machine axis" function, the user deactivates position measuring system 1:
`<Axis>.basic.out.enc1Activation (position measuring system 1) = 0`
 The controller then resets the status signal for the position measuring system:
`<Axis>.basic.in.enc1Active (position measuring system 1 activated) == 0`
- ② The user waits for the status signal and only now removes the attachment head "B" from the spindle.
- ③ Attachment head "A" is now mounted on the spindle.
- ④ The user activates position measuring system 2:
`<Axis>.basic.out.enc2Activation (position measuring system 2) = 1`
 As a consequence, position measuring system 1 is also simultaneously activated; This is because the "Park passive position measuring system" function is not active for position measuring system 1 (motor measuring system!). Position measuring system 1 becomes a passive position measuring system.
 The controller then sets the status signals for the position measuring systems:
`<Axis>.basic.in.enc1Active (position measuring system 1 activated) == 1`
`<Axis>.basic.in.enc2Active (position measuring system 2 activated) == 1`

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.out.enc1Activation	LBP_Axis*.A_PosMeas1	DB31,DBX1.5
<Axis>.basic.out.enc2Activation	LBP_Axis*.A_PosMeas2	DB31,DBX1.6
<Axis>.basic.in.enc1Active	LBP_Axis*.E_PosMeas1	DB31,DBX102.5
<Axis>.basic.in.enc2Active	LBP_Axis*.E_PosMeas2	DB31,DBX102.6

6.9.4 Example: Changing an attachment head for two direct position measuring systems

Initial situation

- Attachment head "A" has an encoder E3.
- Attachment head "B" does not have an encoder.
- Spindle "SP" has two encoders E1 and E2.
- One of the following two telegram types is configured in MD13060 \$MN_DRIVE_TELEGRAM_TYPE (standard telegram type for PROFIdrive):
 - Telegram 118 (two external encoders)
or
 - Telegram 138 (two external encoders, plus torque precontrol)

Further information:

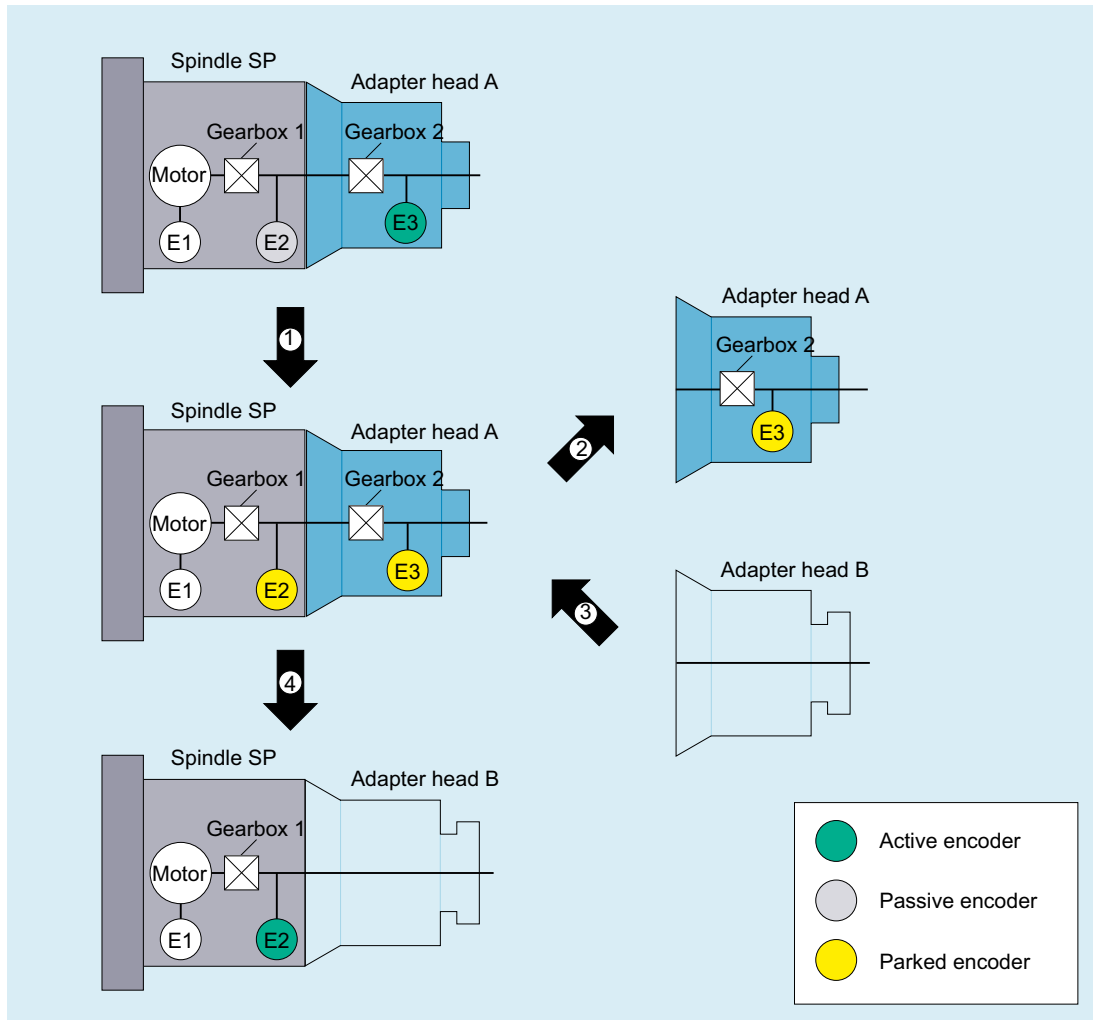
Information regarding encoder assignment, see:
Commissioning Manual CNC Commissioning: NC, PLC, drive; communication between NC and drive > Drives: Assign axis

- The following position measuring systems are configured in the spindle "SP":
 - Direct encoder E2 as position measuring system 1
 - Direct encoder E3 as position measuring system 2
- Attachment head "A" with encoder E3 is currently mounted on the spindle.
- Position measuring system 2 is the active measuring system:
`<Axis>.basic.out.enc2Activation = 1`
 Position measuring system 1 is passive.
- The "Park passive position measuring system" function is:
 - Not active for position measuring system 1:
`MD31046 $MA_ENC_PASSIVE_PARKING [ 0 ] = 0`
 - Active for position measuring system 2:
`MD31046 $MA_ENC_PASSIVE_PARKING [ 1 ] = 1`

Objective

The user would like to change from attachment head "A" to attachment head "B."

Execution



- Before changing an attachment head, the user must deactivate all position measuring systems of the machine axis using the "Parking a machine axis (Page 186)" function:

```
<Axis>.basic.out.enc1Activation (position measuring system 1) = 0
```

```
<Axis>.basic.out.enc2Activation (position measuring system 2) = 0
```

The controller then resets the status signals for the position measuring systems:

```
<Axis>.basic.in.enc1Active (position measuring system 1 activated) == 0
```

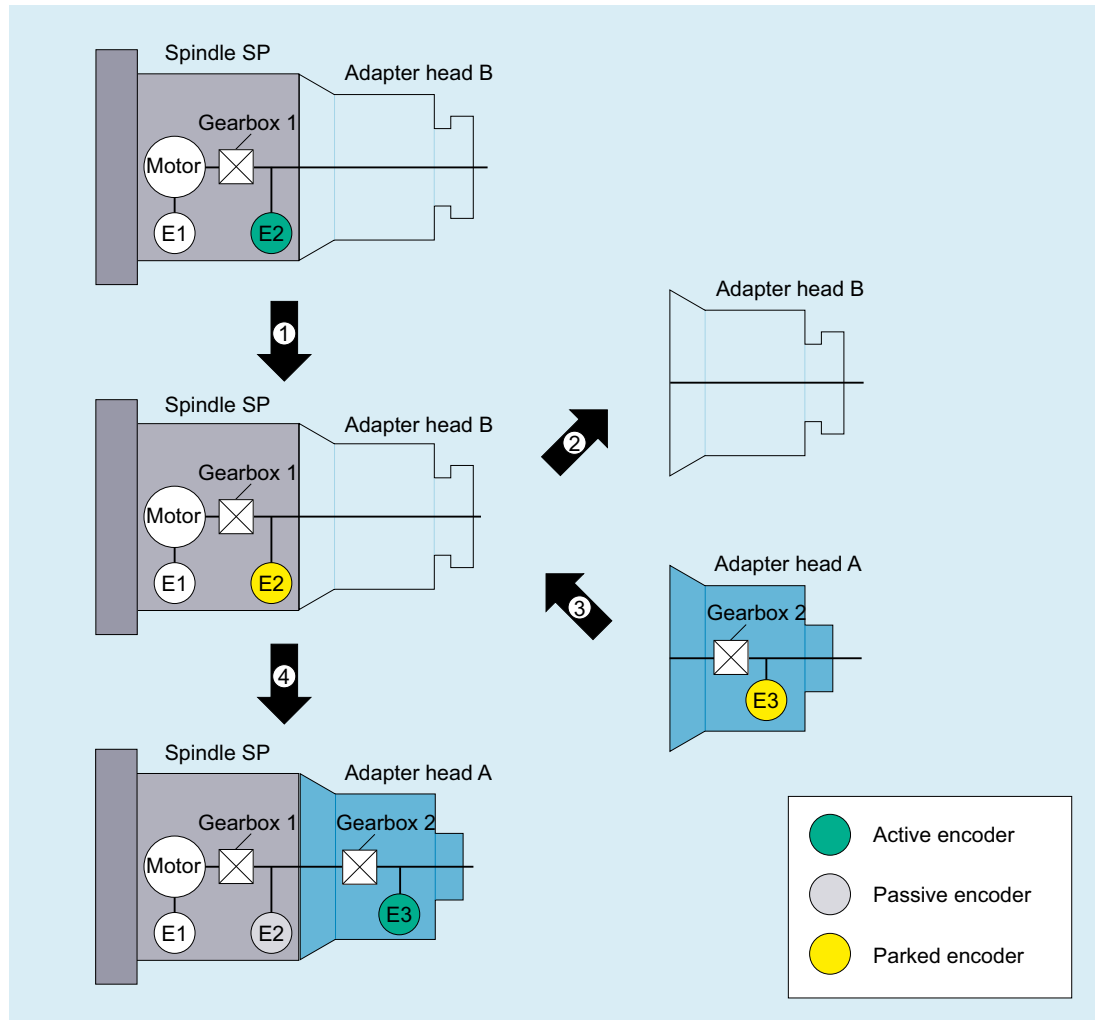
```
<Axis>.basic.in.enc2Active (position measuring system 2 activated) == 0
```
- The user waits for the status signals and only now removes the attachment head "A" from the spindle. This electrically disconnects the encoder cable between the attachment head "A" and coupling. The absence of encoder E3 does not generate any NC or drive faults.

- ③ Attachment head "B" is now mounted on the spindle.
- ④ The user only activates position measuring system 1:
`<Axis>.basic.out.enclActivation (position measuring system 1) = 1`
The control sets the status signal:
`<Axis>.basic.in.enclActive (position measuring system 1 activated) == 1`
The "Park passive position measuring system" function is active for position measuring system 2.
This means that position measuring system 2 does not become passive, but remains in the "Park" state.

Objective

The user now wishes to remount attachment head "A" again.

Execution



- ① Using the "Park machine axis" function, the user deactivates position measuring system 1:
`<Axis>.basic.out.enc1Activation (position measuring system 1) = 0`
 The controller then resets the status signal for the position measuring system:
`<Axis>.basic.in.enc1Active (position measuring system 1 activated) == 0`
- ② The user waits for the status signal and only now removes the attachment head "B" from the spindle.
- ③ Attachment head "A" is now mounted on the spindle.
- ④ The user activates position measuring system 2:
`<Axis>.basic.out.enc2Activation (position measuring system 2) = 1`
 As a consequence, position measuring system 1 is also simultaneously activated; This is because the "Park passive position measuring system" function is not active for position measuring system 1. Position measuring system 1 becomes a passive position measuring system.
 The controller then sets the status signals for the position measuring systems:
`<Axis>.basic.in.enc1Active (position measuring system 1 activated) == 1`
`<Axis>.basic.in.enc2Active (position measuring system 2 activated) == 1`

PLC signals

Basic Program Plus	Basic Program	
<Axis>.basic.out.enc1Activation	LBP_Axis*.A_PosMeas1	DB31,DBX1.5
<Axis>.basic.out.enc2Activation	LBP_Axis*.A_PosMeas2	DB31,DBX1.6
<Axis>.basic.in.enc1Active	LBP_Axis*.E_PosMeas1	DB31,DBX102.5
<Axis>.basic.in.enc2Active	LBP_Axis*.E_PosMeas2	DB31,DBX102.6

6.9.5 Example: Measuring system switchover when encoders are missing in certain parts of the range

In the following example, the direct linear position measuring system is only available in the machining zones, while only the motor measuring system is available outside the machining zones.

Initial situation

- Linear axis "X" has two incremental encoders:
 - Motor encoder E1
 - Direct linear encoder E2
- Direct linear encoder E2 is only available in the machining range.
- One of the following two telegram types is configured in MD13060 \$MN_DRIVE_TELEGRAM_TYPE (standard telegram type for PROFIdrive):
 - Telegram 116 (motor encoder + an external encoder)
 - or
 - Telegram 136 (motor encoder + an external encoder, and torque precontrol)
- The following position measuring systems are configured for the machine axis:
 - Motor encoder E1 as position measuring system 1
 - Direct linear encoder E2 as position measuring system 2
- When the machine is switched on, the user activates both position measuring systems:
 - <Axis>.basic.out.enc1Activation (position measuring system 1) = 1
 - <Axis>.basic.out.enc2Activation (position measuring system 2) = 1

When both position measuring systems are simultaneously activated, then the control selects position measuring system 1 as active position measuring system.
- Position measuring system 2 is selected for machining:
 - <Axis>.basic.out.enc1Activation (position measuring system 1) = 0
 - <Axis>.basic.out.enc2Activation (position measuring system 2) = 1

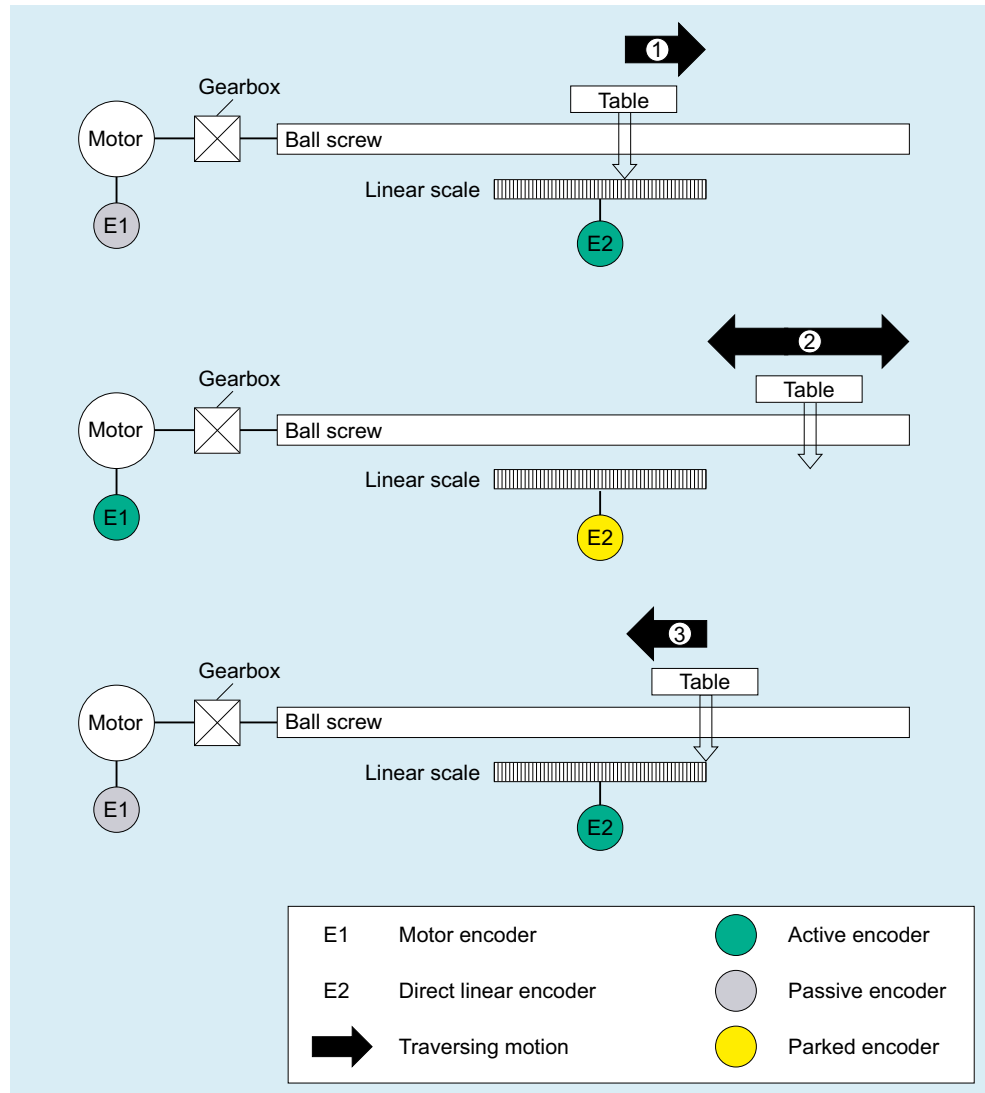
6.9 Parking the passive position measuring system

- The "Park passive position measuring system" function is:
 - Not active for position measuring system 1:
MD31046 \$MA_ENC_PASSIVE_PARKING [0] = 0
 - Active for position measuring system 2:
MD31046 \$MA_ENC_PASSIVE_PARKING [1] = 1
- Transfer of the position and the "referenced" status is activated for position measuring system 2:
MD34210 \$MA_ENC_REFP_STATE[1] = 2

Objective

When passing through the range outside linear position measuring system E2, the missing encoder signals should not initiate any faults in the drive and in the control.

Execution



- ① Before the table reaches the end of the linear position measuring system, a switchover must be made to the motor measuring system. The user does this by activating both position measuring systems:

```
<Axis>.basic.out.enc1Activation (position measuring system 1) = 1
```

```
<Axis>.basic.out.enc2Activation (position measuring system 2) = 1
```

Using the "Park passive position measuring system" function, the linear position measuring system, which is passive after the measuring system switchover, is parked by the control.

The control resets the status signal:

```
<Axis>.basic.in.enc2Active (position measuring system 2 activated) == 0
```

The user waits for the status signal before continuing to traverse in the range outside the linear position measuring system.

- ② Traversing through the range outside the linear position measuring system with the motor measuring system.

6.10 Switching over encoder data sets

- ③ If the table returns to the linear position measuring system range, at standstill, the user switches from the motor measuring system to the linear position measuring system:
- ```
<Axis>.basic.out.enc1Activation (position measuring system 1) = 0
```
- ```
<Axis>.basic.out.enc2Activation (position measuring system 2) = 1
```
- The control sets the status signal:
- ```
<Axis>.basic.in.enc2Active (position measuring system 2 activated) == 1
```
- The motor measuring system becomes a passive position measuring system.

## Result

Both position measuring systems are referenced. The position of the linear position measuring system corresponds to the position of the motor measuring system at the switchover instant. The linear position measuring system must be rereferenced if the position accuracy is not sufficient.

## PLC signals

| Basic Program Plus              | Basic Program        |                     |
|---------------------------------|----------------------|---------------------|
| <Axis>.basic.out.enc1Activation | LBP_Axis*.A_PosMeas1 | DB31, ... .DBX1.5   |
| <Axis>.basic.out.enc2Activation | LBP_Axis*.A_PosMeas2 | DB31, ... .DBX1.6   |
| <Axis>.basic.in.enc2Active      | LBP_Axis*.E_PosMeas2 | DB31, ... .DBX102.6 |

## 6.10 Switching over encoder data sets

## Application

Different attachment heads may be used in succession on one and the same power unit in order to perform different machining tasks.

One motor data set (MDS) and one encoder data set (EDS) must be programmed for each attachment head that is equipped with a motor or an encoder. These data sets must be switched over in the PLC program whenever the attachment head is changed. An MDS/EDS switchover can only be implemented indirectly in this regard via switchover of the drive data set (DDS).

## Function

**NOTICE****Machine damage**

If the following drive parameters and machine data have different settings, the axis might not traverse at the programmed speed or to the programmed position.

The machine data must be changed simultaneously and consistently with encoder data switchover in the parked state.

Encoder data switchover in the control is limited to the following SINAMICS drive parameters.

- p0408 (rotary encoder pulse number)
- p0418 (fine resolution of encoder emulation Gx\_XIST1 (in bits))
- p0419 (fine resolution absolute value Gx\_XIST2 (in bits))

With these parameters, it is possible to switch over encoder data sets with the same encoder type but a different number of encoder pulses.

Switchover in the control is activated in the following machine data:

MD31700 \$MA\_ENC\_EDS\_ACTIVE (activate EDS use)

| Value | Meaning                                      |
|-------|----------------------------------------------|
| 0     | Encoder data set switchover EDS is not used. |
| 1     | Encoder data set switchover EDS is used.     |

A machine data item is provided for each of the drive parameters p0408, p0418, and p0419, which must be parameterized in accordance with the active encoder data set:

- MD31710 \$MA\_ENC\_RESOL\_EDS (encoder pulses per revolution for EDS use)
- MD31720 \$MA\_ENC\_PULSE\_MULT\_EDS (encoder multiplication (high resolution) for EDS use)
- MD31730 \$MA\_ABS\_INC\_RATIO\_EDS (absolute encoder: ratio between absolute resolution and incremental resolution for EDS use)

## Effectiveness

If MD31700 \$MA\_ENC\_EDS\_ACTIVE = 1, the following machine data no longer take effect:

- MD30260 \$MA\_ABS\_INC\_RATIO (absolute encoder: ratio between absolute resolution and incremental resolution)
- MD31020 \$MA\_ENC\_RESOL (encoder pulses per revolution)
- MD31025 \$MA\_ENC\_PULSE\_MULT (encoder multiplication (high resolution))

The drive data switchover in `<Axis>.drive.out.driveDataSetSelection` also switches over the encoder data sets. Switchover is performed in the parked state (see chapters "Parking a machine axis (Page 186)" and "Parking the passive position measuring system (Page 188)").

### Note

If MD31700 \$MA\_ENC\_EDS\_ACTIVE = 1, no plausibility check between the values set in the drive and control is performed.

### NC/PLC interface signals

- `<Axis>.drive.out.enablePulses` (pulse enable)
- `<Axis>.drive.out.disableSpeedCtrlIntegrator` (integrator disable speed controller)

## 6.10 Switching over encoder data sets

- `<Axis>.drive.out.motorChangeoverDone` (motor selection)
- `<Axis>.drive.out.driveDataSetSelection` (request to switch over the drive data set)
- `<Axis>.drive.out.motorDataSetSelection` (request to switch over the motor data set)
- The interface for switching over the drive/motor data sets can be parameterized via:  
`<Axis>.drive.in.numberOfDriveDataSets`

## Constraints

- Only machine data for rotary encoders are available.
- Traversing range extension for absolute-value encoders  
MD30270 \$MA\_ENC\_ABS\_BUFFERING = 0 is not permissible.

## More information

Function Manual SINAMICS S120 Drive Functions;  
Basic information about the drive system > Data sets

- DDS: Drive data set
- EDS: Encoder data set

## PLC signals

| Basic Program Plus                                             | Basic Program                            |                                     |
|----------------------------------------------------------------|------------------------------------------|-------------------------------------|
| <code>&lt;Axis&gt;.drive.out.driveDataSetSelection</code>      | <code>LBP_Axis*.A_ParA ... C</code>      | <code>DB31, ... .DBX21.0..2</code>  |
| <code>&lt;Axis&gt;.drive.out.motorDataSetSelection</code>      | <code>LBP_Axis*.A_MotA ... B</code>      | <code>DB31, ... .DBX21.3..4</code>  |
| <code>&lt;Axis&gt;.drive.out.motorChangeoverDone</code>        | <code>LBP_Axis*.A_MotOK</code>           | <code>DB31, ... .DBX21.5</code>     |
| <code>&lt;Axis&gt;.drive.out.disableSpeedCtrlIntegrator</code> | <code>LBP_Axis*.A_IntegratDisable</code> | <code>DB31, ... .DBX21.6</code>     |
| <code>&lt;Axis&gt;.drive.out.enablePulses</code>               | <code>LBP_Axis*.A_PulseEnable</code>     | <code>DB31, ... .DBX21.7</code>     |
| <code>&lt;Axis&gt;.drive.in.numberOfDriveDataSets</code>       | ---                                      | <code>DB31, ... .DBX130.0..4</code> |

## Other machine data

In addition to the machine data of the axis monitoring function, the following additional machine data should be set or checked:

## All machine axes

- MD31030 \$MA\_LEADSCREW\_PITCH (leadscrew pitch)
- MD31050 \$MA\_DRIVE\_AX\_RATIO\_DENOM (denominator load gearbox)
- MD31060 \$MA\_DRIVE\_AX\_RATIO\_NUMERA (numerator load gearbox)
- MD31070 \$MA\_DRIVE\_ENC\_RATIO\_DENOM (denominator measuring gearbox)
- MD31080 \$MA\_DRIVE\_ENC\_RATIO\_NUMERA (numerator measuring gearbox)

- MD32810 \$MA\_EQUIV\_SPEEDCTRL\_TIME (equivalent time constant speed control loop for feedforward control)
- Encoder resolution: see Function Manual "Axes and Spindles", Section "G2: Velocities, setpoint/actual value systems, closed-loop control".

**Machine axes with analog speed setpoint interface**

- MD32260 \$MA\_RATED\_VELO (rated motor speed)
- MD32250 \$MA\_RATED\_OUTVAL (rated output voltage)



## Contour tunnel monitoring (option)

### Note

The function is an option ("Contour monitoring with tunnel function"), which must be assigned to the hardware via the license management.

### 7.1 Function

"Contour tunnel monitoring" operates in specific channels. The absolute movement of the tool tip in space is monitored.

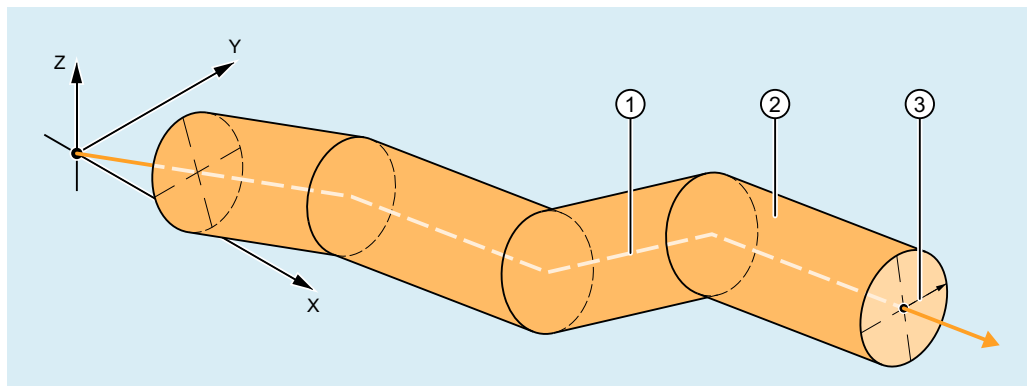
### Note

Monitoring synchronized axes, positioning axes, etc. that are not geometry axes is performed directly on the machine axis plane using function "Contour monitoring (Page 145)".

### Tolerance range

A tolerance range is defined for the programmed path of the tool tip (contour).

For three geometry axes, this tolerance range can be considered to be a tunnel (contour tunnel) through whose center the tool tip path runs and whose radius defines the permissible tolerance:



- ① Contour (programmed path of the tool tip)
- ② "Tunnel" placed around the contour (contour tunnel)
- ③ Permissible tolerance

For only two geometry axes, the tolerance range is defined using lines parallel to the programmed path of the tool tip.

### Response

As long as the calculated actual position of the tool tip remains inside the defined tolerance range, motion continues in the normal way. If the calculated actual position leaves the tolerance range (in the default setting), an alarm is triggered and the axes are stopped along a braking ramp.

The response when exiting the tolerance range can be set, and can be changed as follows:

- Alarm; however, motion continues
- Quick stop

### Activating / deactivating

The monitoring will only become active if the following conditions are met:

- The threshold value (tunnel radius) for the contour tunnel monitoring is greater than 0.0.
- At least two geometry axes have been defined.

The following setting can be used to deactivate the monitoring: Threshold value = 0.0

### Commissioning

"Contour tunnel monitoring" is commissioned using machine data.

More information: Chapter "Parameterization (Page 210)"

### Quality control

The calculated clearance between the programmed path and the actual positions can be transferred to an analog output to analyze the progression of the contour errors during normal operation.

## 7.2 Commissioning

### 7.2.1 Parameterization

#### Tolerance limit

The contour tunnel monitoring responds if the distance between the tool tip (actual value) and the programmed path (setpoint) exceeds the parameterized threshold value due to axis deviations:

MD21050 \$MC\_CONTOUR\_TUNNEL\_TOL (Response threshold for Contour tunnel monitoring)

There is no monitoring if a value of 0.0 is set for the machine data.

The value of the machine data is transferred to the control for new configurations.

### Note

#### Encoder switchover

Switching between two encoder systems usually causes a sudden change in the actual position of the tool tip. The change resulting solely from the encoder switchover must not be so large as to cause the tool tip to exceed the tolerance range. Therefore, the radius defined in MD21050 must be higher than the permissible tolerance for the actual value encoder switchover:

MD36500 \$MA\_ENC\_CHANGE\_TOL (tolerance for the position actual value switchover)

## Response

The response when the contour tunnel monitoring responds can be set using the following machine data:

MD21060 \$MC\_CONTOUR\_TUNNEL\_REACTION (Reaction upon response of contour tunnel monitoring)

| Value | Meaning                                                            |
|-------|--------------------------------------------------------------------|
| 0     | Display alarm and continue machining                               |
| 1     | Deceleration according to the deceleration ramps (default setting) |
| 2     | Rapid stop (speed setpoint = 0)                                    |

## Analysis output

The values of deviation of the actual value of the tool tip from the programmed path can – for analysis purposes – be output on a fast analog output (accuracy monitoring).

An analog output is assigned using machine data:

MD21070 \$MC\_CONTOUR\_ASSIGN\_FASTOUT

| Value | Meaning                     |
|-------|-----------------------------|
| 0     | No output (default setting) |
| 1     | Output to output 1          |
| 2     | Output to output 2          |
| ...   | ...                         |
| 8     | Output to output 8          |

### Scale

The tunnel radius set in MD21050 corresponds to a voltage of 10 V at the output.

## 7.3 Restrictions

### Coupled motion

If coupled motion between two geometry axes is programmed with contour tunnel monitoring, this always results in activation of the contour tunnel monitoring. In this case, the contour tunnel monitoring must be switched off before programming the coupled motion:

MD21050 \$MC\_CONTOUR\_TUNNEL\_TOL = 0.0

# Compensations

## 8.1 Introduction

### Accuracy errors

The accuracy of machine tools is impaired as a result of deviations from the ideal geometry, power transmission faults and measuring system errors. Temperature differences and mechanical forces often result in great reductions in precision when large workpieces are machined.

### Compensation functions

Some of these deviations can usually be measured during commissioning and then compensated for during operation on the basis of values read by the positional actual-value encoder and other sensory devices. State-of-the-art CNC controls have compensation functions that are active on an axis for axis basis.

### Parameterization

These compensation functions can be set for each machine individually with axis-specific machine data.

### Activation

The compensations are active in all operating modes of the control as soon as the input data are available. Any compensations that require the position actual value are not activated until the axis reaches the reference point.

### Position display

The normal actual-value and setpoint position displays ignore the compensation values and show the position values of an ideal machine. The compensation values are output in the "Diagnosis" operating area of the "Axis/Spindle Service" window.

## 8.2 Temperature compensation

### 8.2.1 Function

#### Deformation due to temperature effects

Heat generated by the drive equipment or high ambient temperatures (e.g. caused by sunlight, drafts) cause the machine base and parts of the machinery to expand. This expansion depends, among other things, on the temperature and on the thermal conductivity of the machine parts.

#### Effects

Due to the thermal expansion of the machinery, the actual positions of the axes change depending on temperature. This has a negative impact on the precision of the workpieces being machined.

#### Temperature compensation

By activating the "temperature compensation" function, actual value changes due to temperature effects can be compensated on an axis-by-axis basis.

#### Sensor equipment

To provide effective temperature compensation, a number of temperature sensors for acquiring a temperature profile are needed in addition to the actual position data from existing encoders.

Since temperature-dependent changes occur relatively slowly, the PLC can acquire and preprocess the temperature profile in a minutes cycle, for example.

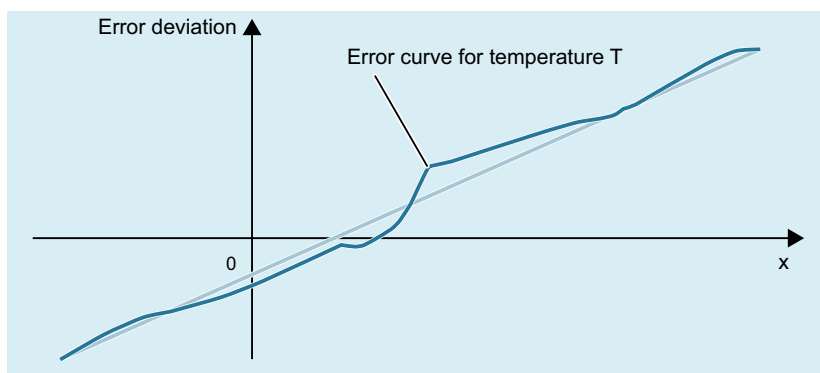
#### Error curves

In order to implement temperature compensation, the actual-value offsets over the positioning range of the axis must be measured at a given temperature (T) and plotted. This produces an error curve for this temperature value. Error curves must be produced for different temperatures.

#### Error curve characteristic

If an axis position reference point  $P_0$  is selected, an offset in the reference point (corresponds to the "position-independent component" of the temperature compensation) can be observed as the temperature changes, and because of the change in length an additional offset in the other position points, which increases with the distance to the reference point (corresponds to the "position-dependent component" of the temperature compensation).

The error curve for a given temperature T can generally be represented with sufficient accuracy by a straight line with a temperature dependent gradient and reference position:



### Compensation equation

The compensation value  $\Delta K_x$  is calculated on the basis of current actual position  $P_x$  of this axis and temperature  $T$  according to the following equation:

$$\Delta K_x = K_0(T) + \tan\beta(T) * (P_x - P_0)$$

The meaning is as follows:

- $\Delta K_x$ : Temperature compensation value of axis at position  $P_x$
- $K_0$ : Position-independent temperature compensation value of axis
- $P_x$ : Actual position of axis
- $P_0$ : Reference position of axis
- $\tan\beta$ : Coefficient for the position-dependent temperature compensation (corresponds to the gradient of the approximated error line)

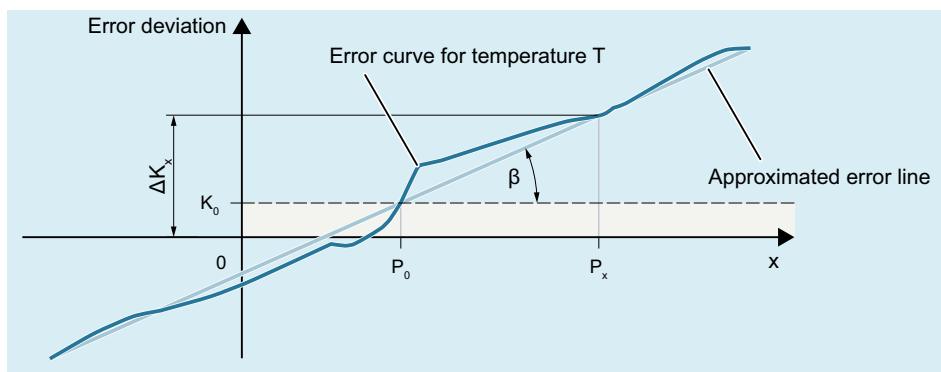


Figure 8-1 Approximated error line for temperature compensation

### Effectiveness

The following conditions must be fulfilled so that the temperature compensation can be activated:

1. The compensation type is selected (MD32750, see "Commissioning (Page 217)").
2. The parameters for the compensation type are defined (see "Commissioning (Page 217)").
3. The axis is referenced.
  - `<Axis>.basic.in.enc1Synchronized == 1` (referenced / synchronized 1) or
  - `<Axis>.basic.in.enc2Synchronized == 1` (referenced / synchronized 2)

As soon as these conditions are fulfilled, the temperature compensation value for the position actual value is added to the setpoint in all modes and the machine axis traverses through this distance. If the compensation value  $\Delta K_x$  is positive, the axis moves in the negative direction.

If the reference position is then lost, e.g. because the encoder frequency has been exceeded, compensation processing is deactivated.

When the encoder frequency is exceeded, the following NC/PLC interface signals are reset:

- `<Axis>.basic.in.enc1Synchronized = 0` (referenced / synchronized 1) or
- `<Axis>.basic.in.enc2Synchronized = 0` (referenced / synchronized 2)

### Determining the compensation values

The compensation values are determined in the interpolator clock cycle.

### Display

The total compensation value calculated from the temperature and sag compensation functions belonging to the actual position is output in the "Diagnosis" operating area of the "Axis/Spindle Service" window.

### Parameter adaptation for temperature changes

Since the approximated error line applies only to the instantaneous temperature value, the parameters of the error lines that are newly generated when the temperature rises or falls must be sent to the NC again. Only in this way can expansion due to heat always be correctly compensated.

When temperature  $T$  changes, the parameters which are temperature-dependent, i.e. ( $K_0$ ,  $\tan\beta$  and  $P_0$ ) also change and can thus always be overwritten by the PLC or by means of a synchronized action.

---

#### Note

When writing via the synchronous action, note that the value is not written in every interpolation cycle.

---

It is thus possible for the machine-tool manufacturer to emulate the mathematical and technological relationship between the axis positions and temperature values via the PLC user program and thus calculate the various parameters for the temperature compensation. The temperature parameters are transferred to the NC using the variable services (FB2 (GET) "Read data" and FB3 (PUT) "Write data").

#### More information

For more information on handling and parameterization of FB2 and FB3, see:  
Function Manual PLC

### Smoothing the compensation values

To prevent overloading of the machine or tripping of monitoring functions in response to step changes in the temperature compensation parameters, the compensation values are distributed over several IPO cycles by an internal control function as soon as they exceed the maximum compensation value specified for each IPO cycle (MD32760, see "Commissioning (Page 217)").

### PLC signals

| Basic Program Plus               | Basic Program       |                    |
|----------------------------------|---------------------|--------------------|
| <Axis>.basic.in.enc1Synchronized | LBP_Axis*.E_RefSyn1 | DB31, ... .DBX60.4 |
| <Axis>.basic.in.enc2Synchronized | LBP_Axis*.E_RefSyn2 | DB31, ... .DBX60.5 |

## 8.2.2 Commissioning

### Temperature-dependent parameters

Error curves for different temperatures can be defined for each axis. For each error curve the following parameters must be determined and then entered in the setting data:

- Position-independent temperature compensation value  $K_0$ :  
SD43900 \$SA\_TEMP\_COMP\_ABS\_VALUE
- Reference position  $P_0$  for position-dependent temperature compensation:  
SD43920 \$SA\_TEMP\_COMP\_REF\_POSITION
- Gradient  $\tan\beta$  for position-dependent temperature compensation:  
SD43910 \$SA\_TEMP\_COMP\_SLOPE

### Temperature compensation type and activation

The temperature compensation type is selected and the temperature compensation activated using the axis-specific machine data:

MD32750 \$MA\_TEMP\_COMP\_TYPE (temperature compensation type)

| Bit | Value | Meaning                                       | Associated parameters                                                             |
|-----|-------|-----------------------------------------------|-----------------------------------------------------------------------------------|
| 0   |       | Position independent temperature compensation | SD43900                                                                           |
|     | 0     | Not active                                    |                                                                                   |
|     | 1     | Active                                        |                                                                                   |
| 1   |       | Position-dependent temperature compensation   | SD43920, SD43910                                                                  |
|     | 0     | Not active                                    |                                                                                   |
|     | 1     | Active                                        |                                                                                   |
| 2   |       | Temperature compensation in tool direction    | MD20390<br>\$MC_TOOL_TEMP_COMP_ON (Activate temperature compensation tool length) |
|     | 0     | Not active                                    |                                                                                   |
|     | 1     | Active                                        |                                                                                   |

### Maximum compensation value per IPO clock cycle

The maximum possible compensation value per IPO cycle, i.e. the maximum distance that can be traversed in an IPO cycle as a result of the temperature compensation, is limited using machine data:

MD32760 \$MA\_COMP\_ADD\_VELO\_FACTOR (velocity increase as a result of compensation)

The specified value acts as a factor and is referred to the maximum axis velocity (MD32000 \$MA\_MAX\_AX\_VELO).

MD32760 also limits the maximum gradient of the error line ( $\tan \beta$ ) of the temperature compensation.

## 8.2.3 Example

### 8.2.3.1 Commissioning the temperature compensation for the Z axis of a lathe

Commissioning of temperature compensation is described below using the example of a Z axis on a lathe.

#### Determining the error characteristic of the Z axis

In order to determine the temperature-dependent error characteristic of the Z axis, proceed as follows:

- Uniform temperature increase by traversing the axis across the whole Z axis traversing range (in the example: from 500 mm to 1500 mm)
- Measuring the axis position in increments of 100 mm
- Measuring the actual temperature at the leadscrew
- Executing a traversing measuring cycle every 20 minutes

The mathematical and technological relationships and the resulting parameters for temperature compensation are derived from the recorded data. The calculated deviation errors for a specific temperature, which refer to the actual position of the Z axis displayed by the NC, are represented in graphic form in the diagram below.

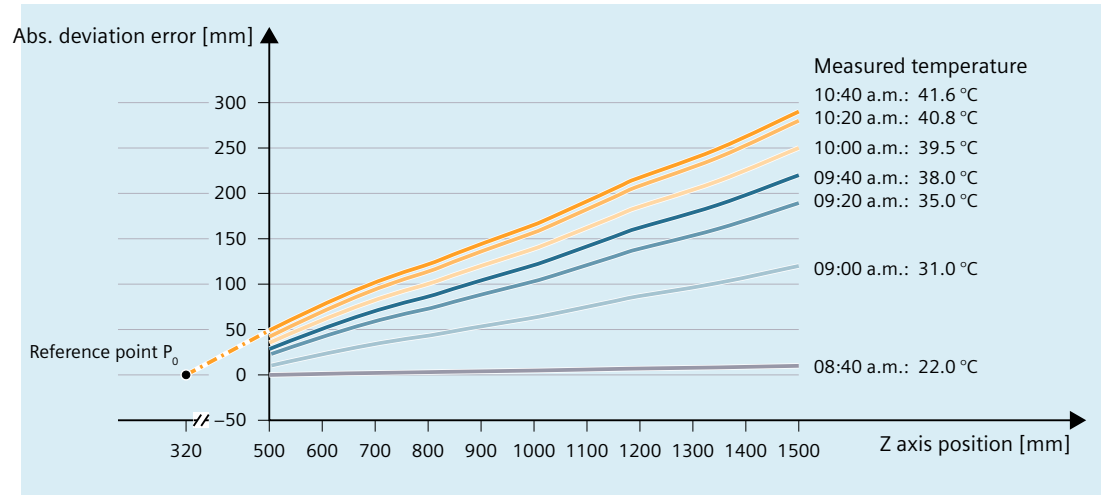


Figure 8-2 Error curves determined for the Z axis

## Specifying parameters

The temperature compensation parameters must now be determined on the basis of the measurement results (see diagram above).

### Reference position $P_0$

As the diagram above illustrates, there are basically two methods of parameterizing reference position  $P_0$ :

1.  $P_0 = 0$  with position-independent temperature compensation value  $K_0 \neq 0$
2.  $P_0 \neq 0$  with position-independent temperature compensation value  $K_0 = 0$

In this case, version 2 is chosen, which means that the position-independent temperature compensation value is always 0. The temperature compensation value therefore only consists of the position-dependent component.

The following parameters are obtained:

- MD32750 \$MA\_TEMP\_COMP\_TYPE = 2  
(only position-dependent temperature compensation active)
- $P_0 = 320 \text{ mm} \rightarrow$  SD43920 \$SA\_TEMP\_COMP\_REF\_POSITION = 320  
(reference position for position-dependent temperature compensation)

### Coefficient $\tan\beta$ (T)

In order to determine the dependency of coefficient  $\tan\beta$  of the position-dependent temperature compensation on the temperature, the error curve gradient is plotted against the measured temperature:

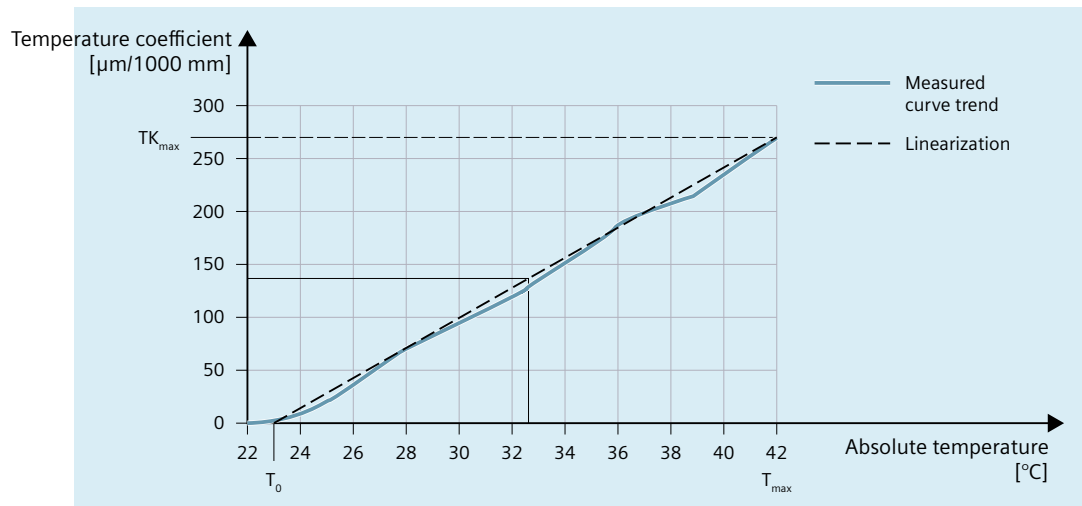


Figure 8-3 Characteristic of coefficient  $\tan\beta$  as a function of measured temperature  $T$

With the appropriate linearization, coefficient  $\tan\beta$  depends on  $T$  as follows:

$$\tan\beta(T) = (T - T_0) * TK_{\max} * 10^{-6} / (T_{\max} - T_0)$$

with

$T_0$  = temperature at which position-dependent error = 0; [degrees]

$T_{\max}$  = maximum measured temperature; [degrees]

$TK_{\max}$  = temperature coefficient at  $T_{\max}$ ; [ $\mu\text{m}/1000 \text{ mm}$ ]

Therefore, based on the values from the above diagram:

$$T_0 = 23^\circ$$

$$T_{\max} = 42^\circ$$

$$TK_{\max} = 270 \mu\text{m}/1000 \text{ mm}$$

and  $\tan\beta(T)$  is therefore:

$$\begin{aligned} \tan\beta(T) &= (T - 23) [\text{degrees}] * 270 [\mu\text{m}/1000 \text{ mm}] * 10^{-6} / (42 - 23) [\text{degrees}] \\ &= (T - 23) [\text{degrees}] * 14.21 [\mu\text{m}/1000 \text{ mm}] * 10^{-6} \end{aligned}$$

Example:

At a temperature of  $T = 32.3$  degrees, therefore:  $\tan\beta = 0.000132$

### PLC user program

The formula given above must be used in the PLC user program to calculate the coefficient  $\tan\beta(T)$  which corresponds to the measured temperature; this must then be written to the following NC setting data:

SD43910 \$SA\_TEMP\_COMP\_SLOPE (gradient for position-dependent temperature compensation)

According to the example above:

$$\text{SD43910 \$SA\_TEMP\_COMP\_SLOPE} = 0.000132$$

## 8.3 Backlash compensation

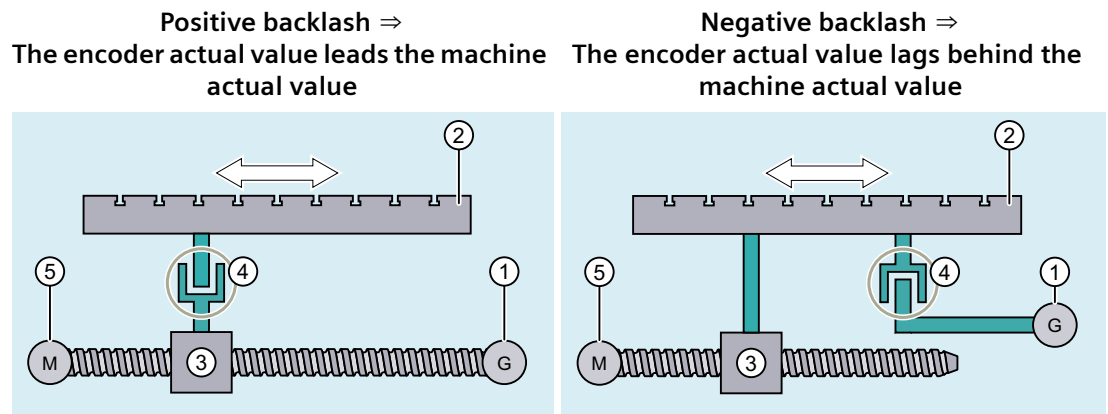
### 8.3.1 Mechanical backlash compensation

#### 8.3.1.1 Function

Mechanical backlash can occur in moving machine parts in the drive train (machine axes), e.g. at a ballscrew or in the connection to the measuring system.

##### Effects

For a machine axis with indirect measuring system, mechanical backlash results in a difference between the actual position of the NC determined using the measuring system and the actual position of the machine part. When the direction reverses, the machine axis traverses through a distance that is incorrect by the amount of the backlash:



The table does not move far enough because the encoder actual value has already changed (leading) due to the backlash while the table still remains stationary.

- ① Encoder
- ③ ballscrew
- ⑤ motor

The table moves too far because the encoder actual value has not yet changed (lagging) due to the backlash while the table is already moving.

- ② moved machine part (table)
- ④ mechanical backlash

##### Compensation

To compensate for the mechanical backlash, the actual value of the machine axis is corrected every time the axis reverses by the axis-specific compensation value set during commissioning (Page 222).

##### Effectiveness

The mechanical backlash compensation of a machine axis is active in all operating modes.  
Requirement:

- Incremental measuring system: Encoder state == "referenced"
- Absolute encoder: Encoder state == "synchronized"

**Displaying the compensation values**

The compensation values active at the actual position of the machine axis are displayed on the user interface for each individual axis.

SINUMERIK Operate: "Diagnostics" operating area > ETC key (">") > "Axis diagnostics" > "Service axis" >

- "Absolute compensation value measuring system 1"
- "Absolute compensation value measuring system 2"
- "Compensation value from LEC"

**8.3.1.2 Commissioning: Axis-specific machine data****Compensation value**

The compensation value of the mechanical backlash is entered in the machine data.

MD32450 \$MA\_BACKLASH[ <active measuring system> ] = <compensation value>

**Measurement**

- Traversing the machine axis and/or the machine part to any measurement position at a high velocity.
- Measuring the actual position of the machine part
- Calculating the backlash compensation value  $K_L$   
 $K_L = \text{"displayed actual position of the machine axis"} - \text{"measured actual position of the machine part"}$ 
  - $K_L > 0$  (**positive backlash**)  $\Rightarrow$  **positive** compensation value
  - $K_L < 0$  (**negative backlash**)  $\Rightarrow$  **negative** compensation value

**Checking**

To check the effect of the compensation, we recommend that the mechanical backlash is measured after the machine data has been activated.

**Circularity test**

The circularity test integrated in the user interface can be used to visualize the mechanical backlash.

SINUMERIK Operate: "Commissioning" operating area > "Optimization/Test" > "Circularity test"

**Second measuring system**

If an axis has a second measuring system, the compensation value must also be determined for this and entered in the machine data:

MD32450 \$MA\_BACKLASH[ <measuring system 2> ]

When the measuring system is switched over, the associated compensation value is automatically used.

### Evaluation factor dependent on the parameter set

For a mechanical backlash dependent on the parameter set, the parameter set-specific factor is entered into the following machine data, which is then applied to the compensation value of the backlash (MD32450 \$MA\_BACKLASH).

MD32452 \$MA\_BACKLASH\_FACTOR[ <parameter set 1 ... n> ] = <factor>

### Effective compensation value

The effective compensation value  $K_p$  of a parameter set is calculated as follows:

$$K_p = ( \text{MD32450 } \$MA\_BACKLASH[ \text{<active measuring system> } ] ) * ( \text{MD32452 } \$MA\_BACKLASH\_FACTOR[ \text{<parameter set> } ] )$$

### Backlash compensation mode

The response of mechanical backlash compensation of a machine axis is set using machine data:

MD32454 \$MA\_BACKLASH\_MODE, Bit 0 and 1:

| Bit | Meaning                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                   |
|-----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------|
| 0   | Restores the last effective backlash compensation after the control system has powered up                                                                                                                                                                                                                                                                                                                         |                                                                                                   |
|     | Val-ue                                                                                                                                                                                                                                                                                                                                                                                                            | Meaning                                                                                           |
|     | 0                                                                                                                                                                                                                                                                                                                                                                                                                 | The backlash compensation value is <b>not</b> restored (default setting)                          |
|     | 1                                                                                                                                                                                                                                                                                                                                                                                                                 | The backlash compensation is restored.                                                            |
|     | <b>Requirement</b><br>The measuring system active after the control powers up must be adjusted and synchronized: <ul style="list-style-type: none"> <li>• MD34210 \$MA_ENC_REFP_STATE[&lt;active measuring system&gt;] == 2</li> <li>• &lt;Axis&gt;.basic.in.enc1Synchronized == 1 (referenced, synchronized 1) or</li> <li>• &lt;Axis&gt;.basic.in.enc2Synchronized == 1 (referenced, synchronized 2)</li> </ul> |                                                                                                   |
| 1   | Effect of other compensation functions (temperature compensation, sag or angularity error compensation)                                                                                                                                                                                                                                                                                                           |                                                                                                   |
|     | Val-ue                                                                                                                                                                                                                                                                                                                                                                                                            | Meaning                                                                                           |
|     | 0                                                                                                                                                                                                                                                                                                                                                                                                                 | The backlash compensation responds to other compensatory motion (default setting).                |
|     | 1                                                                                                                                                                                                                                                                                                                                                                                                                 | The backlash compensation does <b>not</b> respond to other compensatory motion (default setting). |

### PLC signals

| Basic Program Plus               | Basic Program       |                    |
|----------------------------------|---------------------|--------------------|
| <Axis>.basic.in.enc1Synchronized | LBP_Axis*.E_RefSyn1 | DB31, ... .DBX60.4 |
| <Axis>.basic.in.enc2Synchronized | LBP_Axis*.E_RefSyn2 | DB31, ... .DBX60.5 |

## 8.3.2 Dynamic backlash compensation

### 8.3.2.1 Function

#### Dynamic backlash

A dynamic backlash can occur for machine types with sliding guides. Depending on the axial dynamic response (velocity, jerk, etc.) used to approach an end position, the machine slide reaches the programmed end position or stops earlier because of the static friction. The resulting position error is direction-symmetric.

#### Compensation

To compensate for the dynamic backlash, half the signed compensation value (MD32456Commissioning: Axis-specific machine data (Page 225), see "") is applied in accordance with the relevant traversing direction of the axis. Compensation value is applied as ramp.

#### Activation

The dynamic backlash compensation is activated by the PLC only in required the situations:

`<Axis>.basic.out.dynBacklashComp` (activate dynamic backlash compensation)

---

#### Note

The machine tool manufacturer specifies in the PLC user program the "required" situations for the activation of the dynamic backlash compensation. Such situations result with traversal of the axes with G1, in the JOG mode or with the handwheel.

---

The NC uses the following interface to return the required activation to the PLC:

`<Axis>.basic.in.dynBacklashCompActive` (dynamic backlash compensation active)

#### Requirement

The axes to be compensated must be homed.

#### Display

The compensation value that belongs to the current actual position is displayed in the "Diagnostics" operator area of the "Axis/Spindle Service" window as the total compensation calculated from LEC, mechanical and dynamic backlash compensation.

## PLC signals

| Basic Program Plus                    | Basic Program        |                     |
|---------------------------------------|----------------------|---------------------|
| <Axis>.basic.out.dynBacklashComp      | LBP_Axis*.A_DynBComp | DB31, ... .DBX25.0  |
| <Axis>.basic.in.dynBacklashCompActive | LBP_Axis*.E_DynBComp | DB31, ... .DBX102.0 |

### 8.3.2.2 Commissioning: Axis-specific machine data

#### Compensation value

As precondition to commission the dynamic backlash compensation, the mechanical backlash compensation must have already been commissioned (see Chapter "Commissioning: Axis-specific machine data (Page 222)"). To determine the compensation value for the dynamic backlash compensation, the measurement described there should be repeated with **low** traversing velocity. The compensation value determined in this way is entered in the machine data for the corresponding measuring system:

MD32456 \$MA\_BACKLASH\_DYN[ <active measuring system> ] = <compensation value for dynamic backlash compensation>

#### Maximum tolerance for position actual value switchover

You are provided the option of applying the backlash compensation value gradually in several increments when the relevant axis changes direction. This prevents a setpoint step change on the axes from causing corresponding errors. The contents of the axis-specific machine data determine the increment with which the backlash compensation value (MD32450) is applied:

MD36500 \$MA\_ENC\_CHANGE\_TOL (max. tolerance on actual position value switchover)

Please note that the backlash compensation is only fully calculated after <n> servo cycles (<n> = MD32450 / MD36500). An excessive time span can cause the triggering of standstill monitoring alarms. If MD36500 > MD32450, then the compensation is performed in one servo cycle.

#### Compensation value change

Using the machine data, the velocity with which the compensation value is traversed through is set as a percentage of the maximum axis velocity (MD32000 \$MA\_MAX\_AX\_VELO):

MD32457 \$MA\_BACKLASH\_DYN\_MAX\_VELO = <percentage of the maximum axis velocity>

### 8.3.3 Dual position feedback

For the "Dual position feedback" function, contrary to conventional (mechanical or dynamic) backlash compensation, two measuring systems, which are mechanically coupled through a gearbox without backlash, are used for the closed-loop position control. The advantages of a direct measuring system are combined with the advantages of an indirect measuring system:

- Direct measuring system: The closed-loop position control with direct, i.e. encoder on the load side, "automatically" corrects the backlash.
- Indirect measuring system: The closed-loop position control with indirect, i.e. encoder on the motor side, is "rugged" and "stable" regarding discontinuous backlash.

Each time the setpoint changes, initially only the encoder data from the indirect measuring system is used for rugged control without oscillation. With a parameterizable delay time, the closed-loop control smoothly transitions to monitor the direct measuring system, therefore, achieving the required accuracy on the load side. The control operations should, at this point in time, only involve short distances. This is because in the meantime the tooth flanks are in contact with one another, and the backlash has been moved through.

When using the "Dual position feedback" function, it is not necessary to measure and mathematically compensate the backlash.

#### Requirements

The following preconditions must be satisfied for an axis to be compensated:

- Direct and indirect measuring system, mechanically coupled:
  - MD30200 \$MA\_NUM\_ENCS = 2
  - MD31040 \$MA\_ENC\_IS\_DIRECT[ 0 ] == 0 or 1
  - MD31040 \$MA\_ENC\_IS\_DIRECT[ 1 ] == 1 or 0
- Measuring system calibration has been released:  
MD34102 \$MA\_REFP\_SYNC\_ENCS == 1
- Both measuring systems are referenced:
  - <Axis>.basic.in.enc1Synchronized (referenced/synchronized 1) == 1
  - <Axis>.basic.in.enc2Synchronized (referenced / synchronized 2) == 1

#### Application

Typical applications include main spindles on turning machines which feature a mechanical system without backlash between motor and main spindle (e.g. gear). Nevertheless, these axes require a direct measuring system for precision in C-axis operation (on the load side).

Due to mechanical factors such as backlash and rigidity, however, the axis cannot traverse with the same position control dynamic with encoder on the load side (in comparison to position control with motor encoder). The solution here is provided by the dual position feedback.

The servo gain factor does not have to be reduced and it can nevertheless be positioned on the direct encoder.

## PLC signals

| Basic Program Plus               | Basic Program       |                   |
|----------------------------------|---------------------|-------------------|
| <Axis>.basic.in.enc1Synchronized | LBP_Axis*.E_RefSyn1 | DB31, ... DBX60.4 |
| <Axis>.basic.in.enc2Synchronized | LBP_Axis*.E_RefSyn2 | DB31, ... DBX60.5 |

### 8.3.3.1 Commissioning: Axis-specific machine data

#### Delay time

The delay time, during which the closed-loop control smoothly transitions from the indirect to the direct measuring system, is entered in the machine data.

MD32960 \$MA\_POSCTRL\_DUAL\_FEEDBACK\_TIME = <delay time>

If the delay time is zero, the function is inactive and only the active measuring system is effective for position control:

- <Axis>.basic.out.enc1Activation (position measuring system 1) or
- <Axis>.basic.out.enc2Activation (position measuring system 2)

#### Note

After activating the "Dual position feedback" function, all of the existing measuring system compensations and monitoring functions remain unchanged and active; this means that they may have to be deactivated by the user (e.g. backlash compensation values deleted).

## PLC signals

| Basic Program Plus              | Basic Program        |                   |
|---------------------------------|----------------------|-------------------|
| <Axis>.basic.out.enc1Activation | LBP_Axis*.A_PosMeas1 | DB31, ... .DBX1.5 |
| <Axis>.basic.out.enc2Activation | LBP_Axis*.A_PosMeas2 | DB31, ... .DBX1.6 |

### 8.3.3.2 Supplementary conditions

#### Referencing and flying measurement

Also when the "Dual position feedback" function is active, the "Referencing" and "Measuring" functions refer to the active measuring system:

- <Axis>.basic.out.enc1Activation == 1 (position measuring system 1) or
- <Axis>.basic.out.enc2Activation == 1 (position measuring system 2)

#### More information

- Function Manual Axes and Spindles; Referencing
- Function Manual Technologies, Measuring

## PLC signals

| Basic Program Plus              | Basic Program        |                   |
|---------------------------------|----------------------|-------------------|
| <Axis>.basic.out.enc1Activation | LBP_Axis*.A_PosMeas1 | DB31, ... .DBX1.5 |
| <Axis>.basic.out.enc2Activation | LBP_Axis*.A_PosMeas2 | DB31, ... .DBX1.6 |

## 8.4 Nodding compensation

### 8.4.1 Description of the function

#### 8.4.1.1 Options

The "Nodding compensation" function is an option subject to license. The following versions are available:

- **Nodding compensation ECO:** 6FC5800-0AS20-0YB0  
Nodding compensation in **one** machine axis with **one** accelerating axis:
- **Nodding compensation ADVANCED:** 6FC5800-0AS21-0YB0  
Nodding compensation in **any arbitrary number** of machine axes

#### 8.4.1.2 Features

Nodding compensation is used to compensate dynamic position deviations in a machine axis due to this or another machine axis accelerating. The position deviations are obtained as a result of the mechanical compliance within the machine.

This is demonstrated in the following schematic of a 3-axis traveling column machine, as example. If the Y axis is accelerated with  $a_y$ , as a result of the mechanical compliance in the machine, position deviations  $\Delta_z$  occur in the Z axis.

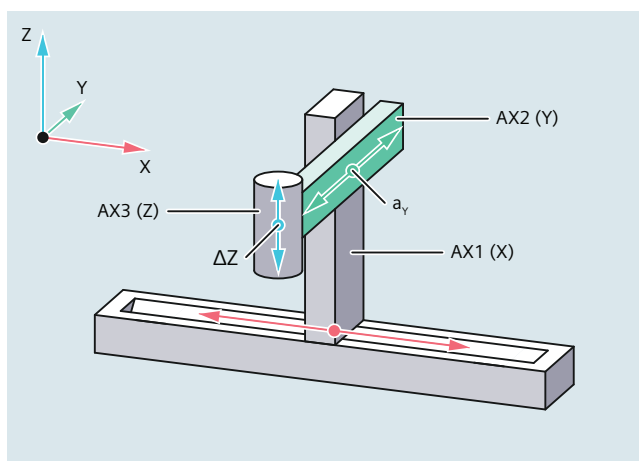


Figure 8-4 3-axis traveling column machine

**Definition of terms**

- Nodding compensation axes  
The machine axes involved in the nodding compensation are designated as follows: // // // //
  - **Compensating axis**  
A machine axis, resulting from other machine axes accelerating, results in position deviations, which can be compensated by nodding compensation. In the diagram: Compensating axis  $\Rightarrow$  Z axis
  - **Accelerating axis**  
Machine axis, whose acceleration as a result of compliance in the machine, results in a position deviation in the compensating axis. In the diagram: Accelerating axis  $\Rightarrow$  Y axis
  - **Adaptation axis**  
Machine axis, whose position when traversing the accelerating axis influences the magnitude of the position deviation in the compensating axis. In the diagram: Adaptation axis  $\Rightarrow$  X axis
- Compliance factor  
Compliance factor N defines the magnitude of the position deviation  $\Delta$  of the compensating axis for traversing motion of the accelerating axis with acceleration a. Compliance factor N is the quotient of position deviation  $\Delta$  of the compensating axis and the acceleration a of the accelerating axis that actually causes the deviation:  $N = \Delta / a$

---

**Note****Further information**

Determining the compliance factor is described in detail in

- Commissioning Manual, CNC Commissioning: NC, PLC, drive; Commissioning NC > Nodding compensation
- Online help  
The online help for nodding compensation is called from the SINUMERIK Operate user interface via: Commissioning > NC > Nodding compensation > Help button on the operator panel front

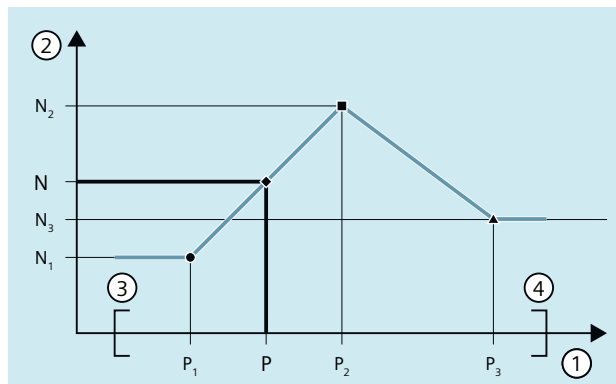
- 
- Compensation value  
The actual compensation value  $\Delta$  is obtained by multiplying the actual compliance factor N with the actual acceleration a of the axis causing the deviation

**Boundary conditions**

- **Rotary axes**  
It is not permissible for the compensating, the accelerating or the adaptation axis to be a rotary axis.
- **Differences to the compile cycle "RMCC/NOCO nodding compensation"**  
(6FC5800-0AN63-0YB0)
  - The functionality of system function "Nodding compensation" is, contrary to compile cycle "RMCC/NOCO", exclusively limited to compensating nodding motion. The functions provided beyond this in the compile cycle, such as compensating centrifugal forces, taking into account rotary axis positions as well as comprehensive position tables, are not available with the system function.
  - Either system function "Nodding compensation" can be used – or compile cycle "RMCC/NOCO", but not together.
  - It is not possible to use the parameterizing data of compile cycle "RMCC/NOCO" to parameterize the "Nodding compensation" system function.

**8.4.1.3 Position-dependent compliance factor**

If the machine compliance depends on the position of another machine axis (adaptation axis), then up to three different positions of the adaptation axis can be selected as interpolation points of the adaptation characteristic; the effective compliance factors can then be determined from the characteristic. Positions (P1, P2, P3) and the compliance factors effective there (N1, N2, N3) must be saved in the machine data of the compensating axis. When traversing the accelerating axis, the effective compliance factor is calculated depending on the position of the adaptation axis by linearly interpolating between the interpolation points.



- ① Traversing range of the adaptation axis with  $P_1 < P_2 < P_3$
- ② Compliance factor of the compensating axis
- ③ Traversing range limit -
- ④ Traversing range limit +

Figure 8-5 Compensation characteristic

| Actual position: P                                   | Compliance factor: N                                            |
|------------------------------------------------------|-----------------------------------------------------------------|
| P less than $P_1$ up to traversing range limit minus | $N = N_1$                                                       |
| P between $P_1$ and $P_2$                            | $N = \text{linear interpolation between } N_1 \text{ and } N_2$ |

| Actual position: P                                                       | Compliance factor: N                                            |
|--------------------------------------------------------------------------|-----------------------------------------------------------------|
| P equal to $P_2$                                                         | $N = N_2$                                                       |
| P between $P_2$ and $P_3$                                                | $N = \text{linear interpolation between } N_2 \text{ and } N_3$ |
| P greater than or equal to $P_3$ - up to the traversing range limit plus | $N = N_3$                                                       |

#### 8.4.1.4 Compensation relationships

A compensation relationship defines the following dependencies of the compensation value:

- On the acceleration of the accelerating axis
- On the position of the adaptation axis
- On the machine compliance

Up to three compensation relationships can be parameterized in the machine data of a compensating axis. The following data can be set for each compensation relationship:

- Accelerating axis
- Adaptation axis
- Number of positions of the adaptation axis
- Positioning the adaptation axis
- Compliance factors at the positions

##### Accelerating axes or adaptation axes

Any **linear** machine axis of the NC can be used as accelerating axis or adaptation axis. Even the compensating axis itself.

##### Total compensation value

The total compensation value acting on the compensating axis is the sum of the compensation values of the individual compensation relationships.

## 8.4.2 Commissioning: Machine data

### 8.4.2.1 Overview

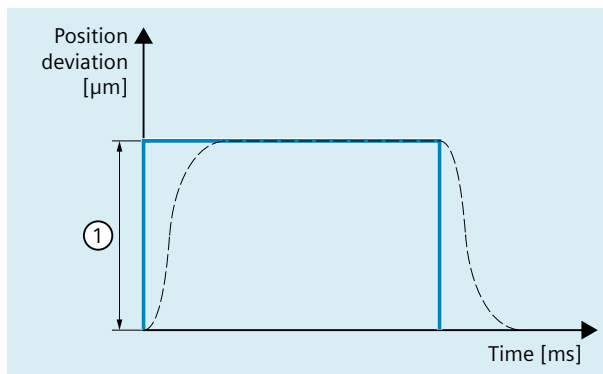
The following table provides an overview with which machine data the nodding compensation is parameterized.

| Number                             | Name: \$MA_      | Meaning                                                            |
|------------------------------------|------------------|--------------------------------------------------------------------|
| MD37302                            | NOCO_FILTER_TIME | Time constant for smoothing nodding compensation values (Page 232) |
| <b>Compensation relationship 1</b> |                  |                                                                    |
| MD37310                            | NOCO_INPUT_AX_1  | Machine axis that causes nodding motion (Page 233)                 |
| MD37312                            | NOCO_ADAPT_AX_1  | Machine axis, whose position influences nodding motion (Page 234)  |

| Number                             | Name: \$MA_       | Meaning                                                                               |
|------------------------------------|-------------------|---------------------------------------------------------------------------------------|
| MD37314                            | NOCO_ADAPT_NUM_1  | Number of positions, adaptation characteristic of the nodding compensation (Page 235) |
| MD37316                            | NOCO_ADAPT_POS_1  | Positions, adaptation characteristic of the nodding compensation (Page 236)           |
| MD37318                            | NOCO_COMPLIANCE_1 | Compliance factors for nodding compensation (Page 237)                                |
| <b>Compensation relationship 2</b> |                   |                                                                                       |
| MD37320                            | NOCO_INPUT_AX_2   | See MD37310                                                                           |
| MD37322                            | NOCO_ADAPT_AX_2   | See MD37312                                                                           |
| MD37324                            | NOCO_ADAPT_NUM_2  | See MD37314                                                                           |
| MD37326                            | NOCO_ADAPT_POS_2  | See MD37316                                                                           |
| MD37328                            | NOCO_COMPLIANCE_2 | See MD37318                                                                           |
| <b>Compensation relationship 3</b> |                   |                                                                                       |
| MD37330                            | NOCO_INPUT_AX_3   | See MD37310                                                                           |
| MD37332                            | NOCO_ADAPT_AX_3   | See MD37312                                                                           |
| MD37334                            | NOCO_ADAPT_NUM_3  | See MD37314                                                                           |
| MD37336                            | NOCO_ADAPT_POS_3  | See MD37316                                                                           |
| MD37338                            | NOCO_COMPLIANCE_3 | See MD37318                                                                           |

#### 8.4.2.2 Time constant for smoothing nodding compensation values

Using the machine data, the time constant is set with which the total nodding compensation value is smoothed when it is entered.



- ① Position deviation
- Blue: Setpoint
  - Red: Actual value

---

MD37302 \$MA\_NOCO\_FILTER\_TIME[<comp. axis>] = <time constant>

---

#### Note

##### Deactivating smoothing

Smoothing is deactivated - or is not active - if the time constant is set to less than 10 % of the position controller clock cycle:

Time constant < 0,1 \* MD10061 \$MN\_POSCTRL\_CYCLE\_TIME

---

### 8.4.2.3

#### Accelerating axis

The number of the machine axis, whose acceleration causes position deviations in the compensating axis, must be entered in the machine data.

The number of the accelerating axis refers to the machine axes of the NC parameterized in machine data MD10002 \$MN\_AXCONF\_LOGIC\_MACHAX\_TAB. The machine axis number is the corresponding machine data index + 1.

MD37310 \$MA\_NOCO\_INPUT\_AX\_1[<comp. Achse>] = <machine axis number>

#### Example

As a result of the machine compliance, when the Y axis accelerates, a position deviation occurs in the Z axis.

- Compensating axis: AX3 (Z axis)
- Accelerating axis: AX2 (Y axis)

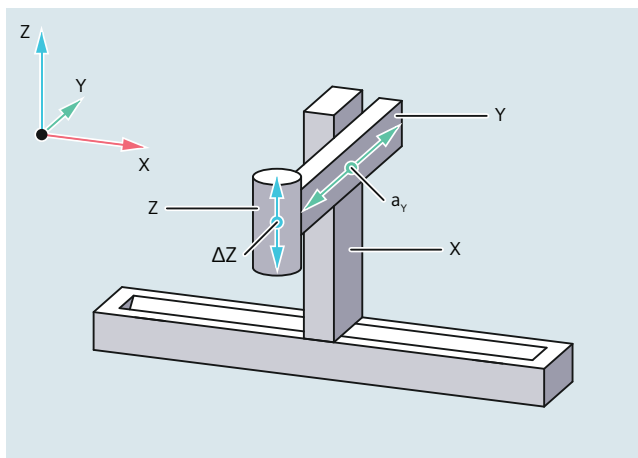


Figure 8-6 Accelerating axis AX2 (Y axis)

```
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[0] = "AX1" ; X axis
```

```
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[1] = "AX2" ; Y axis
```

```
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[2] = "AX3" ; Z axis
```

```
MD37310 $MA_NOCO_INPUT_AX_1[AX3] = 2 ; accelerating axis = Y axis
```

#### 8.4.2.4 Adaptation axis

The number of the machine axis, whose actual position influences the position deviation in the compensating axis, must be entered in the machine data.

The machine axis number refers to the machine axes of the NC parameterized in machine data MD10002 \$MN\_AXCONF\_LOGIC\_MACHAX\_TAB. The machine axis number is the corresponding machine data index + 1.

MD37312 \$MA\_NOCO\_ADAPT\_AX\_1[<comp. Achse>] = <machine axis number>

#### Note

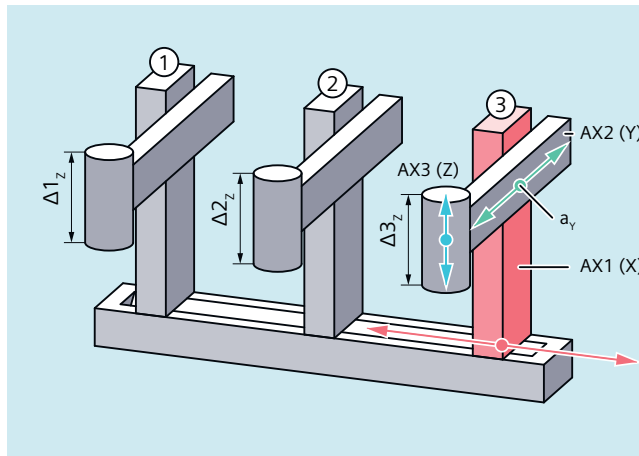
##### Activating the position-dependent adaptation

The position-dependent adaptation only becomes active if one adaptation axis and more than one position (MD37314 \$MA\_NOCO\_ADAPT\_NUM\_1 (Page 235)) are parameterized.

#### Example

As a result of the machine compliance, when the Y axis accelerates, a position deviation occurs in the Z axis. Further, the position deviation depends on the position of the X axis.

- Compensating axis: AX3 (Z axis)
- Accelerating axis: AX2 (Y axis)
- Adaptation axis: AX1 (X axis)



①, ②, ③ Axis positions of the adaptation axis AX1 (X axis)

Figure 8-7 Adaptation axis AX1 (X axis)

```
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[0] = "AX1" ; X axis
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[1] = "AX2" ; Y axis
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[2] = "AX3" ; Z axis
```

```
MD37312 $MA_NOCO_ADAPT_AX_1[AX3] = 1 ; adaptation axis = X axis
```

### 8.4.2.5 Number of positions of the adaptation characteristic

The number of axis positions, for which another compliance is effective, must be entered in the machine data for the position-dependent adaptation.

MD37314 \$MA\_NOCO\_ADAPT\_NUM\_1[<comp. Axis>] = <number>

#### Note

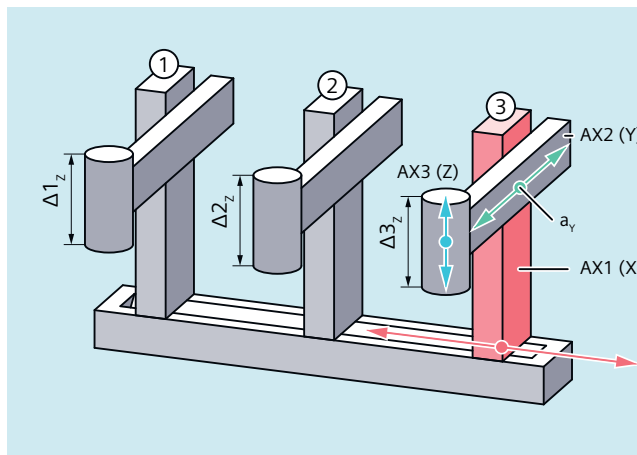
#### Activating the position-dependent adaptation

The position-dependent adaptation only becomes active if more than one position and one adaptation axis (MD37312 \$MA\_NOCO\_ADAPT\_AX\_1 (Page 234)) are parameterized.

#### Example

Interpolation points of the adaptation characteristic are to be parameterized at three positions along the traversing range of the adaptation axis (X axis).

- Compensating axis: AX3 (Z axis)
- Accelerating axis: AX2 (Y axis)
- Adaptation axis: AX1 (X axis)



①, ②, ③

Axis positions of the adaptation axis AX1 (X axis)

Figure 8-8 Number of axis positions of the adaptation axis AX1 (X axis)

```
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[0] = "AX1" ; X axis
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[1] = "AX2" ; Y axis
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[2] = "AX3" ; Z axis
```

```
MD37314 $MA_NOCO_ADAPT_NUM_1[AX3] = 3 ; number of axis positions
```

#### See also:

- Axis positions: MD37316 \$MA\_NOCO\_ADAPT\_POS\_1 (Page 236)
- Compliance factors: MD37318 \$MA\_NOCO\_COMPLIANCE\_1 (Page 237)

### 8.4.2.6 Positions of the adaptation characteristic

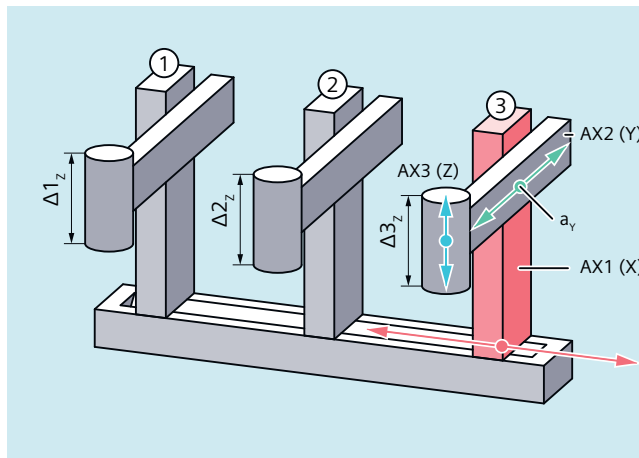
The positions of the adaptation axis, at which a dedicated compliance factor is parameterized, must be entered in the machine data for the interpolation points of the adaptation characteristic:

MD37316 \$MA\_NOCO\_ADAPT\_POS\_1[<index>, <comp. Axis>] = <position>

#### Example

Interpolation points are parameterized at positions -100.0, 0.0, and 120.0 of adaptation axis AX1 (X axis) for the adaptation characteristic.

- Compensating axis: AX3 (Z axis)
- Accelerating axis: AX2 (Y axis)
- Adaptation axis: AX1 (X axis)



- ① Position 1: -100.0
- ② Position 2: 0.0
- ③ Position 3: 120.0

Figure 8-9 Axis positions of the adaptation axis AX1 (X axis)

```
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[0] = "AX1" ; X axis
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[1] = "AX2" ; Y axis
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[2] = "AX3" ; Z axis
```

```
MD37316 $MA_NOCO_ADAPT_POS_1[0, AX3] = -100.0
MD37316 $MA_NOCO_ADAPT_POS_1[1, AX3] = 0.0
MD37316 $MA_NOCO_ADAPT_POS_1[2, AX3] = 120.0
```

#### See also:

- Number of axis positions: MD37314 \$MA\_NOCO\_ADAPT\_NUM\_1 (Page 235)
- Compliance factors: MD37318 \$MA\_NOCO\_COMPLIANCE\_1 (Page 237)

### 8.4.2.7 Compliance factors

The compliance factors associated with the particular positions of the adaptation characteristic (MD37316 \$MA\_NOCO\_ADAPT\_POS\_1 (Page 236)) must be entered in the machine data.

MD37318 \$MA\_NOCO\_COMPLIANCE\_1[<index>, <comp. Achse>] = <compliance factor>

#### Without position-dependent adaptation

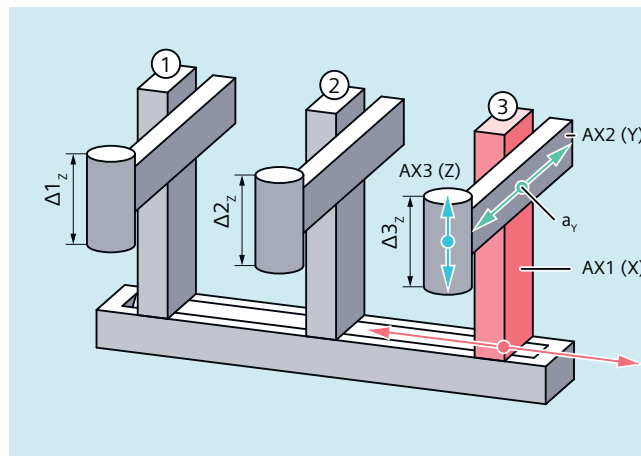
If no position-dependent adaptation is parameterized (MD37312 \$MA\_NOCO\_ADAPT\_AX\_1 (Page 234) == 0), then the compliance factor must be entered into index 0:

MD37318 \$MA\_NOCO\_COMPLIANCE\_1[ 0, <comp. Achse>] = <compliance factor>

#### Example

At positions -100.0, 0.0, and 120.0 of the traversing range of the adaptation axis (X axis), for the compensating axis AX3 (Z axis) compliance factors -12 , -25 and -10  $\mu\text{m} / \text{m/s}^2$  are determined.

- Compensating axis: AX3 (Z axis)
- Accelerating axis: AX2 (Y axis)
- Adaptation axis: AX1 (X axis)



- ① Compliance factor at position 1: -12  $\mu\text{m} / \text{m/s}^2$
- ② Compliance factor at position 2: -25  $\mu\text{m} / \text{m/s}^2$
- ③ Compliance factor at position 3: -10  $\mu\text{m} / \text{m/s}^2$

Figure 8-10 Adaptation axis: AX1 (X axis)

```
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[0] = "AX1" ; X axis
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[1] = "AX2" ; Y axis
MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[2] = "AX3" ; Z axis
```

```
MD37318 $MA_NOCO_COMPLIANCE_1[0, AX3] = -0.000012
MD37318 $MA_NOCO_COMPLIANCE_1[1, AX3] = -0.000025
MD37318 $MA_NOCO_COMPLIANCE_1[2, AX3] = -0.000010
```

#### 8.4 Nodding compensation

**See also:**

- Number of axis positions: MD37314 \$MA\_NOCO\_ADAPT\_NUM\_1 (Page 235)

### 8.4.2.8 Function diagram

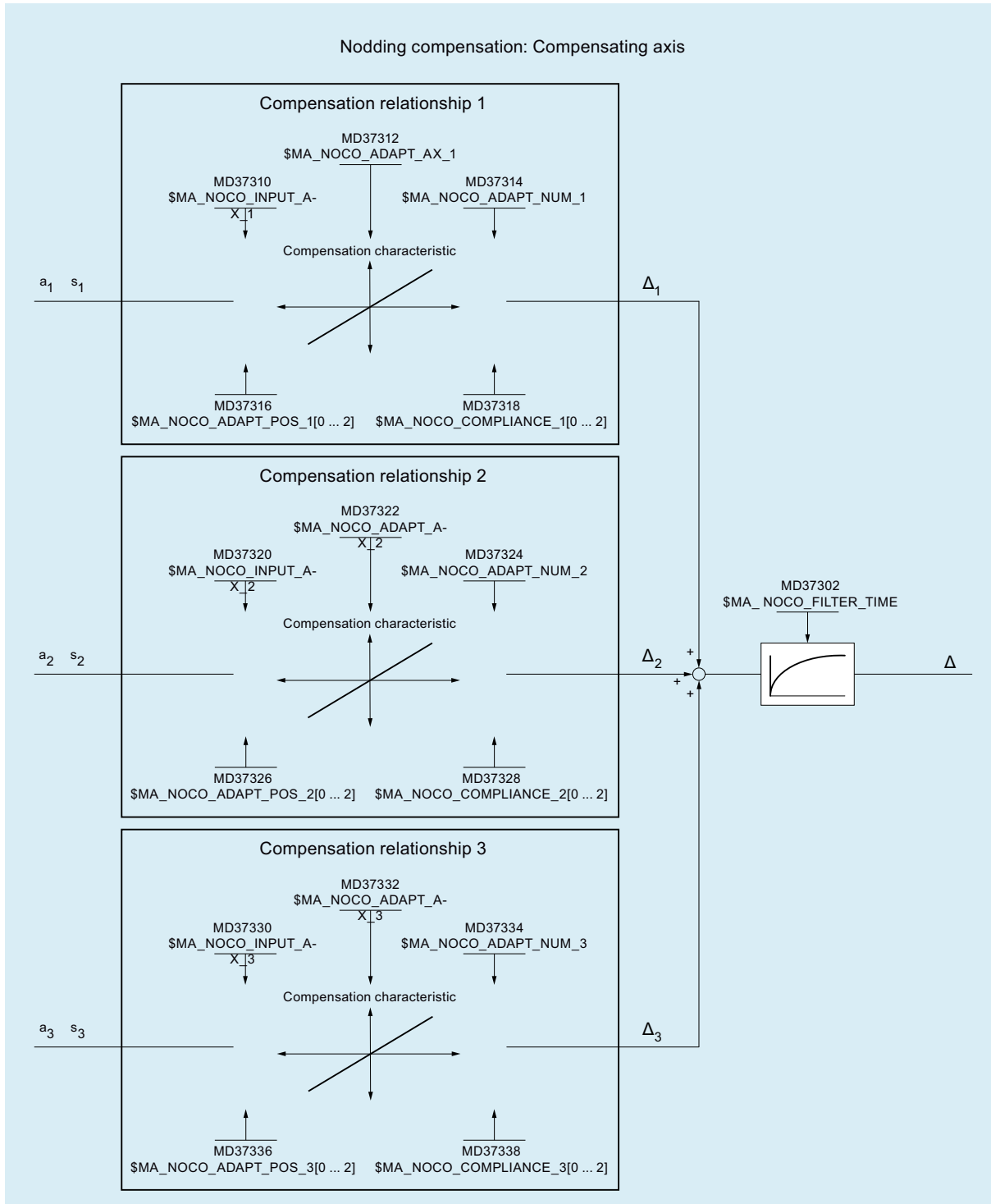


Figure 8-11 Function diagram of a compensating axis

## 8.5 Interpolatory compensation (IC) General properties

### Function

With "interpolating compensation," deviations between the desired and the actual position of an axis can be compensated by leadscrew, measuring system, sag and angularity errors. This is done by determining the deviation by measurement and storing the corresponding compensation values in one or more compensation tables in the NC. During operation, the relevant compensation value is then determined from the compensation table or tables for a compensation axis based on its current set position. Linear interpolation is performed between the interpolation points of the compensation tables.

Within "interpolating compensation", a distinction is made between the two following compensation methods:

- Compensation of leadscrew errors and measuring system errors
- Compensation of sag and angularity errors

### Terms

- Compensation value  
Additional setpoint that is applied to the compensation axis so that the difference between the desired and actual axis position is zero.
- Basic axis  
Axis, via whose set and actual position, the compensation value is determined from the compensation value.
- Compensation axis  
Axis, to whose set or actual position the compensation value is applied.
- Interpolation point  
Value pair in a compensation table, consisting of a set and actual position of the basic axis (interpolation point) and the associated compensation value (interpolation value).
- Compensation table  
Defined number of interpolation points for a defined traversing range of the basic axis.
- Compensation relation  
Unit comprising the basic axis, compensation axis, and compensation table.

### Entering compensation tables

The size of the compensation table, i.e. the number of interpolation points, must first be defined in a machine data. After the next POWER ON, the compensation tables are generated by the NC and preassigned a value of "0".

The compensation values and additional table parameters are entered in the compensation tables using special system variables. Data can be loaded in two different ways:

- By starting an NC program with the parameter values.
- By transferring the compensation tables from an external computer to the control.

---

#### Note

The respective compensation tables can only be loaded if the corresponding compensation function is **not** active for **all** axes:

- MD32700 \$MA\_ENC\_COMP\_ENABLE[<axis>] == 0
  - MD32710 \$MA\_CEC\_ENABLE[<axis>] == 0
- 

The compensation data is also retained when the control system is switched off.

---

#### Note

When changing machine data:

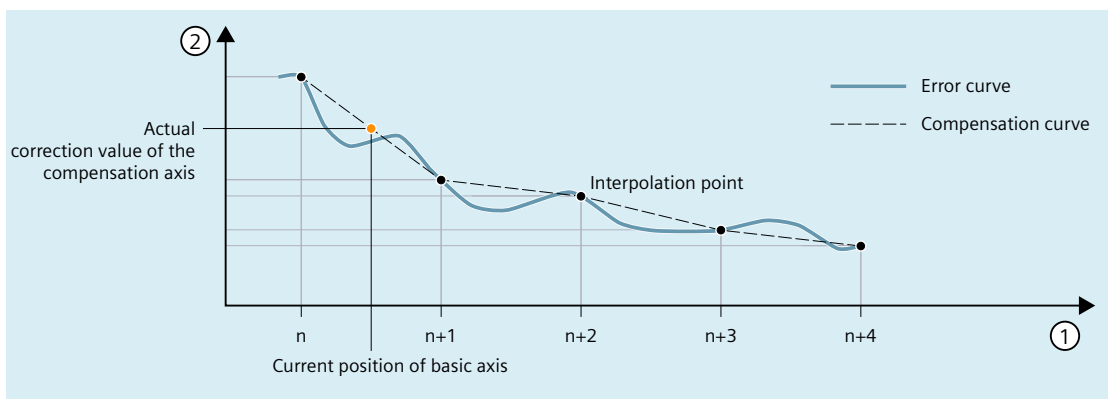
- MD18342 \$MN\_MM\_CEC\_MAX\_POINTS
- MD38000 \$MA\_MM\_ENC\_COMP\_MAX\_POINTS

the static user memory is formatted the next time the system is booted (see Function Manual "Basic Functions", Chapter "S7: Memory configuration").

---

### Interim value calculation

The traversing distance defined via the start and end position within which compensation is to apply, is divided into several sub-paths of equal size. The number of sub-paths is defined depending on the error curve and the desired precision. The positions that limit these sub-paths are referred to as interpolation points below. An interpolation and a compensation value must be assigned to each interpolation point. Between two interpolation points, the effective compensation value is determined by **linear interpolation**.



- ① Position of basic axis  
 ② Compensation values of the compensation axis

Figure 8-12 Intermediate value calculation by linear interpolation

## Constraints

### Compensation value at reference point

It is recommended that a compensation table be structured in such a way that the compensation value has the value "0" at the reference point of the axis.

## 8.6 IC: Leadscrew error and measuring system error compensation

### 8.6.1 Function

#### Leadscrew and measuring system errors

The measuring principle of "indirect measurement" on NC-controlled machines is based on the assumption that the lead of the ball screw is constant at any point within the traversing range, so that the actual position of the axis can be derived from the position of the drive spindle (ideal case). However, manufacturing tolerances result in dimensional deviations of varying degrees of severity on spindles (so-called leadscrew errors).

Added to this are the dimensional deviations (differences in reference division) caused by the measuring system as well as its mounting on the machine (so-called measuring system errors), plus any machine-dependent error sources.

#### Compensation

With "measuring system error compensation" (referred to below as **MSEC**), the base and compensation axes are always identical. It is therefore an **axial compensation** for which a definition of the base axis and compensation axis in the compensation table is not necessary.

---

#### Note

The leadscrew error compensation (**LEC**) is part of the measuring system error compensation.

---

The principle of the MSEC is to modify the axis-specific position actual value by the assigned compensation value in the interpolator clock cycle and to apply this value to the machine axis for immediate traversal. A positive compensation value causes the corresponding machine axis to move in the negative direction.

The magnitude of the compensation value is not limited and is not monitored. In order to avoid impermissibly high velocities and accelerations caused by compensation, small compensation values must be selected. Large compensation values can cause other axis monitoring functions to output alarms (e.g. contour monitoring, speed setpoint limitation).

If the axis to be compensated has a 2nd position measuring system, a separate compensation table must be created and activated for each measuring system. The correct table is automatically used when switching between measuring systems.

**Preconditions / activation**

The MSEC is only active until the following pre-conditions:

- The compensation values are stored in the static user memory and are active (after POWER ON).
- The function has been activated for the relevant machine axis:  
MD32700 \$MA\_ENC\_COMP\_ENABLE[ 0 / 1] = 1:
  - **0:** Measuring system 1
  - **1:** Measuring system 2
- The axis has been referenced:
  - `<Axis>.basic.in.enc1Synchronized` (referenced/synchronized 1) == 1
  - `<Axis>.basic.in.enc2Synchronized` (referenced / synchronized 2) == 1

As soon as these conditions have been fulfilled, the axis-specific actual value is modified by the compensation value in all modes and traversed by the machine axis immediately.

If the reference is then lost, e.g. because the encoder frequency has been exceeded (`<Axis>.basic.in.enc1Synchronized` or `.enc2Synchronized == 0`), compensation processing is deactivated.

**PLC signals**

| Basic Program Plus                                  | Basic Program                    |                    |
|-----------------------------------------------------|----------------------------------|--------------------|
| <code>&lt;Axis&gt;.basic.in.enc1Synchronized</code> | <code>LBP_Axis*.E_RefSyn1</code> | DB31, ... .DBX60.4 |
| <code>&lt;Axis&gt;.basic.in.enc2Synchronized</code> | <code>LBP_Axis*.E_RefSyn2</code> | DB31, ... .DBX60.5 |

**8.6.2 Commissioning****Machine data****Number of compensation interpolation points**

For each machine axis as well as for each measuring system (if a 2nd measuring system is available), the number of reserved interpolation points must be specified for the compensation table:

MD38000 \$MA\_MM\_ENC\_COMP\_MAX\_POINTS[<e>,<AXi>]

- <e>: Position measuring system
- <AXi>: axis

The number of interpolation points of a compensation table is calculated from the accompanying system variables (see below):

- End position: `$AA_ENC_COMP_MAX[...]`
- Start position: `$AA_ENC_COMP_MIN[...]`
- Distance between interpolation points: `$AA_ENC_COMP_STEP[...]`

Number of interpolation points = ((end position - start position) / interpolation point distance) + 1

## System variables

For each machine axis as well as for each measuring system (if a 2nd measuring system exists), the position-related compensation values as well as additional table parameters should be saved for every compensation relationship in the form of system variables:

- **\$AA\_ENC\_COMP\_MIN[<e>,<AXi>] (initial position)**  
 The initial position is the axis position at which the compensation table for the relevant axis begins ( $\hat{=}$  interpolation point 0).  
 The compensation value for the initial position is \$AA\_ENC\_COMP[<e>,0,<AXi>].  
 The compensation value of interpolation point 0 is used for all positions smaller than the initial position (does not apply for tables with modulo function).
- **\$AA\_ENC\_COMP\_MAX[<e>,<AXi>] (end position)**  
 The end position is the axis position at which the compensation table for the relevant axis ends ( $\hat{=}$  interpolation point <k>).  
 The compensation value for the end position is \$AA\_ENC\_COMP[<e>,<k>,<AXi>].  
 The compensation value of interpolation point is used for all positions larger than the end position (exception for table with modulo function).  
 The following supplementary conditions apply to interpolation point <k>:
  - for k = MD38000 - 1:  
 The compensation table is fully utilized!
  - for k < MD38000 - 1:  
 The compensation table is not fully utilized. Compensation values entered in the table that are greater than k are ignored.
  - for k > MD38000 - 1:  
 The compensation table is limited by a control function which reduces the end position. Compensation values that are greater than k are ignored.
- **\$AA\_ENC\_COMP\_STEP[<e>,<AXi>] (distance between interpolation points)**  
 The distance between interpolation points defines the distance between the compensation values in the relevant compensation table.

- **\$AA\_ENC\_COMP[<e>,<N>,<AXi>] (correction value for interpolation point N of the compensation table)**  
 <N> = interpolation point (axis position)  
 For every individual interpolation point the compensation value must be entered in the table.  
 <N> is limited by the maximum number of interpolation points of the particular compensation table (MD38000 \$MA\_MM\_ENC\_COMP\_MAX\_POINTS):  
 $0 \leq N \leq \text{MD38000} - 1$   
 The size of the compensation value is not limited.

**Note**

The first and last compensation values remain active over the entire traversing range; i.e. these values should be set to "0" if the compensation table does not cover the entire traversing range.

- **\$AA\_ENC\_COMP\_IS\_MODULO[<e>,<AXi>] (compensation with modulo function)**  
 System variable to activate/deactivate the compensation with modulo function:
  - \$AA\_ENC\_COMP\_IS\_MODULO[<e>,<AXi>] = 0: Compensation **without** modulo function
  - \$AA\_ENC\_COMP\_IS\_MODULO[<e>,<AXi>] = 1: Compensation **with** modulo function

When compensation with modulo function is activated, the compensation table is repeated cyclically, i.e. the compensation value at position \$AA\_ENC\_COMP\_MAX ( $\hat{=}$  interpolation point \$AA\_ENC\_COMP[<e>,<k>,<AXi>]) is immediately followed by the compensation value at position \$AA\_ENC\_COMP\_MIN ( $\hat{=}$  interpolation point \$AA\_ENC\_COMP[<e>,<0>,<AXi>]). For rotary axes with modulo 360° degrees it is therefore suitable to program 0° (\$AA\_ENC\_COMP\_MIN) as the initial position and 360° (\$AA\_ENC\_COMP\_MAX) as the end position.  
 The compensation values entered for these two positions should be the same as otherwise the compensation value jumps from MAX to MIN at the transition point and vice versa.

 **CAUTION**
**Wrong correction values**

When writing the correction values for a correction table, it must be ensured that all interpolation points within the parameterized range are assigned a value. Correction values which are not described otherwise contain random values.

**Note**

Table parameters containing position information are automatically converted when the system of units is changed (change from MD10240 \$MN\_SCALING\_SYSTEM\_IS\_METRIC).

The position information is always interpreted in the current measuring system. Conversion must be implemented externally.

Automatic conversion of the position data can be configured as follows:

MD10260 \$MN\_CONVERT\_SCALING\_SYSTEM = 1

External conversion is no longer necessary.

**Further information**

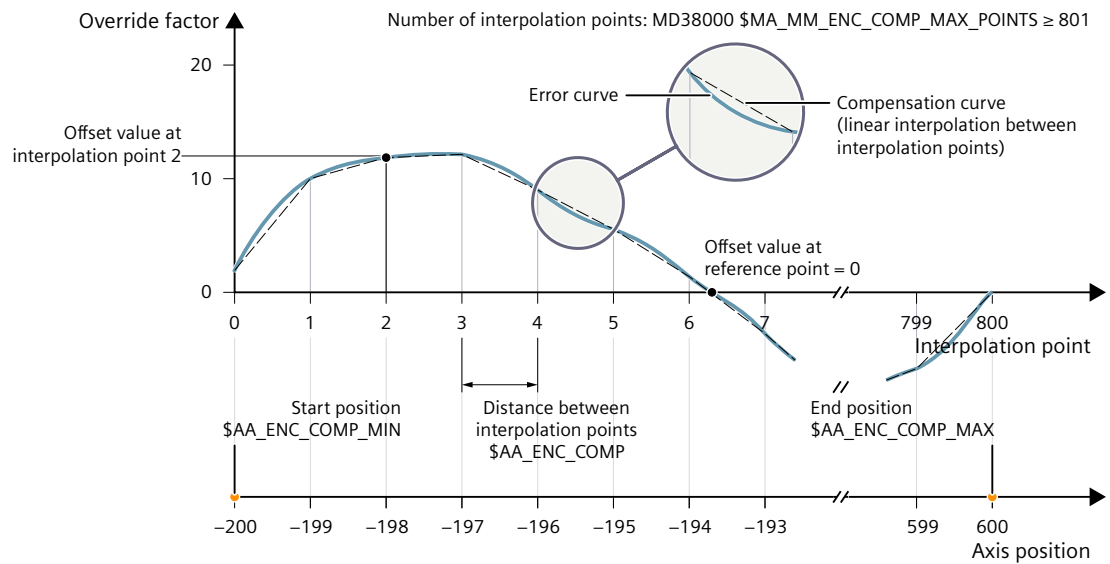
Function Manual Basic Functions; Velocities, setpoint/actual value systems, closed-loop control

### 8.6.3 Example

Sample parameterization of a compensation table:

- Machine axis: X1
- Measuring system: 1
- Starting position: -200 mm
- End position: 600 mm
- Distance between interpolation points: 1 mm
- Number of interpolation points: MD38000 \$MA\_MM\_ENC\_COMP\_MAX\_POINTS =  $((600 - -200) / 1) + 1 = 801$

The memory requirement in the static user memory is:  $801 * 8 \text{ Byte} = 6408 \text{ Byte}$



#### Program for writing system variables

| Program code                   | Comment                                                |
|--------------------------------|--------------------------------------------------------|
| %_N_AX_EEC_INI                 |                                                        |
| CHANDATA(1)                    |                                                        |
| \$AA_ENC_COMP[0,0,X1]=0.003    | 1st compensation value (interpolation point 0): +3μm   |
| \$AA_ENC_COMP[0,1,X1]=0.01     | 2nd compensation value (interpolation point 1): +10μm  |
| \$AA_ENC_COMP[0,2,X1]=0.012    | 3rd compensation value (interpolation point 2): +12μm  |
| ...                            |                                                        |
| \$AA_ENC_COMP[0,800,X1]=-0.0   | Last compensation value (interpolation point 800): 0μm |
| \$AA_ENC_COMP_STEP[0,X1]=1.0   | Distance between interpolation points 1.0 mm           |
| \$AA_ENC_COMP_MIN[0,X1]=-200.0 | Compensation starts at -200.0 mm                       |
| \$AA_ENC_COMP_MAX[0,X1]=600.0  | Compensation ends at +600.0 mm                         |

| Program code                    | Comment                              |
|---------------------------------|--------------------------------------|
| \$AA_ENC_COMP_IS_MODULO[0,X1]=0 | Compensation without modulo function |
| M17                             |                                      |

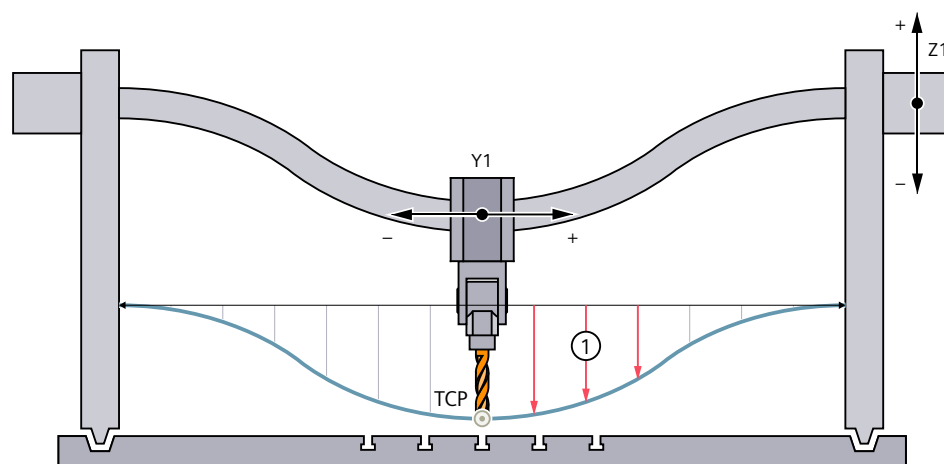
## 8.7 IC: Sag and angularity error compensation

### 8.7.1 General information

#### Function

The sag and angularity error compensation (cross error compensation, CEC) is an axis-specific compensation, where a correction and/or compensation value is added to the the setpoint position of the **compensation axis**. The compensation value is derived from the current setpoint position of one or several **basic axes**.

- **Sag errors**  
In the case of sag errors, the weight of a machine part or workpiece leads to a position-dependent error in the tool center point (TCP).
- **Angularity errors**  
For angularity errors, a deviation in the ideal angle in the geometric axes in relation to each other leads to a position-dependent error in the tool center point (TCP).



① Position error in Z1 as a function of the position of Y1

Figure 8-13 Example: Sag errors

#### Error compensation

For error compensation, at various interpolation points (setpoint positions) of the basic axis (Y1), the particular position error should be determined and entered into the compensation table. When the basic axis (Y1) traverses, in the interpolation clock cycle, the control system calculates the actual compensation value for the compensation axis (Z1) by linearly interpolating between the points. The compensation value is added to the setpoint of the

compensation axis (Z1). A positive compensation value causes the compensation axis (Z1) to traverse in the negative direction.

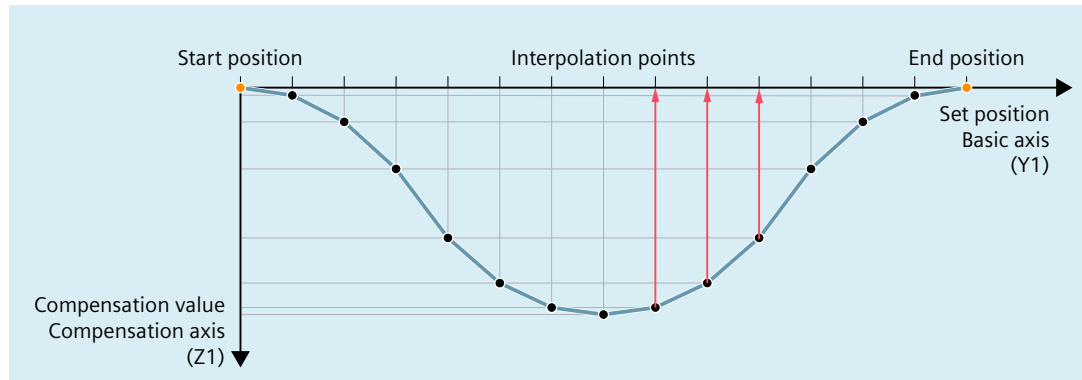


Figure 8-14 Compensation table

### Summed compensation value

Compensation values from several compensation tables can be applied to a compensation axis. The resulting summed compensation value is the sum of all of the individual compensation values.

### Parameterization options

- An axis can be defined as a basic axis for **several** compensation tables.
- **Several** compensation tables can act on **one** compensation axis. The total (summed) compensation value is derived from the sum of the compensation values of the individual compensation tables.
- An axis can be both a base axis and a compensation axis at any one time. The programmed position setpoint is used to calculate the compensation values.
- The scope of action of the compensation (starting and end position of the base axis) and the distance between the interpolation points can be defined for every compensation table.
- The compensation can act **as a function of the direction**.
- Every compensation table has a **modulo function** for cyclic evaluation.
- A **weighting factor** can be taken into account for each compensation table, with which the value in the table is multiplied.
- Using **table multiplication**, the actual compensation value  $K_A$  of compensation table A can be multiplied with the actual compensation value  $K_x$  of any arbitrary compensation table X, i.e. also with itself. The result of the table multiplication is added to the actual compensation value  $K_A$  of compensation table A, and this is then the sum compensation value  $SK_A$ , which is then effective in the compensation axis.  

$$SK_A = K_A + K_A * K_x$$

- **Axes-specific** activation for all compensation relationships of the axis via the machine data:  
MD32710 \$MA\_CEC\_ENABLE[<axis>]
- **Table-specific** activation via the setting data:  
SD41300 \$SN\_CEC\_TABLE\_ENABLE[<table>]  
Application example: Processing-dependent change to the compensation ratio by switching the active compensation table via part program/synchronous actions or PLC user program.

## Application examples

### Compensation of repetitive errors with short wavelength

To compensate for repetitive errors with short wavelength in an axis, a compensation table with modulo function for the repetitive error components (short wavelength) together with a second compensation table without modulo function for the aperiodic error component are parameterized for the same axis.

### User-specific leadscrew error compensation

For user-specific leadscrew error compensation, a compensation table with the same axis is parameterized as basic and compensation axis.

However, in this case there is a disadvantage in so much that contrary to the standard function a measuring system switchover cannot be automatically taken into consideration.

## 8.7.2 Commissioning: Machine data

| Number                     | Identifier                     | Meaning                                               |
|----------------------------|--------------------------------|-------------------------------------------------------|
| NC-specific machine data   |                                |                                                       |
| MD10240                    | \$MN_SCALING_SYSTEM_IS_METRIC. | Metric basic system                                   |
| MD10260                    | \$MN_CONVERT_SCALING_SYSTEM    | Basic system switchover active                        |
| MD18342                    | \$MN_MM_CEC_MAX_POINTS         | Number of interpolation points per compensation table |
| Axis-specific machine data |                                |                                                       |
| MD32710                    | \$MA_CEC_ENABLE                | Enabling sag compensation                             |
| MD32711                    | \$MA_CEC_SCALING_SYSTEM_METRIC | System of units for sag compensation                  |
| MD32720                    | \$MA_CEC_MAX_SUM               | Maximum compensation value for sag compensation       |
| MD32730                    | \$MA_CEC_MAX_VELO              | Maximum velocity change for CEC                       |

### System of units (MD10240, MD10260, MD32711)

The system of units effective in the control system (metric or inch) is defined using:

MD10240 \$MN\_SCALING\_SYSTEM\_IS\_METRIC = <system of units>

**System of units switchover without automatic conversion**

Table parameters with position data are automatically converted if the system of units changes.

The position data is always interpreted in the actual system of units. Conversion must be implemented externally.

**System of units switchover with automatic conversion**

The automatic conversion of position data is activated using:

MD10260 \$MN\_CONVERT\_SCALING\_SYSTEM = TRUE

As a consequence, the system of units set in the following machine data is effective for all compensation tables of the axis:

MD32711 \$MA\_CEC\_SCALING\_SYSTEM\_METRIC = <system of units>

As a consequence, all position data is interpreted together with the summed compensation value in the configured system of units. External conversion of position data is no longer necessary when the system of units changes.

**Axes-specific activation (MD32710)**

Axis-specific activation for all compensation relationships of the axis is realized using:

MD32710 \$MA\_CEC\_ENABLE[<axis>] = TRUE

**Number of interpolation points per compensation table (MD18342)**

The number of interpolation points per compensation table is parameterized using:

MD18342 \$MN\_MM\_CEC\_MAX\_POINTS[<table index>] = <number of interpolation points>

The number of interpolation points required in a compensation table is calculated from the maximum, minimum and step width (increment) set in System variables (Page 251) of table:

$\text{<Number of interpolation points>} = (\$AN\_CEC\_MAX - \$AN\_CEC\_MIN) / \$AN\_CEC\_STEP + 1$

**Monitoring (MD32720, MD32730)****Absolute limiting**

To avoid inadmissibly high compensation motion of the compensation axis, the summed compensation value is monitored against the maximum value specified in the machine data:

MD32720 \$MA\_CEC\_MAX\_SUM[<compensation axis>] = <maximum value>

When the value is exceeded, the summed compensation value is limited to the maximum value and alarm 20124 "Summed compensation value too high" is displayed.

**Limiting the change**

To avoid inadmissibly high dynamic loads on the compensation axis, the summed compensation value change is monitored against the maximum value specified in machine data:

MD32730 \$MA\_CEC\_MAX\_VELO[<compensation axis>] = <percentage of the maximum axis velocity (MD32000 \$MA\_MAX\_AX\_VELO)>

When exceeded, the summed compensation value change is limited to the maximum value and alarm 20125 "Compensation value change too fast" is displayed.

The reduced traversing distance as result of the limiting is output as soon as the compensation value is no longer being limited.

### 8.7.3 Commissioning: Setting data

| Number  | Identifier            | Meaning                   |
|---------|-----------------------|---------------------------|
| SD41300 | \$SN_CEC_TABLE_ENABLE | Compensation table enable |
| SD41310 | \$SN_CEC_TABLE_WEIGHT | Weighting factor          |

#### Compensation table enable (SD41300)

The evaluation of the compensation table is enabled using the setting data:

SD41300 \$SN\_CEC\_TABLE\_ENABLE[<table index>] = TRUE

The compensation value determined using the compensation table is only added to the summed compensation value of the compensation axis when the evaluation is enabled.

#### Weighting factor (SD41310)

The weighting factor should be entered in the setting data, with which the compensation value determined from the compensation table is multiplied:

SD41310 \$SN\_CEC\_TABLE\_WEIGHT[<table index>] = <weighting factor>

As standard, the weighting factor has a value of 1.0. The compensation table becomes inactive with a weighting factor of 0.0.

### 8.7.4 Commissioning: System variables

| Identifier                   | Meaning                |
|------------------------------|------------------------|
| NC-specific system variables |                        |
| \$AN_CEC                     | Compensation values    |
| \$AN_CEC_DIRECTION           | Direction dependency   |
| \$AN_CEC_INPUT_AXIS          | Basic axis             |
| \$AN_CEC_INPUT_NCU           | Basic axis on the NCU: |
| \$AN_CEC_IS_MODULO           | Modulo function        |
| \$AN_CEC_MAX                 | End position           |
| \$AN_CEC_MIN                 | Starting position      |
| \$AN_CEC_MULT_BY_TABLE       | Multiplication         |

| Identifier                     | Meaning                               |
|--------------------------------|---------------------------------------|
| \$AN_CEC_OUTPUT_AXIS           | Compensation axis                     |
| \$AN_CEC_OUTPUT_NCU            | Compensation axis on the NCU          |
| \$AN_CEC_STEP                  | Distance between interpolation points |
| \$AN_CEC_TYPE                  | Table type                            |
| Axis-specific system variables |                                       |
| \$VA_CEC_COMP_VAL              | Actual compensation value             |

### Compensation values (\$AN\_CEC)

The compensation values of the compensation table should be entered into system variable:

$\$AN\_CEC[\text{<table index>, <interpolation point index>}] = \text{<compensation value>}$

$\text{<Interpolation point index>} = 0 \leq x \leq (\text{value of MD18342[<table index>]} - 1)$

#### Note

Before writing to system variable \$AN\_CEC, all of the compensation functions for all of the axes must be deactivated:

- MD32700 \$MA\_ENC\_COMP\_ENABLE[<axis>] = 0
- MD32710 \$MA\_CEC\_ENABLE[<axis>] = 0

### Basic axis (\$AN\_CEC\_INPUT\_AXIS)

The basic axis should be entered into the system variable, i.e. the name of the axis whose position setpoint is to be used as the input for the compensation table.

$\$AN\_CEC\_INPUT\_AXIS[\text{<table index>}] = \text{"<channel axis name>" or "<machine axis name>"}$

### Compensation axis (\$AN\_CEC\_OUTPUT\_AXIS)

The compensation axis should be entered in the system variable, i.e. the name of the axis to whose setpoint the compensation value is added.

$\$AN\_CEC\_OUTPUT\_AXIS[\text{<table index>}] = \text{"<channel axis name>" or "<machine axis name>"}$

#### Note

If the names of channel and machine axes are the same in multi-channel systems, the standard axis names AX1, AX2, etc. must be used.

### Distance between interpolation points (\$AN\_CEC\_STEP)

The distance between two interpolation points should be entered into the system variable (position values of the basic axis) of the compensation table.

$\$AN\_CEC\_STEP[\text{<table index>}] = \text{<distance between interpolation points>}$

Within a compensation table, the distance between interpolation points remains constant.

**Starting position (\$AN\_CEC\_MIN)**

The setpoint position of the basic axis for the first interpolation point or the start of the compensation table should be entered in the system variable.

$\$AN\_CEC\_MIN[\text{<table index>}] = \text{<starting position>}$

---

**Note****Setpoint position less than the starting position**

The compensation value of the first interpolation point is used for all setpoint positions less than the starting position.

Exception: Compensation tables with modulo function

---

**End position (\$AN\_CEC\_MAX)**

The setpoint position of the basic axis for the last interpolation point or the end of the compensation table should be entered in the system variable.

$\$AN\_CEC\_MAX[\text{<table index>}] = \text{<end position>}$

---

**Note****Setpoint position greater than the end position**

The compensation value of the last interpolation point is used for all setpoint positions higher than the starting position.

Exception: Compensation tables with modulo function

---

**Direction-dependent compensation (\$AN\_CEC\_DIRECTION)**

The traversing direction of the basic axis, where the compensation should become effective, should be entered into the system variable.

$\$AN\_CEC\_DIRECTION[\text{<table index>}] = \text{<direction>}$

- 0: in both traversing directions
- 1: in the positive traversing direction only
- -1: in the negative traversing direction only

**Table multiplication (\$AN\_CEC\_MULT\_BY\_TABLE)**

Using table multiplication, the actual compensation value  $K$  of the compensation table can be multiplied with the actual compensation value  $K_x$  of any arbitrary compensation table  $X$ , i.e. also with itself. The result of the table multiplication is added to the actual compensation value  $K$  of compensation table, and this is the summed compensation value  $SK$ , which is then effective in the compensation axis.

$$SK = K + K * K_x$$

The number of the compensation table should be entered in the system variable, with whose compensation value the actual compensation value should be multiplied.

`$AN_CEC_MULT_BY_TABLE[<table index>] = <table number X>`

`<table number X> = <table index X> + 1`

### Modulo function (`$AN_CEC_IS_MODULO`)

If the modulo function is activated for a compensation table, then the input value, i.e. the setpoint position of the basic axis, is calculated, modulo with the input range of the compensation table. This means that the compensation value of the end position is followed again by the compensation value of the starting position or for an inverse run-through direction, the compensation value of the starting position is followed again by the compensation value of the end position.

`$AN_CEC_IS_MODULO[<table index>] = <value>`

- 0: Compensation without modulo function
- 1: Compensation with modulo function

---

#### Note

##### Equal compensation values

When the modulo function is active, we recommend that you set the compensation values of the starting and end positions the same.

---



---

#### Note

##### Modulo rotary axis

For a modulo rotary axis, the modulo function of the compensation must be activated.

---

#### Example: Parameterization for a modulo rotary axis

```
$MA_IS_ROT_AX[AX1] = 1 ; rotary axis
$MA_ROT_IS_MODULO[AX1] = 1 ; modulo 360°
$AN_CEC_INPUT_AXIS[0] = AX1
$AN_CEC_MIN[0] = 0.0
$AN_CEC_MAX[0] = 360.0
$AN_CEC_STEP[0]=1.0
$AN_CEC_IS_MODULO[0] = 1
$MN_MM_CEC_MAX_POINTS = 361
$AN_CEC[0, 0] = $AN_CEC[0, 360] = 0.1
```

### Table type (`$AN_CEC_TYPE`)

The table type, i.e. the compensation type should be entered into the system variable.

\$AN\_CEC\_TYPE[<table index>] = <Type>

- 0: General interpolating compensation
- 1: Cylinder error compensation

### Actual compensation value (\$VA\_CEC\_COMP\_VAL)

The system variable supplies the currently effective compensation value of the axis:

<Actual compensation value> = \$VA\_CEC\_COMP\_VAL[<Axis>]

## 8.7.5 Commissioning: Basic procedure

The compensation tables must be defined in the first commissioning step. To do this, the required number of interpolation points must be set for the particular compensation table. The compensation tables will be created in the control system and populated with default values at the next warm restart.

The second commissioning step involves parameterizing the compensation data using system variables. This can be performed in two different ways:

- Start an NC program in which the system variables will be written.
- Transfer the compensation tables from an external computer to the control.

---

### Note

All compensation functions for all axes must be deactivated before loading the compensation tables:

- MD32700 \$MA\_ENC\_COMP\_ENABLE[<Achse>] == 0
  - MD32710 \$MA\_CEC\_ENABLE[<axis>] == 0
- 

## Sequence

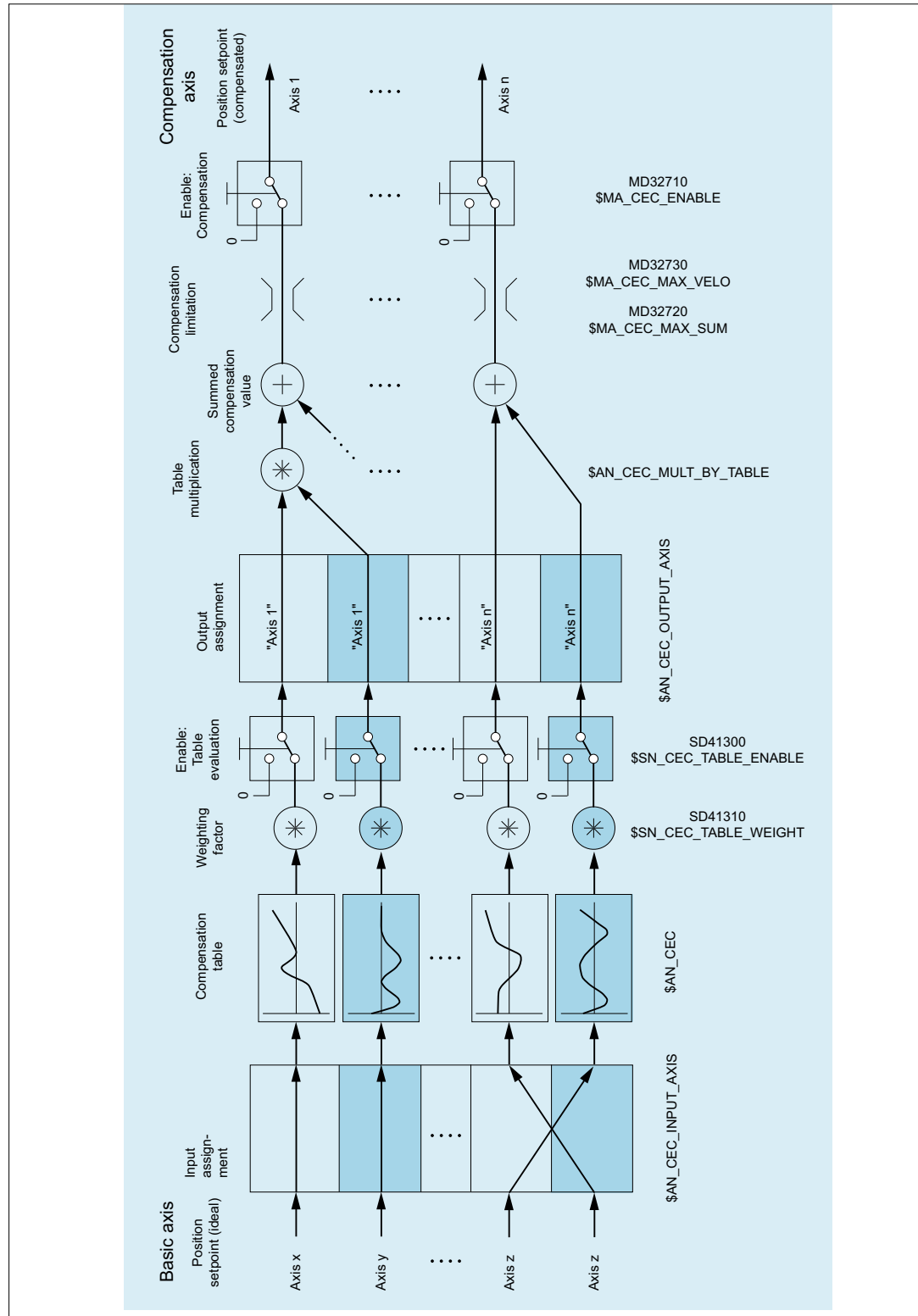
1. Parameterize the number of interpolation points for the compensation tables:  
MD18342 \$MN\_MM\_CEC\_MAX\_POINTS
2. Parameterize the monitoring functions:
  - Absolute limiting: MD32720 \$MA\_CEC\_MAX\_SUM
  - Limiting the change: MD32730 \$MA\_CEC\_MAX\_VELO
3. Parameterize the system of units switchover
  - without automatic conversion:  
MD10240 \$MN\_SCALING\_SYSTEM\_IS\_METRIC
  - with automatic conversion:
    - MD10240 \$MN\_SCALING\_SYSTEM\_IS\_METRIC
    - MD10260 \$MN\_CONVERT\_SCALING\_SYSTEM
    - MD32711 \$MA\_CEC\_SCALING\_SYSTEM\_METRIC
4. Initiate a **warm restart** of the NC to activate the machine data changes

5. Parameterize the weighting factors for the compensation tables:  
SD41310\$SN\_CEC\_TABLE\_WEIGHT
6. Parameterize the table parameters in system variables \$AN\_CEC\_...
7. Check or reference the actual measuring system of the basic and compensation axis:
  - <Axis>.basic.in.enc1Synchronized (referenced/synchronized 1)
  - <Axis>.basic.in.enc2Synchronized (referenced / synchronized 2)
8. Enable compensation:
  - Table-specific: SD41300 CEC\_TABLE\_ENABLE
  - Axis-specific: MD32710 \$MA\_CEC\_ENABLE

### PLC signals

| Basic Program Plus               | Basic Program       |                    |
|----------------------------------|---------------------|--------------------|
| <Axis>.basic.in.enc1Synchronized | LBP_Axis*.E_RefSyn1 | DB31, ... .DBX60.4 |
| <Axis>.basic.in.enc2Synchronized | LBP_Axis*.E_RefSyn2 | DB31, ... .DBX60.5 |

## 8.7.6 Commissioning: Overview diagram



### 8.7.7 Example 1: Sag compensation

Depending on the position of the axis Y1, an additional compensation value is applied to the set position of axis Z1.

Compensation table used: Table 1  $\Rightarrow$  Index = 0

#### Compensation parameters

- Starting position: -400.0
- End position: 400.0
- Distance between interpolation points: 8.0

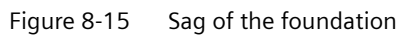
#### Number of interpolation points

MD18342 \$MN\_MM\_CEC\_MAX\_POINTS[ 0 ] = (400.0 - -400.0) / 8.0 + 1 = 101

The memory required in the static user memory is at least 808 bytes (8 bytes per compensation value).

| Program code                | Comment                                                    |
|-----------------------------|------------------------------------------------------------|
| %_N_NC_CEC_INI              | ; Writing the compensation data                            |
| CHANDATA(1)                 | ; Compensation table 1, index 0                            |
| \$AN_CEC[0,0]=0             | ; 1st Compensation value = 0 $\mu$ m                       |
| \$AN_CEC[0,1]=0.01          | ; 2nd Compensation value = 10 $\mu$ m                      |
| \$AN_CEC[0,2]=0.012         | ; 3rd Compensation value = 12 $\mu$ m                      |
| ...                         |                                                            |
| \$AN_CEC[0,100]=0           | ; 101st Compensation value = 0 $\mu$ m                     |
| \$AN_CEC_INPUT_AXIS[0]=AX2  | ; Basic axis Y1 $\Rightarrow$ machine axis name AX2        |
| \$AN_CEC_OUTPUT_AXIS[0]=AX3 | ; Compensation axis Z1 $\Rightarrow$ machine axis name AX3 |
| \$AN_CEC_STEP[0]=8.0        | ; distance between interpolation points 8.0 mm             |
| \$AN_CEC_MIN[0]=-400.0      | ; Starting position: Y1 = -400mm                           |
| \$AN_CEC_MAX[0]=400.0       | ; End position: Y1 = +400mm                                |
| \$AN_CEC_DIRECTION[0]=0     | ; Compensate in both traversing directions of Y1           |
| \$AN_CEC_MULT_BY_TABLE[0]=0 | ; No table multiplication                                  |
| \$AN_CEC_IS_MODULO[0]=0     | ; No modulo function                                       |

Compensation of sag of the foundation of a drilling machine with table multiplication.



The compensation in axis X1 depends on:

- The position of axis X1 because it determines the angle of inclination  $\beta$
- The position of axis Z1, in which the drill is located.

$$\Delta X1_{\text{total}} = \Delta Z1 * \Delta X1 = \Delta Z1 * \sin\beta(X1) \approx \Delta Z1 * \beta(X1)$$


- Basic axis: X1
- Compensation axis: X1
- Compensation values: Sine of the position-dependent tilt angle  $\beta$ , with  $\beta = f(\text{position of X1})$

- Basic axis: Z1
- Compensation axis: X1
- Compensation values: Reaction of the position of axis Z1 on the measured position of axis X1

**Table multiplication**

For compensation relationship 1 (table index: 0) the table multiplication should be set with the compensation relationship 2:

`$AN_CEC_MULT_BY_TABLE[ 0 ] = 2`

**8.7.9****Example 3: 2-dimensional array of compensation values**

For flat-bed machines, the use case often arises in practice in which the sag compensation values of the Z-axis depend on the axis positions of the X axis and Y axis. This is why it makes sense to organize compensation values in a 2-dimensional array.

In this example, a possible way of implementing sag compensation using a grid, having a size of 4 x 5 (rows x columns), is explained in more detail. The size of the complete measuring range is 2000 x 900 mm<sup>2</sup>. The compensation values are each determined in increments of 500 mm along the X axis and 300 mm increments along the Y axis. The interpolation points with the relevant compensation values are positioned at the intersections of the grid (X-Y plane). Compensation values between these interpolation points are linearly interpolated.

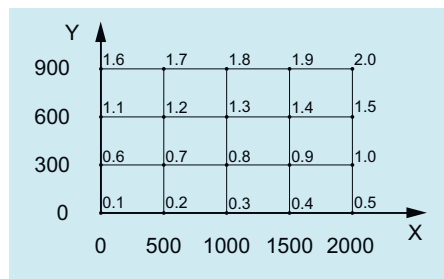


Figure 8-17 Compensation values of the Z axis

**Note****SINUMERIK Operate user interface**

The currently effective compensation value of the Z axis as a result of sag compensation is displayed on the SINUMERIK Operate user interface under:

Operating area: "Diagnostics" > "Axis diagnostics" > "Service axis" > Signal: "Compensation, sag + temperature"

**Implementation**

For each line of the four-line the grid, a compensation table is set up with five interpolation points. In the first compensation table, starting at interpolation point 1, in ascending sequence, the compensation values 0.1 to 0.5 are entered in the first line. Using the same procedure, in the first compensation table, compensation values 0.6 to 1.0 are entered in the second line, etc. These compensation tables, relevant for the position of the X axis, are now designated with **f tables** and the table values as **f<sub>i</sub>(x)**, with i: Table number.

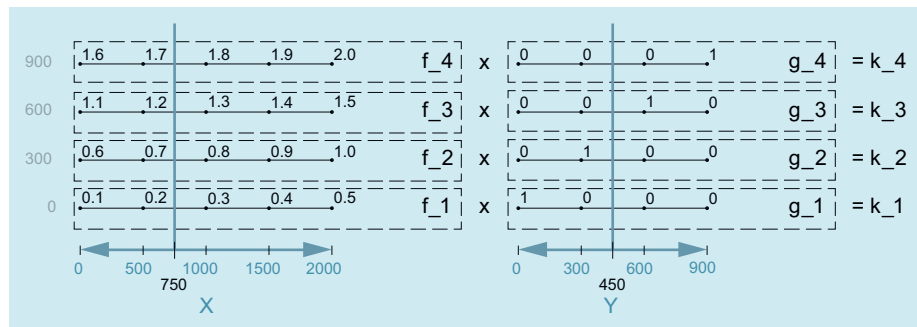
Additional compensation tables are required to take into account the position of the Y axis. These compensation tables are now designated as **g tables** and the table values as **g<sub>i</sub>(y)**. The number of f tables and g tables is equal. In the example, four:

In the g tables, one compensation value is set to 1 - and all the others to 0. The position of compensation value 1 within the table is determined by the table number. In the first g table, compensation value 1 is positioned at the first interpolation point and, in the second g table, at the second interpolation point, etc. By multiplying g tables by f tables, the correct compensation value in each f table is selected by multiplying it by 1. All irrelevant compensation values are concealed through multiplication by 0.

The summed compensation value  $D_z$  at position (x/y) is calculated according to the following equation:

$$D_z(x/y) = f_1(x) * g_1(y) + f_2(x) * g_2(y) + f_3(x) * g_3(y) + f_4(x) * g_4(y) = k_1 + k_2 + k_3 + k_4$$

At location  $D_z(750/450)$ , the summed compensation value of the Z axis is calculated to be:



$$\begin{aligned}
 D_z(750/450) &= f_1(750) * g_1(450) + f_2(750) * g_2(450) + \\
 &\quad f_3(750) * g_3(450) + f_4(750) * g_4(450) \\
 &= 0.25 * 0.0 + 0.75 * 0.5 + \\
 &\quad 1.25 * 0.5 + 1.75 * 0.0 \\
 &= 1.0
 \end{aligned}$$

## Parameterization

### Parameterization of machine data using the NC program

| Program code                                                             | Comment                          |
|--------------------------------------------------------------------------|----------------------------------|
| ; Number of interpolation points for the compensation tables             |                                  |
| \$MN_MM_CEC_MAX_POINTS[0]=5                                              | ; Compensation table 1           |
| \$MN_MM_CEC_MAX_POINTS[1]=5                                              | ; Compensation table 2           |
| \$MN_MM_CEC_MAX_POINTS[2]=5                                              | ; Compensation table 3           |
| \$MN_MM_CEC_MAX_POINTS[3]=5                                              | ; Compensation table 4           |
| \$MN_MM_CEC_MAX_POINTS[4]=4                                              | ; Compensation table 5           |
| \$MN_MM_CEC_MAX_POINTS[5]=4                                              | ; Compensation table 6           |
| \$MN_MM_CEC_MAX_POINTS[6]=4                                              | ; Compensation table 7           |
| \$MN_MM_CEC_MAX_POINTS[7]=4                                              | ; Compensation table 8           |
| ; Monitoring functions for axis Z1 corresponding to the 3rd machine axis |                                  |
| \$MA_CEC_MAX_SUM[AX3]=10.0                                               | ; Max. summed compensation value |

| Program code                 | Comment       |
|------------------------------|---------------|
| \$MA_CEC_MAX_VELO[AX3]=100.0 | ; Max. change |

### Setting the table parameters (system variables) using the NC program

| Program code                                                  | Comment                    |
|---------------------------------------------------------------|----------------------------|
| ; To write to the compensation tables, the                    |                            |
| ; compensation for axis Z1 (compensation axis)                |                            |
| ; must first be deactivated.                                  |                            |
| \$MA_CEC_ENABLE[Z1] = FALSE                                   |                            |
| NEWCONF                                                       | ; Activate \$MA_CEC_ENABLE |
| ; Define values f <sub>i</sub> (x) in the f tables:           |                            |
| ; Function values f <sub>1</sub> (x) for table with index [0] |                            |
| \$AN_CEC[0,0]=0.1                                             |                            |
| \$AN_CEC[0,1]=0.2                                             |                            |
| \$AN_CEC[0,2]=0.3                                             |                            |
| \$AN_CEC[0,3]=0.4                                             |                            |
| \$AN_CEC[0,4]=0.5                                             |                            |
| ; Function values f <sub>2</sub> (x) for table with index [1] |                            |
| \$AN_CEC[1,0]=0.6                                             |                            |
| \$AN_CEC[1,1]=0.7                                             |                            |
| \$AN_CEC[1,2]=0.8                                             |                            |
| \$AN_CEC[1,3]=0.9                                             |                            |
| \$AN_CEC[1,4]=1.0                                             |                            |
| ; Function values f <sub>3</sub> (x) for table with index [2] |                            |
| \$AN_CEC[2,0]=1.1                                             |                            |
| \$AN_CEC[2,1]=1.2                                             |                            |
| \$AN_CEC[2,2]=1.3                                             |                            |
| \$AN_CEC[2,3]=1.4                                             |                            |
| \$AN_CEC[2,4]=1.5                                             |                            |
| ; Function values f <sub>4</sub> (x) for table with index [3] |                            |
| \$AN_CEC[3,0]=1.6                                             |                            |
| \$AN_CEC[3,1]=1.7                                             |                            |
| \$AN_CEC[3,2]=1.8                                             |                            |
| \$AN_CEC[3,3]=1.9                                             |                            |
| \$AN_CEC[3,4]=2.0                                             |                            |
| ; Enable evaluation of f tables with compensation values      |                            |
| \$SN_CEC_TABLE_ENABLE[0]=TRUE                                 |                            |
| \$SN_CEC_TABLE_ENABLE[1]=TRUE                                 |                            |
| \$SN_CEC_TABLE_ENABLE[2]=TRUE                                 |                            |
| \$SN_CEC_TABLE_ENABLE[3]=TRUE                                 |                            |

| Program code                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | Comment |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| <pre> ; Define weighting factor for f tables \$SN_CEC_TABLE_WEIGHT[0]=1.0 \$SN_CEC_TABLE_WEIGHT[1]=1.0 \$SN_CEC_TABLE_WEIGHT[2]=1.0 \$SN_CEC_TABLE_WEIGHT[3]=1.0  ; Changes to the following table parameters do not take effect until ; after a warm restart ; Define basic axis X1 \$AN_CEC_INPUT_AXIS[0]=(X1) \$AN_CEC_INPUT_AXIS[1]=(X1) \$AN_CEC_INPUT_AXIS[2]=(X1) \$AN_CEC_INPUT_AXIS[3]=(X1)  ; Define compensation axis Z1 \$AN_CEC_OUTPUT_AXIS[0]=(Z1) \$AN_CEC_OUTPUT_AXIS[1]=(Z1) \$AN_CEC_OUTPUT_AXIS[2]=(Z1) \$AN_CEC_OUTPUT_AXIS[3]=(Z1)  ; Define distance between interpolation points for compensation values in f tables \$AN_CEC_STEP[0]=500.0 \$AN_CEC_STEP[1]=500.0 \$AN_CEC_STEP[2]=500.0 \$AN_CEC_STEP[3]=500.0  ; Compensation starts at X1=0 \$AN_CEC_MIN[0]=0.0 \$AN_CEC_MIN[1]=0.0 \$AN_CEC_MIN[2]=0.0 \$AN_CEC_MIN[3]=0.0  ; Compensation ends at X1=2000 \$AN_CEC_MAX[0]=2000.0 \$AN_CEC_MAX[1]=2000.0 \$AN_CEC_MAX[2]=2000.0 \$AN_CEC_MAX[3]=2000.0  ; Values of f tables with index [t1] are multiplied by values in g tables ; by the number [t2] ; in accordance with the equation (algorithm) specified above \$AN_CEC_MULT_BY_TABLE[0] = 5 \$AN_CEC_MULT_BY_TABLE[1] = 6 </pre> |         |

## 8.7 IC: Sag and angularity error compensation

| Program code                                                         | Comment |
|----------------------------------------------------------------------|---------|
| \$AN_CEC_MULT_BY_TABLE[2] = 7                                        |         |
| \$AN_CEC_MULT_BY_TABLE[3] = 8                                        |         |
| ; Define the g table values for g_i(y):                              |         |
| ; Function values g_1(x) for table with index [4]                    |         |
| \$AN_CEC[4,0]=1.0                                                    |         |
| \$AN_CEC[4,1]=0.0                                                    |         |
| \$AN_CEC[4,2]=0.0                                                    |         |
| \$AN_CEC[4,3]=0.0                                                    |         |
| ; Function values g_2(x) for table with index [5]                    |         |
| \$AN_CEC[5,0]=0.0                                                    |         |
| \$AN_CEC[5,1]=1.0                                                    |         |
| \$AN_CEC[5,2]=0.0                                                    |         |
| \$AN_CEC[5,3]=0.0                                                    |         |
| ; Function values g_3(x) for table with index [6]                    |         |
| \$AN_CEC[6,0]=0.0                                                    |         |
| \$AN_CEC[6,1]=0.0                                                    |         |
| \$AN_CEC[6,2]=1.0                                                    |         |
| \$AN_CEC[6,3]=0.0                                                    |         |
| ; Function values g_4(x) for table with index [7]                    |         |
| \$AN_CEC[7,0]=0.0                                                    |         |
| \$AN_CEC[7,1]=0.0                                                    |         |
| \$AN_CEC[7,2]=0.0                                                    |         |
| \$AN_CEC[7,3]=1.0                                                    |         |
| ; Enable evaluation of g tables with compensation values             |         |
| \$SN_CEC_TABLE_ENABLE[4]=TRUE                                        |         |
| \$SN_CEC_TABLE_ENABLE[5]=TRUE                                        |         |
| \$SN_CEC_TABLE_ENABLE[6]=TRUE                                        |         |
| \$SN_CEC_TABLE_ENABLE[7]=TRUE                                        |         |
| ; Define weighting factor for g tables                               |         |
| \$SN_CEC_TABLE_WEIGHT[4]=1.0                                         |         |
| \$SN_CEC_TABLE_WEIGHT[5]=1.0                                         |         |
| \$SN_CEC_TABLE_WEIGHT[6]=1.0                                         |         |
| \$SN_CEC_TABLE_WEIGHT[7]=1.0                                         |         |
| ; Changes to the following table parameters do not take effect until |         |
| ; a power on is carried out                                          |         |
| ; Define basic axis Y1                                               |         |
| \$AN_CEC_INPUT_AXIS[4]=(Y1)                                          |         |

| Program code                                                                           | Comment                       |
|----------------------------------------------------------------------------------------|-------------------------------|
| \$AN_CEC_INPUT_AXIS[5]=(Y1)                                                            |                               |
| \$AN_CEC_INPUT_AXIS[6]=(Y1)                                                            |                               |
| \$AN_CEC_INPUT_AXIS[7]=(Y1)                                                            |                               |
| <br>; Define compensation axis Z1                                                      |                               |
| \$AN_CEC_OUTPUT_AXIS[4]=(Z1)                                                           |                               |
| \$AN_CEC_OUTPUT_AXIS[5]=(Z1)                                                           |                               |
| \$AN_CEC_OUTPUT_AXIS[6]=(Z1)                                                           |                               |
| \$AN_CEC_OUTPUT_AXIS[7]=(Z1)                                                           |                               |
| <br>; Define distance between interpolation points for compensation values in g tables |                               |
| \$AN_CEC_STEP[4]=300.0                                                                 |                               |
| \$AN_CEC_STEP[5]=300.0                                                                 |                               |
| \$AN_CEC_STEP[6]=300.0                                                                 |                               |
| \$AN_CEC_STEP[7]=300.0                                                                 |                               |
| <br>; Compensation starts at Y1=0                                                      |                               |
| \$AN_CEC_MIN[4]=0.0                                                                    |                               |
| \$AN_CEC_MIN[5]=0.0                                                                    |                               |
| \$AN_CEC_MIN[6]=0.0                                                                    |                               |
| \$AN_CEC_MIN[7]=0.0                                                                    |                               |
| <br>; Compensation ends at Y1=900                                                      |                               |
| \$AN_CEC_MAX[4]=900.0                                                                  |                               |
| \$AN_CEC_MAX[5]=900.0                                                                  |                               |
| \$AN_CEC_MAX[6]=900.0                                                                  |                               |
| \$AN_CEC_MAX[7]=900.0                                                                  |                               |
| <br>; Activate compensation again                                                      |                               |
| \$MA_CEC_ENABLE[Z1]=TRUE                                                               |                               |
| NEWCONF                                                                                |                               |
| <br>; Carry out a program test to check that the compensation is active                |                               |
| G01 F1000 X0 X0 Z0 G90                                                                 |                               |
| R1=0 R2=0                                                                              |                               |
| LOOP_Y:                                                                                |                               |
| LOOP_X:                                                                                |                               |
| STOPRE                                                                                 |                               |
| X=R1 Y=R2                                                                              |                               |
| M0                                                                                     | ; Wait to check the CEC value |
| R1=R1+500                                                                              |                               |
| IF R1 <=2000 GOTOB LOOP_X                                                              |                               |
| R1=0                                                                                   |                               |
| R2=R2+300                                                                              |                               |
| IF R2<=900 GOTOB LOOP_Y                                                                |                               |

## 8.8 IC: Direction-dependent leadscrew error compensation

### 8.8.1 Description of functions

If the direction-dependent differences at the compensation points are excessively high, for an inconsistent backlash or for extremely high demands placed on the precision, then it may be necessary to apply direction-dependent compensation of the leadscrew error or measuring system error (for direct position sensing).

#### Direction-dependent leadscrew error compensation

For the "direction-dependent leadscrew error compensation" ("direction-dependent LEC" or also "Bidirectional LEC"), two compensation tables are used for each axis. One compensation table for the positive and one compensation table for the negative traversing direction. The deviation at the particular compensation point is entered as difference between the ideal setpoint and measured actual value in the compensation tables. The control automatically calculates compensation values of intermediate values using linear interpolation.

#### Preconditions / activation

The "direction-dependent LEC" is implemented in the SINUMERIK control as a special case of "sag compensation". This is the reason that the preconditions and conditions of "sag compensation" apply (see "IC: Sag and angularity error compensation (Page 247)").

The activation of the compensation can be checked using a reference measurement, e.g. using the laser interferometer or in the simplest case, using the service display of the particular axis.

---

**Note**

If the "direction-dependent LEC" is used in parallel to the sag compensation and compensation of the angularity, then the secondary conditions of these functions must be taken into consideration together, e.g. the assignment of tables <t> to the particular function.

---

### 8.8.2 Commissioning

#### Measuring the error or compensation values

When commissioning the "direction-dependent LEC" - just the same as when commissioning the "direction-dependent LEC" - direction-dependent error curves for each axis are determined using a suitable measuring device (e.g. laser interferometer) (see Section "IC: Leadscrew error and measuring system error compensation (Page 242)"). A part program with measurement points and wait times should be generated in order to perform the measurement (see Section "Example (Page 269)": Program "BI\_SSF\_K\_MESS\_AX1\_X.MPF").

Because the various measuring devices offer different support options for the practical implementation in conjunction with a SINUMERIK control, this process is only generally described in the following referred to a control.

#### Note

The measurement for determining the leadscrew error should only be carried out during the first commissioning if, in the machine data, the traversing directions of the axes in relation to the machine coordinate system have been correctly set.

### Commissioning (principle)

1. Specify the number of compensation interpolation points (also see Section "Compensation for droop and angularity error: Commissioning (Page 255)")  
For the directional leadscrew error compensation, a compensation table for the positive and a compensation table for the negative traversing directions must be assigned to each axis. The number of compensation interpolation points of a table is defined in:  
MD18342 \$MN\_MM\_CEC\_MAX\_POINTS[ <compensation table index> ]

#### CAUTION

##### Possible data loss

A change to the machine data MD18342 \$MN\_MM\_CEC\_MAX\_POINTS, which configures memory, reconfigures the NC memory the next time the control starts. This can result in the loss of all user-specific data. See Function Manual "Basic Functions", Section "Memory configuration".

Creating a commissioning archive:

Operating area: "Commissioning" > "ETC" key > "Commissioning archive" > "Create commissioning archive" > "OK" > Selection: "NC data"

#### Example

- X axis: positive traversing direction, table 1, 11 interpolation points
- X axis: negative traversing direction, table 2, 11 interpolation points

Machine data:

- MD18342 \$MN\_MM\_CEC\_MAX\_POINTS[ 0 ] = 11
- MD18342 \$MN\_MM\_CEC\_MAX\_POINTS[ 1 ] = 11

2. Reading in a commissioning archive that has been created:  
Operating area: "Commissioning" > "ETC" key > "Commissioning archive" > "Read in commissioning archive" > "OK"  
The compensation tables are then available.
3. To simplify commissioning, create an NC program by which the compensation parameters are written into the machine data and system variables (see Section "Example (Page 269)").
4. Run the NC program on the control:  
Mode: "AUTOMATIC" > Select program > NC start
5. Power-on (warm restart).

6. Now, comparative measurements can be made using the laser interferometer.
7. To further improve the compensation results, it is also conceivable to correct individual compensation values in the program. A POWER ON is no longer necessary when reading-in the table again.

---

**Note****NC\_CEC.INI**

The "NC\_CEC.INI" file copied via "Commissioning" > "System data" (from the folder "NC active data" > "sag angularity comp") includes all negotiated sag/angularity and direction-dependent LEC tables.

---



---

**Note****Backlash**

The backlash should be set to **0**:

- MD32450 \$MA\_BACKLASH [ <measuring system> ] = 0
- 

**Compensation parameters**

The compensation parameters are set via the following system variables:

- \$AN\_CEC[ <table>,<interpolation point> ] (compensation value)
  - \$AN\_CEC\_INPUT\_AXIS[ <table> ] (basic axis)
  - \$AN\_CEC\_OUTPUT\_AXIS[ <table> ] (compensation axis)
- 

**Note**

For the "directional LEC," the basis and compensation axes are **always identical**.

---

- \$AN\_CEC\_STEP[ <table> ] (distance between interpolation points)
  - \$AN\_CEC\_MIN[ <table> ] (starting position)
  - \$AN\_CEC\_MAX[ <table> ] (end position)
  - \$AN\_CEC\_DIRECTION[ <table> ] (direction)
- 

**Note**

The setting \$AN\_CEC\_DIRECTION[<t>] = 0 (table is effective for both traversing directions of the basic axis) is **not** relevant for the "direction-dependent LEC".

---

- \$AN\_CEC\_IS\_MODULO[ <table> ] (compensation with modulo function)
- 

**Note**

For a description of these system variables, see Section "Compensation of sag and angularity error: Commissioning (Page 255)".

---

## System of units

See Section "Compensation for droop and angularity error: Commissioning (Page 255)".

## Monitoring

See Section "Compensation for droop and angularity error: Commissioning (Page 255)".

### 8.8.3 Example

The following examples shows parameterization of the directional compensation tables for an axis (machine axis AX1). All parameter values of the compensation tables are written by means of a program.

#### Compensation parameters

- Basic axis = compensation axis = machine axis AX1
- Distance between interpolation points: 58.0 mm
- Starting position: -585.0 mm
- End position: -5.0 mm

#### Table definition

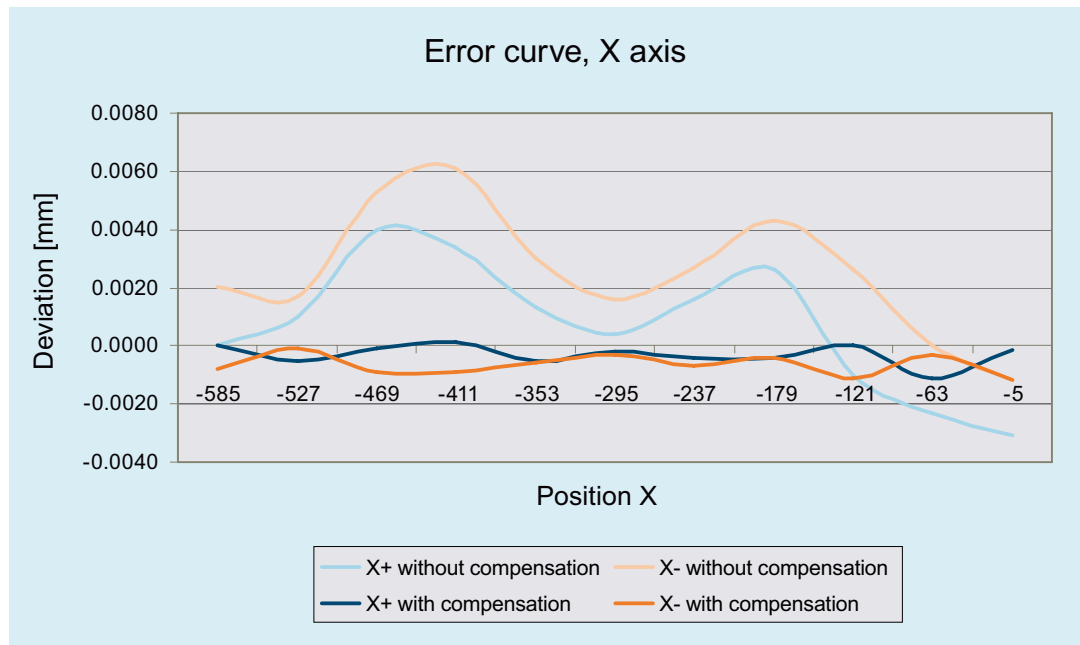
The 1st and 2nd compensation table are defined with 11 compensation interpolation points each for machine axis AX1 as directional compensation tables:

- MD18342 \$MN\_MM\_CEC\_MAX\_POINTS[ 0 ] = 11 (table 1: **positive** traversing direction)
- MD18342 \$MN\_MM\_CEC\_MAX\_POINTS[ 1 ] = 11 (table 2: **negative** traverse direction)

#### Interpolation points and correction values

| Interpolation points |                    | Deviations or correction values |                                | Deviation with compensation    |                                |
|----------------------|--------------------|---------------------------------|--------------------------------|--------------------------------|--------------------------------|
| Index                | Position [mm]      | pos. Traversing direction [mm]  | neg. Traversing direction [mm] | pos. Traversing direction [mm] | neg. Traversing direction [mm] |
| 0                    | -585 <sup>1)</sup> | 0.0000                          | 0.0020                         | 0.0000                         | -0.0008                        |
| 1                    | -527               | 0.0010                          | 0.0017                         | -0.0005                        | -0.0001                        |
| 2                    | -469               | 0.0040                          | 0.0053                         | -0.0001                        | -0.0009                        |
| 3                    | -411               | 0.0034                          | 0.0061                         | 0.0001                         | -0.0009                        |
| 4                    | -353               | 0.0013                          | 0.0030                         | -0.0005                        | -0.0006                        |
| 5                    | -295               | 0.0004                          | 0.0016                         | -0.0002                        | -0.0003                        |
| 6                    | -237               | 0.0016                          | 0.0027                         | -0.0004                        | -0.0007                        |
| 7                    | -179               | 0.0026                          | 0.0043                         | -0.0004                        | -0.0004                        |
| 8                    | -121               | -0.0010                         | 0.0026                         | 0.0000                         | -0.0011                        |
| 9                    | -63                | -0.0023                         | 0.0000                         | -0.0011                        | -0.0003                        |

| Interpolation points                          |                  | Deviations or correction values |                                | Deviation with compensation    |                                |
|-----------------------------------------------|------------------|---------------------------------|--------------------------------|--------------------------------|--------------------------------|
| Index                                         | Position [mm]    | pos. Traversing direction [mm]  | neg. Traversing direction [mm] | pos. Traversing direction [mm] | neg. Traversing direction [mm] |
| 10                                            | -5 <sup>2)</sup> | -0.0031                         | -0.0012                        | -0.0001                        | -0.0012                        |
| 1) Starting position: \$AC_CEC_MIN[ <Table> ] |                  |                                 |                                |                                |                                |
| 2) End position: \$AC_CEC_MAX[ <Table> ]      |                  |                                 |                                |                                |                                |



## Programming

The following actions are performed by program "BI\_SSFk\_TAB\_AX1\_X.MPF":

- Deactivation of the compensation
- Deactivation of the compensation tables to be written (active tables cannot be written).
- Writing the compensation values into the compensation tables for the positive and negative traversing direction of the X axis
- Writing the compensation parameters

```

; directional LEC
; 1st axis AX1
; table 1 - positive traversing direction
; table 2 - negative traversing direction
;--- Deaktivierung der Kompensation und der Tabellen
CHANDATA(1)
$MA_CEC_ENABLE[AX1]=0 ; compensation OFF
$SN_CEC_TABLE_ENABLE[0]=0 ; lock Table 1
$SN_CEC_TABLE_ENABLE[1]=0 ; lock Table 2

```

## 8.8 IC: Direction-dependent leadscrew error compensation

```

NEWCONF
;--- 1. Kompensationstabelle, positive Verfahrriichtung
;----- Kompensationswerte
$AN_CEC[0,0]=0 ; correction value interpolation point 0
$AN_CEC[0,1]=0.001 ; correction value interpolation point 1
$AN_CEC[0,2]=0.004 ; correction value interpolation point 2
$AN_CEC[0,3]=0.0034 ; correction value interpolation point 3
$AN_CEC[0,4]=0.0013 ; correction value interpolation point 4
$AN_CEC[0,5]=0.0004 ; correction value interpolation point 5
$AN_CEC[0,6]=0.0016 ; correction value interpolation point 6
$AN_CEC[0,7]=0.0026 ; correction value interpolation point 7
$AN_CEC[0,8]=-0.001 ; correction value interpolation point 8
$AN_CEC[0,9]=-0.0023 ; correction value interpolation point 9
$AN_CEC[0,10]=-0.0031 ; correction value interpolation point 10
; ----- Kompensationsparame-
ter
$AN_CEC_INPUT_AXIS[0]=(AX1) ; basic axis
$AN_CEC_OUTPUT_AXIS[0]=(AX1) ; compensation axis
$AN_CEC_STEP[0]=58.0 ; distance between interpolation points
$AN_CEC_MIN[0]=-585.0 ; starting position
$AN_CEC_MAX[0]=-5.0 ; end position
$AN_CEC_DIRECTION[0]=1 ; table applies to positive traversing directions
$AN_CEC_MULT_BY_TABLE[0]=0 ; no multiplication (not relevant here)
$AN_CEC_IS_MODULO[0]=0 ; compensation without modulo function
;--- 2. Kompensationstabelle, negative Verfahrriichtung
;----- Kompensationswerte
$AN_CEC[1,0]=0.002 ; correction value interpolation point 0
$AN_CEC[1,1]=0.0017 ; correction value interpolation point 1
$AN_CEC[1,2]=0.0053 ; correction value interpolation point 2
$AN_CEC[1,3]=0.0061 ; correction value interpolation point 3
$AN_CEC[1,4]=0.003 ; correction value interpolation point 4
$AN_CEC[1,5]=0.0016 ; correction value interpolation point 5
$AN_CEC[1,6]=0.0027 ; correction value interpolation point 6
$AN_CEC[1,7]=0.0043 ; correction value interpolation point 7
$AN_CEC[1,8]=0.0026 ; correction value interpolation point 8
$AN_CEC[1,9]=0.000 ; correction value interpolation point 9
$AN_CEC[1,10]=-0.0012 ; correction value interpolation point 10
; ----- Kompensationsparame-
ter
$AN_CEC_INPUT_AXIS[1]=(AX1) ; basic axis
$AN_CEC_OUTPUT_AXIS[1]=(AX1) ; compensation axis
$AN_CEC_STEP[1]=58.0 ; distance between interpolation points
$AN_CEC_MIN[1]=-585.0 ; starting position
$AN_CEC_MAX[1]=-5.0 ; end position
$AN_CEC_DIRECTION[1]=-1 ; table applies to negative traversing directions
$AN_CEC_MULT_BY_TABLE[1]=0 ; no multiplication (not relevant here)

```

## 8.9 IC: Multi Error Compensation (MEC)

```

$AN_CEC_IS_MODULO[1]=0 ; compensation without modulo function (for rotary
 axes only)
;--- Aktivierung der Kompensation und der Tabellen
$MA_CEC_ENABLE[AX1]=1 ; compensation ON
$SN_CEC_TABLE_ENABLE[0]=1 ; enable table 1
$SN_CEC_TABLE_ENABLE[1]=1 ; enable table 2
NEWCONF
M17 ; end of program

```

## 8.9 IC: Multi Error Compensation (MEC)

### 8.9.1 Preconditions

#### Option model

Function "Multi Error Compensation" (MEC) is an option that requires a license. The following versions are available:

#### Multi Error Compensation Eco

For basic turning machines with the following properties:

- 2 input axes
- 2 output axes
- 1 table
- 80 points per table (on average, 8 points per axis)

#### Multi Error Compensation Basic

For basic milling machines, for compensation in the X, Y and Z axes or for milling machines with head kinematics for head compensation for an "average tool length" with the following properties:

- 3 input axes
- 3 output axes
- 1 table
- 500 points per table (on average, 8 points per axis)

#### Multi Error Compensation Advanced

For large milling machines, for compensation in the X, Y and Z axes or for milling machines with head kinematics for head compensation depending on the compensation values of the tool length. In this case, two correction tables with weighting are required.

Properties:

- 5 input axes (3 linear axis und 2 orientation axes)
- 3 output axes (3 linear axes)
- 2 tables
- 4600 points per table (8 points per linear axis, 3 points per orientation axis)

#### Note

The value in brackets is only intended to indicate the approximate number of interpolation points per axis for a uniform distribution. The number of interpolation points per axis is not limited. Only the sum of the measuring points is limited when the option is active.

## 8.9.2 Function description

Function "Multi Error Compensation (MEC)" is an extension of the Sag and angularity error compensation (CEC (Page 247)). For the CEC function, only **one** input axis is linked with **one** output axis. Using the MEC function, up to **five** input axes can be linked with up to **three** output axes. This creates a correction grid with one up to a maximum of 5 dimensions and with up to three assigned correction values.

#### Correction grid

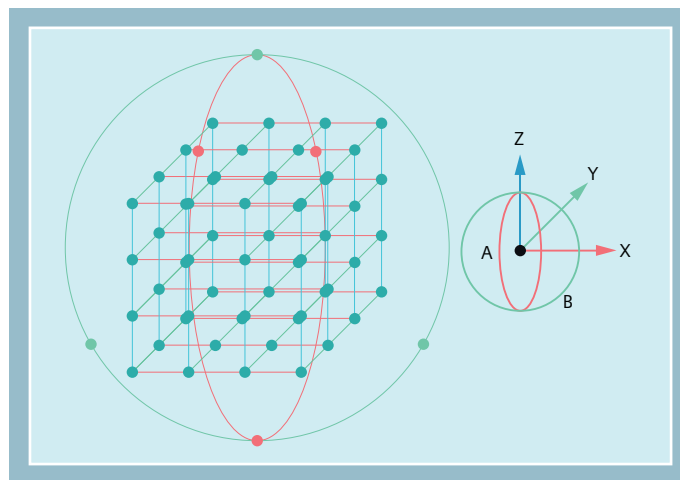


Figure 8-18 5D correction grid

The diagram above shows as **example**, a 5-dimensional correction grid, formed from the following input axes:

- Three linear axes X, Y and Z, each with four correction points distributed over the traversing path of the particular axis
- Two rotary axes A and B, each with three correction points distributed over the traversing path of the particular axis

This results in a total of  $4 \times 4 \times 4 \times 3 \times 3 = 576$  correction points or required measurements for the entire correction space of the machine.

## Properties

The MEC function is characterized by the subsequent features:

- The maximum of five input axes create a correction grid with a maximum of 5 dimensions.
- Each grid point can be assigned between one and a maximum of three correction values.
- The input values for the correction grid are the actual values of the input axes (machine axes) in the machine coordinate system.
- The correction values are determined between the individual grid points using linear interpolation.
- The correction value is added to the respective output axis as compensation value in the machine coordinate system.

### 8.9.3 Commissioning: Configuring the correction table

The correction tables are configured using the subsequently described machine data.

#### Note

The definition of the correction table is described in Chapter "Commissioning: Defining the correction tables (Page 276)"

#### Maximum number: Interpolation points of a correction table

The maximum number of interpolation points for each correction table is configured using the machine data:

MD18344 \$MN\_MM\_MEC\_MAX\_POINTS[<table>] = <value>

| Parameter | Meaning                                                                   |
|-----------|---------------------------------------------------------------------------|
| <table>   | Index i of table, with i = 0 ... ("max. number of correction tables" - 1) |
| <value>   | Number of interpolation points:                                           |

#### Maximum number: Input axes of a correction table

The maximum number of input axes for each correction table is configured using the machine data:

MD18345 \$MN\_MM\_MEC\_MAX\_INPUT[<table>] = <value>

| Parameter | Meaning                                                                   |
|-----------|---------------------------------------------------------------------------|
| <table>   | Index i of table, with i = 0 ... ("max. number of correction tables" - 1) |
| <value>   | Number of input axes                                                      |

#### Maximum number: Output axes of a correction table

The maximum number of output axes for each correction table is configured using the machine data:

MD18346 \$MN\_MM\_MEC\_MAX\_OUTPUT[<table>] = <value>

| Parameter | Meaning                                                                   |
|-----------|---------------------------------------------------------------------------|
| <table>   | Index i of table, with i = 0 ... ("max. number of correction tables" - 1) |
| <value>   | Number of output axes                                                     |

#### 8.9.4 Commissioning: Axis-specific configuring

The axes participating in the MEC function are configured using the subsequently described machine data.

##### Maximum compensation value

The machine data is used to configure the maximum permissible correction value for each output axis:

MD32780 \$MA\_MEC\_MAX\_SUM[<axis>] = <value>

| Parameter | Value                        |
|-----------|------------------------------|
| <axis>    | Channel or machine axis name |

##### Description

To avoid inadmissibly high compensation motion of the output axis, the summed compensation value is monitored against the value specified in the machine data: When the value is exceeded, the summed compensation value is limited to the specified value and alarm 20184 "Summed MEC compensation values too high" is displayed.

##### Maximum velocity change

Using the machine data, the maximum permissible velocity change for each output axis as a result of the correction is configured as a percentage of the maximum axis velocity (MD32000 \$MA\_MAX\_AX\_VELO):

MD32790 \$MA\_MEC\_MAX\_VELO[<axis>] = <value>

| Parameter | Value                        |
|-----------|------------------------------|
| <axis>    | Channel or machine axis name |

##### Description

To avoid inadmissibly high dynamic loads on the output axis, the correction value change is monitored against the maximum value specified in the machine data:

When exceeded, the correction value change is limited to the maximum value and alarm 20185 "Compensation value change too fast" is displayed. The reduced traversing distance as a result of the limiting is output as soon as the compensation value is no longer being limited.

### 8.9.5 Commissioning: Activating the correction table

#### Machine data: Activating the output axes

The respective output axis is activated on an axis-for-axis basis using machine data:

MD32770 \$MA\_MEC\_ENABLE[<axis>] = <status>

| Parameter | Value     | Meaning                              |
|-----------|-----------|--------------------------------------|
| <status>  | TRUE (≠0) | The output axis is active            |
|           | FALSE (0) | The output axis is <b>not</b> active |

#### Setting data: Enabling the correction table

The respective correction table is enabled on a table-for-table bases using the setting data:

SD41360 \$SN\_MEC\_TABLE\_ENABLE[<table>] = <status>

| Parameter | Value                                                                     | Meaning                                   |
|-----------|---------------------------------------------------------------------------|-------------------------------------------|
| <table>   | Index i of table, with i = 0 ... ("max. number of correction tables" - 1) |                                           |
| <status>  | TRUE (≠0)                                                                 | The correction table is active            |
|           | FALSE (0)                                                                 | The correction table is <b>not</b> active |

#### Description

The compensation value determined using the compensation table is only added to the summed compensation value of the output axis when the evaluation is enabled.

### 8.9.6 Commissioning: Defining the correction tables

The correction tables are defined using the subsequently described system variables.

#### Input axes

The input axes of the correction tables are defined using system variables:

\$AN\_MEC\_INPUT\_AXIS[<table>, <input axis>] = <axis>

| Parameter    | Meaning                                                                   |
|--------------|---------------------------------------------------------------------------|
| <table>      | Index i of table, with i = 0 ... ("max. number of correction tables" - 1) |
| <input axis> | Index i of input axis, with i = 0 ... ("max. number of input axes" - 1)   |
| <axis>       | Channel or machine axis name                                              |

## Output axes

The output axes of the correction tables are defined using system variables:

`$AN_MEC_OUTPUT[<table>, <output axis>] = <axis>`

| Parameter     | Meaning                                                                   |
|---------------|---------------------------------------------------------------------------|
| <table>       | Index i of table, with i = 0 ... ("max. number of correction tables" - 1) |
| <output axis> | Index i of output axis, with i = 0 ... ("max. number of output axes" - 1) |
| <axis>        | Channel or machine axis name                                              |

## System of units

The system of units of the correction tables are defined using system variables:

`$AN_MEC_SCALING_METRIC[<table>] = <system of units>`

| Parameter         | Meaning                                                                   |                                  |
|-------------------|---------------------------------------------------------------------------|----------------------------------|
| <table>           | Index i of table, with i = 0 ... ("max. number of correction tables" - 1) |                                  |
| <system of units> | <b>Value</b>                                                              | <b>Meaning</b>                   |
|                   | TRUE<br>(≠0)                                                              | System of units is <b>metric</b> |
|                   | FALSE<br>(0)                                                              | System of units is <b>inch</b>   |

## Measuring positions

The number of measuring positions of the correction tables is defined using system variables:

`$AN_MEC_NUM_POINTS[<table>] = <value>`

| Parameter | Meaning                                                                   |
|-----------|---------------------------------------------------------------------------|
| <table>   | Index n of table, with i = 0 ... ("max. number of correction tables" - 1) |
| <value>   | Number of measuring positions                                             |

## Actual positions of the input axes

The actual positions in the machine coordinate system of the input axes at the particular interpolation point of the correction tables are defined using system variables:

`$AN_MEC_INPUT_POS[<table>, <axis>, <interpolation point>] = <value>`

| Parameter             | Meaning                                                                                        |
|-----------------------|------------------------------------------------------------------------------------------------|
| <table>               | Index i of table, with i = 0 ... ("max. number of correction tables" - 1)                      |
| <axis>                | Index i of <b>input</b> axis, with i = 0 ... ("max. number of input axes" - 1)                 |
| <interpolation point> | Index i of the interpolation point, with i = 0 ... ("max. number of interpolation points" - 1) |
| <value>               | Actual position in the machine coordinate system                                               |

### Compensation values of the output axes

The compensation values in the machine coordinate system for the output axes at the particular interpolation point of the correction tables are defined using system variables:

`$AN_MEC_VAL[<table>, <axis>, <interpolation point>] = <value>`

| Parameter             | Meaning                                                                                        |
|-----------------------|------------------------------------------------------------------------------------------------|
| <table>               | Index i of table, with i = 0 ... ("max. number of correction tables" - 1)                      |
| <axis>                | Index i of <b>output</b> axis, with i = 0 ... ("max. number of output axes" - 1)               |
| <interpolation point> | Index i of the interpolation point, with i = 0 ... ("max. number of interpolation points" - 1) |
| <value>               | Compensation value in the machine coordinate system                                            |

#### 8.9.7 Example

Using an example, the program excerpt illustrates how to define and activate a correction table (table number 8) with three input axes and two output axes:

| Program code                                                          | Comment                          |
|-----------------------------------------------------------------------|----------------------------------|
| <code>\$SN_MEC_TABLE_ENABLE[7] = FALSE</code>                         | ; Deactivate correction table 8  |
| <code>; - - - - -</code>                                              |                                  |
| <code>; Correction table 8, defining the input and output axes</code> |                                  |
| <code>\$AN_MEC_INPUT_AXIS[7, 0] = X</code>                            | ; Input axis X                   |
| <code>\$AN_MEC_INPUT_AXIS[7, 1] = Z</code>                            | ; Input axis Z                   |
| <code>\$AN_MEC_INPUT_AXIS[7, 2] = B</code>                            | ; Input axis B                   |
| <code>;</code>                                                        |                                  |
| <code>\$AN_MEC_OUTPUT_AXIS[7, 0] = X</code>                           | ; Output axis X                  |
| <code>\$AN_MEC_OUTPUT_AXIS[7, 1] = Z</code>                           | ; Output axis Z                  |
| <code>;</code>                                                        |                                  |
| <code>\$AN_MEC_SCALING_METRIC[7] = 1</code>                           | ; System of units is metric.     |
| <code>;</code>                                                        |                                  |
| <code>\$AN_MEC_NUM_POINTS[7] = 2</code>                               | ; 2 interpolation points         |
| <code>;</code>                                                        |                                  |
| <code>; Correction table 8, 1st interpolation point</code>            |                                  |
| <code>\$AN_MEC_INPUT_POS[7,0,0] = 10</code>                           | ; Actual position input axis X   |
| <code>\$AN_MEC_INPUT_POS[7,1,0] = 15</code>                           | ; Actual position input axis Z   |
| <code>\$AN_MEC_INPUT_POS[7,2,0] = 0</code>                            | ; Actual position input axis B   |
| <code>;</code>                                                        |                                  |
| <code>\$AN_MEC_VAL[7,0,0] = 0.123</code>                              | ; Correction value output axis X |
| <code>\$AN_MEC_VAL[7,1,0] = 0.115</code>                              | ; Correction value output axis Z |
| <code>;</code>                                                        |                                  |
| <code>; Correction table 8, 2nd interpolation point</code>            |                                  |
| <code>\$AN_MEC_INPUT_POS[7,0,1] = 15</code>                           | ; Actual position input axis X   |
| <code>\$AN_MEC_INPUT_POS[7,1,1] = 5</code>                            | ; Actual position input axis Z   |
| <code>\$AN_MEC_INPUT_POS[7,2,1] = 45</code>                           | ; Actual position input axis B   |

| Program code                               | Comment                          |
|--------------------------------------------|----------------------------------|
|                                            | ;                                |
| \$AN_MEC_VAL[7,0,0] = 0.053                | ; Correction value output axis X |
| \$AN_MEC_VAL[7,1,0] = 0.012                | ; Correction value output axis Z |
| ; - - - - -                                |                                  |
| SN_MEC_TABLE_ENABLE[7] = TRUE              | ; Activate correction table 8    |
|                                            | ;                                |
| Enable MEC corrections for the output axes |                                  |
| \$MA_MEC_ENABLE[X] = TRUE                  | ; Enable output axis X           |
| \$MA_MEC_ENABLE[Z] = TRUE                  | ; Enable output axis Z           |

## 8.9.8 Constraints

### Behavior at the out range of a correction table

The correction value of the last interpolation point is used if the set position of an input axis lies outside the defined values of the correction table.

#### Correction tables with modulo function

For correction tables with modulo function, the set position is reduced according to the modulo property of the input axis in the defined modulo range.

### Referencing status

#### Change "referenced" → "not referenced"

If the referencing status of the active measuring system of an input axis changes from "Referenced/synchronized" to "Not referenced/synchronized", then the MEC function is deactivated in the corresponding output axes.

#### Change "not referenced" → "referenced"

If the referencing status of the active measuring system of an input axis changes from "Not referenced/synchronized" back to "Referenced/synchronized", then the MEC function is re-activated in the corresponding output axes.

#### While referencing

While referencing, the MEC function in the corresponding output axes is de-activated.

## 8.10 IC: Cylinder error compensation

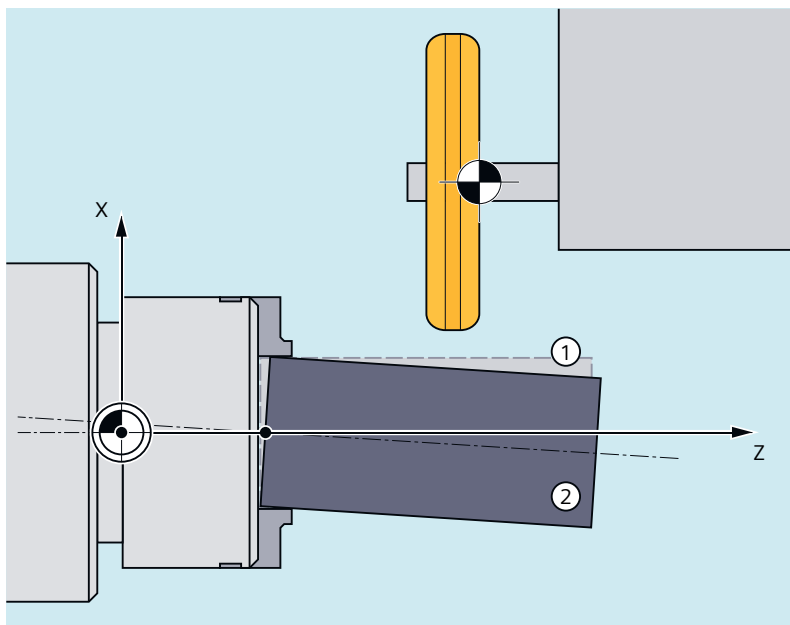
### Note

The following option is required to use the "Cylindrical error compensation":

SINUMERIK Grinding Advanced (article number: 6FC5800-0BS35-0YB0)

### 8.10.1 Function

The cylinder error compensation is used to compensate for clamping errors for cylindrical grinding. In this case, a shift and/or rotation of the workpiece in the machining plane is compensated.



- ① Set position of the cylinder
- ② Actual position of the cylinder

Contrary to IC: Sag and angularity error compensation (Page 247), which is used as basis for cylinder error compensation, only two interpolation points are used to define the compensation function (straight line).

### 8.10.2 Commissioning

| Number  | Identifier            | Meaning                   |
|---------|-----------------------|---------------------------|
| SD41300 | \$SN_CEC_TABLE_ENABLE | Compensation table enable |
| SD41310 | \$SN_CEC_TABLE_WEIGHT | Weighting factor          |

| Number  | Identifier            | Meaning                                             |
|---------|-----------------------|-----------------------------------------------------|
| SD41320 | \$SN_CEC_0            | Compensation value at the starting position         |
| SD41321 | \$SN_CEC_1            | Compensation value at the end position              |
| SD41330 | \$SN_CEC_BAS_0        | Measuring point P1: Base value                      |
| SD41331 | \$SN_CEC_BAS_1        | Measuring point P2: Base value                      |
| SD41335 | \$SN_CEC_BAS_STORE_0  | Bit memories for measuring point P1: Base value     |
| SD41336 | \$SN_CEC_BAS_STORE_1  | Bit memories for measuring point P2: Base value     |
| SD41340 | \$SN_CEC_COMP_0       | Measuring point P1: Cylinder error                  |
| SD41341 | \$SN_CEC_COMP_1       | Measuring point P2: Cylinder error                  |
| SD41350 | \$SN_CEC_COMP_STORE_0 | Bit memories for measuring point P1: Cylinder error |
| SD41351 | \$SN_CEC_COMP_STORE_1 | Bit memories for measuring point P2: Cylinder error |
| SD41355 | \$SN_CEC_CALC         | Start of calculation by 0/1 edge                    |
| SD41356 | \$SN_CEC_CALC_ADD     | Absolute or additive compensation values            |

**Note****Table index**

All of the subsequently described machine data, setting data and system variables with the same **index <t>** belong to the same compensation table.

**Compensation function**

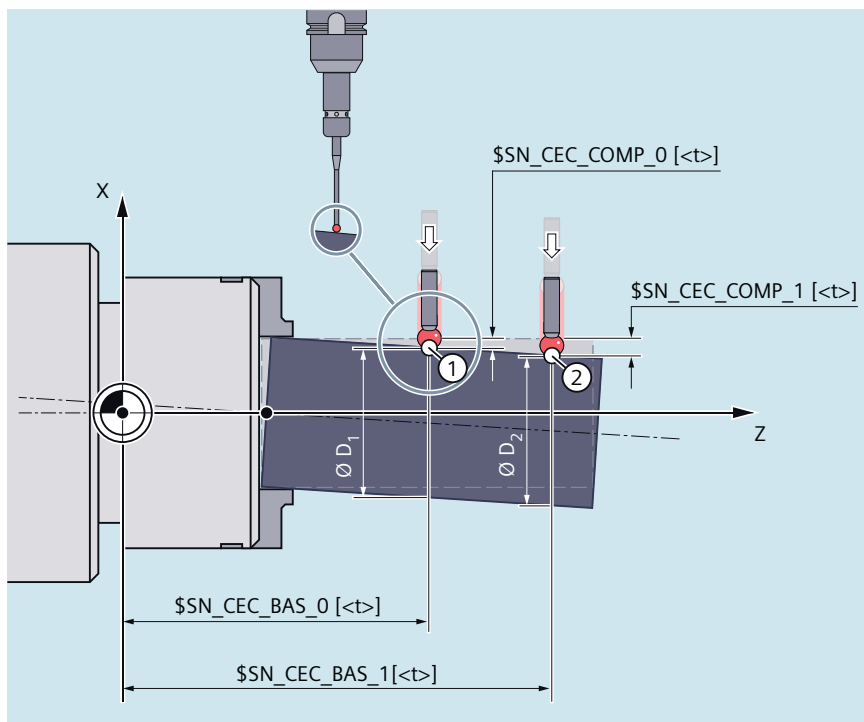
The compensation function of the cylinder error compensation is a straight line with the form:

$$\Delta X = m \cdot Z + b$$

- $\Delta X$  Compensation value for the set position of the compensation axis
- $m$  Gradient of the compensation straight lines
- $Z$  Set position of the basic axis
- $b$  Offset of the compensation straight lines

**Measuring points**

To enable the control to calculate the compensation function, the error in the compensation axis must be measured at two points on the clamped cylindrical workpiece. In the following figure, in accordance with the machining plane G18 (Z/X) that is usual in grinding, the basic axis is designated Z and the compensation axis, X.



① Measuring point P1

② Measuring point P2

\$SN\_CEC\_BAS\_0 / \_1      Positions of the measuring points in the basic axis (Z)

\$SN\_CEC\_COMP\_0 / \_1      Cylinder error at the measuring points in the compensation axis (X)

-Ø D<sub>1</sub> / Ø D<sub>2</sub>      The following applies: Ø D<sub>1</sub> == Ø D<sub>2</sub>

The measuring points must meet the following condition:

$$Z_{P1} < Z_{P2} \Rightarrow \$SN\_CEC\_BAS\_0 < \$SN\_CEC\_BAS\_1$$

## Basic commissioning

Cylinder error compensation is based on sag compensation (CEC) (Page 247). As a consequence, before commissioning the cylinder error compensation, the following compensation parameters must first be set in the system variables of the sag compensation. To this end, the required values are written to the system variables in an NC program.

| Program code                                                               | Comment                                                                                   |
|----------------------------------------------------------------------------|-------------------------------------------------------------------------------------------|
| <code>\$AN_CEC_INPUT_AXIS[&lt;t&gt;] = "&lt;basic axis&gt;"</code>         | ; name of the basic axis                                                                  |
| <code>\$AN_CEC_OUTPUT_AXIS[&lt;t&gt;] = "&lt;compensation axis&gt;"</code> | ; name of the compensation axis                                                           |
| <code>\$AN_CEC_MIN[&lt;t&gt;] = &lt;starting position&gt;</code>           | ; starting position of the traversing range in which the compensation should be effective |
| <code>\$AN_CEC_MAX[&lt;t&gt;] = &lt;end position&gt;</code>                | ; end position of the traversing range in which the compensation should be effective      |

| Program code                                                 | Comment                                                                   |
|--------------------------------------------------------------|---------------------------------------------------------------------------|
| \$AN_CEC_STEP[<t>] = <distance between interpolation points> | ; distance between interpolation point = end position - starting position |
| \$AN_CEC_DIRECTION[<t>] = 0                                  | ; no dependency on the direction                                          |
| \$AN_CEC_IS_MODULO[<t>] = FALSE                              | ; no modulo functionality                                                 |
| \$AN_CEC_MULT_BY_TABLE[<t>] = 0                              | ; no multiplication with another table                                    |
| \$AN_CEC_TYPE = 1                                            | ; compensation type = cylinder error compensation                         |

### Distance between interpolation points (\$AN\_CEC\_STEP[t])

Because the compensation function is a straight line, only two interpolation points are required for the cylinder error compensation. The distance between the interpolation points is therefore the difference between the end and the starting position:

$$\$AN\_CEC\_STEP[<t>] = \$AN\_CEC\_MAX[<t>] - \$AN\_CEC\_MIN[<t>]$$

## Setting the cylinder error compensation

The following shows the principle approach when setting cylinder compensation.

1. Definition of whether the actual compensation values take effect absolutely or relatively:

| Setting data |                        | Meaning                                                             |
|--------------|------------------------|---------------------------------------------------------------------|
| SD41356      | \$SN_CEC_CALC_ADD[<t>] | Absolute or additive compensation values (0: absolute, 1: relative) |

2. Definition of the weighting factor to which the compensation values are added:

| Setting data |                            | Meaning          |
|--------------|----------------------------|------------------|
| SD41310      | \$SN_CEC_TABLE_WEIGHT[<t>] | Weighting factor |

The characteristic curve can be moved vertically via the weighting factor.

3. Determining the **error** in the direction of the compensation axis (X) at two measuring points.
4. Transfer the measuring points (basic value and error or compensation value) to the compensation data:

|                    | Base value (Z)              | Compensation value (X)       |
|--------------------|-----------------------------|------------------------------|
| Measuring point P1 | SD41330 \$SN_CEC_BAS_0[<t>] | SD41340 \$SN_CEC_COMP_0[<t>] |
| Measuring point P2 | SD41331 \$SN_CEC_BAS_1[<t>] | SD41341 \$SN_CEC_COMP_1[<t>] |

5. Triggering a positive edge change (0→1) at the start of calculation of the compensation function:

| Setting data |                    | Meaning                          |
|--------------|--------------------|----------------------------------|
| SD41355      | \$SN_CEC_CALC[<t>] | Start of calculation by 0→1 edge |

6. The control calculates the compensation values for the starting and end points of the compensation straight lines, absolute or relative, depending on setting data SD41356 \$SN\_CEC\_CALC\_ADD[<t>]:

Absolute:

- SD41320 \$SN\_CEC\_0[<t>] = <calculated compensation value at the starting position>
- SD41321 \$SN\_CEC\_1[<t>] = <calculated compensation value at the end position>

Relative:

- SD41320 \$SN\_CEC\_0[<t>] = SD41320 \$SN\_CEC\_0[<t>] + <calculated compensation value at the starting position>
- SD41321 \$SN\_CEC\_1[<t>] = SD41321 \$SN\_CEC\_1[<t>] + <calculated compensation value at the end position>

| Setting data |                 | Meaning                                     |
|--------------|-----------------|---------------------------------------------|
| SD41320      | \$SN_CEC_0[<t>] | Compensation value at the starting position |
| SD41321      | \$SN_CEC_1[<t>] | Compensation value at the end position      |

The measured values are written into bit memories after the calculation and then deleted:

- Bit memory = measured value
- Measured value = 0.0

| Setting data            |                            | Meaning                                             |
|-------------------------|----------------------------|-----------------------------------------------------|
| Bit memory              |                            |                                                     |
| SD41335                 | \$SN_CEC_BAS_STORE_0[<t>]  | Bit memories for measuring point P1: Base value     |
| SD41350                 | \$SN_CEC_COMP_STORE_0[<t>] | Bit memories for measuring point P1: Cylinder error |
| SD41336                 | \$SN_CEC_COMP_STORE_0[<t>] | Bit memories for measuring point P2: Base value     |
| SD41351                 | \$SN_CEC_COMP_STORE_1[<t>] | Bit memories for measuring point P2: Cylinder error |
| Deleted measured values |                            |                                                     |
| SD41330                 | \$SN_CEC_BAS_0[<t>]        | Measuring point P1: Base value                      |
| SD41340                 | \$SN_CEC_COMP_0[<t>]       | Measuring point P1: Cylinder error                  |
| SD41331                 | \$SN_CEC_BAS_1[<t>]        | Measuring point P2: Base value                      |
| SD41341                 | \$SN_CEC_COMP_1[<t>]       | Measuring point P2: Cylinder error                  |

7. Set enable for use of the compensation table:

| Setting data |                            | Meaning                   |
|--------------|----------------------------|---------------------------|
| SD41300      | \$SN_CEC_TABLE_ENABLE[<t>] | Compensation table enable |

#### Note

##### SINUMERIK Toolbox

File CNC\_TAB\_CONF.MPF is also provided in the SINUMERIK Toolbox. It includes an example how cylinder error compensation can be parameterized within an NC program.

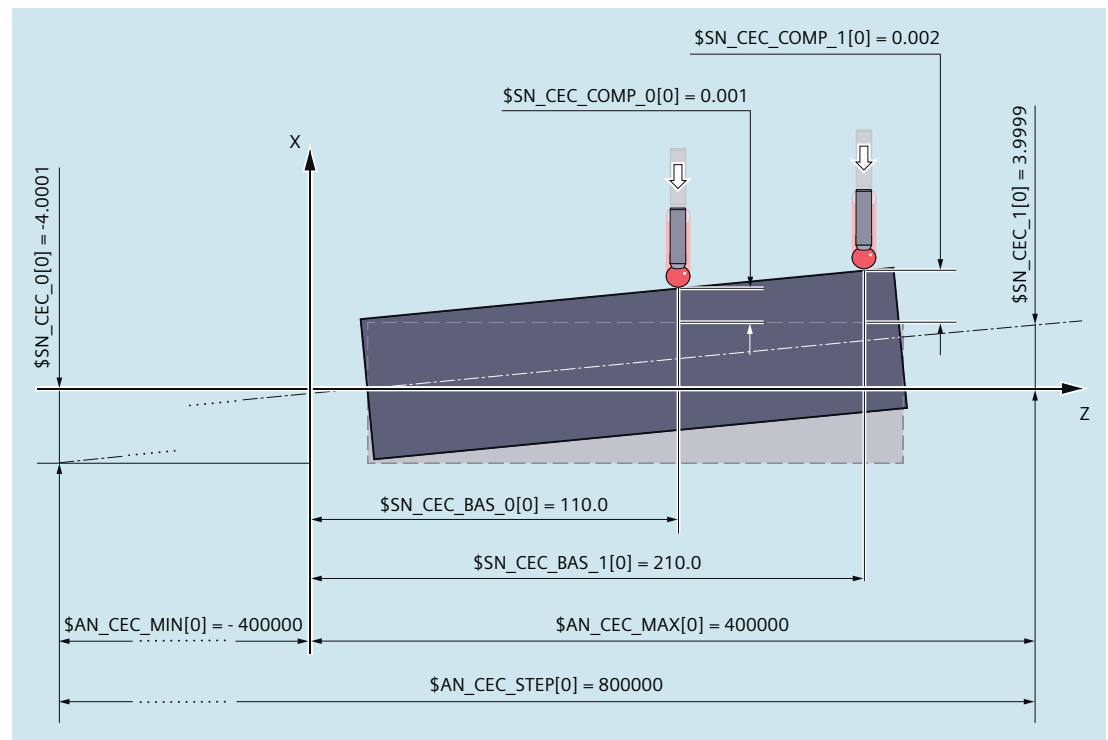
### 8.10.3 Examples

In the following examples, for the cylinder error compensation, the first compensation table (index 0) is used.

#### Example 1: Cylinder error compensation with absolute compensation values

##### Overview of the compensation parameters determining the characteristic

For the example, the compensation parameters shown in the following figure are used.



##### General compensation data

| Setting data \$SN_... |                     | Value | Meaning                                           |
|-----------------------|---------------------|-------|---------------------------------------------------|
| SD41300               | CEC_TABLE_ENABLE[0] | FALSE | Enable                                            |
| SD41310               | CEC_TABLE_WEIGHT[0] | 1     | Weighting factor                                  |
| SD41356               | CEC_CALC_ADD[0]     | FALSE | <b>Absolute</b> compensation values <sup>1)</sup> |

1) The compensation values calculated below are written into the setting data SD41320 \$SN\_CEC\_0[0] and SD41321 \$SN\_CEC\_1[0], i.e. they are applied absolutely. In case of TRUE, they would be added to the existing compensation values.

##### Axes, compensation range, and table properties

| System variable \$AN_... | Value | Meaning           |
|--------------------------|-------|-------------------|
| CEC_INPUT_AXIS[0]        | (Z)   | Basic axis        |
| CEC_OUTPUT_AXIS[0]       | (X)   | Compensation axis |

| System variable \$AN_... | Value       | Meaning                                                                         |
|--------------------------|-------------|---------------------------------------------------------------------------------|
| CEC_MIN[0]               | -400,000 mm | Starting position                                                               |
| CEC_MAX[0]               | 400,000 mm  | End position                                                                    |
| CEC_STEP[0]              | 800,000 mm  | Distance between interpolation points                                           |
| CEC_DIRECTION[0]         | 0           | The compensation table applies to both traversing directions of the basic axis. |
| CEC_IS_MODULO[0]         | FALSE       | No modulo function is active                                                    |
| CEC_MULT_BY_TABLE[0]     | 0           | No table multiplication is active                                               |

### Measuring points

| Setting data \$SN_... | Value    | Meaning                            |
|-----------------------|----------|------------------------------------|
| SD41330 CEC_BAS_0[0]  | 110.0 mm | Measuring point P1: Base value     |
| SD41331 CEC_BAS_1[0]  | 210.0 mm | Measuring point P2: Base value     |
| SD41340 CEC_COMP_0[0] | 0.001 mm | Measuring point P1: Cylinder error |
| SD41341 CEC_COMP_1[0] | 0.002 mm | Measuring point P2: Cylinder error |

### Calculation of the compensation values

Calculation of the compensation values is performed by the control based on a rising edge (0→1) in setting data SD41355 \$SN\_CEC\_CALC[0]. The calculations shown below are therefore only for illustration.

- Compensation function  $\Delta X = f(Z)$ :

$$\Delta X = m \cdot Z + b = \frac{X_{P2} - X_{P1}}{Z_{P2} - Z_{P1}} \cdot Z + b \quad (1)$$

- Calculation of the gradient m:

$$\begin{aligned}
 m &= \frac{\text{SD41341 } \$SN\_CEC\_COMP\_1[0] - \text{SD41340 } \$SN\_CEC\_COMP\_0[0]}{\text{SD41331 } \$SN\_CEC\_BAS\_1[0] - \text{SD41330 } \$SN\_CEC\_BAS\_0[0]} = \\
 &= \frac{0,002 - 0,001}{210,0 - 110,0} = 1 \cdot 10^{-5} \quad (2)
 \end{aligned}$$

- Calculating the offset b → measuring point P1(110.0, 0.001) and (2) used in (1):

$$\begin{aligned}
 0,001 &= 1 \cdot 10^{-5} \cdot 110,0 + b \\
 b &= -1 \cdot 10^{-4} \quad (3)
 \end{aligned}$$

- Resulting compensation function → (3) and (2) used in (1):

$$\Delta X = 1 \cdot 10^{-5} \cdot Z - 1 \cdot 10^{-4}$$

- Calculating the compensation values → starting or end position used in (4):

$$\begin{aligned}
 \text{SD41320 } \$SN\_CEC\_0[0] &= \Delta X_{\text{COMP}_0} = 1 \cdot 10^{-5} \cdot -400000 - 1 \cdot 10^{-4} = -4,0001 \\
 \text{SD41321 } \$SN\_CEC\_1[0] &= \Delta X_{\text{COMP}_1} = 1 \cdot 10^{-5} \cdot 400000 - 1 \cdot 10^{-4} = 3,9999
 \end{aligned}$$

| Setting data \$SN_... |                     | Value      | Meaning                                             |
|-----------------------|---------------------|------------|-----------------------------------------------------|
| SD41320               | CEC_0[0]            | -4.0001 mm | Compensation value at the starting position         |
| SD41321               | CEC_1[0]            | 3.9999 mm  | Compensation value at the end position              |
| SD41330               | CEC_BAS_0[0]        | 0.0 mm     | Measuring point P1: Base value (deleted)            |
| SD41331               | CEC_BAS_1[0]        | 0.0 mm     | Measuring point P2: Base value (deleted)            |
| SD41335               | CEC_BAS_STORE_0[0]  | 110.0 mm   | Bit memories for measuring point P1: Base value     |
| SD41336               | CEC_BAS_STORE_1[0]  | 210.0 mm   | Bit memories for measuring point P2: Base value     |
| SD41340               | CEC_COMP_0[0]       | 0.0 mm     | Measuring point P1: Cylinder error (deleted)        |
| SD41341               | CEC_COMP_1[0]       | 0.0 mm     | Measuring point P2: Cylinder error (deleted)        |
| SD41350               | CEC_COMP_STORE_0[0] | 0.001 mm   | Bit memories for measuring point P1: Cylinder error |
| SD41351               | CEC_COMP_STORE_1[0] | 0.002 mm   | Bit memories for measuring point P2: Cylinder error |

### Example 2: Cylinder error compensation with relative compensation values

Example 2 is based on the compensation data of example 1. Cylinder error compensation is also performed using the first compensation table (index 0).

In example 2, the cylinder error is determined only at two measuring points, unlike example 1. The compensation values are to apply relatively, i.e. they are **added** to the existing compensation values from example 1.

#### General compensation data

| Setting data \$SN_...                                                                                                             |                 | Value | Meaning                                    |
|-----------------------------------------------------------------------------------------------------------------------------------|-----------------|-------|--------------------------------------------|
| SD41356                                                                                                                           | CEC_CALC_ADD[0] | TRUE  | Relative compensation values <sup>1)</sup> |
| 1) The compensation values calculated below are <b>added</b> to the setting data SD41320 \$SN_CEC_0[0] and SD41321 \$SN_CEC_1[0]. |                 |       |                                            |

#### Measuring points

| Setting data \$SN_... |               | Value     | Meaning                            |
|-----------------------|---------------|-----------|------------------------------------|
| SD41330               | CEC_BAS_0[0]  | 10.0 mm   | Measuring point P1: Base value     |
| SD41331               | CEC_BAS_1[0]  | 410.0 mm  | Measuring point P2: Base value     |
| SD41340               | CEC_COMP_0[0] | 0.0 mm    | Measuring point P1: Cylinder error |
| SD41341               | CEC_COMP_1[0] | -0.003 mm | Measuring point P2: Cylinder error |

**Calculation of the compensation values**

Calculation of the compensation values is performed by the control based on a rising edge (0→1) in setting data SD41355 \$SN\_CEC\_CALC[0]:

- Compensation function:

$$\Delta X = m * Z + b = -7,5 * 10^{-6} * Z + 7,5 * 10^{-5}$$

- Calculated compensation values:

$$\begin{aligned}\Delta X_{\text{COMP}_0} &= 3,000075 \text{ mm} \\ \Delta X_{\text{COMP}_1} &= -2,999925 \text{ mm}\end{aligned}$$

- Adding the calculated compensation values to the actual compensation values:

$$\begin{aligned}\text{SD41320 } \$\text{SN\_CEC\_CEC\_0[0]} &= \text{SD41320 } \$\text{SN\_CEC\_CEC\_0[0]} + \Delta X_{\text{COMP}_0} = \\ &= -4.0001 \text{ mm} + 3,000075 \text{ mm} = -1,000025 \text{ mm} \\ \text{SD41321 } \$\text{SN\_CEC\_CEC\_1[0]} &= \text{SD41321 } \$\text{SN\_CEC\_CEC\_1[0]} + \Delta X_{\text{COMP}_1} = \\ &= 3.9999 \text{ mm} - 2,999925 \text{ mm} = 0,999975 \text{ mm}\end{aligned}$$

| Setting data \$SN_CEC_... |                     | Value        | Meaning                                             |
|---------------------------|---------------------|--------------|-----------------------------------------------------|
| SD41320                   | CEC_0[0]            | -1.000025 mm | Compensation value at the starting position         |
| SD41321                   | CEC_1[0]            | 0.999975 mm  | Compensation value at the end position              |
| SD41330                   | CEC_BAS_0[0]        | 0.0 mm       | Measuring point P1: Base value (deleted)            |
| SD41331                   | CEC_BAS_1[0]        | 0.0 mm       | Measuring point P2: Base value (deleted)            |
| SD41335                   | CEC_BAS_STORE_0[0]  | 10.0 mm      | Bit memories for measuring point P1: Base value     |
| SD41336                   | CEC_BAS_STORE_1[0]  | 410.0 mm     | Bit memories for measuring point P2: Base value     |
| SD41340                   | CEC_COMP_0[0]       | 0.0 mm       | Measuring point P1: Cylinder error (deleted)        |
| SD41341                   | CEC_COMP_1[0]       | 0.0 mm       | Measuring point P2: Cylinder error (deleted)        |
| SD41350                   | CEC_COMP_STORE_0[0] | 0.0 mm       | Bit memories for measuring point P1: Cylinder error |
| SD41351                   | CEC_COMP_STORE_1[0] | -0.003 mm    | Bit memories for measuring point P2: Cylinder error |

## 8.11 IC: Constraints

### Compensated actual position

The following functions are based on the compensated actual position:

- Measurement
- Teach In
- Software limit switch

## Displaying the actual position

The actual position of the axis without compensation (ideal machine) is displayed in the actual position display in the machine coordinate system.

The actual position of the axis with compensation (MSEC and backlash compensation) is displayed in the service display "Axis/spindle" ("Diagnostics" operating area).

## Displaying the compensation values

The following compensation values are also output in the "Axis/spindle" service display (operating area "Diagnosis"):

- "Absolute compensation value measuring system 1" or 2  
The value displayed is the sum of the compensation values of MSEC, obtained from the actual position of the basic and compensation axis, and backlash compensation.
- Compensation, sag + temperature  
The value displayed is the sum of the compensation values of sag and temperature compensation, obtained from the actual position of the basic and compensation axis.

## Loss of the referenced status of the basic axis

If the reference status of the active measuring system of the basic axis changes to **"Not referenced/synchronized"**, the MSEC and/or sag compensation is deactivated in the corresponding axes (basic/compensation axis):

- `<Axis>.basic.in.enc1Synchronized 1 → 0` (referenced / synchronized 1) or
- `<Axis>.basic.in.enc2Synchronized 1 → 0` (referenced / synchronized 2)

If the reference state of the active measuring system of the basic axis then changes back to "Referenced/synchronized", the MSEC and/or sag compensation is reactivated in the corresponding axes (basic/compensation axis):

- `<Axis>.basic.in.enc1Synchronized 0 → 1` (referenced / synchronized 1) or
- `<Axis>.basic.in.enc2Synchronized 0 → 1` (referenced / synchronized 2)

## Controller enable signals

If a compensation relationship is active, then the controller enable should always be set the same for the basic and compensation axes:

`<Axis>.basic.out.enableCtrl == 1`

## Traversing signal output

The traversing signals of the compensation axis are only when the compensation function is activated/deactivated and when the number of active compensation tables changes.

Traversing motion of the compensation axis, resulting from motion of the basic axis, does not result in the output of traversing signals in the compensation axis.

**PLC signals**

| Basic Program Plus               | Basic Program           |                    |
|----------------------------------|-------------------------|--------------------|
| <Axis>.basic.in.enc1Synchronized | LBP_Axis*.E_RefSyn1     | DB31, ... .DBX60.4 |
| <Axis>.basic.in.enc2Synchronized | LBP_Axis*.E_RefSyn2     | DB31, ... .DBX60.5 |
| <Axis>.basic.out.enableCtrl      | LBP_Axis*.A_ContrEnable | DB31, ... .DBX2.1  |

**8.12 Dynamic feedforward control (following error compensation)****8.12.1 General properties****Axial following error**

The remaining system deviation of the position controller when traversing a machining axis is known as axial following error. In other words, the axial following error is the difference between the setpoint position and the actual position of the machine axis with a velocity which is not equal to "0".

**Effects**

Particularly during acceleration in contour curvatures, e.g. circles and corners, this following error leads to undesirable, velocity-dependent contour violations.

**Compensation**

The axial following error can be reduced almost to zero with the help of the "dynamic feedforward control". The function is therefore also called "following error compensation". Reduction of the following errors in the interpolating axes also ensures that the contour errors are less velocity-dependent and contour distortions are minimized.

**Methods**

There are two "dynamic feedforward control" methods:

- Speed feedforward control (velocity-dependent)
- Torque feedforward control (acceleration-dependent)

**Activation**

The feedforward control method is selected and activated using the machine data:

MD32620 \$MA\_FFW\_MODE (feedforward control mode)

Setting \$MA\_FFW\_MODE = 4 is recommended to avoid contour errors.

| Value | Meaning                                       |
|-------|-----------------------------------------------|
| 0     | No feedforward control                        |
| 1     | Speed feedforward control with PT1 balancing  |
| 2     | Torque feedforward control with PT1 balancing |
| 3     | Speed feedforward control with Tt balancing   |
| 4     | Torque feedforward control with Tt balancing  |

Modes 1 and 2 are additionally provided for compatibility purposes.

### Activation/deactivation in part program

The following axis-specific machine data can be used to define that the feedforward control for the respective axis/spindle can be activated and deactivated by the part program:

MD32630 \$MA\_FFW\_ACTIVATION\_MODE (activate feedforward control from program)

| Value | Meaning                                                                                                                                                                                      |
|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0     | The feedforward control cannot be activated and deactivated from the part program. This means that the state specified using MD32620 \$MA_FFW_MODE is always effective for the axis/spindle. |
| 1     | The feedforward control can be activated and deactivated from the part program. The operation becomes active immediately.                                                                    |
| 2     | The feedforward control can be activated and deactivated from the part program. The operation only becomes active the next time that the axis comes to a standstill.                         |

The feedforward control is activated/deactivated from the part program using the operations:

FFWON: Feedforward control ON

FFWOF: Feedforward control OFF

The default setting (i.e. M30 even after reset) is entered using the channel-specific machine data:

MD20150 \$MC\_GCODE\_RESET\_VALUES (initial setting of the G groups)

FFWON/FFWOF is active for all axes/spindles in the axis mode, where:

MD32630 \$MA\_FFW\_ACTIVATION\_MODE = 1 (or 2)

and

MD32620 \$MA\_FFW\_MODE = 1, 2, 3 or 4

The identical MD32630 setting should be used for axes that interpolate with each other.

The feedforward control should only be activated or deactivated while the axis/spindle is stationary in the axis mode, in order to prevent jerky motion. Hence the switchover is delayed automatically up to the next standstill through block search stop.

If the axes are optimized with feedforward control, it is recommended that the feedforward control is not switched off in the part program.

MD32630=0 is recommended in this regard.

### Note

A preprocessing stop has no effect for command or PLC axes traversing asynchronously to the part program processing. To ensure that `FFWON/FFWOF` only has an effect on the axis/spindle when it is next stationary in the axis mode, you must explicitly set `MD32630 = 2` for each axis/spindle in the axis mode (see also "Forward feed control for command- and PLC axes (Page 296)").

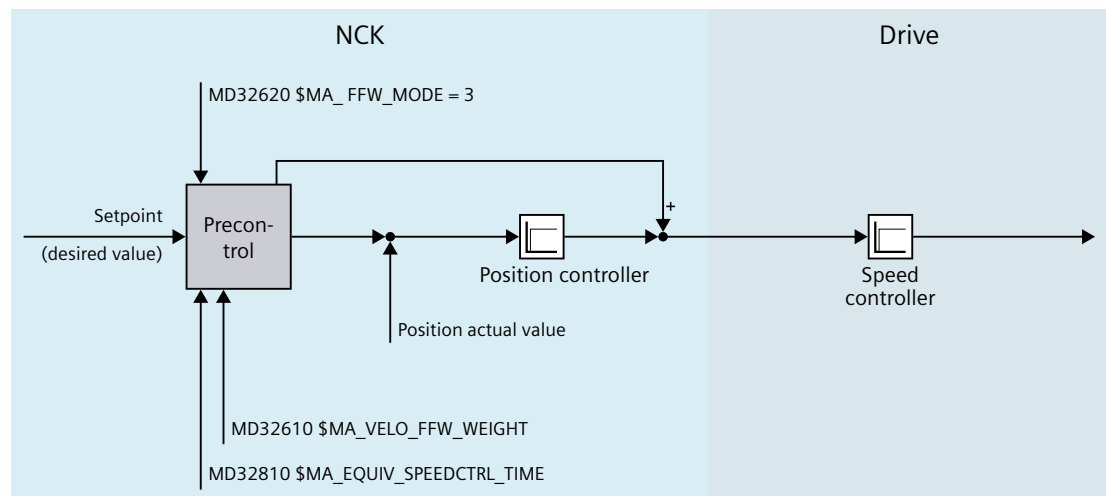
## 8.12.2 Speed feedforward control

### Function

In the case of speed feedforward control, a velocity setpoint is also applied directly to the input of the speed controller. With this value the following error can be reduced to nearly zero (i.e. system deviation is 0) when the velocity is constant.

### Commissioning

The following axis-specific parameters must be defined for the speed feedforward control during commissioning:



### Equivalent time constant of the speed control loop (MD32810)

The equivalent time constant of the speed control loop must be determined accurately (e.g. graphically from a speed setpoint step response) and entered into the following machine data to correctly set the speed feedforward control:

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

**Feedforward control factor for speed feedforward control (MD32610)**

The additional velocity setpoint can be weighted using a factor:

MD32610 \$MA\_VELO\_FFW\_WEIGHT

Value range: 0 ... 1

"0" means: no feedforward control. As standard, the factor has a value of 1 ( $\hat{=}$  100%).

The factor should remain set at 100%, as this value is the optimum setting for an optimally set control loop for the axis/spindle as well as a precisely determined equivalent time constant of the speed control loop.

**Fine adjustment**

The speed feedforward control for the particular axis/spindle can be optimized by making slight changes (fine tuning) to the equivalent time constants of the speed control loop (MD32810).

Fine adjustment (equivalent time constant, moment of inertia) can also be determined automatically via AST (AutoServoTuning). Calibration of the interpolating axes ("Dynamic response adaptation") is likewise performed by the AST.

To make this check, the axis/spindle should be traversed at a constant velocity and in the service display "Axis/spindle", the "System deviation" should be checked.

A small acceleration and a high feedrate should be chosen so that the values can be easily read on the service display. This produces very long acceleration phases from which it is easy to read off the control deviation.

Example:

|                                    |                                                    |
|------------------------------------|----------------------------------------------------|
| MD32300 \$MA_MAX_AX_ACCEL = 0.1    | ; Maximum axis acceleration = 0.1 m/s <sup>2</sup> |
| MD32000 \$MA_MAX_AX_VELO = 20000.0 | ; Maximum axis velocity<br>= 20000.0 mm/min        |

```
; Part program for setting the equivalent time constant
G1 F20000
FFWON
LOOP:
X1000
X0
GOTOB LOOP
M30
```

**Further information**

For detailed information about setting the equivalent time constants of the speed control loop (MD32810) refer to:

- Function Manual Axes and Spindles; Velocities, setpoint/actual value systems, closed-loop control (G2), optimizing the closed-loop control

### 8.12.3 Torque feedforward control

#### Function

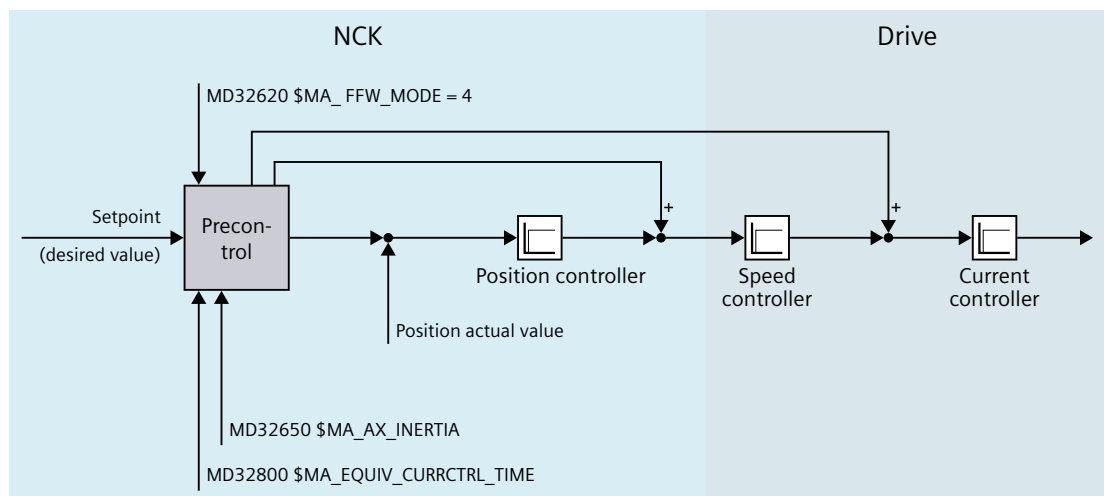
In the case of torque feedforward control, an additional current setpoint proportional to the torque is applied directly to the current controller input. This value is formed using the acceleration and moment of inertia.

#### Application

Torque feedforward control is required to achieve high contour accuracy where the demands on the dynamic response are very high. If set correctly, the following error can almost be completely compensated even during high acceleration.

#### Commissioning

The following axis-specific parameters must be defined during commissioning for torque feedforward control:



#### Equivalent time constant of the current control loop (MD32800)

The equivalent time constant of the current control loop must be determined accurately (e.g. graphically from the step response of the current control loop) and entered in the following machine data in order to correctly set the torque feedforward control:

MD32800 \$MA\_EQUIV\_CURRCTRL\_TIME (equivalent time constant current control loop for feedforward control)

#### Total moment of inertia of axis (MD32650)

The torque feedforward control already offers advantages over the speed feedforward control - even if the actual moment of inertia is not precisely known. It is recommended for load-dependent axes that the empty axis is measured and the resulting total inertia applied. For fine tuning of the AX\_INERTIA::

MD32650 \$MA\_AX\_INERTIA (inertia for torque feedforward control)

**Fine adjustment**

The torque feedforward control for the particular axis/spindle can be optimized by making slight changes (fine tuning) to the values in MD32800 and MD32650. Fine adjustment (equivalent time constant, moment of inertia) can be determined automatically via AST (AutoServoTuning). Calibration of the interpolating axes ("Dynamic response adaptation") is likewise performed by the AST.

The position controller difference can be recorded via the trace functionality as verification. In addition to traversing at a constant velocity, the following error should be monitored especially when the axis/spindle accelerates.

**Note**

As a result of the extremely fast sequences when accelerating, when commissioning the torque feedforward control, the service display cannot be used to check the fine adjustment.

**Further information**

For detailed information about setting the equivalent time constants of the current control loop (MD32810) refer to:

- Function Manual Axes and Spindles; Velocities, setpoint/actual value systems, closed-loop control, optimizing the closed-loop control

**8.12.4 Dynamic response adaptation****Function**

For axes that interpolate with one another, but with different axial control loop response times, dynamic response adaptation can be used to achieve identical time responses of all axes to ensure optimum contour accuracy without loss of control quality.

**Commissioning****Time constant for dynamic response adaptation (MD32910)**

The difference between the equivalent time constants of the "slowest" speed or current control loop and the particular axis should be entered as time constant for the dynamic response adaptation in the following machine data.

MD32910 \$MA\_DYN\_MATCH\_TIME (time constant of dynamic response adaptation)

Example:

Equivalent time constants of the speed control loop (MD32810) for active speed feedforward control of axes 1, 2 and 3:

- Axis 1: 2 ms
- Axis 2: 4 ms (dynamically the slowest axis)
- Axis 3: 1 ms

## 8.12 Dynamic feedforward control (following error compensation)

This means that the following values are obtained for the time constant of the dynamic response adaptation MD32910:

- Axis 1: 2 ms
- Axis 2: 0 ms
- Axis 3: 3 ms

### Activation (MD32900)

The dynamic response adaptation is only active if the following machine data is set:

MD32900 \$MA\_DYN\_MATCH\_ENABLE= 1

### Further filters

Additional filters can be used for dynamic response adaptation:

- MD32890 \$MA\_DESVAL\_DELAY\_ENABLE (axial phase filter setpoint)
- MD32895 \$MA\_DESVAL\_DELAY\_TIME (time constant for the axial phase filter setpoint)

### Further information

Function Manual Axes and Spindles; Velocities, setpoint/actual value systems, closed-loop control, optimizing the closed-loop control

## 8.12.5 Forward feed control for command- and PLC axes

### Function

For command and PLC axes, it must be prevented that the feedforward control is activated/deactivated at higher velocities as follows:

MD32630 \$MA\_FFW\_ACTIVATION\_MODE = 2

With this setting, the FFWON/FFWOF operation only becomes active below the stationary velocity (MD36060 \$MA\_STANDSTILL\_VELO\_TOL) configured for this particular axis.

If the switchover instruction coincides with an axis motion, the required switchover is executed only in the next stoppage condition of the axis. This avoids the following error being suddenly established/reduced.

---

#### Note

A stoppage velocity set to a very high value can lead to the changeover of the feedforward control in the movement. Controls can be activated depending on the existing following error.

---

## Commissioning

We recommend the following procedure when checking the feedforward control for command and PLC axes:

1. Check the stoppage velocity in MD36060.
2. Check the existing following error of the axis in stoppage condition.
3. Setting the changeover condition and activating it:  
MD32630 = 2
4. Traverse axis in the part program using the POSA operation.
5. Execute FFWON during the axis motion.
6. The  $K_v$  factor and following error displayed in the service display "Axis/spindle" must not jump.
7. A higher  $K_v$  factor and a lower following error are only obtained for traversing motion following standstill. However, the feedforward control is active only from the stoppage condition.

Essentially the same as when activating the feedforward control, for deactivation, the following applies:

1. Traverse axis in the part program using the POSA operation.
2. Execute FFWOF during the axis motion.
3. The  $K_v$  factor and following error displayed in the service display "Axis/spindle" must not jump.
4. A lower  $K_v$  factor and a higher following error are only obtained for traversing motion following standstill. However, the feedforward control is inactive only from the stoppage condition.

## Example

In the following program example, axis A is traversed asynchronously to the path. An attempt is made to activate the feedforward control in the channel while traversing. Contrary to the geometry axes X, Y and Z, the feedforward control is not immediately effective for axis A. Here one waits for the stoppage after N60. Axis A then traverses with the feedforward control in N70.

### Program code

```
N10 FFWOF
N20 POSA[A]=1000 FA[A]=10000
N30 G4 F1
N40 FFWON
N50 G0 X10 Y10 Z10
N60 WAITP(A)
N70 POSA[A]=1500 FA[A]=10000
N80 WAITP(A)
M30
```

### 8.12.6 Secondary conditions

#### Axes that are interpolating axes with one another

Also for axes that interpolate with one another, the feedforward control parameter should be optimally set **for each axis**, i.e. also several axes that are interpolating with one another can have different feedforward control parameters.

#### Check contour monitoring

As the two equivalent time constants:

- MD32810 \$MA\_EQUIV\_SPEEDCTRL\_TIME (equivalent time constant speed control loop for feedforward control)

and

- MD32800 \$MA\_EQUIV\_CURRCTRL\_TIME (equivalent time constant current control loop for feedforward control)

also influence the contour monitoring, this should be subsequently checked.

#### Further information

Axis monitoring functions (Page 145)

#### Influence on the servo gain factor ( $K_v$ factor)

When the feedforward control is set correctly, the response to setpoint changes in the controlled system under speed feedforward control is as dynamic as that of the speed control loop or, under torque feedforward control, as that of the current control loop, i.e. the servo gain factor ( $K_v$  factor) entered into MD32200 \$MA\_POS\_CTRLGAIN hardly has any effect on the control behavior (e.g. corner errors, overshoots, circle/radius errors).

On the other hand, feedforward control does not affect the response to disturbances (synchronism). In this case, the servo gain factor ( $K_v$  factor) entered in MD32200 is the active factor.

#### Service display " $K_v$ factor"

When a feedforward control is active, the servo gain ( $K_v$  factor) of the axis (corresponds to servo gain factor ( $K_v$  factor) active as response to setpoint changes) shown in the service display "axis/spindle" is very high.

## 8.13 Friction compensation overview

In addition to the mass inertia and the machining forces, the friction forces in the gearing and guideways of the machine influence the dynamic behavior of a machine axis. In particular, when accelerating a machine axis from standstill, the transition from static to sliding friction has an adverse effect. The resulting sudden change in the friction force results in a briefly increased following error. With interpolating axes, this results in significant contour violations. For circles, these occur especially at the quadrant transitions where one of the participating axes is stationary as its direction is reversed.

To improve the contour accuracy at these points, an additional setpoint pulse (velocity or torque setpoint pulse) is applied to the friction compensation following a direction reversal by the machine axis (during acceleration).

### Function versions

The following function variants are available for friction compensation:

- Friction compensation with a constant compensation value (Page 299)  
Depending on the acceleration of the machine axis, the same pulse is always applied to the velocity setpoint.  
The amplitude and the decay time of the velocity setpoint pulse can be set.
- Friction compensation with adaptive characteristic (Page 306)  
Depending on the acceleration of the machine axis, the velocity setpoint pulse is determined from a parameterizable characteristic.  
Three different acceleration values can be set as well as the minimum and maximum amplitude and the decay time of the velocity setpoint pulse.
- Friction compensation with adaptive characteristics (Page 310) (Licensed option: 6FC5800-0AS06-0YB0)  
Depending on the acceleration of the machine axis, the velocity setpoint pulse is determined from up to three parameterizable characteristics.  
Up to nine different acceleration values can be set. For each acceleration value, weighting factors for the amplitude, the active time, and the decay time of the velocity setpoint pulse can be defined.  
If the velocity setpoint pulse is not sufficient to achieve the desired result, an additional torque setpoint pulse can be parameterized.

## 8.14 Friction compensation with a constant compensation value

### 8.14.1 Description of functions

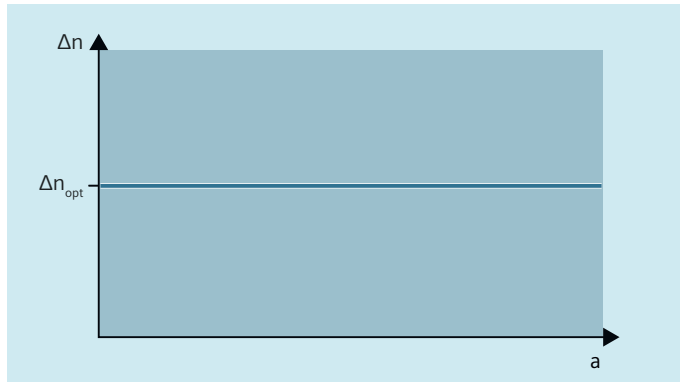
The friction compensation described below with constant compensation value is intended for applications in which the following general conditions apply:

- The velocity setpoint pulse required for friction compensation is **independent** of the acceleration.
- An acceleration-**independent** amplitude and an acceleration-**independent** decay time are sufficient to model the velocity setpoint pulse.

## 8.14 Friction compensation with a constant compensation value

- For friction compensation at the positive and negative reversal point, the same velocity setpoint pulse is sufficient.
- The requirement for contour accuracy at the reversal points is relatively low.

The following figure shows an example characteristic that can be set with this function:



$\Delta n_{\text{opt}}$  Amplitude of the velocity setpoint pulse

$a$  Acceleration at the quadrant transition

## 8.14.2 Commissioning

To determine the axis-specific compensation value  $\Delta n_{\text{opt}}$ , the Circularity test (Page 301) must be used to determine the optimum amplitude of the velocity setpoint pulse  $\Delta n_{\text{opt}_a}$  for each of a number of acceleration values. The different acceleration values should cover the entire dynamic range.

**Calculation of the acceleration at the quadrant transition**

On a circular path, the acceleration  $a$  of a machine axis on direction reversal at the quadrant transition is calculated from the radius  $r$  of the circle and the path velocity  $v$ :

$$a = \frac{v^2}{r}$$

**Note**

The path velocity and therefore the axis-specific acceleration  $a$  can be varied simply via the feedrate override.

**Evaluation of the value pairs determined ( $a$ ,  $\Delta n_{\text{opt}_a}$ )**

To determine the compensation value  $\Delta n_{\text{opt}}$ , we recommend graphically applying the value pairs determined in the circularity test comprising the acceleration  $a$  and the associated optimum amplitude of the velocity setpoint pulse  $\Delta n_{\text{opt}_a}$ :

$$\Delta n_{\text{opt}_a} = f(a)$$

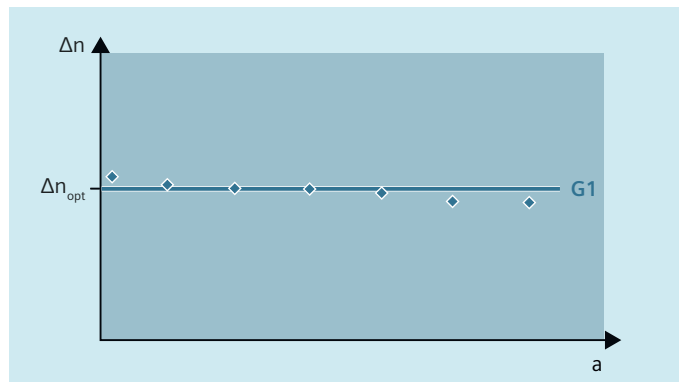


Figure 8-19 Axis-specific characteristic determination

The optimum amplitude of the velocity setpoint pulse  $\Delta n_{\text{opt}}$  is obtained by drawing in the straight line G1.

### Axis-specific machine data

#### Activating friction compensation

Friction compensation is activated with:

- MD32500 \$MA\_FRICT\_COMP\_ENABLE[ <axis> ] = **TRUE (1)**

#### Activating friction compensation with a constant compensation value

Friction compensation with constant compensation value is activated with:

- MD32490 \$MA\_FRICT\_COMP\_MODE[ <axis> ] = **1**
- MD32510 \$MA\_FRICT\_COMP\_ADAPT\_ENABLE[ <axis> ] = **FALSE (0)**

#### Velocity setpoint pulse: Amplitude

The amplitude of the velocity setpoint pulse is set with:

MD32520 \$MA\_FRICT\_COMP\_CONST\_MAX[ <axis> ] = <amplitude>

#### Velocity setpoint pulse: Decay time

The decay time of the velocity setpoint pulse is set via:

MD32540 \$MA\_FRICT\_COMP\_TIME[ <axis> ] = <time constant>

### 8.14.2.1 Circularity test

#### Commissioning via the circularity test

The easiest way to commission the friction compensation is with the circularity test integrated in the user interface. This involves acquiring the circular contour created on the machine as a circle is traversed based on the actual positions of the participating machine axes. Deviations from the programmed ideal circular contour, in particular, at the quadrant transitions, are visualized graphically.

In the SINUMERIK Operate user interface, the circularity test can be found under:  
"Operating area switchover" > "Commissioning" > "Optimization/test" > "Circularity test"

### Sequence

The sequence for commissioning friction compensation for a machine axis is divided into the following steps:

1. Set the acceleration  $a_1$  via path velocity  $v_n$  and circle radius  $r$
2. Perform circularity test **without** friction compensation
3. Perform a circularity test **with** friction compensation with small initial values for amplitude and decay time
4. Optimize the circularity test **with** friction compensation by changing the parameter values step by step
5. Plot the optimized parameter values graphically.  
Example: Amplitude =  $f(a_n)$
6. Setting of the next acceleration  $a_n$  and continuation with point 2.

### Performing circularity test without friction compensation

A circularity test without friction compensation should be performed to determine the initial quality of the circular contour at the quadrant transitions. To do this, switch off the friction compensation temporarily:

MD32500 FRICT\_COMP\_ENABLE[ <axis> ] = 0

The following figure shows a typical example of quadrant transitions without friction compensation:

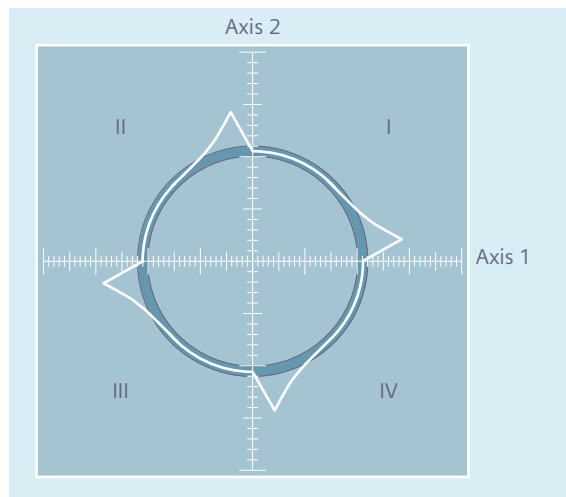


Figure 8-20 Quadrant transitions without friction compensation

The friction compensation must then be activated again:

MD32500 FRICT\_COMP\_ENABLE[ <axis> ] = 1

### Performing circularity test with friction compensation

It is recommended that a small amplitude value and a decay time of just a few position control cycles be set as initial values for the velocity setpoint pulse, e.g.:

- MD32520 \$MA\_FRICT\_COMP\_CONST\_MAX[ <axis> ] = 10 [mm/min]
- MD32540 \$FRICT\_COMP\_TIME[ <axis> ] = 0.008 [s]

The circularity test performed with these values provides an initial assessment of the friction compensation.

#### Amplitude too small

Too small an amplitude value (MD32520) is revealed in the circularity test by insufficient compensation of the contour deviations at the quadrant transitions.

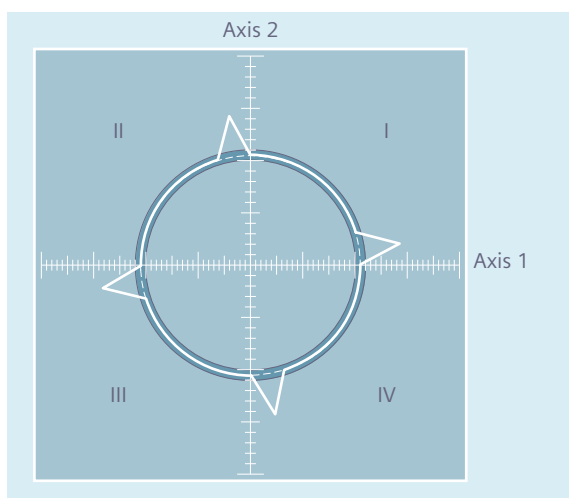


Figure 8-21 Amplitude too small

#### Amplitude too large

Too large an amplitude value (MD32520) is revealed in the circularity test by overcompensation of the contour deviations at the quadrant transitions.

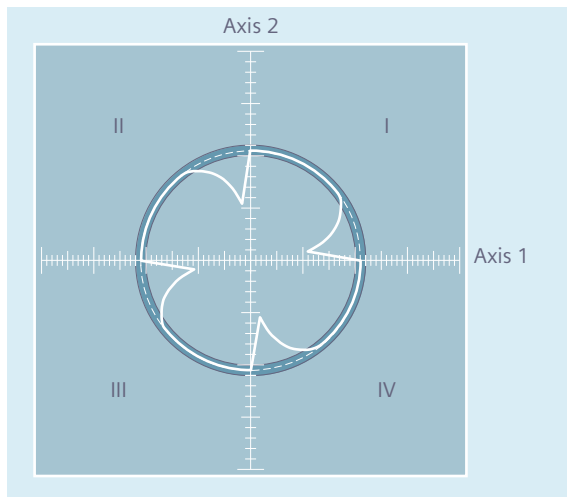


Figure 8-22 Amplitude too large

**Decay time too short**

Too short a decay time (MD32540) is revealed in the circularity test when the contour deviations at the quadrant transitions are compensated for a short time but immediately increase again thereafter.

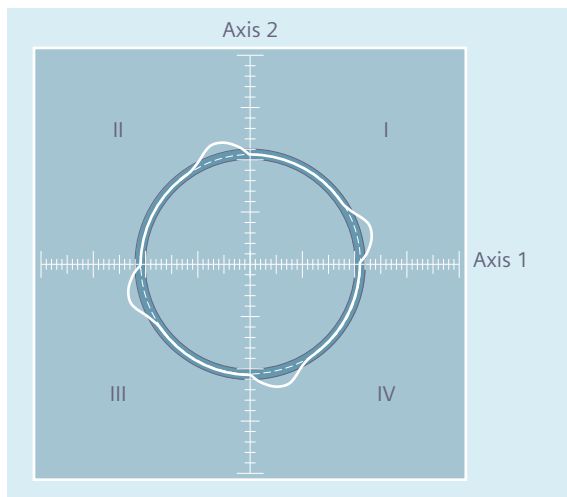


Figure 8-23 Decay time too short

**Decay time too long**

Too long a decay time (MD32540) is revealed in the circularity test when the contour deviation at the quadrant transitions is compensated for at first. This assumes that the optimum amplitude values has already been set. If the decay time is too long, however, the compensation applies for too long and results in overcompensation at the next circular contour.

## 8.14 Friction compensation with a constant compensation value

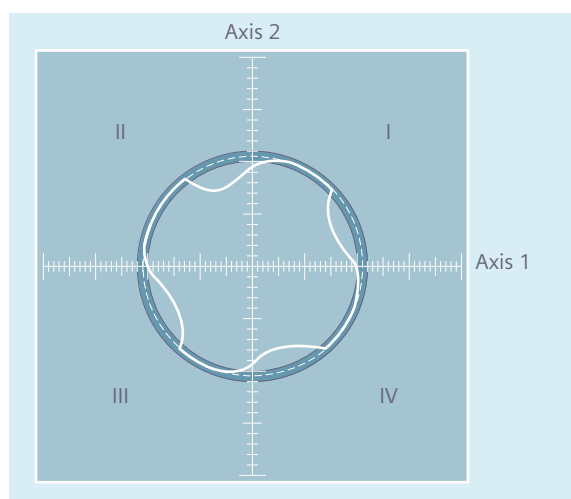


Figure 8-24 Decay time too long

**Good friction compensation setting**

With a good friction compensation setting, "no" contour violations can be detected at the quadrant transitions.

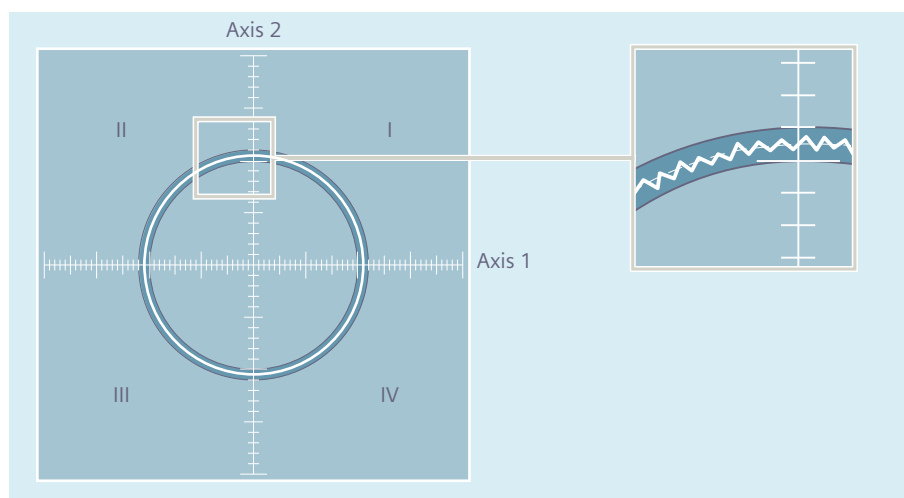


Figure 8-25 Good friction compensation setting

**See also**

Friction compensation with adaptive characteristic (Page 306)

### 8.14.3 Supplementary conditions

#### Reaction of setpoint-related compensation functions

The following setpoint-related compensation functions affect the position setpoint and must be deactivated for the axes that perform a circularity test:

- Sag and angularity compensation (CEC)  
MD32710 \$MA\_CEC\_ENABLE[ <axis> ] = 0
- Directional leadscrew error compensation (LEC):  
MD32710 \$MA\_CEC\_ENABLE[ <axis> ] = 0
- Temperature compensation:  
MD32750 \$MA\_TEMP\_COMP\_TYPE[ <axis> ] = 0

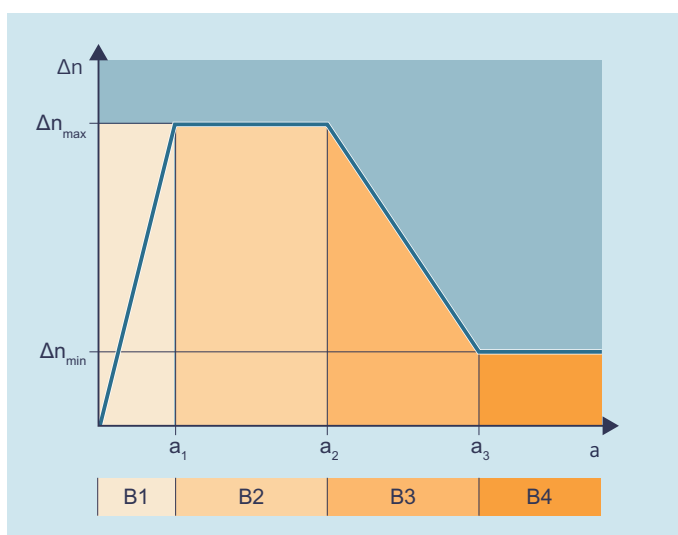
## 8.15 Friction compensation with adaptive characteristic

### 8.15.1 Description of functions

The friction compensation with adaptive characteristic is intended for applications with the following requirements:

- The velocity setpoint pulse required for friction compensation depends on the acceleration.
- It is enough if the characteristic is modeled by three different acceleration interpolation points and a minimum and maximum amplitude of the velocity setpoint pulse.
- An acceleration-**dependent** amplitude and an acceleration-**independent** decay time are sufficient to model the velocity setpoint pulse.
- For friction compensation at the positive and negative reversal point, the same velocity setpoint pulse is sufficient.
- For greater acceleration, a smaller compensation value is required than for smaller acceleration.
- The requirement for contour accuracy at the reversal points is high.

The following figure shows an example characteristic that can be set with this function:



$\Delta n_{\max}$  Maximum amplitude of the velocity setpoint pulse

$\Delta n_{\min}$  Minimum amplitude of the velocity setpoint pulse

$a_1, a_2, a_3$  Acceleration interpolation points 1, 2, and 3

B1 ... B4 Acceleration range 1 ... 4

The amplitude of the velocity setpoint pulse  $\Delta n$  is calculated in the relevant acceleration range B1 to B4 to form:

| Range | Acceleration $a$      | Amplitude of the velocity setpoint pulse $\Delta n$                                                               |
|-------|-----------------------|-------------------------------------------------------------------------------------------------------------------|
| B1    | $a < a_1$             | $\Delta n = \Delta n_{\max} * \frac{a}{a_1}$                                                                      |
| B2    | $a_1 \leq a \leq a_2$ | $\Delta n = \Delta n_{\max}$                                                                                      |
| B3    | $a_2 < a < a_3$       | $\Delta n = \Delta n_{\max} + \left[ \frac{(\Delta n_{\min} - \Delta n_{\max})}{(a_3 - a_2)} \right] * (a - a_2)$ |
| B4    | $a \geq a_3$          | $\Delta n = \Delta n_{\min}$                                                                                      |

## 8.15.2 commissioning

To determine the axis-specific characteristic parameters, the Circularity test (Page 301) must be used to determine the optimum amplitude of the velocity setpoint pulse  $\Delta n_{\text{opt},a}$  each for each of a number of acceleration values. The different acceleration values should cover the entire dynamic range. In particular, a sufficient number of measurements must be ensured for low acceleration values and large circle radii.

### Calculation of the acceleration at the quadrant transition

On a circular path, the acceleration  $a$  of a machine axis on direction reversal at the quadrant transition is calculated from the radius  $r$  of the circle and the path velocity  $v$ :

$$a = \frac{v^2}{r}$$

**Note**

The path velocity and therefore the axis-specific acceleration  $a$  can be varied simply via the feedrate override.

**Evaluation of the value pairs determined ( $a$ ,  $\Delta n_{\text{opt}_a}$ )**

To determine the acceleration interpolation points  $a_1$ ,  $a_2$ , and  $a_3$  and the minimum and maximum amplitude of the velocity setpoint pulse  $\Delta n_{\text{min}}$  and  $\Delta n_{\text{max}}$ , we recommend graphically applying the value pairs determined in the circularity test comprising the acceleration  $a$  and the associated optimum amplitude of the velocity setpoint pulse  $\Delta n_{\text{opt}_a}$ :

$$\Delta n_{\text{opt}_a} = f(a)$$

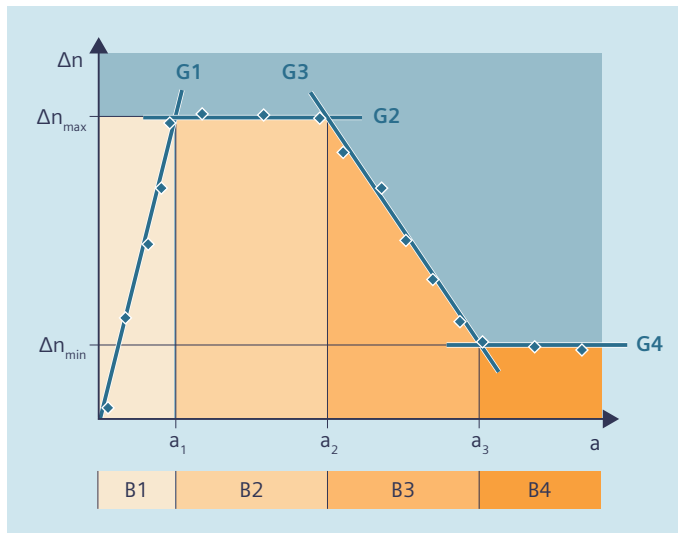


Figure 8-26 Axis-specific characteristic determination

The acceleration interpolation points  $a_1$ ,  $a_2$ , and  $a_3$  and the minimum and maximum amplitude of the velocity setpoint pulse  $\Delta n_{\text{min}}$  and  $\Delta n_{\text{max}}$  result from drawing in the straight lines G1 ... G4.

**Axis-specific machine data****Activating friction compensation**

Friction compensation is activated with:

MD32500 \$MA\_FRICT\_COMP\_ENABLE[ <axis> ] = **TRUE (1)**

**Activating friction compensation with adaptive characteristic**

Friction compensation with adaptive characteristic is activated with:

- MD32490 \$MA\_FRICT\_COMP\_MODE[ <axis> ] = 1
- MD32510 \$MA\_FRICT\_COMP\_ADAPT\_ENABLE[ <axis> ] = **TRUE (1)**

**Characteristic parameters: Acceleration  $a_1$ ,  $a_2$ , and  $a_3$** 

The axis-specific acceleration  $a$  occurring on a circle path on reversal at the quadrant transitions is calculated from the radius  $r$  and the path velocity  $v$  as:

$$a = \frac{v^2}{r}$$

The acceleration values  $a_1$ ,  $a_2$ , and  $a_3$  are to be entered in the machine data in monotonically ascending order ( $a_1 < a_2 < a_3$ ):

- MD32550 \$MA\_FRICT\_COMP\_ACCEL1[ <axis> ] = < $a_1$ >
- MD32560 \$MA\_FRICT\_COMP\_ACCEL2[ <axis> ] = < $a_2$ >
- MD32570 \$MA\_FRICT\_COMP\_ACCEL3[ <axis> ] = < $a_3$ >

**Characteristic parameters: Amplitude  $\Delta n_{\min}$  and  $\Delta n_{\max}$** 

The maximum and minimum amplitude of the velocity setpoint pulse ( $\Delta n_{\max}$ ,  $\Delta n_{\min}$ ) must be entered in the following machine data:

- MD32520 \$MA\_FRICT\_COMP\_CONST\_MAX[ <axis> ] = < $\Delta n_{\max}$ >
- MD32530 \$MA\_FRICT\_COMP\_CONST\_MIN[ <axis> ] = < $\Delta n_{\min}$ >

**Note**

If satisfactory results cannot be obtained for very small path velocities, the computational resolution may have to be increased:

- MD10200 \$MA\_INT\_INCR\_PER\_MM (computational resolution for linear positions)
- MD10210 \$MA\_INT\_INCR\_PER\_DEG (computational resolution for angular positions)

**Velocity setpoint pulse: Decay time**

The decay time of the velocity setpoint pulse is set via:

MD32540 \$MA\_FRICT\_COMP\_TIME[ <axis> ] = <decay time>

### 8.15.3 Supplementary conditions

#### Reaction of setpoint-related compensation functions

The following setpoint-related compensation functions affect the position setpoint and must be deactivated for the axes that perform a circularity test:

- Sag and angularity compensation (CEC)  
MD32710 \$MA\_CEC\_ENABLE[ <axis> ] = 0
- Directional leadscrew error compensation (LEC):  
MD32710 \$MA\_CEC\_ENABLE[ <axis> ] = 0
- Temperature compensation:  
MD32750 \$MA\_TEMP\_COMP\_TYPE[ <axis> ] = 0

## 8.16 Friction compensation with adaptive characteristics

### 8.16.1 Description of the function

Friction compensation with adaptive characteristics has the following properties:

- For fast and simple optimization, commissioning can be performed via the SINUMERIK Operate user interface.
- In guided commissioning, the test movements are optionally generated automatically for one or two axes.
- The compensation values can be set separately for each axis, e.g. for a vertical axis, for each reversal point (direction-dependent).
- In relation to the set position (compensation mode (Page 312) = 3), better results are achieved by early application of the compensation value.
- If the velocity setpoint pulse is insufficient to achieve the desired result, e.g. for gearless drives, a pulse can additionally be applied to the torque setpoint.
- Four compensation characteristics with up to nine acceleration interpolation points can be defined.
- Settable parameters for the velocity setpoint pulse:
  - Delay time
  - Amplitude <sup>1)</sup>
  - Action time <sup>1)</sup>
  - Rise time
  - Velocity threshold
  - Decay time <sup>1)</sup>

- Settable parameters for the torque setpoint pulse:
  - Amplitude <sup>1)</sup>
  - Velocity threshold
  - Delay time
  - Rise time <sup>1)</sup>
- A temperature, leadscrew error, and/or sag compensation that is active in parallel has no effect on the friction compensation.

<sup>1)</sup> Parameters that can be adapted depending on the acceleration

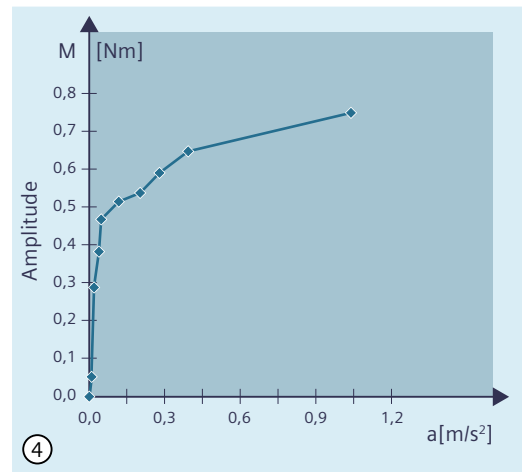
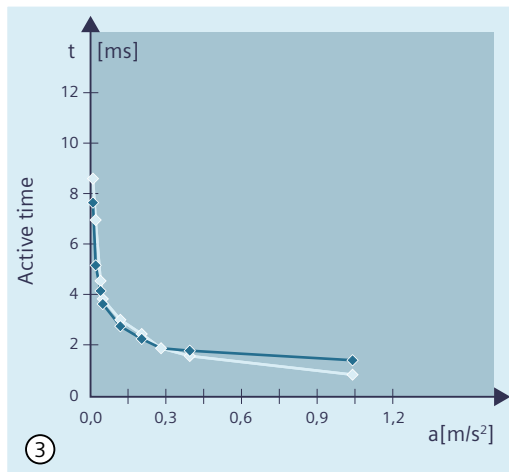
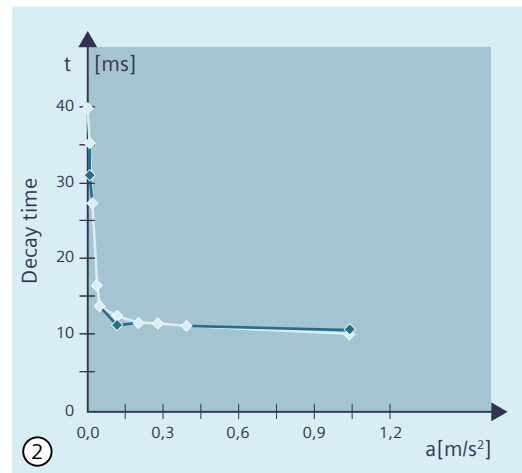
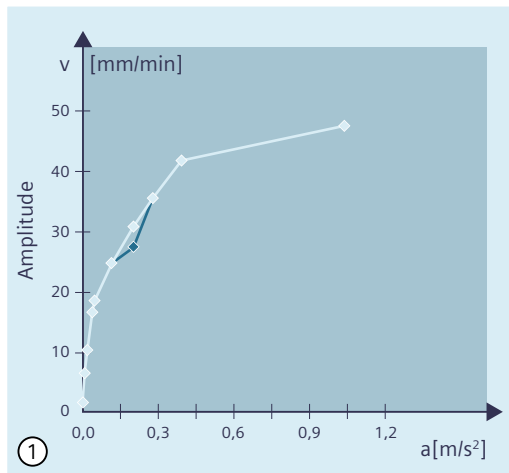
---

**Note**

In most applications, compensation by means of velocity setpoint pulse and compensation characteristics for amplitude and decay time are sufficient. Only if no satisfactory results have been achieved should further options be used.

---

## Examples of adapted characteristics



- ①②③ Velocity injection pulse  
 Light blue: Lower reversal point  
 Dark blue: Upper reversal point  
 ④ Torque setpoint pulse

Compensation values for acceleration between the parameterized interpolation points are interpolated linearly.

## 8.16.2 Commissioning

## 8.16.2.1 Activating the function

## Activating friction compensation

Friction compensation is activated axis-specifically with:

MD32500 \$MA\_FRICT\_COMP\_ENABLE[<axis>] = TRUE

### Activating friction compensation with adaptive characteristics

Friction compensation with adaptive characteristics can be applied to the set or actual position of the machine axis, depending on the dynamics of the axis. The function is activated axis-specifically with:

MD32490 \$MA\_FRICT\_COMP\_MODE = <mode>

| <mode>                                                                         | Meaning                                                                                                                                                                                                        |
|--------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 3                                                                              | For axes with high dynamics and active feedforward control <sup>1)</sup><br>The velocity or torque setpoint pulse is applied if the <b>set</b> position of the machine axis has reached the trigger point.     |
| 4                                                                              | For axes with low dynamics and inactive feedforward control <sup>1)</sup><br>The velocity or torque setpoint pulse is applied if the <b>actual</b> position of the machine axis has reached the trigger point. |
| <sup>1)</sup> Feedforward control: MD32620 FFW_MODE (feedforward control mode) |                                                                                                                                                                                                                |

### Note

#### Commissioning

We urgently recommend performing guided commissioning using the commissioning functions of the SINUMERIK Operate user interface and not changing the generated values in the machine data subsequently.

## 8.16.2.2 Commissioning functions of the SINUMERIK Operate user interface

Commissioning of the friction compensation with adaptive characteristics is performed in the "friction compensation" window:

Operating area: "Commissioning" > "NC" > "**Friction compensation**"

### Note

#### Softkey "Friction compensation"

The "friction compensation" softkey is only displayed if the friction compensation with adaptive characteristics is active (Page 312) for at least one machine axis.

The parameters for the axis-specific velocity and torque setpoint pulses are determined and stored retentively in machine data. Friction compensation by means of velocity setpoint pulse is normally sufficient.

Determination of the characteristic parameters for the velocity setpoint pulses is performed with guidance by the operator interface.

The characteristic parameters for the torque setpoint pulse required only in exceptional cases must be determined manually.

#### Note

##### Friction compensation - axis selection dialog

For the selection of an axis on an NCU other than the currently selected NCU, a switchover is no longer possible once the "Friction compensation" dialog is opened.

To clear the switchover lock, you must first exit the "Friction compensation" dialog.

**More information:** Commissioning Manual Commissioning CNC

## Function

The characteristic parameters are determined using a **circularity test** for up to nine different acceleration values (Page 315). One or two machine axes are alternately continuously traversed via an circularity test program generated automatically by the control and the result is displayed in a circle diagram.

The traversing movements of the circularity program (radius, path velocity, and direction of rotation) are already set in the following setting data. If necessary, they can be modified via the data list (see below paragraph "Further parameters").

| Parameter                                                          | Axis type   | Setting Data                                |
|--------------------------------------------------------------------|-------------|---------------------------------------------|
| Radius                                                             | Linear axis | SD55820 \$SCS_FRICT_OPT_RADIUS              |
|                                                                    | Rotary axis | SD55821 \$SCS_FRICT_OPT_RADIUS_ROT          |
| Path velocity                                                      | Linear axis | SD55822 \$SCS_FRICT_OPT_FEED[ 0 ... 8 ]     |
|                                                                    | Rotary axis | SD55823 \$SCS_FRICT_OPT_FEED_ROT[ 0 ... 8 ] |
| Direction of rotation <sup>1)</sup>                                | ---         | SD55828 \$SCS_FRICT_OPT_DIR_MINUS           |
| <sup>1)</sup> Only effective in the circularity test with two axes |             |                                             |

### Measuring step

After the start of the circularity test program, in each of the up to nine measuring steps of a measurement series, the axes are traversed with the path velocity that is defined in the setting data or with the corresponding acceleration. The deviations of the actual positions from the ideal path are acquired and displayed graphically as a set or actual circular path. The parameters of the velocity setpoint pulse can be varied by the commissioning engineering while the circularity test continues to be performed constantly. The resulting changes are automatically updated in the result graphics.

If the circularity tests is only performed with a single axis, the result is also displayed as a circle. For this purpose, the determined contour deviations of the axis are displayed one horizontally and once vertically. If the circularity test is performed with two axes, both axes must be geometry axes of the same channel.

**Completion**

After optimization of the compensation parameters for the entire measurement series has been completed, the maximum values, characteristics interpolation points, and weighting factors are calculated by the control and written into the following machine data:

| Maximum value | Machine data                          |
|---------------|---------------------------------------|
| Amplitude     | MD32571 \$MA_FRICT_VELO_STEP          |
| Decay time    | MD32574 \$MA_FRICT_V_PULSE_DECAY_TIME |
| Active time   | MD32573 \$MA_FRICT_V_PULSE_CONST_TIME |

| Characteristic interpolation points                       | Machine data                                    |
|-----------------------------------------------------------|-------------------------------------------------|
| Acceleration of the friction compensation characteristics | MD32581 \$MA_FRICT_ADAPT_TABLE_ACCEL[ 0 ... 9 ] |

| Weighting factors         | Machine data                                                                                          |
|---------------------------|-------------------------------------------------------------------------------------------------------|
| Amplitude upper / lower   | MD32582 \$MA_FRICT_ADAPT_V_STEP_PLUS[ 0 ... 9 ]<br>MD32583 \$MA_FRICT_ADAPT_V_STEP_MINUS[ 0 ... 9 ]   |
| Decay time upper / lower  | MD32586 \$MA_FRICT_ADAPT_V_DECAY_PLUS[ 0 ... 9 ]<br>MD32587 \$MA_FRICT_ADAPT_V_DECAY_MINUS[ 0 ... 9 ] |
| Action time upper / lower | MD32584 \$MA_FRICT_ADAPT_V_CONST_PLUS[ 0 ... 9 ]<br>MD32585 \$MA_FRICT_ADAPT_V_CONST_MINUS[ 0 ... 9 ] |

**Additional parameters**

With the vertical softkey "Data list," further machine and setting data are displayed for parameterization of the friction compensation with adaptive characteristics, e.g. also the parameters for the torque setpoint pulse. The setting these parameters is described in the following sections.

**8.16.2.3 Parameterization of the acceleration at the characteristic interpolation points**

From the channel-specific setting data of the circularity test (circle radius and traversing velocity), the control calculates the acceleration values of the characteristic interpolation points.

**Channel-specific setting data**

The channel-specific setting data for circle radius and velocities apply to all machine axes with active friction compensation with adaptive characteristic curves and that are channel axes of the channel in question. They are already assigned typical values. In necessary, they can be adapted machine-specifically.

**Circle radius**

A circle is traversed with the parameterized radius in the automatically generated circularity tests program:

- Linear axes  
SD55820 \$SCS\_FRICT\_OPT\_RADIUS = <radius>
- Rotary axes  
SD55821 \$SCS\_FRICT\_OPT\_RADIUS\_ROT = <radius>

**Traversing velocities**

In the circularity test, the machine axes are traversed in each of the up to nine measurement sections with the respectively parameterized velocity:

- Linear axes  
SD55822 \$SCS\_FRICT\_OPT\_FEED[ 0 ... 8 ] = <velocity 1 ... 9>
- Rotary axes  
SD55823 \$SCS\_FRICT\_OPT\_FEED\_ROT[ 0 ... 8 ] = <velocity 1 ... 9>

**Note**

If fewer than nine characteristic interpolation points ( 1 ... n ) are required, the velocity zero must be entered in each of the corresponding array elements for all unneeded characteristic interpolation points ( (n+1) ... 9 ).

If the velocity is equal to zero in an array element, no following array element must contain a velocity that is not equal to zero.

**Direction of rotation**

In the circularity test with two axes, the direction of rotation of the circle is defined with:

SD55828 \$SCS\_FRICT\_OPT\_DIR\_MINUS = <direction of rotation>

**Axis-specific machine data**

The acceleration  $a$  at the characteristic interpolation point  $n$  is calculated by the control from the setting data for the circle radius  $r$  and the path velocity  $v_{n-1}$  as follows:

$$a_n = \frac{v_{n-1}^2}{r}$$

$$\text{MD32581 \$MA\_FRICT\_ADAPT\_TABLE\_ACCEL[ } n \text{ ]} = \frac{(\text{SD55822 \$SCS\_FRICT\_OPT\_FEED[ } n-1 \text{ ]})^2}{\text{SD55820 \$SCS\_FRICT\_OPT\_RADIUS}}$$

with  $n = 1, 2, 3, \dots, 9$

For acceleration  $a$  at characteristic interpolation point  $n = 0$ , the value zero is always entered:

MD32581 \$MA\_FRICT\_ADAPT\_TABLE\_ACCEL[ 0 ] = 0

### 8.16.2.4 Velocity setpoint pulse

Determination of the characteristic parameters for the velocity setpoint pulses and calculation of the corresponding machine data is fully supported by the user interface (Page 313). We therefore advise **against** determining the characteristic parameters or writing the machine data manually.

### Axis-specific machine data

#### Velocity injection pulse

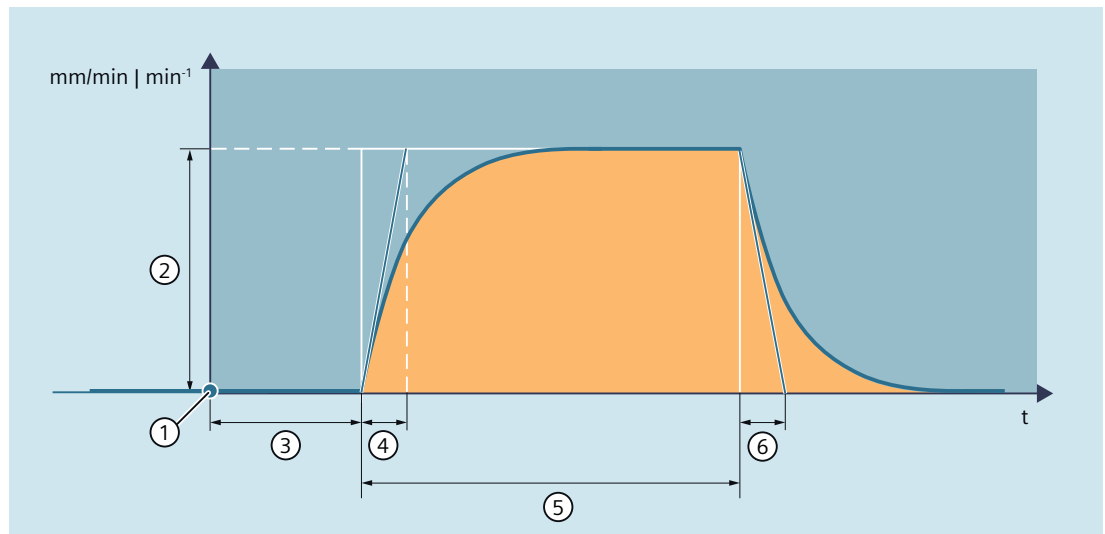


Figure 8-27 Basic pulse shape

The numbers (①, ②, ...) stated in the following tables refer to the figure above.

#### Acceleration-independent parameters

| No. | Machine data                           | Description                                                                |
|-----|----------------------------------------|----------------------------------------------------------------------------|
| ①   | -                                      | Trigger time or acceleration of the axes from standstill                   |
| ③   | MD32572 \$MA_FRICT_V_PULSE_DELAY_TIME  | Delay time                                                                 |
| ④   | MD32575 \$MA_FRICT_V_PULSE_SMOOTH_TIME | Time constant T of the rise time, after 5 * T ⇒ output value ≈ input value |

#### Maximum values of the acceleration-dependently adaptable parameters

| No. | Machine data                          | Description |
|-----|---------------------------------------|-------------|
| ②   | MD32571 \$MA_FRICT_VELO_STEP          | Amplitude   |
| ⑤   | MD32573 \$MA_FRICT_V_PULSE_CONST_TIME | Active time |
| ⑥   | MD32574 \$MA_FRICT_V_PULSE_DECAY_TIME | Decay time  |

**Weighting factors for acceleration-dependent adaptation of the maximum values**

Table 8-1 Lower reversal point

| Machine data                                     | Description                          |
|--------------------------------------------------|--------------------------------------|
| MD32582 \$MA_FRICT_ADAPT_V_STEP_PLUS[ 0 ... 9 ]  | Weighting factor for the amplitude   |
| MD32584 \$MA_FRICT_ADAPT_V_CONST_PLUS[ 0 ... 9 ] | Weighting factor for the action time |
| MD32586 \$MA_FRICT_ADAPT_V_DECAY_PLUS[ 0 ... 9 ] | Weighting factor for the decay time  |

Table 8-2 Upper reversal point

| Machine data                                      | Description                          |
|---------------------------------------------------|--------------------------------------|
| MD32583 \$MA_FRICT_ADAPT_V_STEP_MINUS[ 0 ... 9 ]  | Weighting factor for the amplitude   |
| MD32585 \$MA_FRICT_ADAPT_V_CONST_MINUS[ 0 ... 9 ] | Weighting factor for the action time |
| MD32587 \$MA_FRICT_ADAPT_V_DECAY_MINUS[ 0 ... 9 ] | Weighting factor for the decay time  |

**Calculation of the characteristic parameters**

The commissioning engineer must determine the optimum values for amplitude, action time, and decay time for the velocity setpoint pulse for different acceleration values during the circularity test (Page 313). On completion of the circularity test, the control will calculate the resulting characteristic parameters (maximum values, characteristic interpolation points, and weighting factors) and write them into the corresponding machine data.

**Example: Amplitude characteristic for the lower reversal point**

- Maximum value

$$\text{MD32571 \$MA\_FRICT\_VELO\_STEP} = \max \{ \text{Amplitude 1, Amplitude 2, ..., Amplitude 9} \}$$

- Weighting factors

$$\text{MD32582 \$MA\_FRICT\_ADAPT\_V\_STEP\_PLUS[ n ]} = \frac{\text{Amplitude[ n ]}}{\text{MD32571 \$MA\_FRICT\_VELO\_STEP}}$$

- Effective interpolation points

The effective interpolation points of the amplitude characteristics are the interpolation points to which the following applies:

- MD32581 \$MA\_FRICT\_ADAPT\_TABLE\_ACCEL[ <interpolation point> ] ≠ 0 (acceleration)
- MD32582 \$MA\_FRICT\_ADAPT\_V\_STEP\_PLUS[ <interpolation point> ] ≠ 0 (weighting factor)

The amplitude of the velocity setpoint pulse of an effective interpolation point is calculated as:

$$\begin{aligned} \text{Amplitude[ n ]} &= \text{Maximum value} * \text{Weighting factor[ n ]} \\ &= \text{MD32571 \$MA\_FRICT\_VELO\_STEP} * \text{MD32582 \$MA\_FRICT\_ADAPT\_V\_STEP\_PLUS[ n ]} \end{aligned}$$

**Note****Vertical offset of a characteristic**

Subsequent vertical displacement of a characteristics is most easily achieved by changing the corresponding maximum value in the machine data.

**8.16.2.5 Torque setpoint pulse**

Determination of the characteristic parameters for the torque setpoint pulses and calculation of the corresponding machine data is not supported directly by the user interface. We therefore advise determining the characteristic parameters or writing the machine data manually

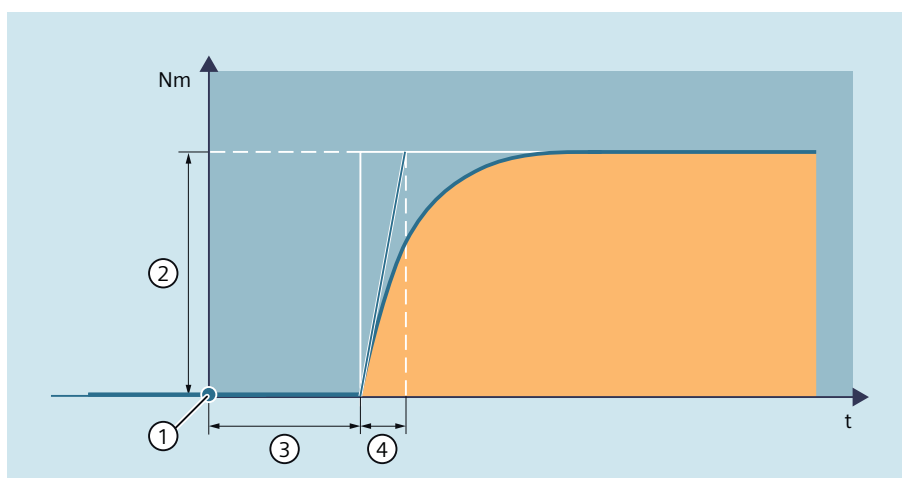
**Axis-specific machine data****Torque setpoint pulse**

Figure 8-28 Basic pulse shape

The numbers (①, ②, ...) stated in the following tables refer to the figure above.

**Acceleration-independent parameters**

| No. | Machine data                          | Description                                              |
|-----|---------------------------------------|----------------------------------------------------------|
| ①   | -                                     | Trigger time or acceleration of the axes from standstill |
| ③   | MD32577 \$MA_FRICT_T_PULSE_DELAY_TIME | Delay time                                               |

**Maximum values of the acceleration-dependently adaptable parameters**

| No. | Machine data                           | Description |
|-----|----------------------------------------|-------------|
| ②   | MD32576 \$MA_FRICT_TORQUE_STEP         | Amplitude   |
| ④   | MD32578 \$MA_FRICT_T_PULSE_SMOOTH_TIME | Rise time   |

**Weighting factors for acceleration-dependent adaptation of the maximum values**

| Machine data                                       | Description                                                    |
|----------------------------------------------------|----------------------------------------------------------------|
| MD32588 \$MA_FRICT_ADAPT_T_STEP[ 0 ... 9 ]         | Weighting factor for the amplitude                             |
| MD32589 \$MA_FRICT_ADAPT_T_SMOOTH_PLUS[ 0 ... 9 ]  | Weighting factor for the rise time in the lower reversal point |
| MD32590 \$MA_FRICT_ADAPT_T_SMOOTH_MINUS[ 0 ... 9 ] | Weighting factor for the rise time in the upper reversal point |

**Commissioning (manual)****Requirement**

The circularity test (Page 313) for determining the characteristic parameter was already been successfully performed completely or at least for the current measuring step.

**Note****Determining the characteristic parameters**

The characteristic parameters for the torque setpoint pulse are most simply determined in the circularity test (Page 313) of the user interface.

Operating area: "Commissioning" > "NC" > "**Friction compensation**"

**Setting the machine data**

The machine data are most simply set via the data list of the circularity test:

Operating area: "Commissioning" > "NC" > "Friction compensation" > "**Data list**" vertical softkey

**Assumption**

The circularity tests has been started, measuring step 1 is active.

**Recommended procedure**

1. Initialization of the weighting factors:
  - MD32588 \$MA\_FRICT\_ADAPT\_T\_STEP[ 0 ] = 1
  - MD32588 \$MA\_FRICT\_ADAPT\_T\_STEP[ 1 ... 9 ] = 0

**Note**

For all measuring steps, the weighting factor with index 0 is used.

2. Setting the amplitude for the current measuring step:
  - MD32576 \$MA\_FRICT\_TORQUE\_STEP = <amplitude>
3. Determining the changed contour deviation in the circle diagram.
4. Optimization of the amplitude value by changing in MD32576 \$MA\_FRICT\_TORQUE\_STEP and checking in the circle diagram.
5. Recording the amplitude value for this measuring step for subsequent determination of the maximum value or the weighting factor of the measuring step.

6. Switching on to the next measuring step and continuing with step 2 until the amplitude values have been determined for all measuring steps.
7. Determining the maximum value from the recorded amplitude values and entering them in the machine data:

$$\text{MD32576 \$MA\_FRICT\_TORQUE\_STEP} = \max \{ \text{Amplitude 1, Amplitude 2, ..., Amplitude 9} \}$$

8. Calculation of the weighting factors for all active interpolation points from the recorded amplitude values and entering them in the machine data:

$$\text{MD32588 \$MA\_FRICT\_ADAPT\_T\_STEP[ n ]} = \frac{\text{Amplitude[ n ]}}{\text{MD32576 \$MA\_FRICT\_TORQUE\_STEP}}$$

That completes commissioning of the amplitude characteristic for the torque setpoint pulse.

### Optional

Setting the parameters that are independent of acceleration:

- MD32577 \$MA\_FRICT\_T\_PULSE\_DELAY\_TIME
- MD32578 \$MA\_FRICT\_T\_PULSE\_SMOOTH\_TIME

## Effective interpolation points

The effective interpolation points of the amplitude characteristics are the interpolation points to which the following applies:

- MD32581 \$MA\_FRICT\_ADAPT\_TABLE\_ACCEL[ <interpolation point> ]  $\neq 0$  (acceleration)
- MD32583 \$MA\_FRICT\_ADAPT\_T\_STEP[ <interpolation point> ]  $\neq 0$  (weighting factor)

The amplitude of the torque setpoint pulse of an effective interpolation point is calculated as:

$$\begin{aligned} \text{Amplitude[ n ]} &= \text{Maximum value} * \text{Weighting factor[ n ]} \\ &= \text{MD32576 \$MA\_FRICT\_TORQUE\_STEP} * \text{MD32588 \$MA\_FRICT\_ADAPT\_T\_STEP[ n ]} \end{aligned}$$

### Note

#### Vertical offset of the characteristic

Subsequent vertical displacement of the amplitude characteristic is most easily achieved by changing the corresponding maximum value in the machine data.

## 8.16.2.6 Velocity threshold for triggering the injection pulses

In order to prevent overcompensation with the smallest axis movements, it may be necessary to specify a velocity threshold for triggering the injection pulses:

MD32591 \$MA\_FRICT\_TRIGGER\_TRESHOLD [<axis>] = <threshold value>

At axis velocities below the set threshold, triggering of the injection pulses is suppressed. Only when the axis velocity reaches the set threshold value are the injection pulses triggered.

#### Note

The specification of a velocity threshold changes the injection time. The velocity threshold should therefore only be adapted if the adaptation options were not sufficient to prevent overcompensation in case of minor axis movements or at low axis velocities.

## 8.17 Compensation functions for suspended axes

### 8.17.1 Electronic counterweight

#### Axis without counterweight

For axes that have a weight load without counterweight, then after the brake is released, the hanging (suspended) axis drops and the following response is obtained:

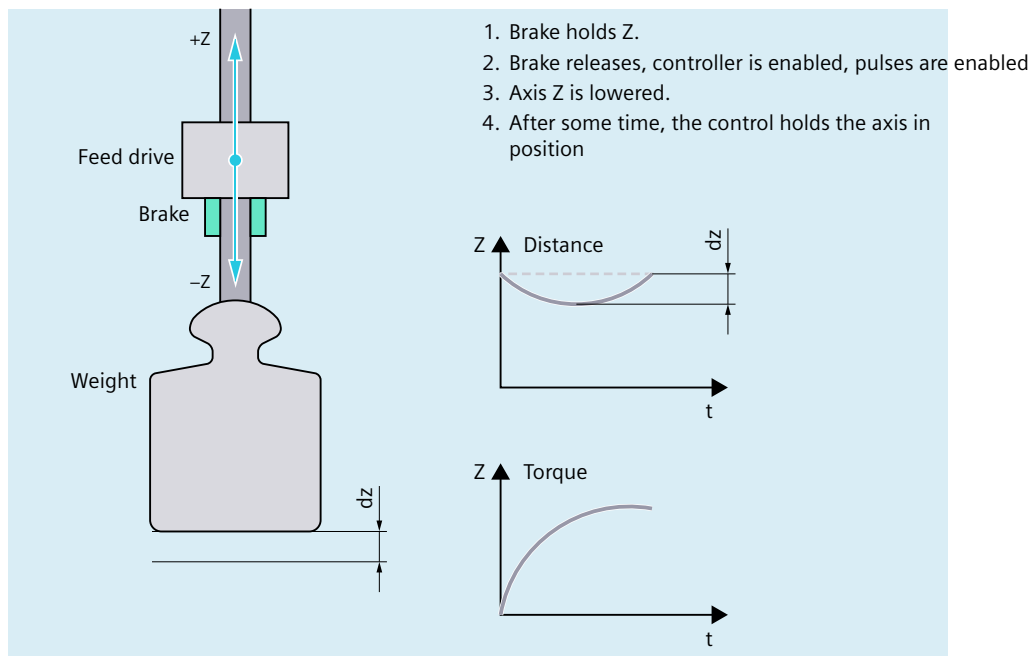


Figure 8-29 Drop of a hanging axis without counterweight

#### "Electronic counterweight" function

A hanging (suspended) axis can almost be completely prevented from dropping (sagging) using the "electronic counterweight" function.

The electronic counterweight prevents axes with a weight load from sagging when the closed-loop control is switched on. After releasing the brake, the constant counterweight torque maintains the position of the vertical axis.

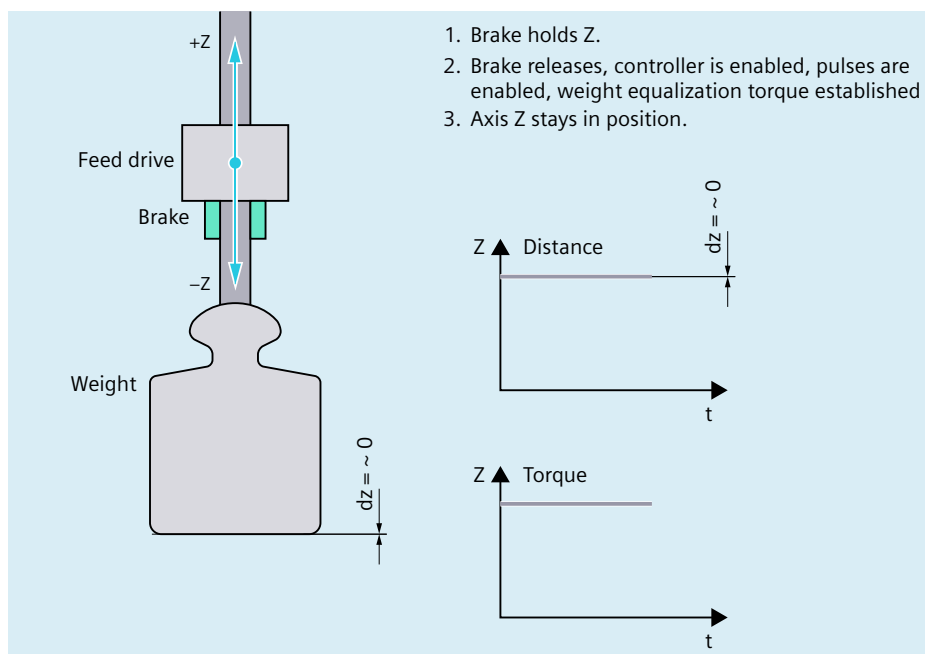


Figure 8-30 Lowering of a vertical axis with electronic weight compensation

## Commissioning

### Note

The "electronic counterweight" is commissioned through the drive!

## Further information

For additional information, see the following:

Function Manual Drive Functions SINAMICS S120

## 8.17.2 Special function: Reboot delay

### Function

For changes, for example, to machine data values to become effective, the NC must be restarted. This is done, for example, on the user interface by triggering an NC reset. If there are suspended axes on the machine, failure of the closed-loop control while the control is starting will result in the axes failing to maintain height.

With the "Reboot delay" function, the request to reboot the NC (NC reset) is communicated to the NC via the operator interface as previously. Rebooting, during which the closed-loop control of the axes is deactivated, is then delayed on the NC by a time that can be parameterized. During this time, user-specific actions, such as engaging the holding brakes of the suspended axes, are performed.

---

**Note**

The reboot delay is only effective on a request to reboot the NC (NC reset) via the **user interface**.

In the case of a power-on reset by switching the control off and on again, pressing the reset button on the front of the NCU, or power failure, a parameterized reboot delay time will have no effect.

---

**Alarm 2900 "Reboot after a delay"**

When a reboot request is detected, alarm 2900 "Reboot after a delay" is triggered.

**Alarm responses**

The following responses are triggered by the alarm 2900:

- The NC/PLC interface signals are reset:
  - `<ModeGroup>.basic.in.ready = 0` (mode group ready) ; all mode groups
  - `<Chan>.basic.in.ready = 0` (channel ready) for **all** channels
  - `<Axis>.basic.in.ready = 0` (axis ready) for **all** axes
- Braking the axis / spindles at the current limit.  
For further details, see machine data:
  - MD36610 \$MA\_AX\_EMERGENCY\_STOP\_TIME (braking ramp time when errors occur)
  - MD36620 \$MA\_SERVO\_DISABLE\_DELAY\_TIME (OFF delay of the controller enable)

The NC/PLC interface signal "NC ready" remains set:

```
<Nc>.basic.in.ncReady == 1
```

**Alarm suppression**

With the machine data, **display** of the alarm 2900 "Reboot after a delay" will be suppressed on the user interface:

```
MD11410 $MN_SUPPRESS_ALARM_MASKBit 20 = 1
```

The alarm responses are not affected by this.

**Controlling holding brakes**

During the reboot operation of the PLC, the PLC outputs defined as 0 are reset. Control of the holding brakes must therefore be connected on the user side in such a way that the brakes engage or remain engaged on a control signal == 0 and are released or remain released on a control signal == 1.

## Parameterization

The reboot delay time is set in machine data:

MD10088 \$MN\_REBOOT\_DELAY\_TIME = <reboot delay time>

If the parameterized reboot delay time is 0.0, the function is deactivated.

## System variables

The time remaining until the NC is rebooted can be read in the system variable:

\$AN\_REBOOT\_DELAY\_TIME

While no request for a reboot of the NC (NC reset) has been triggered from the user interface, the system variable has the value 0.0.

A value greater than 0.0 indicates that a reboot request (NC reset) has been triggered from the user interface and also the time remaining in the NC or PLC until the reboot.

### Application example

Evaluating the system variables in a static synchronized action

Condition part: Check for a value greater than 0.0 because then a request for a reboot of the NC (NC reset) has been made from the user interface.

Action part: e.g. triggering "safe standstill" as part of the "Safety Integrated" function.

## PLC signals

| Basic Program Plus         | Basic Program         |                    |
|----------------------------|-----------------------|--------------------|
| <ModeGroup>.basic.in.ready | LBP_ModeGroup.E_MGOK  | DB11.DBX6.3        |
| <Chan>.basic.in.ready      | LBP_Chan*.E_ChanRO    | DB21, ... .DBX36.5 |
| <Axis>.basic.in.ready      | LBP_Axis*.E_AxisReady | DB31, ... .DBX61.2 |
| <Nc>.basic.in.ncReady      | LBP_NC.E_NCready      | DB10.DBX108.7      |



# Appendix

## A.1 List of abbreviations

| A     |                                                                                                              |
|-------|--------------------------------------------------------------------------------------------------------------|
| A     | Output                                                                                                       |
| AFIS  | Automatic Filter Switch: Automatic filter switch                                                             |
| ASCII | American Standard Code for Information Interchange: American coding standard for the exchange of information |
| ASIC  | Application Specific Integrated Circuit: User switching circuit                                              |
| ASUP  | Asynchronous subprogram                                                                                      |
| AUTO  | Operating mode "Automatic"                                                                                   |
| AUXFU | Auxiliary Function: Auxiliary function                                                                       |
| AWL   | Statement list                                                                                               |

| B    |                                                               |
|------|---------------------------------------------------------------|
| BAG  | Mode group                                                    |
| BCD  | Binary Coded Decimals: Decimal numbers encoded in binary code |
| BICO | Binector Connector                                            |
| BIN  | Binary Files: Binary files                                    |
| BKS  | Basic coordinate system                                       |
| BM   | Operating alarm                                               |
| BO   | Binector Output                                               |
| BTSS | Operator panel interface                                      |

| C   |                                                                           |
|-----|---------------------------------------------------------------------------|
| CLC | Clearance control                                                         |
| CNC | Computerized Numerical Control: Computerunterstützte numerische Steuerung |
| COM | Communication                                                             |
| CP  | Communication Processor                                                   |
| CPU | Central Processing Unit: Central processing unit                          |
| CST | Configured Stop: Configured stop                                          |

| D   |                                                                            |
|-----|----------------------------------------------------------------------------|
| DDS | Drive Data Set: Drive data set                                             |
| DIR | Directory: Directory                                                       |
| DO  | Drive Object                                                               |
| DRF | Differential Resolver Function: Differential resolver function (handwheel) |

| D     |                                 |
|-------|---------------------------------|
| DRY   | Dry Run: Dry run feedrate       |
| DWORD | Double word (currently 32 bits) |

| E   |                                                     |
|-----|-----------------------------------------------------|
| E   | Input                                               |
| EES | Execution from External Storage                     |
| E/A | Input/Output                                        |
| ESR | Extended stop and retract                           |
| ETC | ETC key ">"; Softkey bar extension in the same menu |

| F      |                                                                                                                                       |
|--------|---------------------------------------------------------------------------------------------------------------------------------------|
| FDD    | Feed Disable: Feed disable                                                                                                            |
| FdStop | Feed Stop: Feed stop                                                                                                                  |
| 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 |
| FM     | Error message                                                                                                                         |
| FUP    | Control system flowchart (PLC programming method)                                                                                     |
| FW     | Firmware                                                                                                                              |

| G   |                                    |
|-----|------------------------------------|
| GEO | Geometry, e.g. geometry axis       |
| GP  | Basic program (PLC)                |
| GUD | Global User Data: Global user data |

| H    |                                                   |
|------|---------------------------------------------------|
| HEX  | Abbreviation for hexadecimal number               |
| HiFu | Auxiliary function                                |
| HMI  | Human Machine Interface: SINUMERIK user interface |
| HSA  | Main spindle drive                                |
| HT   | Handheld Terminal                                 |
| HW   | Hardware                                          |

| I   |                                      |
|-----|--------------------------------------|
| IBN | Commissioning                        |
| INC | Increment: Increment                 |
| INI | Initializing Data: Initializing data |
| IPO | Interpolator                         |

| <b>J</b> |                     |
|----------|---------------------|
| JOG      | JOGging: Setup mode |

| <b>K</b> |                                         |
|----------|-----------------------------------------|
| KOP      | Ladder diagram (PLC programming method) |

| <b>L</b> |                                            |
|----------|--------------------------------------------|
| LED      | Light Emitting Diode: Light-emitting diode |
| LMS      | Position measuring system                  |
| LR       | Position controller                        |

| <b>M</b> |                                              |
|----------|----------------------------------------------|
| Main     | Main program: Main program (OB1, PLC)        |
| MCP      | Machine Control Panel: Machine control panel |
| MD       | Machine Data                                 |
| MDA      | Manual Data Automatic: Manual input          |
| MDS      | Motor Data Set: Motor data set               |
| MELDW    | Message word                                 |
| MKS      | Machine coordinate system                    |
| MM       | Motor Module                                 |
| MPF      | Main Program File: Main program (NC)         |
| MPI      | Multi Point Interface                        |
| MSTT     | Machine control panel                        |

| <b>N</b> |                                                                                     |
|----------|-------------------------------------------------------------------------------------|
| NC       | Numerical Control: Numerical control with block preparation, traversing range, etc. |
| NCU      | Numerical Control Unit: NC hardware unit                                            |
| NCK      | Numerical Control Kernel                                                            |
| NCSD     | NC Start Disable                                                                    |
| NST      | Interface signal                                                                    |
| NV       | Work offset                                                                         |
| NX       | Numerical Extension: Axis expansion module                                          |

| <b>O</b> |                                      |
|----------|--------------------------------------|
| OB       | Organization block in the PLC        |
| OEM      | Original Equipment Manufacturer      |
| OP       | Operation Panel: Operating equipment |

| P        |                                                                                                                      |
|----------|----------------------------------------------------------------------------------------------------------------------|
| PCU      | PC Unit: PC box (computer unit)                                                                                      |
| PG       | Programming device                                                                                                   |
| PI       | Program instance                                                                                                     |
| PLC      | Programmable Logic Control: Controller                                                                               |
| PN       | PROFINET                                                                                                             |
| PO       | Power On                                                                                                             |
| POS      | Position/positioning                                                                                                 |
| 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                                                                                                         |
| PTP      | Point to Point: Point-to-point                                                                                       |
| PZD      | Process data: Process data part of a PPO                                                                             |

| R     |                                                             |
|-------|-------------------------------------------------------------|
| REF   | REference point approach function                           |
| REPOS | REPOSition function                                         |
| RESU  | Retrace support                                             |
| RID   | Read In Disable                                             |
| RP    | R Parameter, arithmetic parameter, predefined user variable |

| S       |                                                           |
|---------|-----------------------------------------------------------|
| SA      | Synchronized action                                       |
| SBL     | Single Block: Single block                                |
| SBT     | Safe Brake Test                                           |
| SCC     | Safety Control Channel                                    |
| SCL     | Structured Control Language                               |
| SD      | Settingdatum or setting data                              |
| SDI     | Safe Direction                                            |
| SERUPRO | Search-Run by Program Test: Block search via program test |
| SIC     | Safety Info Channel                                       |
| SKP     | Skip: Function for skipping a part program block          |
| SLP     | Safe Limited Position                                     |
| SLS     | Safely Limited Speed                                      |
| SMI     | Sensor Module Integrated                                  |
| SOS     | Safe Operating Stop                                       |
| SPF     | Sub Program File: Subprogram (NC)                         |
| SS1     | Safe Stop 1                                               |
| SS2     | Safe Stop 2                                               |
| STO     | Safe Torque Off                                           |

| <b>S</b> |                                 |
|----------|---------------------------------|
| STW      | Control word                    |
| SUG      | Grinding wheel peripheral speed |
| SW       | Software                        |

| <b>T</b> |                                                                                                         |
|----------|---------------------------------------------------------------------------------------------------------|
| TCU      | Thin Client Unit                                                                                        |
| TIA      | Totally Integrated Automation                                                                           |
| TM       | Terminal Module (SINAMICS)                                                                              |
| TO       | Tool Offset: Tool offset                                                                                |
| TOA      | Tool Offset Active: Identifier (file type) for tool offsets                                             |
| TOFF     | Online tool length offset                                                                               |
| TRANSMIT | Transform Milling Into Turning: Coordination transformation for milling operations on a turning machine |

| <b>U</b> |                      |
|----------|----------------------|
| UP       | Subprogram           |
| USB      | Universal Serial Bus |

| <b>V</b> |                                                     |
|----------|-----------------------------------------------------|
| VDI      | Internal communication interface between NC and PLC |

| <b>W</b> |                                           |
|----------|-------------------------------------------|
| WKS      | Workpiece coordinate system               |
| WPD      | Work Piece Directory: Workpiece directory |
| WZ       | Tool                                      |
| WZV      | Tool management                           |

| <b>Z</b> |                        |
|----------|------------------------|
| ZSW      | Status word (of drive) |



# Index

## \$

\$AA\_COLLPOS, 86  
\$AA\_DTBREB, 87  
\$AA\_DTBREB\_CMD, 87  
\$AA\_DTBREB\_CORR, 87  
\$AA\_DTBREB\_DEP, 87  
\$AA\_DTBREM, 88  
\$AA\_DTBREM\_CMD, 88  
\$AA\_DTBREM\_CORR, 88  
\$AA\_DTBREM\_DEP, 88  
\$AA\_ENC\_COMP, 245  
\$AA\_ENC\_COMP\_IS\_MODULO, 245  
\$AA\_ENC\_COMP\_MAX, 244  
\$AA\_ENC\_COMP\_MIN, 244  
\$AA\_ENC\_COMP\_STEP, 244  
\$AC\_COLLPOS, 86  
\$AN\_ACTIVATE\_COLL\_CHECK, 86  
\$AN\_CEC, 252  
\$AN\_CEC\_DIRECTION, 253  
\$AN\_CEC\_INPUT\_AXIS, 252  
\$AN\_CEC\_IS\_MODULO, 254  
\$AN\_CEC\_MAX, 253  
\$AN\_CEC\_MIN, 253  
\$AN\_CEC\_MULT\_BY\_TABLE, 254  
\$AN\_CEC\_OUTPUT\_AXIS, 252  
\$AN\_CEC\_STEP, 252  
\$AN\_COLL\_CHECK\_OFF, 86  
\$AN\_COLL\_IPO\_ACTIVE, 86  
\$AN\_COLL\_IPO\_LIMIT, 86  
\$AN\_COLL\_LOAD, 86  
\$AN\_COLL\_STATE, 86  
\$AN\_COLL\_STATE\_COND, 86  
\$AN\_COLL\_TIMEOUT\_CNTR, 86  
\$AN\_REBOOT\_DELAY\_TIME, 325  
\$NP\_1ST\_PROT, 33  
\$NP\_BIT\_NO, 37  
\$NP\_CHAIN\_ELEM, 31  
\$NP\_COLL\_PAIR, 63, 84  
\$NP\_COLOR, 46  
\$NP\_D\_LEVEL, 47  
\$NP\_DIR, 62  
\$NP\_FILENAME, 56  
\$NP\_INDEX, 39  
\$NP\_INIT\_STAT, 37, 38  
\$NP\_NAME, 42  
\$NP\_NEXT, 44  
\$NP\_NEXTP, 45  
\$NP\_OFF, 60  
\$NP\_PARA, 59  
\$NP\_PROT\_COLOR, 35  
\$NP\_PROT\_D\_LEVEL, 36  
\$NP\_PROT\_NAME, 30  
\$NP\_PROT\_TYPE, 32  
\$NP\_SAFETY\_DIST, 85  
\$NP\_TYPE, 54  
\$NP\_USAGE, 48  
\$P\_WORKAREA\_CS\_COORD\_SYSTEM, 182  
\$SC\_PA\_ACTIV\_IMMED, 107, 119  
\$SC\_PA\_CENT\_ABS, 108  
\$SC\_PA\_CENT\_ORD, 108  
\$SC\_PA\_CONT\_ABS, 108  
\$SC\_PA\_CONT\_NUM, 107  
\$SC\_PA\_CONT\_ORD, 108  
\$SC\_PA\_CONT\_TYP, 108  
\$SC\_PA\_LIM\_3DIM, 107  
\$SC\_PA\_MINUS\_LIM, 107  
\$SC\_PA\_ORI, 107  
\$SC\_PA\_PLUS\_LIM, 107  
\$SC\_PA\_T\_W, 107  
\$SN\_PA\_ACTIV\_IMMED, 107, 119  
\$SN\_PA\_CENT\_ABS, 108  
\$SN\_PA\_CENT\_ORD, 108  
\$SN\_PA\_CONT\_ABS, 108  
\$SN\_PA\_CONT\_NUM, 107  
\$SN\_PA\_CONT\_ORD, 108  
\$SN\_PA\_CONT\_TYP, 108  
\$SN\_PA\_LIM\_3DIM, 107  
\$SN\_PA\_MINUS\_LIM, 107  
\$SN\_PA\_ORI, 107  
\$SN\_PA\_PLUS\_LIM, 107  
\$SN\_PA\_T\_W, 107  
\$VA\_ABSOLUTE\_ENC\_DELTA\_INIT, 171  
\$VA\_ENC\_ZERO\_MON\_ERR\_CNT, 168, 171

## A

Accelerating axis, 229  
Adaptation axis, 229  
axis  
    Basic, 240  
    Compensation, 240  
Axis monitoring functions  
    Actual velocity, 163  
    Boundary conditions, 206  
    Following error, 146

Speed setpoint, 162  
Zero speed, 150

## B

Backlash, 222  
-Compensation, dynamic, 224  
Dynamic, 224

## C

Collision avoidance  
Fundamentals example, 92  
Collision pairs, 83  
Collision tolerance, 70  
Compensating axis, 229  
Compensation  
Following error, 290  
Interpolating, 240  
Leadscrew error, 242  
Measuring system error, 242  
Compensation value, 229  
Compliance factor, 229  
Contour error, 145  
Contour tunnel  
-radius, 210  
CPROT, 116  
CPROTDEF, 113

## D

Deformation  
due to temperature effects, 214  
Dynamic backlash, 224  
Dynamic response  
-adaptation, 295

## E

Encoder monitoring functions, 166  
Encoder frequency, 166  
Error  
Leadscrew, 242  
Measuring system, 242  
-temperature compensation curves, 214

## F

Feedforward control, 290  
Speed, 292  
Torque, 294  
Following error, 290

## G

G25, 178  
G26, 178

## H

Hardware limit switch, 172

## I

Interpolation point, 240

## L

LEC, 242  
Limit-switch monitoring, 172  
Linear signal distortions, 145

## M

MD10088, 325  
MD10200, 309  
MD10210, 309  
MD10240, 249  
MD10260, 245, 249, 250  
MD10618, 112  
MD10619, 80  
MD10622, 80  
MD10710, 180  
MD11410, 324  
MD18190, 112  
MD18890, 28, 81  
MD18892, 28, 81  
MD18893, 28  
MD18894, 29, 81  
MD18895, 29, 81  
MD18896, 81  
MD18897, 28  
MD18898, 81  
MD18899, 29

- 
- MD18951, 81, 82  
 MD18952, 82  
 MD19830, 79  
 MD19840, 79  
 MD20150, 180  
 MD20390, 218  
 MD21020, 177  
 MD21050, 210  
 MD21060, 211  
 MD21070, 211  
 MD21600, 173  
 MD21601, 173  
 MD21602, 174  
 MD28200, 112  
 MD28210, 112  
 MD28212, 112  
 MD28600, 181  
 MD30260, 205  
 MD30270, 206  
 MD31020, 205  
 MD31025, 205  
 MD31030, 206  
 MD31046, 189  
 MD31050, 206  
 MD31060, 206  
 MD31070, 206  
 MD31080, 206  
 MD31700, 205  
 MD31710, 205  
 MD31720, 205  
 MD31730, 205  
 MD32000, 146  
 MD32200, 146, 149  
 MD32250, 207  
 MD32260, 207  
 MD32300, 146, 154  
 MD32402, 146  
 MD32410, 146  
 MD32450, 222  
 MD32452, 223  
 MD32454, 223  
 MD32456, 225  
 MD32457, 225  
 MD32490, 301, 309  
 MD32500, 301, 302, 308  
 MD32510, 301, 309  
 MD32520, 303, 309  
 MD32530, 309  
 MD32540, 303  
 MD32550, 309  
 MD32560, 309  
 MD32570, 309  
 MD32610, 146, 293  
 MD32620, 146, 290  
 MD32630, 291  
 MD32650, 294  
 MD32700, 241, 252, 255  
 MD32710, 241, 249, 252  
 MD32711, 249, 250  
 MD32720, 250  
 MD32730, 250  
 MD32750, 218, 219, 306, 310  
 MD32760, 218  
 MD32800, 146, 294  
 MD32810, 146, 207, 292  
 MD32900, 296  
 MD32910, 146, 295  
 MD32960, 227  
 MD34210, 223  
 MD36010, 149  
 MD36012, 151  
 MD36020, 149  
 MD36030, 150, 152  
 MD36040, 150  
 MD36050, 152, 161  
 MD36051, 152  
 MD36052, 153, 154, 156, 158, 160  
 MD36100, 173  
 MD36110, 173  
 MD36120, 173  
 MD36130, 173  
 MD36200, 163  
 MD36210, 162  
 MD36220, 162  
 MD36300, 166  
 MD36310, 165, 168  
 MD36312, 168  
 MD36400, 146  
 MD36500, 211  
 MD36600, 172  
 MD36610, 147, 150, 151, 154, 163, 164, 167, 169, 324  
 MD36620, 324  
 MD38000, 243  
 MD51160, 82  
 MD51161, 82  
 MD51162, 82  
 MSEC, 242  
 Multiplication  
     Table, 248
- N**
- Non-linear signal distortions, 145

NPROT, 116  
NPROTDEF, 113

## P

Parking, 186  
PROTA, 88  
PROTD, 90  
Protection areas, 105  
    Restrictions, 134  
Protection zones, 113  
PROTS, 89

## S

Safety clearance, 70  
SD41300, 249  
SD43400, 179  
SD43410, 179  
SD43420, 178  
SD43430, 178  
SD43900, 217  
SD43910, 217  
SD43920, 217, 219  
Setup  
    Collision avoidance, 67  
Signal distortions, 145  
Software limit switch, 173  
States  
    Protection areas, 68  
Switching over the encoder data set, 204

## T

table  
    Compensation, 240  
Temperature  
    -compensation, 214  
    -influence, 214  
Temperature compensation  
    Coefficient  $\tan\beta(T)$ , 219  
Time constant  
    Dynamic response adaptation, 295

## W

WALCS0, 182  
WALIMOF, 179  
WALIMON, 179

Working area limitation, 176  
    in BCS, 178  
    in WCS/SZS, 180  
Working area limitation group, 180

## Z

Zero mark diagnostics, 168