Description of Functions 11/2002 Edition

sinumerik

Special Functions (Part 3)
SINUMERIK 840D/840Di/810D (CCU2)

SIEMENS

SIEMENS

# SINUMERIK 840D/840Di
# SINUMERIK 810D (CCU2)

# Special Functions (Part 3)

## Description of Functions

**Valid for**

| *Control* | *Software Version* |
|---|---|
| SINUMERIK 840D | 6 |
| SINUMERIK 840DE (export version) | 6 |
| SINUMERIK 840D powerline | 6 |
| SINUMERIK 840DE powerline | 6 |
| SINUMERIK 840Di | 2 |
| SINUMERIK 840DiE (export version) | 2 |
| SINUMERIK 810D | 3 |
| SINUMERIK 810DE (export version) | 3 |
| SINUMERIK 810D powerline | 6 |
| SINUMERIK 810DE powerline | 6 |

**11.02 Edition**

# SINUMERIK® Documentation

## Printing history

Brief details of this edition and previous editions are listed below.

The status of each edition is shown by the code in the “Remarks” column.

*Status code in the “Remarks” column:*

**A** . . . . . New documentation.
**B** . . . . . Unrevised reprint with new order no.
**C** . . . . . Revised edition with new status.
If factual changes have been made on the page in relation to the same software version, this is indicated by a new edition coding in the header on that page.

| Edition | Order No. | Remarks |
|---|---|---|
| 04.95 | 6FC5 297-2AC80-0BP0 | **A** |
| 03.96 | 6FC5 297-3AC80-0BP0 | **C** |
| 08.97 | 6FC5 297-4AC80-0BP0 | **C** |
| 12.97 | 6FC5 297-4AC80-0BP1 | **C** |
| 12.98 | 6FC5 297-5AC80-0BP0 | **C** |
| 08.99 | 6FC5 297-5AC80-0BP1 | **C** |
| 04.00 | 6FC5 297-5AC80-0BP2 | **C** |
| 10.00 | 6FC5 297-6AC80-0BP0 | **C** |
| 09.01 | 6FC5 297-6AC80-0BP1 | **C** |
| 11.02 | 6FC5 297-6AC80-0BP2 | **C** |

This book is part of the documentation on CD-ROM **(DOCONCD)**

| Edition | Order No. | Remarks |
|---|---|---|
| 11.02 | 6FC5 298-6CA00-0BG3 | **C** |

**Trademarks**
SIMATIC®, SIMATIC HMI®, SIMATIC NET®, SIROTEC®, SINUMERIK® and SIMODRIVE® are trademarks of Siemens. Other product names used in this documentation may be trademarks which, if used by third parties, could infringe the rights of their owners.

Further information is available on the Internet under:
http://www.ad.siemens.de/sinumerik

This publication was produced with Interleaf V7.



Other functions not described in this documentation might be executable in the control. However, no claim can be made regarding the availability of these functions when the equipment is first supplied or for service cases.

We have checked that the contents of this document correspond to the hardware and software described. Nonetheless, differences might exist and therefore we cannot guarantee that they are completely identical. The information contained in this document is, however, reviewed regularly and any necessary changes will be included in the next edition. We welcome suggestions for improvement.

Subject to changes without prior notice.

Order No. 6FC5-297-6AC80-0BP2
Printed in Germany

Siemens Aktiengesellschaft

# Preface

## Reader's note

The SINUMERIK documentation is structured in 4 levels:

- General documentation
- User documentation
- Manufacturer/service documentation
- OEM documentation.

This documentation is intended for machine tool manufacturers. It gives a detailed description of the functions available in the SINUMERIK controls.

The descriptions of functions are only valid for the software versions specified. For new software versions, the relevant descriptions are available on request. Earlier descriptions of functions are only partly applicable for new software versions.

For more detailed information on SINUMERIK 840D/840Di/810D and other publications covering all SINUMERIK controls (e.g. universal interface, measuring cycles ...), please contact your local Siemens office.

---

**Note**

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

---

## Hotline

If you have any questions about the control, please contact the hotline:

A&D Technical Support
Phone: ++49-180-5050-222
Fax: ++49-180-5050-223
Email: adsupport@siemens.com

Please send any questions about the documentation (suggestions for improvement, corrections) to the following fax number or email address:
Fax: ++49-9131-98-2176
Email: motioncontrol.docu@erlf.siemens.de

Fax form: see reply form at the end of the manual.

## Internet address SINUMERIK

http://www.ad.siemens.de/sinumerik

**SINUMERIK 840D powerline**

As from 09/2001

- SINUMERIK 840D powerline and
- SINUMERIK 840DE powerline

will be available with improved performance. A list of all available **powerline** modules you will find in Section 1.1 of the Hardware Description /PHD/.

**SINUMERIK 810D powerline**

As from 12/2001

- SINUMERIK 810D powerline aund
- SINUMERIK 810DE powerline

will be available with improved performance. A list of all available **powerline** modules you will find in Section 1.1 of the Hardware Description /PHC/.

**Objective**

The Descriptions of Functions provide the information required to configure and start-up the control.

**Target groups**

The Descriptions of Functions include information for:

- the configuring engineer
- the PLC programming engineer for the creation of the PLC user program with the signals listed
- the start-up engineer after system configuration and installation
- the service technician for checking and interpreting the status displays and alarms.

**Information on using this manual**

This Function Manual is structured as follows:

- Overall table of contents (main headings) of the manual
- Descriptions of functions in alphanumeric sequence according to the Function Description codes
- Appendix with index.

---

**Note**

The **Basic Machine** (Part 1) Description of Functions contains a glossary as well as lists of abbreviations, terms used and references.

The footer on each page contains the following information:
Part of the Description of Functions / Manual / Section – Page

---

If you require more information about a particular function, you can find the function and its sorting code on the inside front page of the manual.

If you require information on a certain term, please refer to the index in the appendix. The index lists the code of the Description of Functions, the section number and the page on which information relevant to the term you are seeking can be found.

Chapters 4 and 5 in each individual Description of Functions provide definitions of “Effectiveness, data format, input limits, etc.” for the various signals and data. These definitions are explained below under “Technical information”.

!

**Important**

This documentation is valid for:

- SINUMERIK 840D control,
  SW 6
- SINUMERIK 810D control,
  SW 6
- SINUMERIK 840Di control
  SW 2

**Specification of software version**

The software versions specified in this documentation refer to the SINUMERIK 840D control system. The equivalent software version for the SINUMERIK 810D (if the function is enabled, refer to /BU/, Catalog NC 60) is not explicitly specified. In this case, the following applies:

Table 1-1 Corresponding software version

| SINUMERIK 840D | | SINUMERIK 810D | SINUMERIK 810D powerline | SINUMERIK 840Di |
|---|---|---|---|---|
| 6.3 (09.01) | corresponds to | – | 6.1 (12.01) | 2.1 (07.01) |
| 5.3 (04.00) | corresponds to | 3.3 (04.00) | – | 1.1 (07.00) |
| 3.7 (03.97) | corresponds to | 1.7 (03.97) | – | – |

## Explanation of symbols

**Important**

This symbol always appears in the documentation when important information is being conveyed.

**Ordering data option**

In this documentation, you will find this symbol with a reference to an order data option. The function described is executable only if the control contains the designated option.

**Machine manufacturer**

This symbol appears in this documentation whenever the machine manufacturer can influence or modify the described functional behavior. Please observe the information provided by the machine manufacturer.

**Danger**

Indicates an imminently hazardous situation which, if not avoided, **will** result in death or serious injury or in substantial property damage.

**Warning**

Indicates a potentially hazardous situation which, if not avoided, **could** result in death or serious injury or in substantial property damage.

**Caution**

Used with the safety alert symbol indicates a potentially hazardous situation which, if not avoided, **may** result in minor or moderate injury or in property damage.

**Caution**

Used without safety alert symbol indicates a potentially hazardous situation which, if not avoided, **may** result in property damage.

**Notice**

Used without the safety alert symbol indicates a potential situation which, if not avoided, **may** result in an undesirable result or state.

## Technical information

**Notations**

The following notations and abbreviations are used in this documentation:

- PLC interface –> IS "Signal name" (Signal date)
  Example: – IS "MMC-CPU1 ready" (DB10, DBX108.2) means that the signal is filed in data block 10, data byte 108, bit 2.
  – IS "Feedrate/spindle override" (DB31–48, DBB0) means that the signals are located referring to spindles and axes in the data blocks 31 to 48, data block byte 0.
- Machine data –> MD: MD_NAME (English designation)
- Setting data –> SD: SD_NAME (English designation)
- The symbol "≙" means "corresponds to".

**Explanation of terms and abbreviations in Chapters 4 and 5**

Chapter 4 and 5 of each Description of Functions contains a description of the data and signals which are important for the relevant function. Some of the terms and abbreviations used in this tabulated, explanatory text are explained in more detail below.

Values in the table

The machine data used in the Description of Functions are always values for an NCU 572.

The values for another NCU (for example, NCU 570, NCU 571, NCU 573) are in the list manual.

**References:** /LIS/ "Lists"

Default value

The machine/setting data is automatically preset to this value during start-up. If the default values depending on the channels are different, these are specified by " / ".

Range of values (minimum and maximum values)

Specifies the input limits. If no value range is defined, the input limits are defined by the data type and the field is characterized by "∗∗∗".

Effectiveness of changes

Changes to machine data, setting data, or similar parameters do not necessarily take immediate effect in the control. For this reason, the conditions for activating a new setting are always specified. The following list shows the applied conditions in prioritized order:

- POWER ON (po) "RESET" key on front panel of NCU module or disconnection/connection of power supply
- NEW_CONF (cf) – "New configuration" function via PLC interface
  – "RESET" key on control unit or
- RESET (re) "RESET" key on control unit or
- Immediately (in) after value entry

Protection level

There are protection levels 0 to 7. Levels 0 to 3 (4 to 7) can be enabled by entering a password (keyswitch setting). The operator only has access to information on the level for which he is authorized and for all lower levels. Various protection levels are assigned to machine data as standard.

The table only lists the write protection level. However, there is a fixed assignment between write and read levels:

| Write protection level | Read protection level |
|---|---|
| 0 | 0 |
| 1 | 1 |
| 2 | 4 |

**References:** /BA/, Operator's Guide
/FB/, A2, "Various Interface Signals"

Unit

The unit refers to the default settings of the machine data SCALING_FACTOR_USER_DEF_MASK and
SCALING_FACTOR_USER_DEF.
If the MD is not governed by any particular physical unit, then the symbol "–" is entered in the unit field.

Data type

The following data types are used in the control system:

- DOUBLE
  Real or integer values (decimal values or integers)
  Input limits from $+/-4.19*10^{-307}$ to $+/-1.67*10^{308}$
- DWORD
  Integer values
  Input limits from $-2,147*10^{9}$ to $+2,147*10^{9}$
- BOOLEAN
  Possible input values: true or false and/or 0 or 1
- BYTE
  Integer values from –128 to +127
- STRING
  consisting of max. 16 ASCII characters (capital letters, digits and underscore).

**Data management**

The explanations of the PLC interface in the individual Function Descriptions are based on a theoretical maximum number of components:

- 4 operating mode groups (associated signals stored in DB11)
- 8 channels (associated signals stored in DB21–30)
- 31 axes (associated signals stored in DB31–61).

For the number of components which can actually be implemented in the relevant software version, please refer to
**References:** /BU/, "Ordering document" Catalog NC 60

■

# SINUMERIK 840D/840Di/810D Description of Functions Special Functions (Part 3)

# 3-Axis to 5-Axis Transformation (F2)

■

# Notes

# Brief Description 1

## 1.1 5-axis transformation

**Functionality**

The "5-axis transformation" machining package (see Section 2.1) is designed for machining sculptured surfaces with machine tools that have two rotary axes in addition to the three linear axes X, Y and Z:
This package thus allows an axially symmetrical tool (milling cutter, laser beam) to be oriented in any desired relation to the workpiece in the machining space.

The path and path velocity are programmed in the same way as for 3-axis tools. The tool orientation is programmed additionally in the motion blocks.
The real-time transformation performs the calculation of the resulting motion of all 5 axes. The generated machining programs are therefore not machine specific. Kinematic-specific post-processors are not used for the 5-axis machining operation.
A selection of various transformations is available for adapting the control to various machine kinematics. Parts program commands can be issued in operation to switch over between two transformations parameterized during start-up.

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

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

The calculation also includes a tool length compensation.

Since the orientation in relation to the workpiece surface is stored in a separate FRAME, a tool retraction operation with vertical orientation to the workpiece is also possible.

**Tool orientation**

The tool orientation can be specified in two ways:

- In relation to machine
- In relation to workpiece.

Machine-related orientation

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

Workpiece-related orientation

The workpiece-related orientation is not dependent on the machine kinematics. It is programmed by means of:

- Euler angles
- RPY angles
- Vector components.

These elements define the direction of the tool in the workpiece coordinate system. It is possible to program a specific component of the tool in its orientation to the workpiece. In most cases, this will be a longitudinal axis of the tool with the tool tip (Tool Center Point TCP), which is also termed "TCP programming".

System variables for orientation

Parts programs and synchronized actions have read access to system variables that provide the following information:

- End orientation of block (run-in value)
- Orientation setpoint (SW 6.4 and higher)
- Actual orientation (SW 6.4 and higher)
- Angle between setpoint and actual orientation (SW 6.4 and higher)
- Status for actual orientation variable (SW 6.4 and higher)

Chapter 4 contains a detailed description of this.

**Further transformations**

The transformations described in the following sections are to be regarded as special cases of the general 5-axis transformation described above:

- 3-axis and 4-axis transformations
  With 2 or 3 linear axes and **one** rotary axis.
- Swiveling linear axis
  One of the rotary axis rotates the 3rd linear axis.
- Universal milling head
  The two rotary axes are positioned at a configurable angle in relation to one another.

For an overview of these functions, please refer to sections 1.2 to 1.4; for a more detailed description, refer to 2.2 to 2.4. Knowledge of the general 5-axis transformation is a prerequisite for all of these transformations.

## 1.2 3-axis and 4-axis transformation

**Definition**

In contrast to the transformations described in Section 1.1, the 3-axis and 4-axis transformations have the following characteristics:

- 3-axis transformation
  - Two translatory axes
  - One rotary axis
- 4-axis transformation
  - Three translatory axes
  - One rotary axis.

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

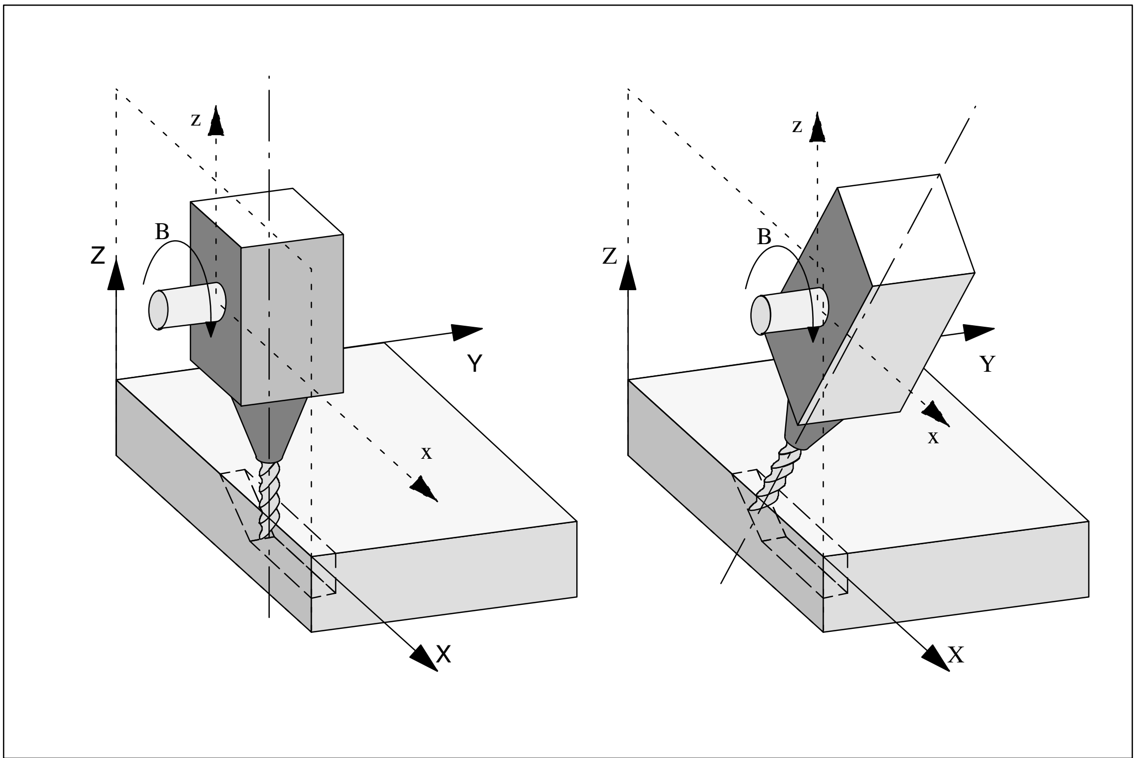



Fig. 1-1 Schematic diagram of 3-axis transformation

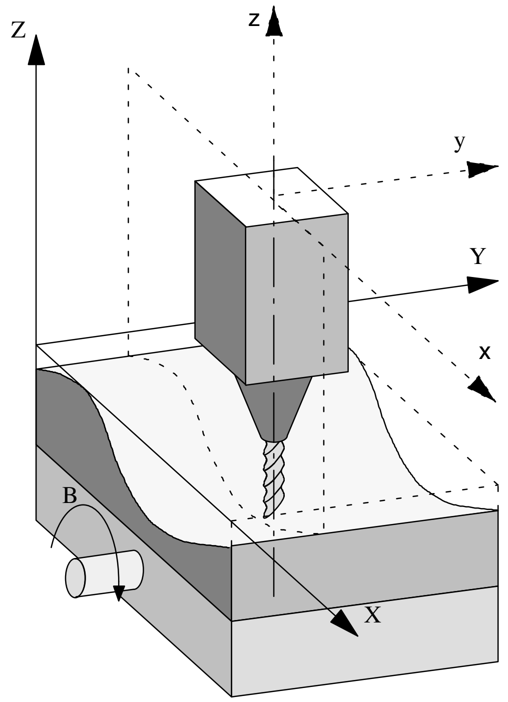



Fig. 1-2 Schematic diagram of 4-axis transformation with movable workpiece

A detailed description of the possible kinematics for 3-axis and 4-axis transformations can be found in Section 2.2.

## 1.3 Orientation transformation with swiveling linear axis

**Introduction**

This type of transformation is similar to the 5-axis transformation for machine type 3 described in Section 1.1. However, the 3rd linear axis is not always perpendicular to the plane defined by the other two linear axes.

**Features of machine kinematics**

Machine kinematics, for which the orientation transformation described in the following section applies, can be described as follows:

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

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

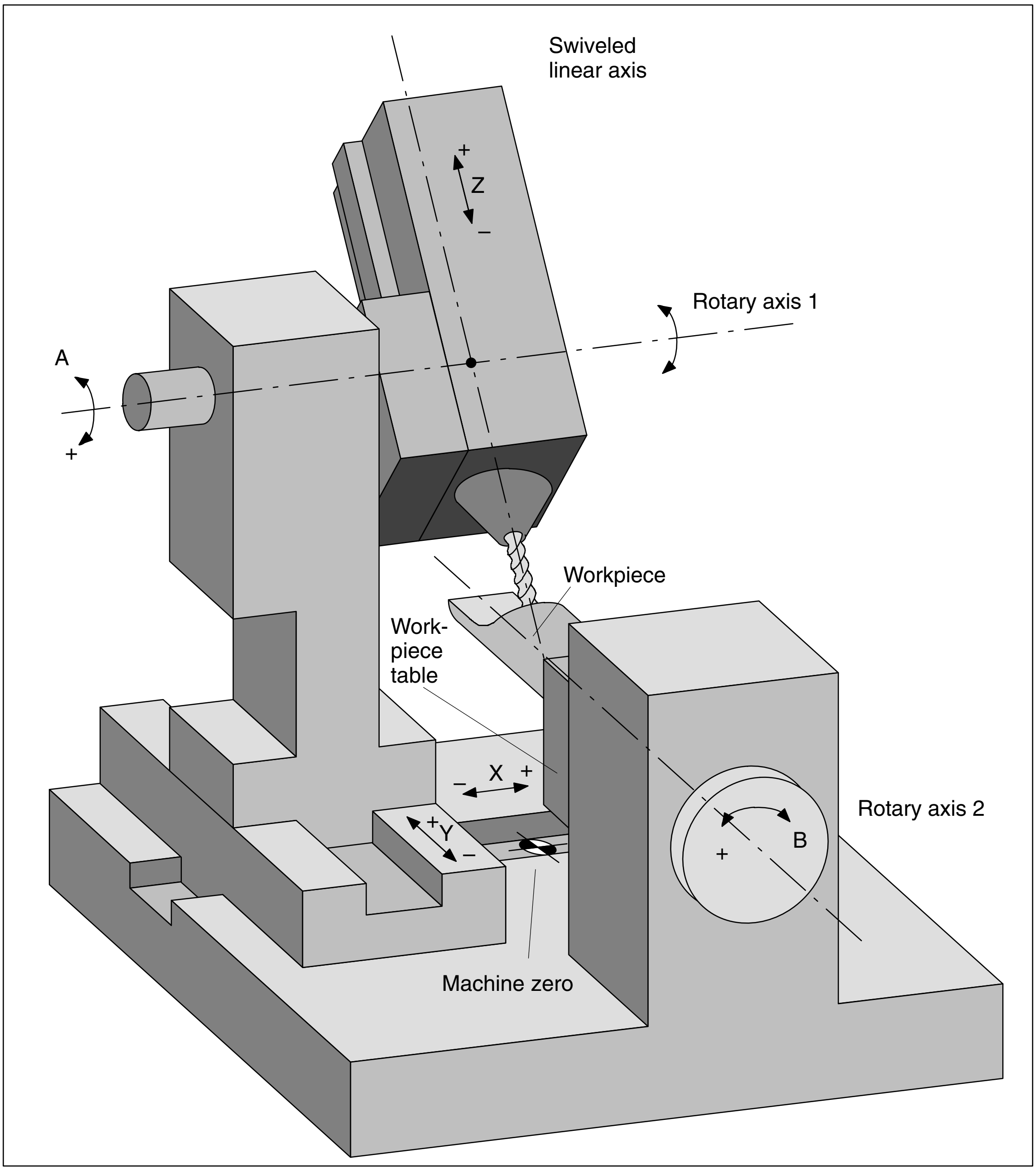



Fig. 1-3 Schematic diagram of a machine with swiveling linear axis

## 1.4 Universal milling head

**Features**

A machine tool with universal milling head is characterized by the following features:

The machine tool for the universal milling head has at least 5 axes

- 3 linear axes (for linear motions) [X, Y, Z] move the machining point to any desired position in the machining space.
- 2 rotary swivel axes that are arranged at a configurable angle (usually 45 degrees), allow the tool to swivel to positions in space that are limited to a half sphere with the 45 degrees configuration.

## 1.5 Orientation axes (SW 5.3 and higher)

**Introduction**

With regard to the kinematics of robots, hexapods or nutators, there is no such simple correlation between axis motion and change in orientation as is the case on conventional 5-axis machines.

For this reason, the change in orientation is defined by a model that is created independently of the actual machine. This model defines three virtual orientation axes which can be visualized as rotations about the coordinate axes of a rectangular coordinate system.

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

**Definition**

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

– Geometry axes

– Orientation axes

**Geometry axes** describe the working point.
**Orientation axes** describe the orientation of the tool in space.

**Tool orientation**

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

– Direct Programming of rotary axis positions A, B, C

– For 5-axis transformation, by programming:
the Euler or RPY angle in degrees via A2, B2, C2 or
the direction vector via A3, B3, C3

– Programming via leading angle LEAD and tilt angle TILT

## 1.6 Cartesian manual travel (SW 6.3 and higher)

The Cartesian manual travel function allows you to set the

Basic Coordinate System (BCS),
Workpiece Coordinate System (WCS),
and the Tool Coordinate System (TCS)

separately as reference system both for the translation and for the orientation.

## 1.7 Cartesian PTP travel (SW 5.3 and higher)

PTP = **P**oint-**t**o-**P**oint motion

**Introduction**

This function makes it possible to program a position in a Cartesian coordinate system (workpiece coordinate system), while the machine traverses in the machine coordinate system.

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

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

## 1.8 Generic 5-axis transformation (SW 5.2 and higher)

**Introduction**

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

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

Detailed description given in Section 2.6.

## 1.9 On-line tool length offset (SW 6.4 and higher)

**Introduction**

You can use the system variable $AA_TOFF[ ] to overlay the effective tool lengths in 3-D at runtime. These offsets are active for

active orientation transformation (TRAORI) or an
active tool carrier in the relevant tool direction.

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

Detailed description given in Section 2.6.

## 1.10 Activation via parts program/softkey (SW 5.2 and higher)

Most of the machine data relevant to kinematic transformations were activated by power ON in earlier versions.

As of SW 5.2, you can also activate transformations MDs via the parts program/softkey and it is not necessary to boot the control.

Detailed Description given in

**References:** /FB/ 2, M1, “Kinematic transformation”, Section 2.5.

## 1.11 Compression of orientation (SW 6.3 and higher)

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

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

**Solution up to SW 6.1**

Compressors COMPON, COMPCURV and COMPCAD can only be used in conjunction with special NC blocks in software versions up to SW 6.1.

- Only NC blocks in which the feed is programmed (with F) in addition to the axis motion, are compressed.
- Positions for the axes must be specified directly and cannot be programmed via assignments.
- In the case of NC programs for 5-axis machines, the tool orientation must be programmed by specifying rotary axis positions in order to activate the compressor.

This means that you can only run 5-axis programs with the compressor if the orientation is programmed directly from the rotary axis motion, independent of the kinematics.

**Solution option as of SW 6.3**

You can program the tool orientation **independent of the kinematics** by using direction vectors.

NC programs with such direction vectors can be executed with compressors COMPON, COMPCURV and COMPCAD.

You will find a detailed description in Section 2.7.

■

## Notes

# Detailed Description 2

## 2.1 5-axis transformation

### 2.1.1 Kinematic transformation

**Task of orientation transformation**

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

**Applications**

The “5-axis transformation” machining package is provided for machine tools which have not only three linear axes X, Y and Z, but also two additional rotary axes (rotation about the linear axes): This package thus allows an axially symmetrical tool (milling cutter, laser beam) to be oriented in any desired relation to the workpiece in every point of the machining space.

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

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

The kinematic transformation does not act on positioning axes.

## 2.1.2 Machine types for 5-axis transformation

**Kinematics of machines for 5-axis transformation**

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

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

   Example:

   – A parallel to X
   – B parallel to Y
   – C parallel to Z
3. Rotary axes are positioned vertically one above the other.
4. Rotary axes turn
   – Tool with two-axis swivel head (machine type 1)
   – Workpiece with two-axis rotary table (machine type 2)
   – Tool and workpiece with single-axis rotary table and swivel head (machine type 3)
5. The following applies to machine types 1 and 2:
   – Rotary axis 1 is treated as the 4th machine axis of the transformation.
   – Motion of 1st rotary axis changes the orientation of the 2nd rotary axis.
   – Rotary axis 2 is treated as the 5th machine axis of the transformation.
   – Motion of 2nd rotary axis does not change the orientation of the 1st rotary axis.
6. The following applies to machine type 3:
   – 1st rotary axis (4th machine axis of transformation) turns the tool.
   – 2nd rotary axis (5th machine axis of transformation) turns the workpiece.
7. Basic tool position:
   – In negative Z direction.

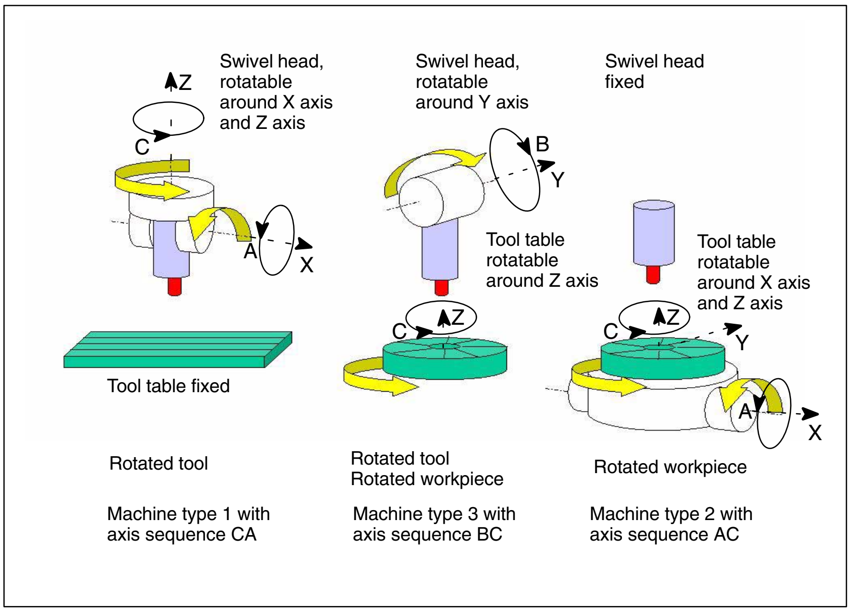



Fig. 2-1 Machine types for 5-axis transformation
Machine type 1 Two-axis swivel head
Machine type 2 Two-axis rotary table
Machine type 3 Single-axis swivel head and Single-axis rotary table

---

**Note**

Transformations not fulfilling all the conditions specified here are described in separate sections:
3-axis and 4-axis transformations under 2.2
Swiveling linear axis under 2.3
Universal milling head under 2.4.

---

### 2.1.3 Configuration of a machine for 5-axis transformation

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

- Machine type
- Axis assignment
- Geometry information
- Direction of rotation assignment.

**Machine type**

The machine types have been designated above as types 1 to 3 and are stored in machine data

$MC_TRAFO_TYPE_1 ... $MC_TRAFO_TYPE_8

as a two-digit number.

Table 2-1 gives a list of machine types which are suitable for 5-axis transformation.

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

Table 2-1 Overview of machine types which are suitable for 5-axis transformation

| **Machine type** | **1** | **2** | **3** |
|---|---|---|---|
| Swivel/rotatable | Tool | Workpiece | Tool/workpiece |
| Axis sequence | | | |
| | | | |
| AB | 16 | 32 | 48 |
| AC | x | 33 | 49 |
| | | | |
| BA | 18 | 34 | 50 |
| BC | x | 35 | 51 |
| | | | |
| CA | 20 | x | x |
| CB | 21 | x | x |

**Identification of axis sequence**

The axis sequence is identified in the following way:

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

**Axis assignment**

The axis assignment at the input of the 5-axis transformation defines which axis is imaged internally on a channel axis by the transformation. This assignment is defined in MD: $MC_TRAFO_AXES_IN_1 ... $MC_TRAFO_AXES_IN_8.

**Geometry information**

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

MD: $MC_TRAFO5_PART_OFFSET_1
Workpiece-oriented offset

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

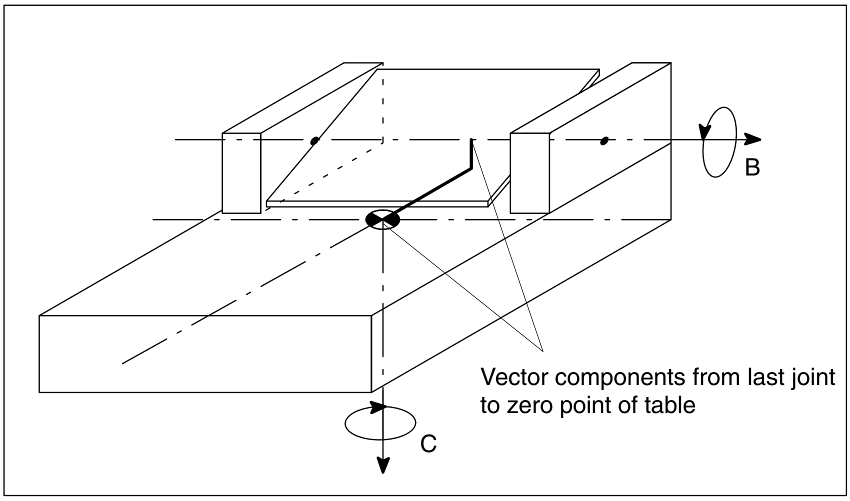



Fig. 2-2 Machine data $MC_TRAFO5_PART_OFFSET_1 for machine type 2

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

MD: $MC_TRAFO5_JOINT_OFFSET_1
Vector from the first to the second joint (machine type 1 and 2).
Vector from machine zero to joint of table (machine type 3).

MD: $MC_TRAFO5_ROT_AX_OFFSET_1
Angle offset of first or second rotary axis

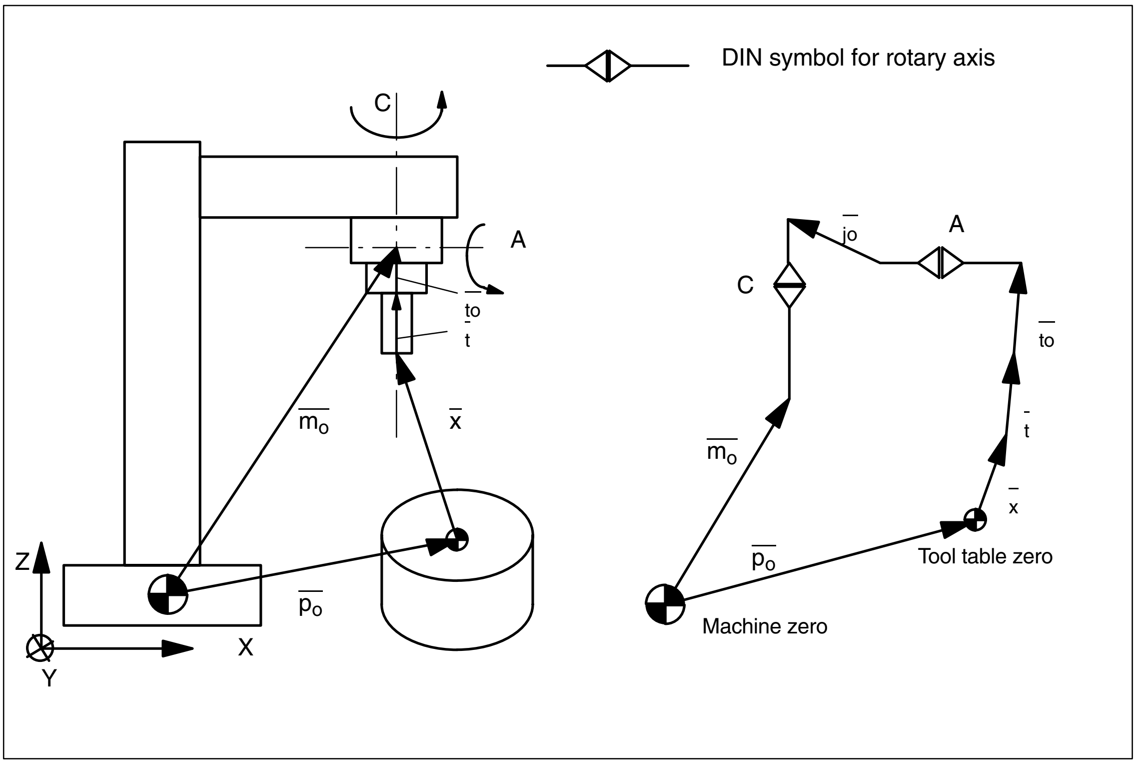



Fig. 2-3 Schematic diagram of CA kinematics, moved tool

| | |
|---|---|
| $\overline{mo}$ | Position vector in MCS |
| $\overline{po}$ | $MC_TRAFO5_PART_OFFSET_n[0 ..2] |
| $\overline{x}$ | Vector of programmed position in BCS |
| $\overline{t}$ | Tool offset vector |
| $\overline{to}$ | $MC_TRAFO5_BASE_TOOL_n[0 .. 2] |
| $\overline{jo}$ | $MC_TRAFO5_JOINT_OFFSET_n[0 .. 2] |

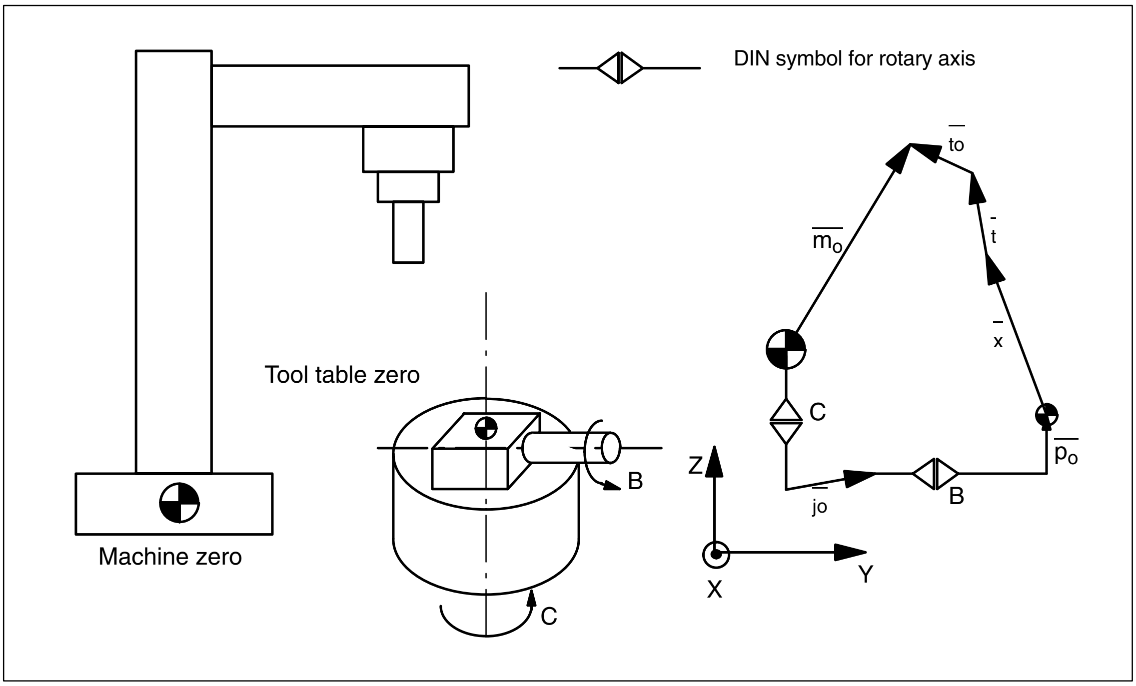



Fig. 2-4 Schematic diagram of CB kinematics, moved workpiece

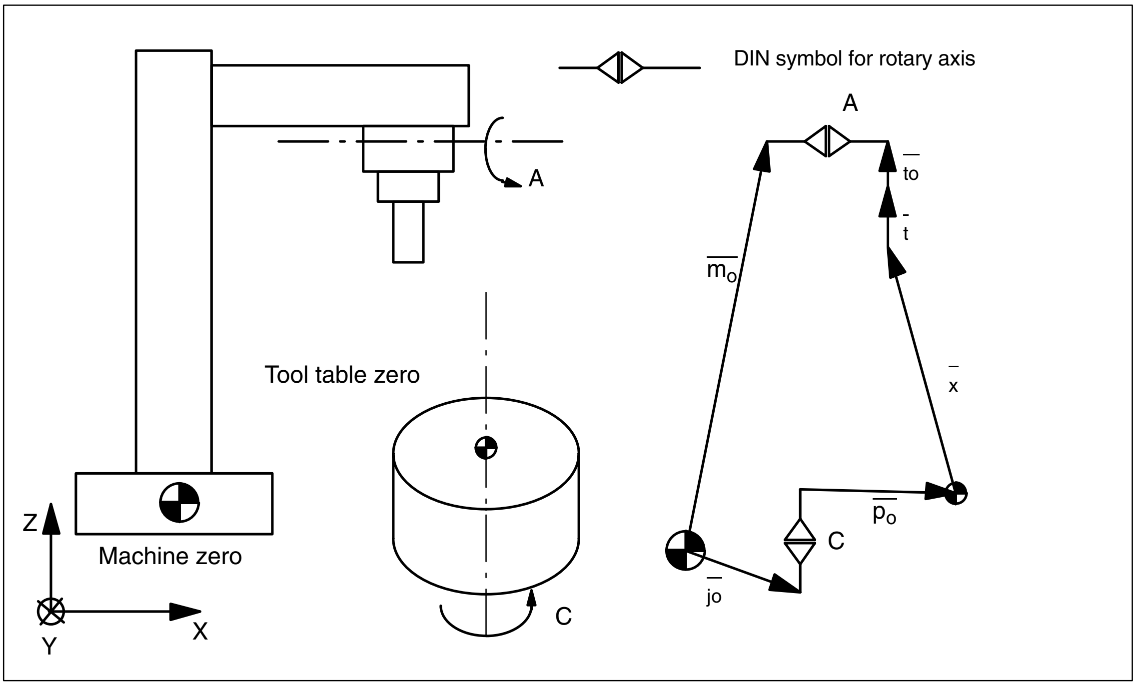



Fig. 2-5 Schematic diagram of AC kinematics, moved tool, moved workpiece

**Direction of rotation assignment**

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

MD 24520: TRAFO5_ROT_SIGN_IS_PLUS_1[n] 1st rotary axis
MD 24620: TRAFO5_ROT_SIGN_IS_PLUS_2[n] 2nd rotary axis

**Transformation types**

Eight transformation types per channel can be configured in machine data TRAFO_TYPE_1 to 8. Of these eight types, only two may be 5-axis transformations.

**Activation**

Activation of the 5-axis transformations is described in Section 2.5.

## 2.1.4 Tool orientation

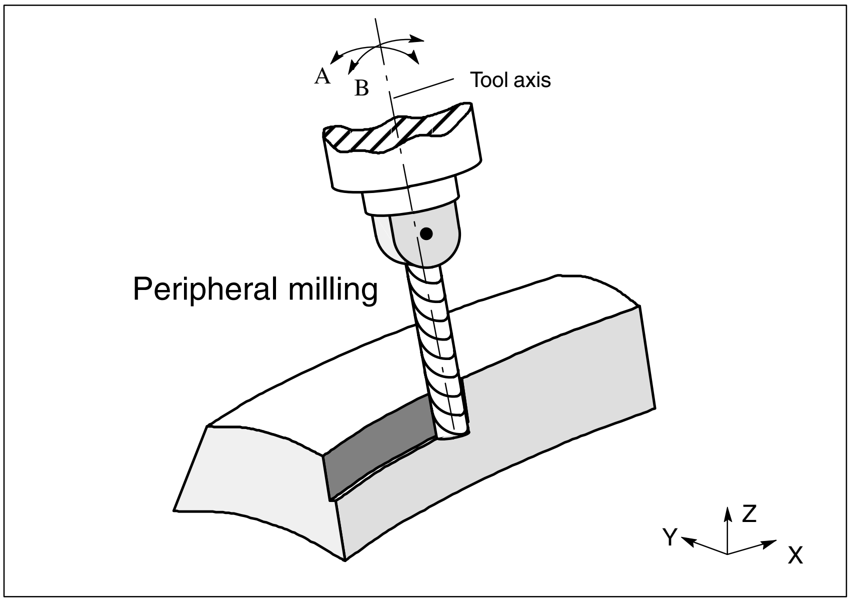



Fig. 2-6 Machining of workpieces with 5-axis transformation

**Programming**

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

- Directly as rotary axes A, B, C
- Indirectly for 5-axis transformation
  specifying the Euler or RPY angle in degrees via A2, B2, C2
- Indirectly for 5-axis transformation via direction vector A3, B3, C3.

The designators of Euler angles or directional vectors can be set in machine data:

| | |
|---|---|
| Euler angle via | MD 10620: EULER_ANGLE_NAME_TAB |
| Direction vector via | MD 10640: DIR_VECTOR_NAME_TAB |

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

**Euler or RPY**

Via MD 21100: ORIENTATION_IS_EULER it is possible to switch between Euler and RPY input.

**Orientation reference**

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

- In the workpiece coordinate system with command ORIWKS
- In the machine coordinate system with command ORIMKS.

**ORIWKS command**

The tool orientation is programmed in the workpiece coordinate system and is thus not dependent on the machine kinematics.
In the case of a change in orientation of a tool tip at a fixed point in space, the tool moves along a large arc on the plane stretching from the start vector to the end vector.

**ORIMKS command**

The tool orientation is programmed in the machine coordinate system and is thus dependent on the machining kinematics.
In the case of a change in orientation of a tool tip at a fixed point in space, linear interpolation takes place between the rotary axis positions.

The orientation is selected via NC language commands ORIWKS and ORIMKS.

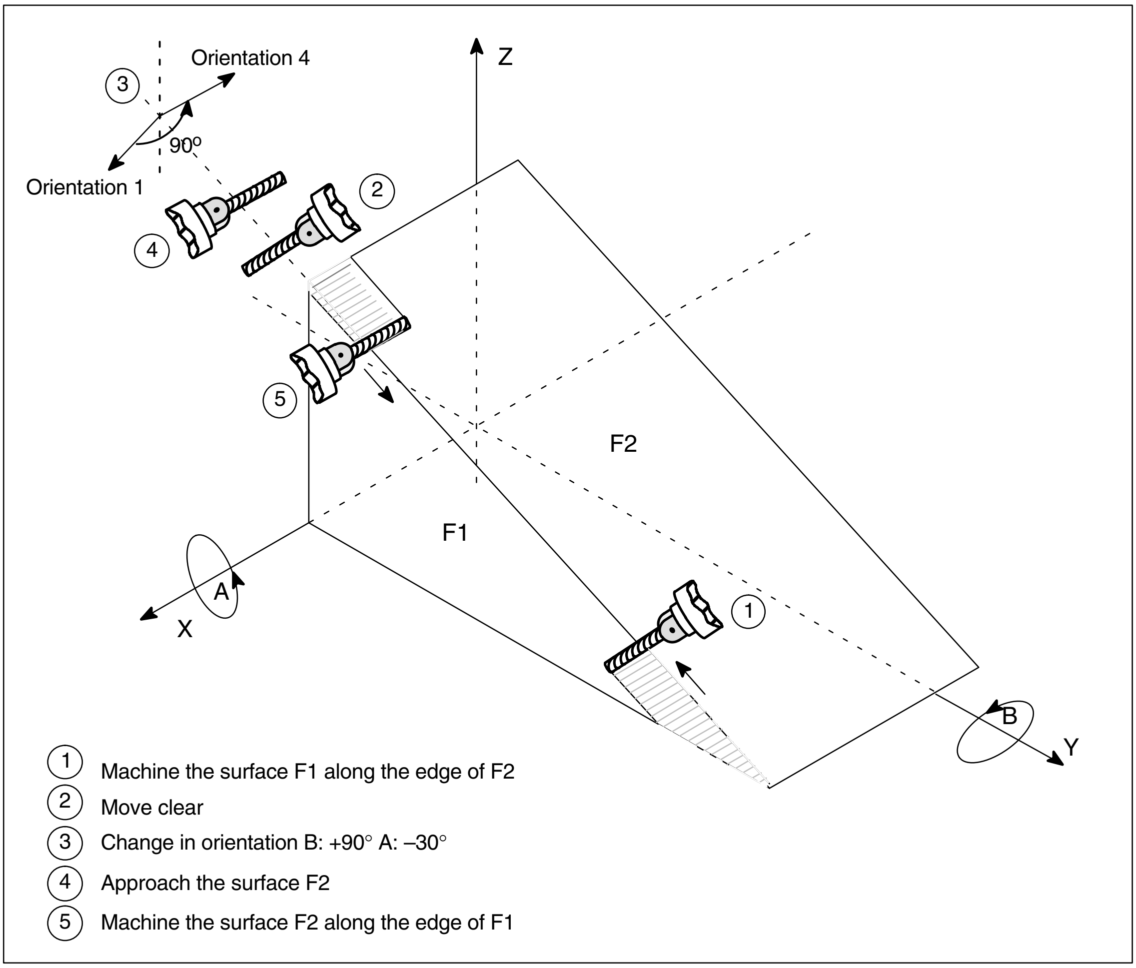



Fig. 2-7 Change in cutter orientation while machining inclined edges

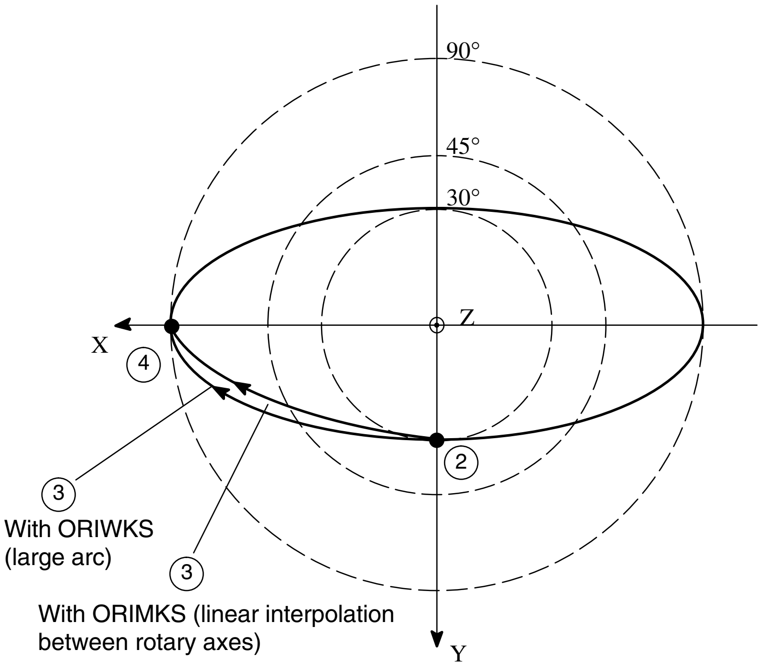



Fig. 2-8 Change in orientation while machining inclined edges

Basic setting is ORIMKS (SW 2 and higher).
The initial setting can be altered in MD 20150: GCODE_RESET_VALUES
MD 20150: GCODE_RESET_VALUES [24] = 1 ⇒ ORIWKS is the initial setting
MD 20150: GCODE_RESET_VALUES [24] = 2 ⇒ ORIMKS is the initial setting

**Improper tool orientation**

If the tool orientation is programmed in conjunction with the following functions

- G04 Dwell time
- G33 Thread cutting with constant lead
- G74 Approach reference point
- G75 Approach fixed point
- REPOSL Repositioning
- REPOSQ Repositioning
- REPOSH Repositioning.

Then alarm 12130 "Illegal tool orientation" is output when Euler angles and directional vectors are selected. The NC program stops (this alarm can also occur in connection with G331, G332 and G63). To remedy this situation, the tool orientation can be programmed with axis end values.

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

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

If a transformation changeover (power ON, power Off or change of transformation) is executed when tool radius compensation is active, alarm 14400 is output.
In the reverse situation, i.e. a tool radius compensation is selected or deselected when a transformation is active, no alarm message is output.

**Multiple input of tool orientation**

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

N50 A3=1 B3=1 C3=1

If the tool orientation is input several times, e.g. with directional vectors and Euler angles:

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

then error message 12240 "Channel X Block Y tool orientation xx defined more than once" is output and the NC parts program stops.

**Tool orientation using orientation vectors**

**As of SW 5.3** polynomials can also be programmed for the modification of the orientation vector.

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

The interpolation of orientation vectors can be programmed with polynomials up to the 5th degree. The polynomial interpolation of orientation vectors is described in Subsection 2.9.1.

---

**Note**

Further explanations of tool orientation using orientation vectors and their handling at machine tools are given in:

**References:** /FB/, W1 "Tool offset, orientable toolholder"

---

### 2.1.5 Singular positions and handling

**Extremely high velocity**

If the path runs in close vicinity to a pole (singularity), one or several axes may traverse at a very high velocity.

In this case, alarm 10910 “Excessive velocity of one axis” is output. The programmed velocity is then reduced to a value which does not exceed the maximum axis velocity.

**Behavior at pole**

Unwanted behavior of fast compensating movements can be controlled by making an appropriate selection of the following machine data (see Fig. 2-9):

MD 24530 or MD24630: TRAFO5_NON_POLE_LIMIT_1 or 2
MD 24540 or MD 24640: TRAFO5_POLE_LIMIT_1 or 2

---

**Note**

Singularities are dealt with differently in SW 5.2 and higher: There is now only one relevant machine data $MC_TRAFO5_POLE_LIMIT (see Subsection 2.6.7 or Programming Guide “Advanced” [PGA], Subsection 7.1.3).

---

**$MC_TRAFO5_NON_POLE_LIMIT**

This MD designates a limit angle for the fifth axis of the first (MD:
MD 24530: TRAFO5_NON_POLE_LIMIT_1 or the second
MD 24630: TRAFO5_NON_POLE_LIMIT_2 5-axis transformation with the following characteristics:

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

With the 5-axis transformation, a coordinate system consisting of circles of longitude and latitude is spanned over a spherical surface by the two orientation axes of the tool.
If, as a result of orientation programming (i.e. the orientation vector is positioned on one plane), the path passes so close to the pole that the angle is less than the value defined in this MD, then a deviation from the specified interpolation is made such that the interpolation passes through the pole.

**$MC_TRAFO5_POLE_LIMIT**

This MD designates a limit angle for the fifth axis of the first
MD 24540: TRAFO5_POLE_LIMIT_1 or the second
MD 24640: TRAFO5_POLE_LIMIT_2 5-axis transformation with the following characteristics:

With interpolation through the pole point, only the fifth axis moves; the fourth axis remains in its start position. If a movement is programmed which does not pass exactly through the pole point, but is to pass within the tolerance defined by $MC_TRAFO5_NON_POLE_LIMIT in the vicinity of the pole, a deviation is made from the specified path because the interpolation runs exactly through the pole point. As a result, the position at the end point of the fourth axis (pole axis) deviates from the programmed value.

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

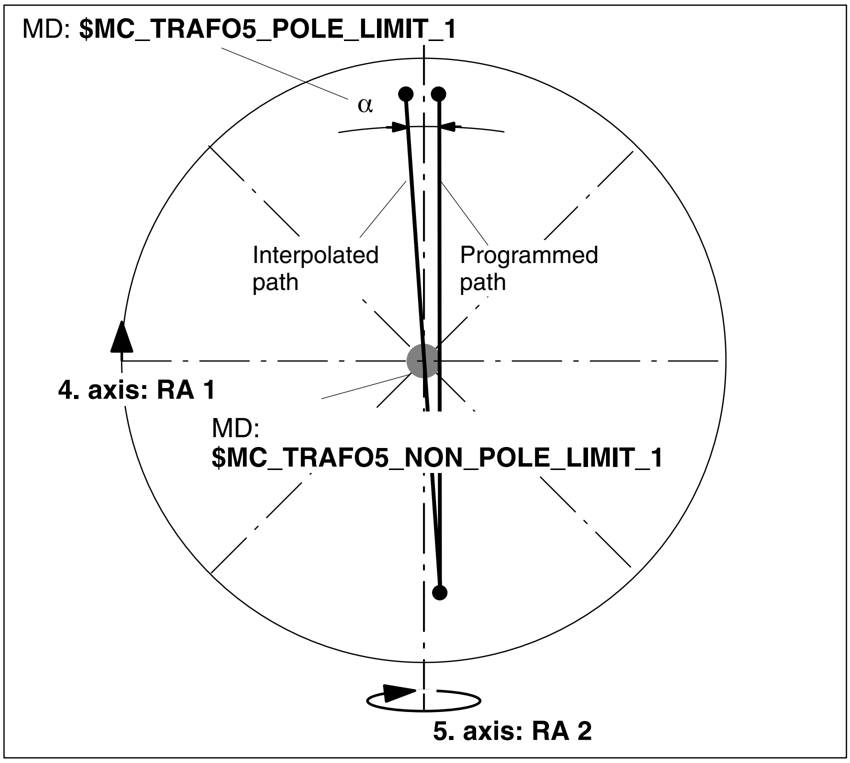



Fig. 2-9 5-axis transformation; orientation path in pole vicinity.
Example of machine type 1: 2-axis swivel head with rotary axis RA 1 (4th axis of transformation) and rotary axis RA 2 (5th axis of transformation)

**MD 21108**

Machine data MD 21108: POLE_ORI_MODE can be used to set the response for large circle interpolation in pole position as follows:

Does not define the treatment of orientation changes during large circle interpolation unless the starting orientation is equal to the pole orientation or approximates to it and the end orientation of the block is outside the tolerance circle defined in MD TRAFO5_NON_POLE_LIMIT_1/2.

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

Machine data MD 21108: POLE_ORI_MODE is decimal coded.
The **unit digits** define the behavior if the start orientation precisely matches the pole orientation and the
**ten digits** the behavior if the start orientation of the block is outside the tolerance circle defined in MD TRAFO5_NON_POLE_LIMIT_1/2.
All setting values are described in the section entitled “Channel-specific machine data”.

## 2.2 3-axis and 4-axis transformations

**Introduction**

The 3-axis and 4-axis transformations are special types of the 5-axis transformation described in Section 2.1. Orientation of the tool is possible only in the plane perpendicular to the rotary axis. The transformation supports machine types with movable tool and movable workpiece.

**Kinematics variants**

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

Table 2-2 Variants of 3-axis and 4-axis transformations

| Machine type | Swiveling/rotary | Rotary axis is parallel | Orientation plane | MD: $MC_TRAFO _TYPE_n | Tool orientation at zero position |
|---|---|---|---|---|---|
| 1 | Tool | X | Y–Z | 16 | Z |
| | | Y | X–Z | 18 | |
| | | Z | X–Y | 20 | Y |
| | | Z | X–Y | 21 | X |
| | | Any | Any * | 24 | Any |
| 2 | Workpiece | X | Y–Z | 32, 33 | Z |
| | | Y | X–Z | 34, 35 | |
| | | Any | Any * | 40 | Any |

Note: with reference to types 24 and 40 *
In the case of transformation types 24 and 40, the axis of rotation and the tool orientation can be set such that the orientation change takes place at the outside of a taper and not in a plane.

**Zero position**

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

**Axis assignments**

The three translatory axes included in the transformation are assigned to any channel axes via the machine data $MC_TRAFO_GEOAX_ASSIGN_TAB_n[0..2] and $MC_TRAFO_AXES_IN_n[0..2]. The following must apply for the assignment of channel axes to geometry axes for the transformation:

$MC_TRAFO_GEOAX_ASSIGN_TAB_n[0] = $MC_TRAFO_AXES_IN_n[0]
$MC_TRAFO_GEOAX_ASSIGN_TAB_n[1] = $MC_TRAFO_AXES_IN_n[1]
$MC_TRAFO_GEOAX_ASSIGN_TAB_n[2] = $MC_TRAFO_AXES_IN_n[2]

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

**Parameter assignment procedure**

- Enter the type of transformation according to Table 2-2 as machine data $MC_TRAFO_TYPE_n.
- Assign channel axes to the geometry axes of the transformation.
- For a 3-axis transformation, set the values for the axis which is not required:
  - $MC_TRAFO_GEOAX_ASSIGN_TAB_n[geoax] = 0
  - $MC_TRAFO_AXES_IN_n[geoax] = 0
    $MC_TRAFO_AXES_IN_n[4] = 0 ; there is no 2nd rotary axis
- For a 4-axis transformation, set the following for the 3 linear axes
  - $MC_TRAFO_GEOAX_ASSIGN_TAB_n[geoax] = ...
  - $MC_TRAFO_AXES_IN_n[geoax] = ...
    $MC_TRAFO_AXES_IN_n[4] = 0 ; there is no 2nd rotary axis.

Complete examples can be found in Section 6.2.

## 2.3 Transformation with swiveled linear axis

**Applications**

You can use a transformation with swiveling linear axis if your application is characterized by the kinematics described in Section 1.3 and only a small swivel range (<< ± 90 degrees) is crossed by the first rotary axis.

**Kinematics variants**

The orientation transformation with swiveled linear axis forms a transformation group of its own. It is defined in the machine data $MC_TRAFO_TYPE_n (n = 1, 2, 3, 4) by the following values:

| Transform. type | 1st rotary axis | 2nd rotary axis | Swiveled linear axis |
|---|---|---|---|
| 64 | A | B | Z |
| 65 | A | C | Y |
| 66 | B | A | Z |
| 67 | B | C | X |
| 68 | C | A | Y |
| 69 | C | B | X |

**Pole**

The corresponding transformation has a pole with a tool orientation parallel to the second rotary axis. Singularity occurs in the pole position because the third linear axis is parallel to the plane of the first two linear axes, excluding the possibility of compensating movements perpendicular to this plane.

**Parameter assignments**

The following machine data with the following meanings are used to adjust the transformation equations to the machine (n=1,2):

| | |
|---|---|
| $MC_TRAFO5_PART_OFFSET_n | Vector from the second rotary axis to the workpiece table zero |
| $MC_TRAFO5_ROT_AX_OFFSET_n | Axis positions of the two rotary axes at the initial position of the machine |
| $MC_TRAFO5_ROT_SIGN_IS_PLUS_n | Sign with which the rotary axis positions are included in the transformation |
| $MC_TRAFO5_JOINT_OFFSET_n | Vector from the machine zero to the second rotary axis |
| $MC_TRAFO5_BASE_TOOL_n | Vector from the toolholder (flange) to the first rotary axis (measured at machine initial position) |
| $MC_TRAFO5_TOOL_ROT_AX_OFFSET_n<br><br>(from SW 3.2) | Vector from machine zero to the first rotary axis (measured at machine initial position) |

**Definition of required values**

As an aid for defining the values for the above-mentioned machine data, the following two sketches show the basic interrelations between the vectors.

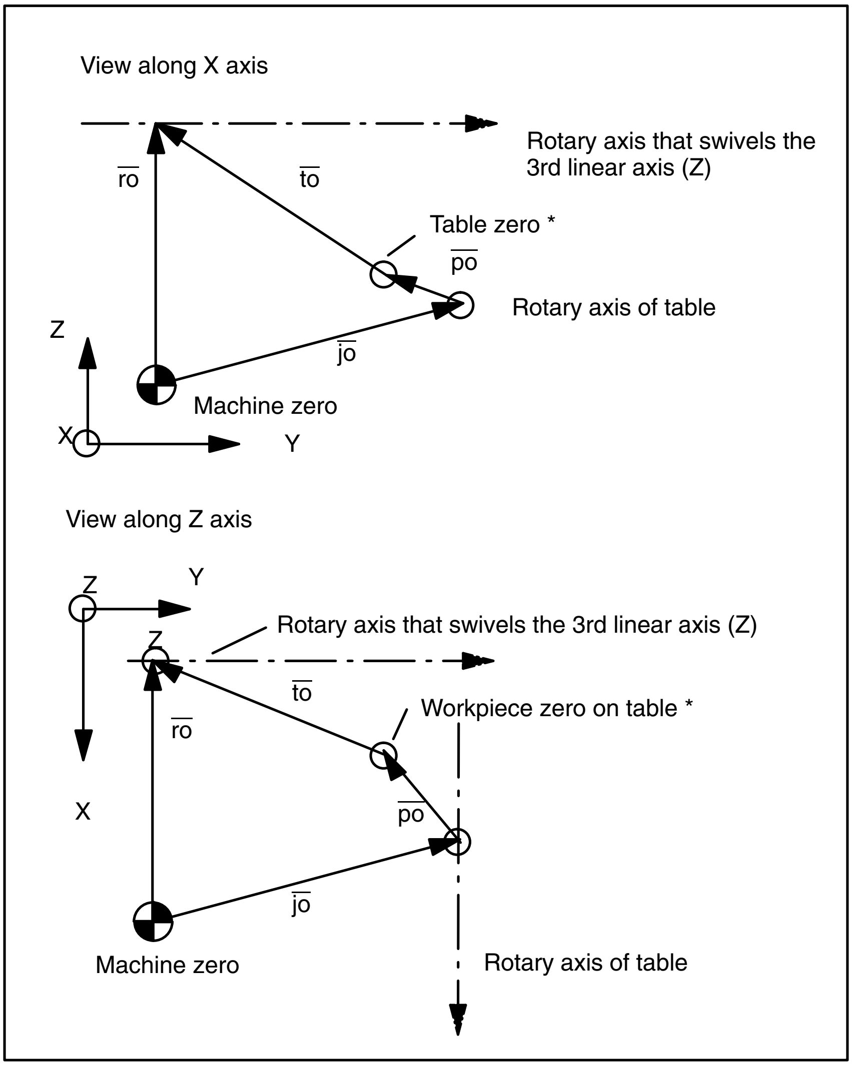



Fig. 2-10 Projections of the vectors to be set in MD

Meanings for the vector designations:

| | |
|---|---|
| $MC_TRAFO5_PART_OFFSET_n | $\overline{po}$ |
| $MC_TRAFO5_TOOL_ROT_AX_OFFSET_n | $\overline{ro}$ |
| $MC_TRAFO5_JOINT_OFFSET_n | $\overline{jo}$ |
| $MC_TRAFO5_BASE_TOOL_n | $\overline{to}$ |

---

**Note**

For the schematic diagram shown in Fig. 2-10, it has been assumed that the machine has been traversed so that the tool holding flange is in line with the table zero (marked by *). If this cannot be implemented for geometric reasons, the values for $\overline{to}$ must be corrected by the deviations.

---

Fig. 2-12 shows the vector components for the machine represented in Fig. 1-3 with their respective designations.

---

**Note**

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

---

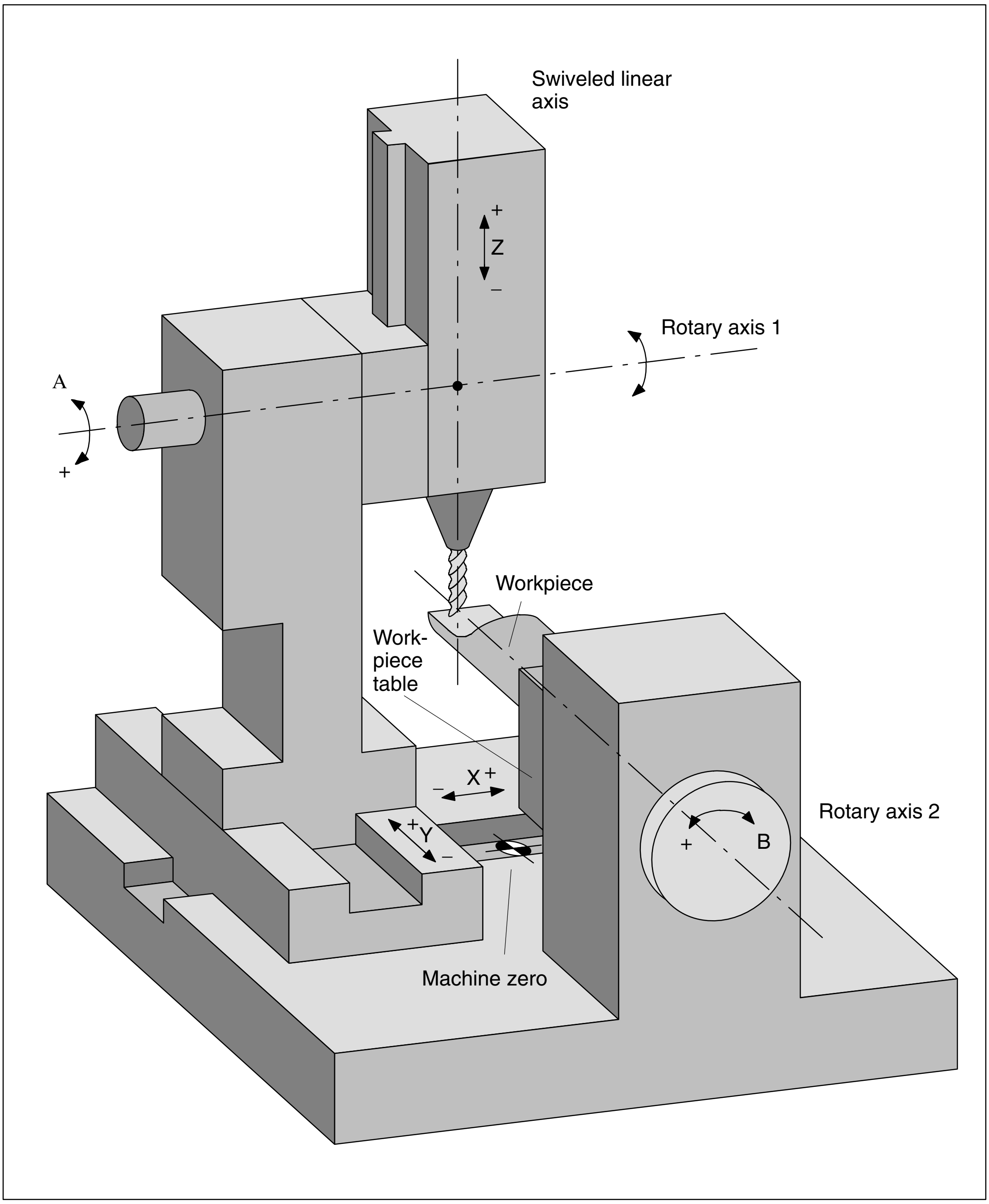



Fig. 2-11 Machine with swiveled linear axis in position zero

The following conversion of the geometry into the machine data to be specified is based on the example in Fig. 2-11.

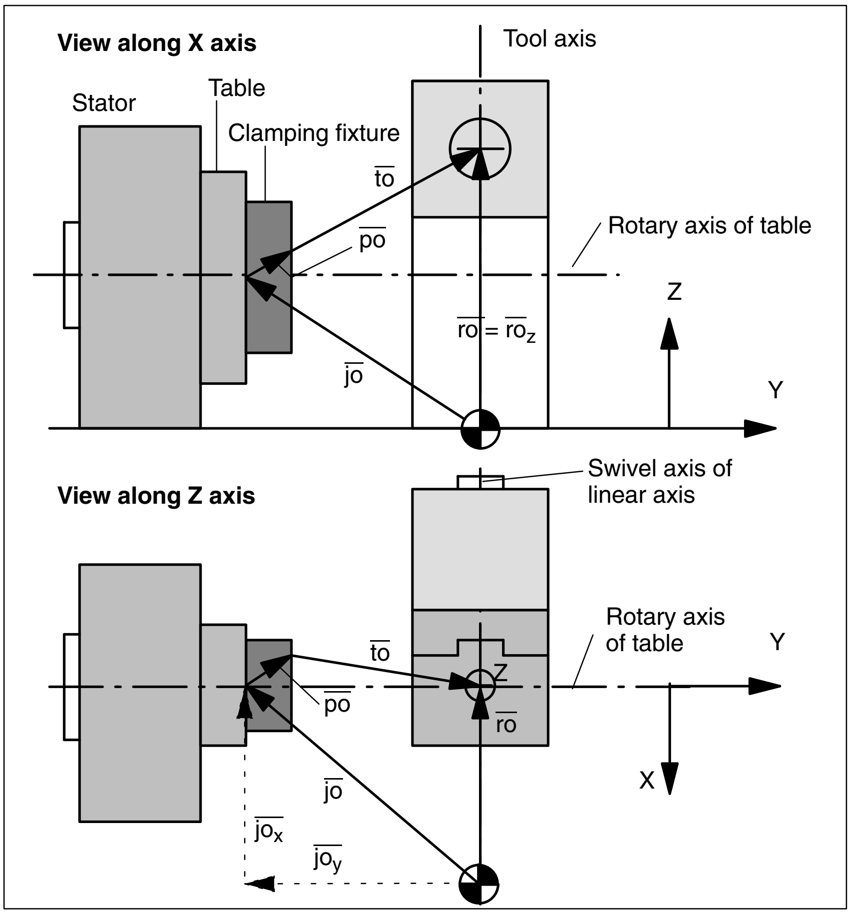



Fig. 2-12 Example of vector designations for MD settings for Fig. 2-11

**Procedure for setting MD**

Proceed as follows:

– Determine the x and y components of the vectors indicated, as shown in the lower section of Fig. 2-12 for vector $\overline{jo}$.

– Determine the z fractions of the corresponding vectors, as shown in the upper section for $\overline{ro}_z$.

– Set the 4 machine data correspondingly.
$MC_TRAFO5_PART_OFFSET_n
$MC_TRAFO5_TOOL_ROT_AX_OFFSET_n
$MC_TRAFO5_JOINT_OFFSET_n
$MC_TRAFO5_BASE_TOOL_n.

This procedure can be used for all kinematics specified under “Kinematics variants”. Observe the notes on Fig. 2-10.

**Zero components**

With certain geometries or machine zero positions, individual components or complete vectors can become zero.

**Type of machine**

The machine shown in Fig. 2-11 corresponds to version 1. Therefore, type of transformation 64 must be set in machine data $MC_TRAFO_TYPE_n (4<NBS>least-significant bits in MD).

**Activation**

The transformation for a swiveled linear axis is activated in the same way as the 5-axis transformations. Details are described in Section 2.5.

**Tool orientation**

With regard to tool orientation, the same applies as described in Subsection 2.1.4.

# 2.4 Universal milling head

## 2.4.1 Fundamentals of universal milling head

**Note**

The following description of the universal milling head transformation has been formulated on the assumption that the reader has already read and understood the general 5-axis transformation described in Section 2.1. Please note that where no specific statements relating to the universal milling head are made in the following section, the statements relating to the general 5-axis transformation apply.

**Applications**

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

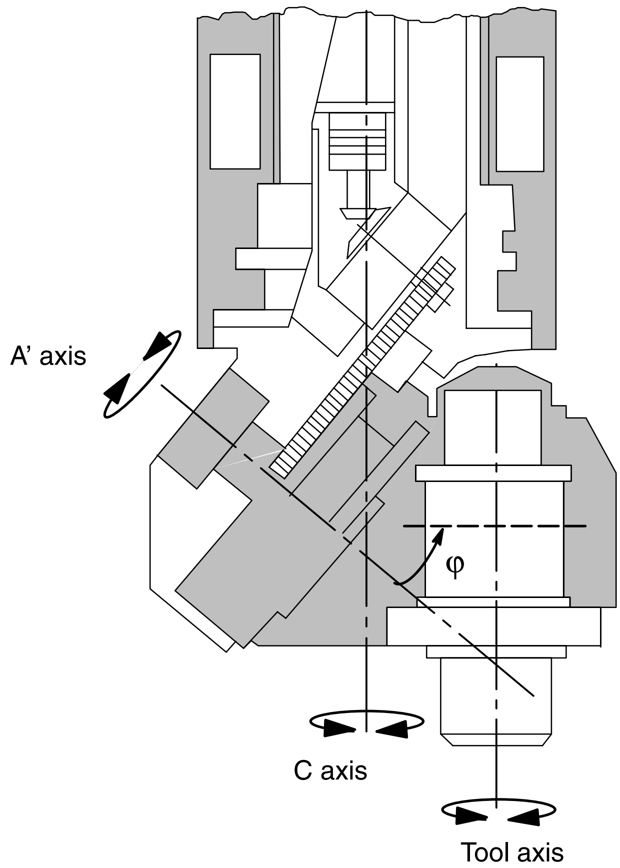



Fig. 2-13 Universal milling head

**Configuring the nutator angle φ**

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

| | |
|---|---|
| $MC_TRAFO5_NUTATOR_AX_ANGLE_1 | For the first orientation transformation |
| $MC_TRAFO5_NUTATOR_AX_ANGLE_2 | For the second orientation transformation |

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

**Tool orientation**

Tool orientation at zero position can be specified as follows:

– Parallel to the first rotary axis or

– perpendicular to it, and in the plane of the specified axis sequence

**Types of kinematics**

The axis sequence of the rotary axes and the orientation direction of the tool at zero position are set for the different types of kinematics by means of machine data $MC_TRAFO_TYPE_1 ... $MC_TRAFO_TYPE_8.

**Naming scheme for axes**

As is the case for the other 5-axis transformations, the following applies:
The rotary axis ...

| | |
|---|---|
| A is parallel to X | A' is below the angle φ to the X axis |
| B is parallel to Y | B' is below the angle φ to the Y axis |
| C is parallel to Z | C' is below the angle φ to the Z axis |

**Angle definition**

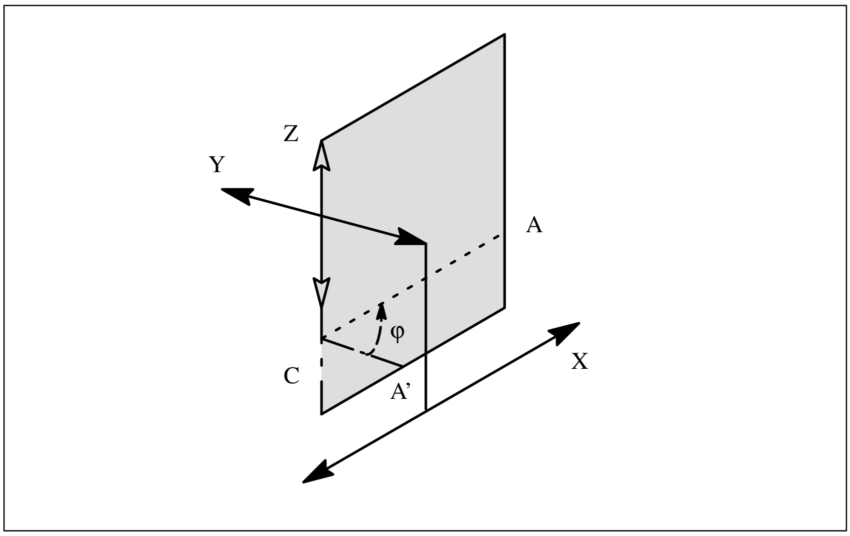



Fig. 2-14 Position of axis A'

Axis A' is positioned in the plane spanned by the rectangular axes of the designated axis sequence. If, for example, the axis sequence is CA', then axis A' is positioned in plane Z-X. The angle φ is then the angle between axis A' and the X axis.

## 2.4.2 Parameterization

**Setting the type of transformation**

The following table gives the data required in order to set machine data $MC_TRAFO_TYP_n appropriately for any given machine kinematics (general concept).

Table 2-3 MD $MC_TRAFO_TYPE_n

| Bit | Decimal | Description |
|---|---|---|
| 8 | 128 | Bit to identify transformation type:<br>1: Transformation for universal milling head |
| 7<br>6 | 0<br>32<br>64 | 00: Moving tool<br>01: Moving workpiece<br>10: Moving tool and workpiece |
| 5<br>4 | <br>0<br>8<br>16 | Orientation of tool in position zero<br>00: X direction<br>01: Y direction<br>10: Z direction |
| 3<br>2<br>1 | <br>0<br>1<br>2<br>3<br>4<br>5 | Axis sequence<br>000: AB'<br>001: AC'<br>010: BA'<br>011: BC'<br>100: CA'<br>101: CB' |

Among the full range of options specified in the general concept above, the settings highlighted in gray in the following table are implemented in software version 3.1, the others in software version 3.2 and higher.

Table 2-4 Implemented combinations; the table below gives the values for $MC_TRAFO_TYPE_n for the configurable axis sequences and for the orientation direction of the tool in position zero, showing separate data for moving tool, moving workpiece and moving tool and workpiece. The transformation does not support any table elements which do not contain a preset value.

| | Direction of orientation of tool in position zero | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Tool | | | Workpiece | | | Tool / Workpiece | | |
| Axis sequence | X | Y | Z | X | Y | Z | X | Y | Z |
| AB' | 128 | 136 | | | | | | | |
| AC' | 129 | | 145 | | | | | | |
| BA' | 130 | 138 | | | | | | | |
| BC' | | 139 | 147 | | | | | | |
| CA' | 132 | | 148 | | | | | | |
| CB' | | 141 | 149 | | | | | | |

**Example of transformation type**

$MC_TRAFO_TYPE = 148 means for example:

The 1st rotary axis is parallel to the Z axis, the 2nd rotary axis is an inclined X axis and the tool orientation in position zero is pointing towards Z. Only the tool is moved by the two rotary axes

| | |
|---|---|
| Bit 8 = 1 | Universal milling head |
| Bits 6 and 7 = 00 | Moving tool |
| Bits 5 and 4 = 10 | Orientation in position zero Z direction |
| Bit 3–1 = 100 | Axis sequence CA' |

**Active machining plane**

Since the tool orientation in position zero can be set in directions other than just the Z direction, the user must ensure that he sets the active machining level such that the tool length compensation takes effect in the tool orientation direction. The active machining plane should always be the plane according to which the tool orientation is set in position zero.

**Other settings**

The geometry information used by the universal milling head transformation for calculation of the axis values is set analogously to that of the other 5-axis transformations.

### 2.4.3 Traversal of universal milling head in JOG mode

**JOG**

The linear axes can be traversed normally in JOG mode. It is, however, difficult to set the orientation correctly by traversing these axes.

**Activation of universal milling head**

The transformation for universal milling head in the program is activated as described in the following Section 2.5.

## 2.5 Call and application of the 3-axis to 5-axis transformation

**Switching on**

The 3-axis to 5-axis transformations (including the transformations for swiveled linear axis and universal milling head) are switched on with the command TRAORI(n), with n representing the number of the transformation (n=1 or 2).

Once execution of the command TRAORI(n) is completed and the transformation thus activated, the IS "Transformation active" (DB21–30, DBX33.6) switches to "1".

If the machine data for a called transformation group have not been defined, the NC program stops and the control outputs the alarm 14100 "Orientation transformation not available".

**Switching off**

The currently active 3-axis to 5-axis transformation is switched off with TRAFOOF or TRAFOOF(). The IS "Transformation active" (DB21–30, DBX33.6) switches to "0".

**Switching over**

You can switch from one active transformation to another transformation configured in the same channel. To do this you must again input the command TRAORI(n) with a new value for n.

**RESET/ end of program**

The behavior of the control with regard to 3-axis/5-axis transformations after power up, end of program or RESET depends on the
MD 20110 RESET_MODE_MASK
Bit 7: Reset behavior of "Active kinematic transformation"

Bit 7=0: Initial setting for active transformation after end of parts program or RESET according to MD 20140: TRAFO_RESET_VALUE is defined with the following meaning:

| | |
|---|---|
| 0: | No transformation is active after RESET |
| 1 to 8: | The transformation preset in MD |
| 24100: | TRAFO_TYPE_1 to MD 24460: TRAFO_TYPE_8 is active. |

Bit 7=1: The current setting for the active transformation remains unchanged after a RESET or end of parts program.

**Option**

The "5-axis transformation" and its special types described in this Description of Functions are available only in the form of an option. If this option is not implemented in the control and a transformation is called with the command TRAORI, the error message 14780 "Block uses a function that has not been enabled" appears and the NC program stops.

If 3-axis to 5-axis transformation is not specified in machine data
MD 24100: TRAFO_TYPE_1 ... MD 24460: TRAFO_TYPE_8, programming the TRAORI (1 or 2) command triggers alarm 14100 "Channel x block y orientation transformation not available".

If the MD: $MC_TRAFO_TYPE_n is set without the 5-axis transformation option being enabled there is no alarm.

# 2.6 Generic 5-axis transformation (SW 5.2 and higher)

## 2.6.1 Functionality

**Scope of functions**

The scope of functions provided by the generic 5-axis transformation package covers the 5-axis transformation implemented in SW 5.1 and lower (see Section 2.1) for perpendicular rotary axes and the transformations for the universal milling head (one rotary axis parallel to a linear axis, the second rotary axis at any angle to it, see Section 2.4).

**Field of application**

In certain cases, it is not possible to compensate the conventional transformation accuracy, e.g. when

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

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

**A programming example**
for the generic 5-axis transformation is described in Section 6.6.

**Activation**

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

The transformation can be deselected implicitly by selecting another transformation or explicitly with TRAFOOF.

## 2.6.2 Description of machine kinematics

**Machine types**

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

1. Machine type: Rotatable tool
   Both rotary axes change the orientation of the tool.
   The orientation of the workpiece is fixed.
2. Machine type: Rotatable tool
   Both rotary axes change the orientation of the workpiece.
   The orientation of the tool is fixed.
3. Machine type: Rotatable tool and rotatable workpiece
   One rotary axis changes the tool orientation and the other the workpiece orientation.

**Configurations**

The machine configurations are defined as in earlier versions (see Subsection 2.1.3) in machine data $MC_TRAFO_TYPE_1, ..., _8. Additional types have been introduced for the generic 5-axis transformation:

Table 2-5 Overview of machine types for generic 5-axis transformation

| Machine type | 1 | 2 | 3 |
|---|---|---|---|
| Swivel/rotatable | Tool | Workpiece | Tool/workpiece |
| | 24 | 40 | 56 |

**Rotary axis direction**

The direction of the rotary axes is defined by machine data $MC_TRAFO5_AXIS1_n (1st rotary axis) and $MC_TRAFO5_AXIS2_n (2nd rotary axis). In this case, n is 1 or 2 for the first or second 5-axis transformation in the system. The machine data specified above are arrays with three values which can describe that axis direction (analogous to description of rotary axes for orientatable toolholder. The absolute value of the vectors is insignificant; only the defined direction is relevant.

Example:

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

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

## 2.6.3 Generic orientation transformation variants (SW 6.1 and higher)

**Extension**

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

**Variant 1**

### 4-axis transformations

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

$MC_TRAFO_AXES_IN_1[4] = 0 or
$MC_TRAFO_AXES_IN_2[4] = 0

**Variant 2**

### 3-axis transformations

In a 3-axis transformation one of the geometry axes is missing in addition. This is achieved by entering a zero in the field:

$MC_TRAFO_GEOAX_ASSIGN_TAB_1[n] or
$MC_TRAFO_GEOAX_ASSIGN_TAB_2[n]

**Transformation types**

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

- 3-or 4-axis transformation with rotatable tool
  $MC_TRAFO_TYPE_n = 24
- 3-or 4-axis transformation with rotatable workpiece
  $MC_TRAFO_TYPE_n = 40

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

**Effects on orientations**

The generic 3-or 4-axis transformation has the following effect on the various orientations:

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

Priority:

| | |
|---|---|
| High | Programmed orientation, |
| Medium | Tool orientation and |
| Low | Basic orientation |

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

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

---

**Note**

Please refer to the following documentation for further information on programmable tool orientation and basic tool orientation:

**References:** /FB/, W1, "Tool Compensation" Orientable Toolholders
/PG/, "Programming Guide Fundamentals", Chapter 8

---

**Differences**

Please note the following differences as distinct from the existing 3-and 4-axis transformations described in Section 2.2:

| Description | SW 6.1 and higher |
|---|---|
| Position of the rotary axis | 1. Any position possible.<br>2. Does not have to be parallel to a linear axis. |
| Direction of the rotary axis | 3. Must be defined with $MC_TRAFO5_AXIS1_1[n] or $MC_TRAFO5_AXIS1_2[n]. |
| Basic tool orientation | 4. Must be defined with $MC_TRAFO5_BASE_ORIENT_1[n] or $MC_TRAFO5_BASE_ORIENT_2[n]. |
| Selection of a generic 3/4-axis transformation | 5. An optional tool orientation can be transferred as in the case of a generic 5-axis transformation. |

### 2.6.4 Online tool length offset (SW 6.4 and higher)

**Functionality**

The effective tool lengths can be changed in real time so that these changes in length are also considered for orientation changes in the tool. System variable $AA_TOFF[ ] applies the tool length offsets in 3-D according to the three tool directions.

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

The geometry identifiers are used as index. The number of active offset directions must be the same as the number of active geometry axes. All offsets can be active at the same time.

**Application**

The online tool length offset function can be used for:

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

---

**Note**

The online tool length compensation is optional and must be enabled beforehand. This function is only practical in conjunction with an active orientation transformation or an active orientable toolholder.
**References** /FB/, W1, "Tool Compensation" Orientable Toolholders

---

**Block preparation**

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

The tool offset is always known at the time of run-in when the tool length offsets are not changed after program start or if more blocks have been processed after changing the tool length offsets than the IPO buffer can accommodate between run-in and main run. This ensures that the axis velocities are quickly considered correctly.

The dimension for the difference between the currently active offset in the interpolator and the offset that was active at the time of block preparation can be polled in the system variable $AA_TOFF_PREP_DIFF[ ].

---

**Note**

The change of the effective tool length through the online tool length offset produces changes in the compensatory movements of the transformation of the axes involved. The resulting velocities can be higher or lower depending on the machine kinematics and the current axis position.

---

**MD 21190: TOFF_MODE**

Machine data MD 21190: TOFF_MODE can be used to set whether the contents of the synhronization variable $AA_TOFF[ ] is to be approached as an absolute value or whether an integrating behavior is to take place. The integrating behavior of $AA_TOFF[ ] allows a 3D distance control. The integrated value is available via the system variable $AA_TOFF_VAL[ ].

The following machine data and setting data are available for configuring the online tool length offset:

| Machine data / setting data | Meaning for online tool length offset |
|---|---|
| MD 21190: TOFF_MODE | The contents of $AA_TOFF[ ] are traversed as an absolute value or integrated |
| MD 21194: TOFF_VELO | Velocity of online tool length offset |
| MD 21194: TOFF_ACCEL | Acceleration of online tool length offset |
| SD 42970: TOFF_LIMIT | Upper limit of tool length offset value |

When planning the velocity, 20% is reserved for the online tool length offset, which can be changed via machine data MD 20610: ADD_MOVE_ACCEL_RESERVE.

**Activation**

The TOFFON statement can be used to activate the online tool length compensation from the parts program for at least one tool direction if the option is available. When activated, an offset value can be specified for the corresponding offset direction and applied immediately.
Example: TOFFON(Z, 25).

Repeated programming of the statement TOFFON( ) with an offset causes the new offset to be applied. The offset value is added to variables $AA_TOFF[ ] as an absolute value.

---

**Note**

For more information and programming examples, please refer to:
**References:** /PGA/, Chapter 7 “Transformations”

---

As long as the online tool length offset is active, the VDI signal of the interface NCK →PLC IS “TOFF active” (DB21, ... DBX318.2) is set to 1.

During an offset motion, the VDI →signal IS “TOFF motion active” (DB21, ... DBX318.3) is set to 1.

**Resetting**

The offset values can be reset with the command TOFFOF( ). This statement triggers a stop preprocessing.

The tool length offsets set up are cleared and incorporated in the basic coordinate system. The run-in synchronizes with the current position in main run. Since no axes can be traversed here, the values of $AA_IM[ ] do not change. Only the values of the variables $AA_IW[ ] and $AA_IB[ ] are changed. These variables now contain the deselected share of the tool length offset.

After deselection of the “Online tool length offset” for a tool direction, the value of system variable $AA_TOFF[ ] or $AA_TOFF_VAL[ ] is zero for this tool direction.
The IS “TOFF active” (DB21, ... DBX318.2) is set to 0.

**Alarm 21670**

An existing tool length offset must be deleted via TOFFOF( ) so that Alarm 21670 “Channel %1 block %2, impermissible change of tool direction active due to $AA_TOFF active” is suppressed:

- When the transformation is deactivated with TRAFOOF
- If you switchover from CP to PTP travel
- If a tool length offset exists in the direction of the geometry axis during geometry replacement
- If a tool length offset is present during change of plane
- When changing from axis/specific manual traversing in JOG mode to PTP as long as the tool length offset is active. There is no switchover to PTP.

**Mode change**

The tool length compensation remains active even if the mode is changed and can be executed in any mode.

If a tool length offset is interpolated on account of $AA_TOFF[ ] during mode change, the mode change cannot take place until the interpolation of the tool length offset has been completed. Alarm 16907 “Channel %1 Action %2 <ALNX> possible only in Stop state” is issused.

**Behavior with REF and block search**

The tool length offset is not considered during reference point approach REF in JOG mode.

The statements TOFFON( ) and TOFFOF( ) are not collected and output in an action block during block search.

**System variable**

In the case of online tool length offset, the following system variables are available to the user:

| System variables | Meaning for online tool length offset |
|---|---|
| $AA_TOFF[ ] | Position offset in the tool coordinate system |
| $AA_TOFF_VAL[ ] | Integrated position offset in the WCS |
| $AA_TOFF_LIMIT[ ] | Query whether the tool length offset is close to the limit |
| $AA_TOFF_PREP_DIFF[ ] | Size of the difference between the currently active value of $AA_TOFF[ ] and the value prepared as the current motion block. |

**References:** /PGA/, Programming Guide Production Planning, List of system variables”

**Boundary conditions**

The online tool length offset function is an option and is available during “generic 5-axis transformation” per default and for “Orientable tool carriers”.

If the tool is not vertical to the workpiece surface during machining, or the contour contains curvatures whose radius is smaller than the offset dimension, deviations compared to the actual offset surface are produced. It s not possible to produce exact offset surfaces with one tool length offset alone.

### 2.6.5 Orientation

**Differences compared to SW 5.1**

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

By means of the generic 5-axis transformation it is possible to enable any basic tool orientation, i.e. the space orientation of the tool is arbitrary with axes in basic positions.

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

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

**Definition**

A basic orientation can be defined by one of the following three methods:

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

**Via the transformation call**

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

The first parameter specifies the transformation number. If the first transformation must be activated, the number can be omitted; when an orientation is specified in the call, however, the uniqueness of the parameters must be assured by inserting a blank space for the transformation number, e.g. TRAORI(, 0., 1., 5.).

The absolute value of the vector is insignificant; only the direction is relevant. Unprogrammed vector elements can be set to zero.

---

**Note**

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

---

Please note that if all three vector components are zero (because they have been set explicitly so or not specified at all), the basic orientation is not defined by data in the TRAORI(...) call, but by one of the methods described below.

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

**Via the orientation of the active tool**

Re 2.:
The basic orientation is determined by the tool

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

The orientation of a tool is dependent on the selected plane. The plane is parallel to Z with G17, parallel to Y with G18 and parallel to X with G19. It can be modified in any way by orientatable toolholders (see SINUMERIK 840D/810D/FM-NC Description of Functions, Basic Machine [Part1] Description of Functions).

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

If the tool is deselected, thereby canceling the definition of a tool orientation, the basic orientation programmed in machine data becomes operative.

**Via a machine data**

With reference to 3.:
If the basic orientation is not defined by either of the two variants described above, it is specified with reference to machine data $MC_TRAFO5_BASE_ORIENT_n. This machine data must not be set to a zero vector or else an alarm will be generated during control runup when a transformation is active.

If a basic orientation is programmed in machine data $MC_TRAFO5_BASE_ORIENT_n when a transformation is active and a tool is subsequently activated, the basic orientation is re-defined by the tool.

---

**Note**

The range of settable orientations depends on the directions of the rotary axes involved and the basic orientation. The rotary axes must be mutually perpendicular if all possible orientations are to be used. If this condition is not fulfilled, so-called “dead” ranges occur.

---

Examples:

1. Extreme example: A machine with rotatable tool has a C axis as its first rotary axis and an A axis as its second. If the basic orientation is defined in parallel to the A axis, the orientation can only be changed in the X-Y plane (when the C axis is rotating), i.e. an orientation with a Z component unequal to zero is not possible in this instance. The orientation does not change when the A axis rotates.
2. Realistic example: A machine with nutator kinematics (universal head) with an axis inclined at less than 45° in a basic orientation parallel to the Z axis can only assume orientations within a semi-circle: The top semi-circle with basic orientation towards +Z and the bottom with basic orientation towards –Z.

### 2.6.6 Orientation movements with axis limits (SW 6.1 and higher)

**Calculating the rotary axis position**

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

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

The control chooses the solution which represents the shortest distance from the current starting point allowing for the programmed interpolation type.

**Determining the permissible axis limits**

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

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

- A generic 5-axis transformation of type 24, 40 or 56 must be active.
- The axis must be referenced.
- The axis must not be a modulo rotary axis.
- Machine data MD 21180: ROT_AX_SWL_CHECK_MODE must be not equal to zero.

Machine data MD 21180: ROT_AX_SWL_CHECK_MODE specifies when the modification of the rotary axis positions is allowed:

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

**Change-over to axis interpolation**

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

**Example**

An example is shown in Section 6.6 for modifying the rotary axis motion of a 5-axis machine with a rotatable tool.

## 2.6.7 Singularities of orientation

**Description of problem**

As described in Subsection 2.1.5 for SW 5.2 and lower, singularities (poles) are constellations in which the tool is orientated in parallel to the first rotary axis. If the orientation is changed when the tool is in or close to a singularity (as is the case with large-circle interpolation ORIWKS), the rotary axis positions must change by large amounts to achieve small changes in orientation. In extreme cases, a jump in the rotary axis position would be needed.

This type of situation is handled in the following way:
Only one machine data is relevant
MD 24540: TRAFO5_POLE_LIMIT_1 or
MD 24640: TRAFO5_POLE_LIMIT_2,

describing a circle around the pole (as before).

For further information on how singularities are handled, please refer to:
**References:** /PGA/, “Programming Guide Advanced” Subsection 7.1.3

**Example for machine type 1**

Rotatable tool
Both rotary axes modify the tool orientation.
The orientation of the workpiece is fixed.

2-axis inclinable head with
rotary axis RA 1 (4th axis of transformation) and
rotary axis RA 2 (5th axis of transformation).

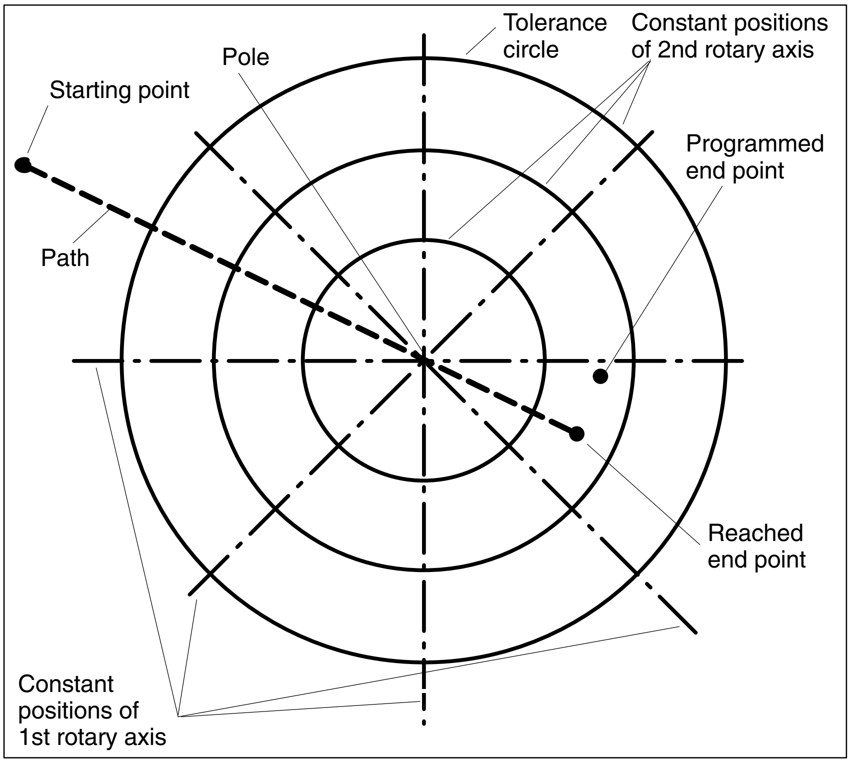



Fig. 2-15 Generic 5-axis transformation; end point of orientation inside tolerance circle

**End point within the circle**

If the end point is within the circle, the first axis comes to a standstill and the second axis moves until the difference between the target and actual orientation is minimal. However, since the first rotary axis does not move, the orientation will generally deviate from the programmed value (see Fig. 2-15). However, the programmed orientation can at least be reached exactly if the first rotary axis happens to be positioned correctly.

**Note**

In Fig. 2-15 the resulting path is a straight line because the position of the first rotary axis is constant on that path. This representation is always correct, irrespective of the angle between the two rotary axes. The orientation vector only moves in a plane, however, if the two rotary axes and the basic orientation are all mutually perpendicular. In all other cases, the orientation vector describes the lateral area of a cone.

**End point outside the circle**

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

## 2.7 Compression of orientation (SW 6.3 and higher)

### Introduction

Up to SW 6.2, the use of the compressors was subject to restrictions with reference to the orientation. Some of these restrictions have been overcome by the options described below.

### Extended function

Compressors COMPON, COMPCURV and COMPCAD have been extended such that NC programs containing orientations programmed by means of direction vectors can also be compressed to a definable tolerance.

### Requirements

The “compressor for orientation” function is only implemented if the orientation transformation option is available.

### Previous function

The compressor is only active for linear blocks (G1). The compression procedure is interrupted by any other NC instruction, such as an auxiliary function output, but not by parameter calculations. The blocks to be compressed can only contain the following elements:

– Block number
– G1
– Axis addresses
– Feedrate
– Comment

N... G1 X... Y... Z... A... B... F...        ; comment

### Function SW 6.3 and higher

The position values do not have to be programmed directly, but can be specified via parameter assignments. The general format is:

N... G1 X=<...> Y=<...> Z=<...> A=<...> B=<...> F=<...> ; comment

<...> can contain any parameter expression, e.g. X=R1*(R2+R3).

### Programming options

The tool orientation can be programmed in the following (kinematic-independent) ways for 5-axis machines if an orientation transformation (TRAORI) is active:

1. Programming of the direction **vector**:
   A3=<...> B3=<...> C3=<...>
2. Programming of the Euler **angle** or RPY **angle**:
   A2=<...> B2=<...> C2=<...>

**Large circle interpolation**

The orientation motion is only compressed if large circle interpolation is active, i.e. change in tool orientation takes place in the plane defined by the start and end orientation.
Large circle interpolation is performed under the following conditions:

1. For MD 21104: ORI_IPO_WITH_G_CODE = 0 if ORIWKS is active and the orientation is programmed as a vector (with A3, B3, C3 or A2, B2, C2).
2. For MD 21104: ORI_IPO_WITH_G_CODE = 1 if ORIVECT or ORIPLANE is active.
   The tool orientation can be programmed either as a direction vector or with rotary axis positions. If either of the G codes ORICONxx or ORICURVE is active or polynomials are programmed for the orientation angle (PO[PHI] and PO[PSI]) large circle interpolation does not take place.

**Rotation of the tool**

On **6-axis** machines, the rotation of the tool can be programmed in addition to the tool orientation.
The angle of rotation is programmed with the THETA identifier. (THETA=<...>).
NC blocks which also contain a rotation can only be compressed if the angle of rotation changes in **linear** fashion. In other words, PO[THT]=(...) may not be used to program a polynomial for the angle of rotation.

The general format of an NC block which can be compressed is:
N... X=<...> Y=<...> Z=<...> A3=<...> B3=<...> C3=<...> THETA=<...> F=<...>

or
N... X=<...>Y=<...> Z=<...> A2=<...> B2=<...> C2=<...> THETA=<...> F=<...>

However, if the tool orientation is specified by rotary axis positions, e.g. in the following format:

N... X=<...> Y=<...> Z=<...> A=<...> B=<...> THETA=<...> F=<...>

the compression is performed in two different ways, depending on whether or not large circle interpolation is performed. If no rotary axis interpolation takes place, the compressed orientation change is represented in the usual way by axial polynomials for the rotary axes.

**Accuracy**

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

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

**Axis accuracy**

The compressor generates a spline curve for each axis. The maximum deviation of the spline curve from the programmed end points of each axis is defined by the value set in MD 33100: COMPRESS_POS_TOL.

**Contour accuracy**

The maximum deviations are not defined separately for each axis. Instead, the maximum geometric deviation of the contour (geometry axes) and of the tool orientation are checked.
This is performed using the following setting data:

1. SD 42475: COMPRESS_CONTUR_TOL: maximum tolerance for the contour.
2. SD 42476: COMPRESS_ORI_TOL: maximum angular displacement for the tool orientation.
3. SD 42477: COMPRESS_ORI_ROT_TOL: maximum angular displacement for the angle of rotation of the tool (only available for 6-axis machines).

**Using the setting data**

MD 20482: COMPRESSOR_MODE can be used to set a particular type of tolerance specification:

Value 0: axial tolerances with MD 33100: COMPRESS_POS_TOL for all axes (geometry axes and orientation axes).

Value 1: contour tolerance is specified by SD 42475: COMPRESS_CONTUR_TOL, tolerance for the orientation is specified by axial tolerances with MD 33100: COMPRESS_POS_TOL.

Value 2: the maximum angular displacement for the tool orientation is specified by SD 42476: COMPRESS_ORI_TOL, tolerance for the orientation is specified by axial tolerances with MD 33100: COMPRESS_POS_TOL.

Value 3: contour tolerance is specified by SD 42475: COMPRESS_CONTUR_TOL and the maximum angular displacement for the tool orientation is specified by SD 42476: COMPRESS_ORI_TOL.

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

**Activation**

The orientation compressor is activated by one of the G codes COMPON, COMPCURV and COMPCAD.

**Example**

**Programming example**
For the compression of a circle approximated by a polygon definition, please see Section 6.7.

## 2.8 Orientation axes (SW 5.3 and higher)

**Direction**

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

– ORIMKS: Reference system = basic coordinate system

– ORIWKS: Reference system = workpiece coordinate system

**Order of rotations**

The order of rotation of the orientation axes is defined by MD<NBS>21120: ORIAX_TURN_TAB_1[0..2].

1. First rotation around the rotated axis of the reference system specified in MD<NBS>21120: ORIAX_TURN_TAB_1[0]
2. Second rotation around the rotated axis of the reference system specified in MD 21120:ORIAX_TURN_TAB_1[1]
3. Third rotation around the rotated axis of the reference system specified in MD 21120: ORIAX_TURN_TAB_1[2]

**Direction of the tool vector**

The direction of the tool vector in the basic machine setting is defined by
MD 24580: TRAFO5_TOOL_VECTOR_1 or
MD 24680: TRAFO5_TOOL_VECTOR_2.

**Assignment to channel axes**

Machine data MD 24585: TRAFO5_ORIAX_ASSIGN_TAB_1[0..2] are used to assign up to a total of 3 virtual orientation axes to the channel, which are set as input variables in machine data $MC_TRAFO_AXES_IN_n[4..6].

As regards assigning channel axes to orientation axes, please note the following:

- $MC_TRAFO5_ORIAX_ASSIGN_TAB_n[0] = $MC_TRAFO_AXES_IN_n [4]
- $MC_TRAFO5_ORIAX_ASSIGN_TAB_n[1] = $MC_TRAFO_AXES_IN_n [5]
- $MC_TRAFO5_ORIAX_ASSIGN_TAB_n[2] = $MC_TRAFO_AXES_IN_n [6]

Orientation transformation 1:
MD 24585: TRAFO5_ORIAX_ASSIGN_TAB_1[n] n = channel axis [0..2]

Orientation transformation 2:
MD 24685: TRAFO5_ORIAX_ASSIGN_TAB_2[n] n = channel axis [0..2]

Transformation [1..4]
MD 24110: TRAFO5_AXES_IN_1[n] n = Axis index [0..7]
to
MD 24410: TRAFO5_AXES_IN_4[n]
Transformation [5..8]
MD 24432: TRAFO5_AXES_IN_5[n] n = axis index [0..7]
to
MD 24462: TRAFO5_AXES_IN_8[n]

**Example**

For orientation axes, please see Section 6.4 “Example for orientation axes”.

### 2.8.1 JOG mode

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

- The orientation axis must be defined as such, that is, a value must be set in MD $MC_TRAFO5_ORIAX_ASSIGN_TAB.
- A transformation must be active (TRAORI command).

**Axis traversal using traverse keys**

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

If more than one orientation axis is moved, alarm 20062 “Channel 1 axis 2 already active” is output.

**Axis traversal using handwheels**

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

**Feedrate in JOG**

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

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

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

- MD 21150: JOG_VELO_RAPID_ORI[n]
- MD 21155: JOG_VELO_ORI[n]
- MD 21160: JOG_VELO_RAPID_GEO[n]
- MD 21165: JOG_VELO_GEO[n]

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

**Acceleration**

MD 21170: The acceleration for the orientation axes can be set using ACCEL_ORI[n].

## 2.8.2 Programming

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

**Programming of orientation**

Orientation axes are programmed by means of axis identifiers **A2, B2** and **C2**.

Euler and RPY values are distinguished on the basis of G group 50:

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

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

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

MD 21102: ORI_DEF_WITH_G_CODE defines whether
MD 21100: ORIENTATION_IS_EULER is active (default) or G group 50.

The following four variants are available for programming the orientation:

1. A, B, C: Input of machine axis position
2. A2, B2, C2: Angle programming of virtual axes
3. A3, B3, C3: Input of vector components
4. LEAD, TILT: Specification of the lead and side angles with reference to path and surface

---

**Note**

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

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

---

**Example**

Please refer to Section 6.4 “Example of orientation axes” for an example of orientation axes for a kinematic with 6 or 5 transformed axes.

**Interpolation type**

MD 21104: ORI_IPO_WITH_G_CODE defines which type of interpolation is used:

- ORIMKS or ORIWKS (for description, see Subsection 2.1.4)
- G code group 51 with the commands ORIAXES or ORIVECT
  - ORIAXES:
    Linear interpolation of machine axes or orientation axes.
  - ORIVECT:
    The orientation is controlled by the orientation vector being swiveled in the plane spanned by the start and end vectors (large-circle interpolation). In cases of 6 transformed axes, there is a rotation about the orientation vector in addition to this swiveling motion.
    When ORIVECT is programmed, the orientation axes always traverse along the shortest possible path.

**Value range**

Value range for orientation axes:
−180 degrees < A2 < 180 degrees
−90 degrees < B2 < 90 degrees
−180 degrees < C2 < 180 degrees

All possible angles of rotation can be represented within this value range. Values outside the range are normalized by the control system to within the range specified above.

**Feedrate**

**Feedrate when programming ORIAXES:**

The feedrate for an orientation axis can be limited via the FL[ ] instruction (feed limit).

**Feedrate when programming ORIVECT:**

The feedrate must be programmed with commands FORI1 and FORI2:

- FORI1 Feedrate for swiveling the orientation vector on the large arc
- FORI2 Feedrate for overlaid rotation about the swiveled orientation vector

The lowest of the two feedrates is applied for overlaid turning and swiveling motions. With orientation motions, the feedrate corresponds to an angular velocity [degrees/min]. If geometry axes and orientation axes are traversing along the same path, the traversing motion is determined by the smallest of the two feedrates.

The programmed feedrate override also applies to FORI1 and FORI2.

**References:** /PA/, Programming Guide, Fundamentals

## 2.8.3 Restrictions for kinematics and interpolation

**Fewer than 6 axes**

Not all degrees of freedom are available for the orientation. The following special rules therefore apply:

**5-axis kinematics**

This has only two degrees of freedom for the orientation. The assignment of the orientation axes and the tool vector direction must be selected such that there is no rotation about the tool vector itself. As a result, only two orientation angles are required to describe the orientation. If the axis is traversed by ORIVECT, the tool vector performs a pure swiveling motion.

**3-and 4-axis kinematics**

Only one degree of freedom is available for the orientation in the case of 3-axis and 4-axis kinematics. The respective transformation determines the relevant orientation angle.

It only makes sense to traverse the orientation axis with ORIAXES. Linear interpolation for the orientation axis is direct.

**Interpolation across several blocks**

Machine tools with the kinematics of an orientable toolholder are capable of orienting the tool in space. The orientation of the tool is almost always programmed in each block. For example, it is possible to

specify the tool orientation directly with reference to the rotary axis positions.

If orientations of a tool are interpolated over several successive blocks, undesirable abrupt changes in the orientation vector may be encountered at the block transitions. This cause irregular velocity and acceleration changes in the rotary axes at the block transitions.

Large circle interpolation can be used to generate a movement of the orientation axes with continuous velocity and acceleration across several blocks. The orientation axes behave like

normal linear axes if only G1 blocks are interpolated.

In the case of linear axes, a movement with continuous acceleration is achieved by using polynomials for the axis interpolation.

**Tool orientation based on orientation vectors**

A much better method is to use orientation vectors in order to program the tool orientation in space.

The interpolation of orientation vectors can be programmed with polynomials up to a maximum of 5th degree. Please consider the features of polynomial interpolation of orientation vectors described in Subsection 2.9.1.

---

**Note**

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

**References:** /PGA/, Programming Guide Advanced

---

# 2.9 Orientation vectors

## 2.9.1 Polynomial interpolation of orientation vectors (SW 5.3 and higher)

**Programming of polynomials**

**Polynomial programming for axis motion**

In the case of a change in orientation using rotary axis interpolation, linear interpolation normally takes place in the rotary axes. However, it is also possible to program the polynomials as usual for the rotary axes. This enables you to produce generally more homogeneous axis motion.

---

**Note**

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

**References:** /PGA/, Programming Guide Advanced

---

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

- **not active**, the programmed axis end points are traversed in a line.
- **active**, the programmed polynomials are interpolated as polynomials.

**MD 10674**

Machine data MD 10674: PO_WITHOUT_POLY = **FALSE** can be used to set whether the programming of:

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

If MD 10674: PO_WITHOUT_POLY = **TRUE**, programming of:

- PO[...] = (...) is always possible independent of whether POLY is active or not.

MD 10674: PO_WITHOUT_POLY = FALSE is the default setting.

**POLYPATH**

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

| | |
|---|---|
| (“AXES”): | For all path axes and additional axes |
| (“VECT”): | For orientation axes |
| (“AXES”,“VECT”): | For path axes, additional axes and orientation axes |
| (without argument): | **Deactivates** polynomial interpolation for all axis groups. |

Polynomial interpolation is activated for all axis groups per default.

**Programming of orientation vectors**

An orientation vector can be programmed in each block. If polynomials are programmed for the orientation, the orientation vector is usually no longer located in the plane between the start and end vectors, but can be rotated out of this plane.

The orientation vectors can be programmed as follows:

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

**Selection of type of interpolation**

The type of interpolation for orientation axes is selected using the G codes of group 51

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

and is independent of the type of programming of the end vector. If ORIAXES is active, the interpolation of the rotary axis can also take place using polynomials like polynomial interpolation of axes with POLY.

On the other hand, if ORIVECT is active, a "normal" large circle interpolation is carried out through linear interpolation of the angle of the orientation vector in the plane that is clamped from the start and end vector.

**Polynomials for 2 angles**

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

The two PHI and PSI angles are specified in degrees.

POLY — Activate polynomial interpolation for all axis groups.

POLYPATH ( ) — Activate polynomial interpolation for all axis groups. Possible groups are "AXES" and "VECT".

The coefficients $a_n$ and $b_n$ are specified in degrees.

PO[PHI]=($a_2$, $a_3$, $a_4$, $a_5$)

The PHI angle is interpolated as
$PHI(u) = a_0 + a_1*u + a_2*u^2 + a_3*u^3 + a_4*u^4 + a_5*u^5$.

PO[PSI]=($b_2$, $b_3$, $b_4$, $b_5$)

The PSI angle is interpolated as
$PHI(u) = b_0 + b_1*u + b_2*u^2 + b_3*u^3 + b_4*u^4 + b_5*u^5$.

PL — Length of the parameter interval where polynomials are defined. The interval always begins with 0.
**Theoretical value range** for PL: 0.0001 ... 99999.9999.
The PL value applies to the block which contains it.
If no PL is programmed, PL = 1 is active.

**Rotation of the orientation vector**

Orientation changes are possible with ORIVECT, independent of the type of end vector programming. The following situations apply:

**Example 1** The components of the end vector are programmed.

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

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

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

The angle PHI describes the rotation of the orientation vector in the plane between the start and end vectors (large circle interpolation, see Fig. 2-16). The interpolation of the orientation vector is exactly the same as in example 1.

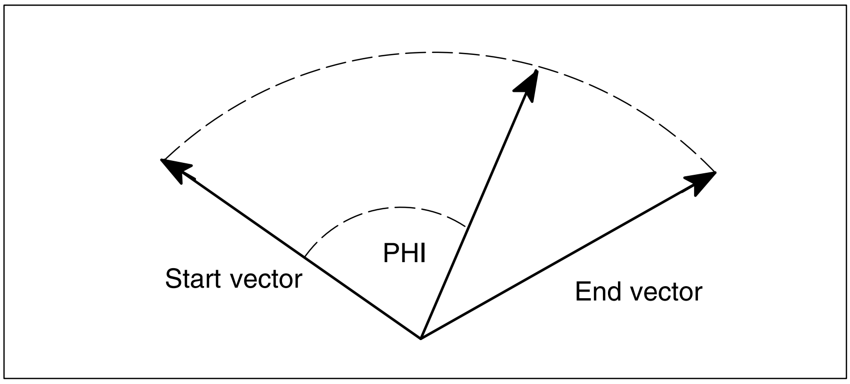



Fig. 2-16 Rotation of the orientation vector in the plane between start and end vector

**PHI and PSI angle**

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

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

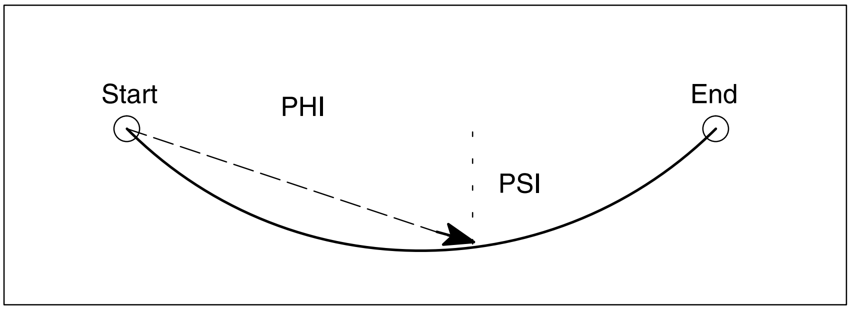



Fig. 2-17 Movement of the orientation vector in the top view

The angle PSI can be used to generate movements of the orientation vector perpendicular to the large circle interpolation plane (see Fig. 2-17).

**Maximum polynomials of 5th degree permitted**

5th degree polynomials are the maximum possible for programming the angles PHI and PSI. The constant and linear coefficient is defined by the start value and end value of the orientation vector in each case.

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

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

**Special features**

If no polynomial is programmed for the PSI,

- the interpolation of the orientation vector is always in the plane spanned by the start and end vectors.
- the PHI angle in this plane is interpolated according to the programmed polynomial for PHI.

This ensures

- that the orientation vector moves through a “normal” large circle interpolation in the plane between the start and end vector and the movement is more or less irregular independent of the programmed polynomial.

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

Further explanations of tool orientation using orientation vectors for machine tools are given in Subsection 2.8.3.

**Supplementary conditions**

The polynomial interpolation of orientation vectors is only possible for control variants in which

- both an orientation transformation

and

- the polynomial interpolation belong to the functional scope.

### 2.9.2 Rotation of the orientation vector (SW 6.1 and higher)

**Functionality**

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

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

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

**Programming of the orientation direction**

The following options are available for programming the tool orientation:

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

You can switch between Euler and RPY angle programming with machine data MD 21100: ORIENTATION_IS_EULER or using G codes ORIEULER and ORIRPY.

**Programming of the orientation direction and rotation**

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

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

1. Direct programming of the rotary axis positions
An additional rotary axis must be defined for the direction of rotation.
2. Programming in Euler angles via A2, B2, C2
Angle C2 must also be programmed. The complete orientation is then defined including the tool rotation.
3. Programming in RPY angles via A2, B2, C2
Additional parameters are not required.
4. Programming of the direction vector via A3, B3, C3
The angle of rotation is programmed with THETA=<value>.

---

**Note**

The following cases do not allow for a programmed rotation:

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

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

---

**Rotation of the orientation vector**

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

- Linear interpolation, i.e. the angle between the current rotation vector and the start vector is a linear function of the path parameter.
- Non-linear by additional programming of a polynomial for the angle of rotation. $\theta$ The polynomial is maximum 5th degree in the format:

  PO[THT] = $(d_2, d_3, d_4, d_5)$

**Interpolation of the angle of rotation**

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

The end value of the angle,
in addition to the constant and linear coefficient $d_n$ of the polynomial cannot be programmed directly in this case.

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

The end angle $\theta_e$
is derived from the programming of the rotation vector.

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

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

---

**Note**

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

---

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

**Formula**

$$\theta(u) = \theta_s + d_1 u + d_2 u^2 + d_3 u^3 + d_4 u^4 + d_5 u^5 \quad (1)$$

This yields the following for the linear coefficients for parameter interval 0 ... 1:

**Formula**

$$d_1 = \theta_e - \theta_s - d_2 - d_3 - d_4 - d_5 \quad (2)$$

**Interpolation of the rotation vector**

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

- ORIROTA (**ori**entation **rot**ation **a**bsolute):
  The angle of rotation THETA is interpreted with reference to an absolute direction in space. The basic direction of rotation is defined by machine data.
- ORIROTR (**ori**entation **rot**ation **r**elative):
  The angle of rotation THETA is interpreted relative to the plane defined by the start and end orientation.
- ORIROTT (**ori**entation **rot**ation **t**angential):
  The angle of rotation THETA is interpreted relative to the change in orientation. That means the rotation vector interpolation is tangential to the orientation change for THETA=0.

  This is different to ORIROTR only if the orientation change does not take place in one plane. This is the case if at least one polynomial was programmed for the “tilt angle” PSI for the orientation. An additional angle of rotation THETA can then be used to interpolate the rotation vector such that it always exhibits a specific angle with reference to the orientation change.

**Activating the rotation**

A rotation of the orientation vector is programmed with the identifier THETA. The following options are available for programming:

| | |
|---|---|
| THETA=<value> | an angle of rotation which is reached at the end of the block. |
| THETA = $\theta_e$ | programmed angle $\theta_e$ can be interpreted either as an absolute dimension (G90 is active) or as an incremental dimension (G91 is active). |
| THETA = AC(...) | non-modal switchover to absolute dimensions. |
| THETA = IC(...) | non-modal switchover to incremental dimensions. |
| PO[THT] = (...) | programming of a polynomial for the angle of rotation THETA. |

The angle THETA is programmed in degrees.

The interpolation of the rotation vector is defined by the modal G codes:

| | |
|---|---|
| ORIROTA | Angle of rotation to an absolute direction of rotation. |
| ORIROTR | Angle of rotation relative to the plane between the start and end orientation. |
| ORIROTT | Angle of rotation relative to the change in the orientation vector. |
| PL | Length of the parameter interval where polynomials are defined. The interval always begins with 0. If no PL is programmed, PL = 1 is active. |

These G codes define the reference direction of the angle of rotation. The meaning of the programmed angle of rotation is interpreted accordingly.

**Supplementary conditions**

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

1. Rotary axis positions
2. Euler angle via A2, B2, C2
3. RPY angles via A2, B2, C2.
4. Direction vector via A3, B3, C3.

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

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

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

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

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

### 2.9.3 Extended interpolation of orientation axes (SW 6.1 and higher)

**Functionality**

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

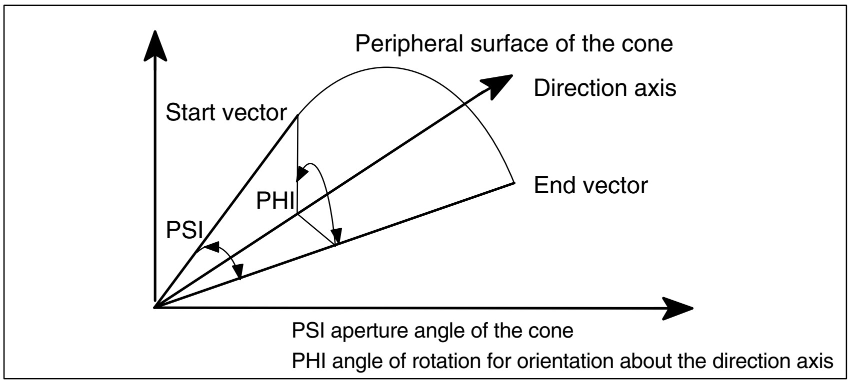



Fig. 2-18 Change in orientation of a peripheral surface of the cone located in space

**Definitions required**

Generally, the following data are required:

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

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

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

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

  In order to define the cone, the **direction vector** or its **aperture angle** must be programmed. Both may not be specified at the same time.
- A further option is to program an **intermediate orientation** that lies between the start and end orientation.

**Programming**

ORIPLANE **ori**entation interpolation in a **plane**
Interpolation in a plane (large circle interpolation)

ORICONCW **ori**entation interpolation on a **con**e **c**lock**w**ise
Interpolation on a peripheral surface of the cone in clockwise direction

ORICONCCW **ori**entation interpolation on a **con**e **c**ounter **c**lock**w**ise
Interpolation on a peripheral surface of the cone in counter clockwise direction.

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

---

**Note**

The programming of an end orientation is **not absolutely** necessary. If no orientation is specified, a full cone surface with 360 degrees is interpolated.

---

The **aperture angle of the cone** is programmed with
NUT= <angle> , where the angle is specified in degrees.

---

**Note**

An end orientation **must** be specified.
A complete outside taper with 360 degrees can be interpolated in this way. The sign of the aperture angle defines whether the traversing angle is to be greater or less than 180 degrees.

---

The meanings are as follows:

NUT = +... traversing angle less than or equal to 180 degrees

NUT = –... traversing angle greater than or equal to 180 degrees

A positive sign can be omitted when programming.

**Data for intermediate orientation**

ORICONIO **ori**entation interpolation on a **con**e with **i**ntermediate **o**rientation Interpolation on a peripheral surface of the cone with intermediate orientation specified.

If this G code is active, it is necessary to specify an **intermediate orientation**

with A7, B7, C7 , and this is specified as a (normalized) vector.

---

**Note**

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

---

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

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

**Rotation angle and aperture angle**

The following may be programmed for the angle of the cone

| | |
|---|---|
| PHI | **Angle of rotation** of the orientation around the direction axis |
| PSI | **Aperture angle** of the cone |

as well as the polynomials of the 5th degree (max.). They are programmed as follows:

| | |
|---|---|
| PO[PHI] = (a2, a3, a4, a5) | The constants and linear coefficients |
| PO[PSI] = (b2, b3, b4, b5) | are determined by the start and end orientation. |

**Further interpolation options**

It is possible to interpolate the orientation on a cone that connects at a tangent to the previous change in orientation.
This orientation interpolation is achieved by programming the G code ORICONTO.

| | |
|---|---|
| ORICONTO | **ori**entation interpolation on a **con**e with **t**angential **o**rientation<br>Interpolation on a peripheral surface of the cone with tangential transition |

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

| | |
|---|---|
| ORICURVE | **ori**entation interpolation with a second **curve**<br>Interpolation of the orientation specifying a movement between two contact points of the tool |

The coordination of the movement of the 2nd contact point of the tool is necessary here. This additional curve in space is programmed with

| | |
|---|---|
| XH, YH, ZH | Except for the relevant end values, you can also program additional polynomials of the form |

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

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

---

**Note**

No circles or involutes are permissible. IT is also possible to activate a spindle interpolation with BSPLINE. The programmed end points of both curves in space are then interpreted as nodes.

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

---

**Boundary conditions**

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

**Activation**

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

| | |
|---|---|
| ORIPLANE | Interpolation in a plane with end orientation specified (corresponds to ORIVECT) |
| ORICONCW | Interpolation on a peripheral surface of the cone in clockwise direction<br>Specification of the end orientation and cone direction or aperture angle of the cone. |
| ORICONCCW | Interpolation on a peripheral surface of the cone in counterclockwise direction.<br>Specification of the end orientation and cone direction or aperture angle of the cone. |
| ORICONIO | Interpolation on a peripheral surface of the cone with specification of the end orientation and an intermediate orientation. |
| ORICONTO | Interpolation a peripheral surface of the cone with tangential transition, specification of the end orientation. |
| ORICURVE | Interpolation of the orientation with specification of the movement of two contact points of the tool. |

**Example**

The various orientation changes are programmed in the following sample program:

```
...
N10 G1 X0 Y0 F5000
N20 TRAORI                              ; Orientation transformation activated.
N30 ORIVECT                             ; Interpolate tool orientation as vector
N40 ORIPLANE                            ; Select large circle interpolation
N50 A3=0 B3=0 C3=1                      ;
N60 A3=0 B3=1 C3=1                      ; Orientation rotated 45 degrees in Y/Z
                                        ; plane, at the end of the block, the
                                        ; orientation (0, 1/√2, 1/√2) is reached.
N70 ORICONCW                            : The orientation vector is interpolated
                                        ; on a cone with the direction
                                        ;
N80 A6=0 B6=0 C6=1 A3=1 B3=0 C3=1       ; (0, 0, 1) up to the orientation
                                        ; (1/√2= 0: 1/√2) in clockwise direction
                                        ; The rotation angle is 270 degrees.
                                        ; The tool orientation turns a full
N90 A6=0 B6=0 C6=1                      ; rotation on the same cone
...
```

## 2.10 Cartesian manual travel (810D, SW 6.1 and higher)

**Functionality**

The Cartesian manual travel function allows you to set axes independently in the Cartesian coordinate systems as reference system for the JOB mode

- Basic Coordinate System BCS MD 21106: Bit0 = 1
- Workpiece Coordinate System WCS MD 21106: Bit1 = 1
- Tool Coordinate System TCS MD 21106: Bit2 = 1

Machine data MD 21106: CART_JOG_SYSTEM with the Cartesian manual travel function is activated to do this.

---

**Note**

The Cartesian manual travel function is implemented in SINUMERIK 810D powerline with CCU3 as of SW 6.1. SINUMERK 840D requires the option “Transformation package handling” SW6.3 or higher.

The workpice coordinate system has been shifted and rotated compared to the basic coordinate system via frames.

**References:** /FB1/, Description of Functions, Basic Machine, K2 Axes, Coordinate Systems, Frames, Reset Behavior

---

Representation of the reference system in the coordinate system:

| | | | |
|---|---|---|---|
| | **WCS** = Workpiece zero | | **TCS** = Tool zero |

**Selecting reference systems**

For JOG motion, you can specify one of three reference systems separately both for the

**Translation** (coarse shift) of the geometry axes and for the
**orientation** for orientation axes using the

SD 42650: CART_JOG_MODE.

If more than one bit is set for the translation or for the orientation reference system, or if an attempt is made to set a reference system that has not been activated via MD 21106: CART_JOG_SYSTEM, Alarm 14148 “Reference system not permissible for Cartesian manual travel” is issued.

**Translation**

A translation movement can be used to move the tool tip (TCP) in parallel and 3-dimensional to the axes of the reference system. The traversing movement is made via the VDI signals of the geometry axes.

Machine data MD 24120: TRAFO_GEOAX_ASSIGN_TAB_x[n] is used to assign the geometry axes. Simultaneous traversing in more than one direction permits the execution of movements that lie parallel to the directions of the reference system.

**Translation in the BCS**

The Basic Coordinate System (BCS) describes the Cartesian zero of the machine.

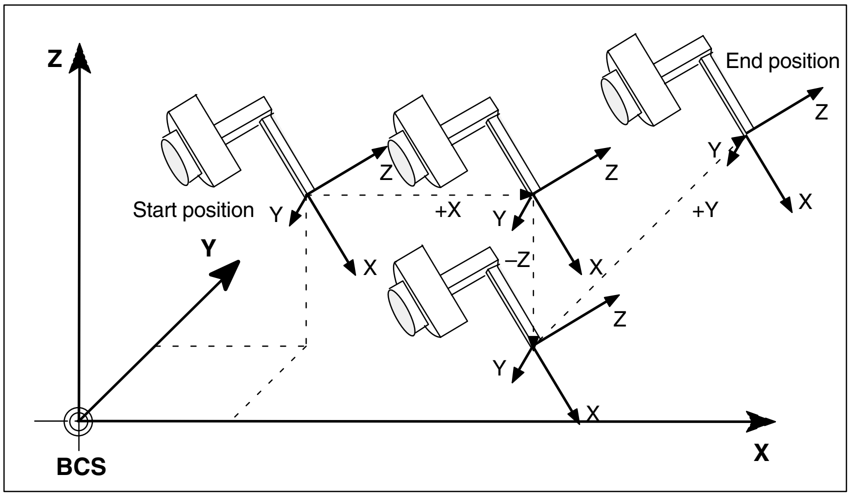



Fig. 2-19 Cartesian manual travel in the Basic Coordinate System (Translation)

**Translation in the WCS**

The Workpiece Coordinate System (WCS) lies in the workpiece zero. The workpiece coordinate system can be shifted and rotated relative to the reference system via frames. As long as the frame rotation is active, the traversing movements correspond to the translation of the movements in the basic coordinate system.

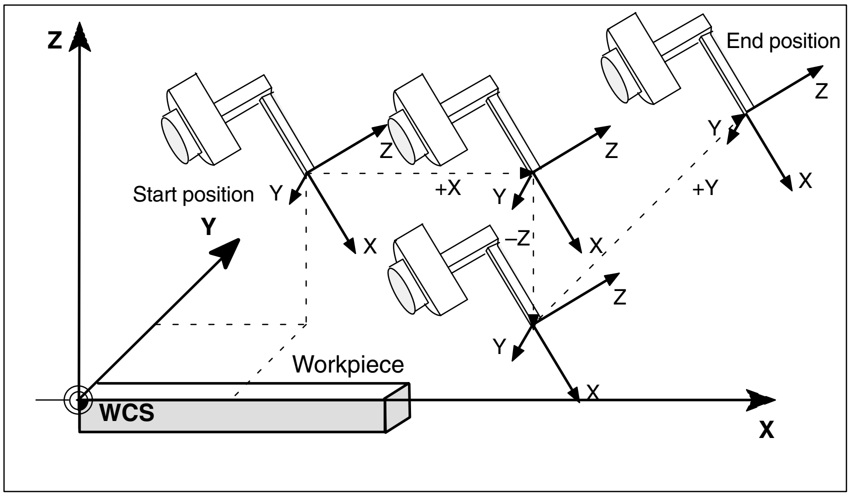



Fig. 2-20 Cartesian manual travel in the Workpiece Coordinate System (Translation)

**Translation in the TCS**

The Tool Coordinate System (TCS) lies in the tool tip. Its direction depends on the current setting of the machine, since the tool coordinate system moves during the motion.

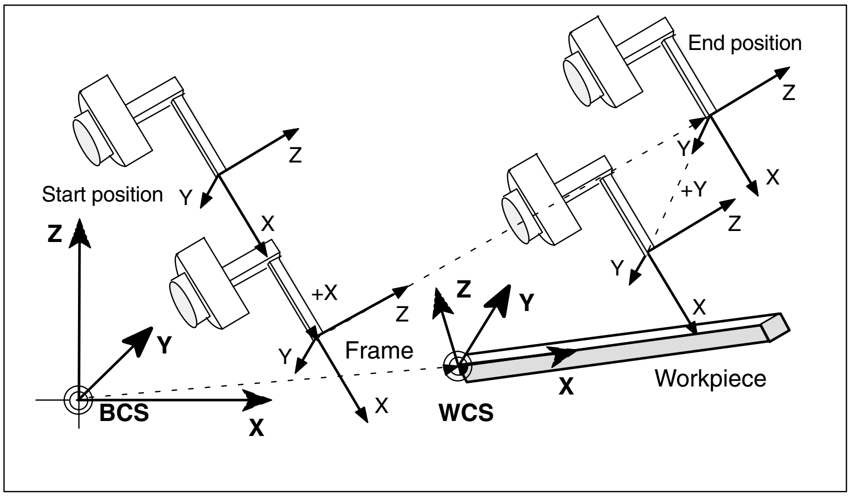



Fig. 2-21 Cartesian manual travel in the Tool Coordinate System (Translation)

**Translation and Orientation in the TCS simultaneously**

If translation and orientation motions are executed at the same time, the translation is always traversed corresponding to the current orientation of the tool. This permits infeed movements that are made directly in the tool direction or movements that run perpendicular to tool direction.

**Orientation**

The tool can be aligned to the component surface via an orientation movement. The orientation movement is given control from the PLC via the VDI signals of the orientation axes (DB21, ... DBB321).

Several orientation axes can be traversed simultaneously. The virtual orientation axes execute rotations around the fixed axes of the relevant reference system.

The **rotations** are identified according to the RPY angles.

- A angle : rotation around Z axis
- B angle : rotation around Y axis
- C angle: rotation around X axis

**Programing rotations:**
The user can define how the rotations are to be executed with the current G codes of group 50 for orientation definition

ORIEULER, ORIRPY, ORIVIRT1 and ORIVIRT2.

With ORIVIRT1, the rotations are executed according to MD 21120: ORIAX_TURN_TAB_1. The assignment of the orientation axes to the channel axes is made via MD 24585: TRAFO5_ORIAX_ASSIGN_TAB_1.

The **direction of rotation** is determined according to the “right hand rule”. The thumb points in the direction of the rotary axis. The finger stipulates the positive direction of rotation.

**Orientation in the WCS**

The rotations are made around the defined directions of the Workpiece Coordinate Systems. If frame rotation is active, the movements correspond to the rotations in the Basic Coordinate System.

**Orientation in the BCS**

The rotations are made around the defined directions of the Basic Coordinate System.

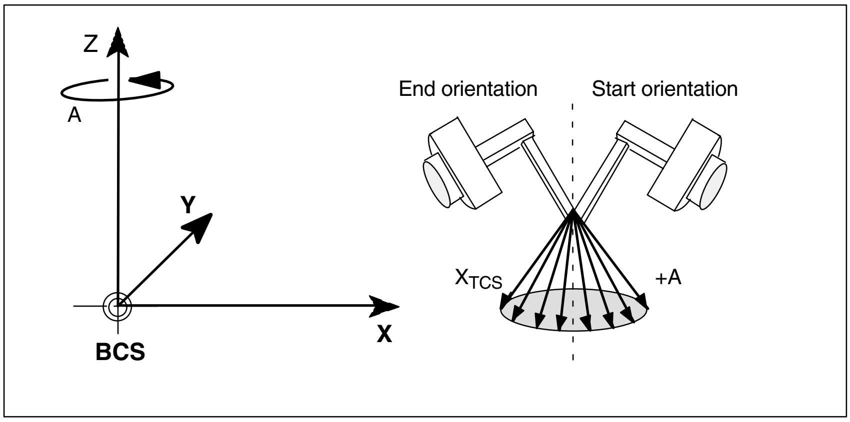



Fig. 2-22 Cartesian manual travel in the basic coordinate system Orientation angle A

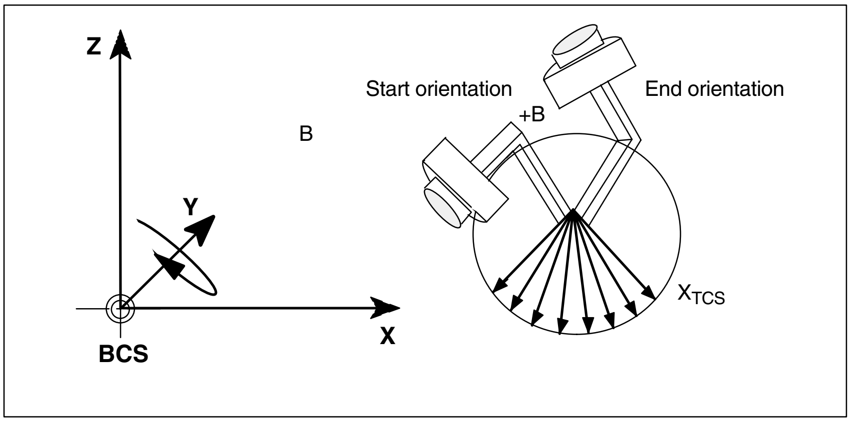



Fig. 2-23 Cartesian manual travel in the basic coordinate system Orientation angle B

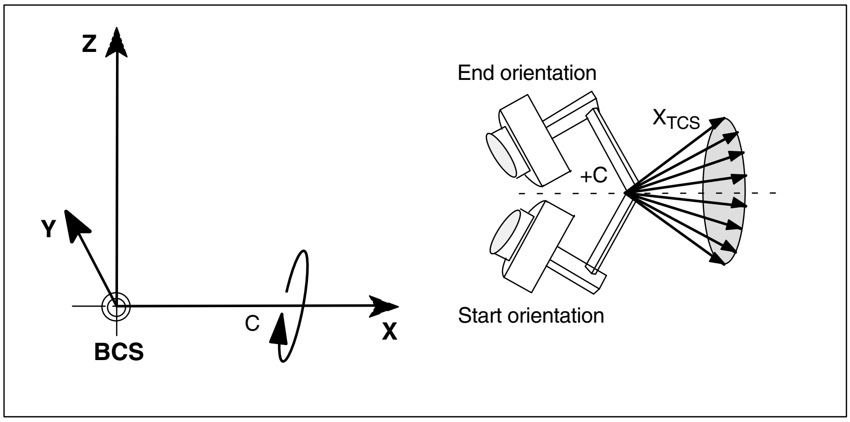



Fig. 2-24 Cartesian manual travel in the basic coordinate system Orientation angle C

**Orientation in TCS**

The rotations are around the moving directions in the Tool Coordinate System. The current homing directions of the tool are always used as rotary axes.

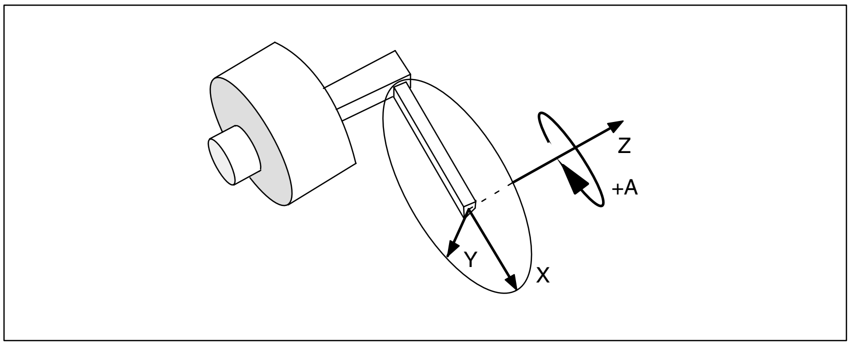



Fig. 2-25 Cartesian manual travel in the Tool Coordinate System, orientation angle A

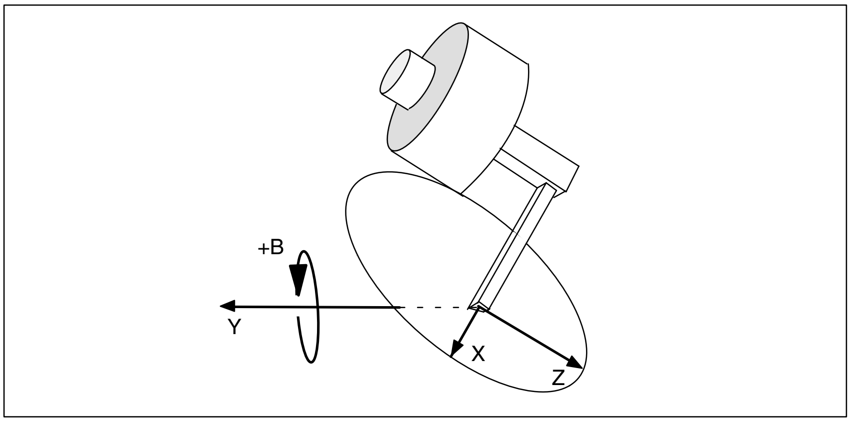



Fig. 2-26 Cartesian manual travel in the Tool Coordinate System, orientation angle B

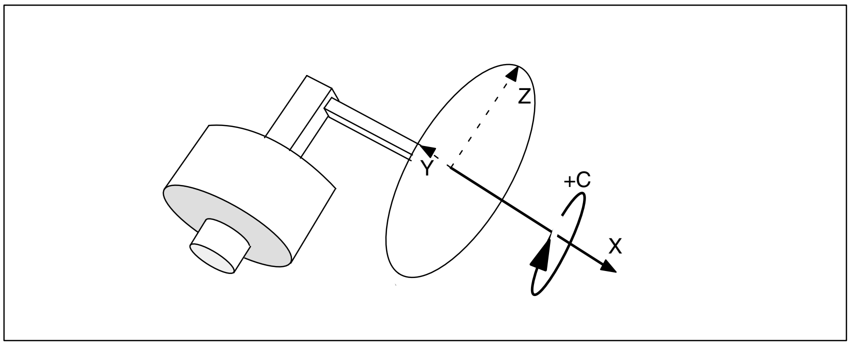



Fig. 2-27 Cartesian manual travel in the Tool Coordinate System, orientation angle C

### Boundary conditions

Only if IS "Transformation active" (DB31, ... DBX33.6) is set to 1, is it possible to execute the Cartesian manual travel function. The following boundary conditions apply:

- The option "Transformation package handling" with 5-axis or 6-axis transformation is required for SINUMERIK 840D SW6.3 and higher.
- Virtual orientation axes must be defined via machine data MD 24585: TRAFO5_ORIAX_ASSIGN_TAB_1[n].
- IS "Activate PTP/CP traversing" (DB31, ... DBX29.4) must be 0.
- Machine data MD 21106: CART_JOG_SYSTEM must be > 0.

Table 2-6 Conditions for Cartesian manual traversing

| Transformation in program active (TRAORI..) | G codes PTP/CP | IS "Activate PTP/CP tranversing" | IS "Transformation active" |
|---|---|---|---|
| FALSE | Not functional! | Not functional! | DB31, ... DBX33.6 = 0 |
| TRUE | CP | DB31, ... DBX29.4 = 0 | DB31, ... DBX33.6 = 1 |
| TRUE | CP | DB31, ... DBX29.4 = 1 | DB31, ... DBX33.6 = 0 |
| TRUE | PTP | DB31, ... DBX29.4 = 0 | DB31, ... DBX33.6 = 1 |
| TRUE | PTP | DB31, ... DBX29.4 = 1 | DB31, ... DBX33.6 = 0 |

The G code PTP/CP currently active in the program does not affect Cartesian manual travel. The VDI interface signals are interpreted in the channel DB for geometry and orientation axes.

### Activation

The reference system for Cartesian manual travel is set as follows:

- The function Cartesian manual travel is activated with machine data MD 21106: CART_JOG_SYSTEM > 0.

  The reference systems BCS, WCS or TCS are enabled by setting the bits in MD 21106: CART_JOG_SYSTEM.
- The JOG traversing motion via SD 42650: CART_JOG_MODE

  Standard response as before: Bits 0 to 2 = 0, Bits 8 to 10 = 0

  Reference system for translation via reference system for orientation via Bits 0–2 and the Bits 8–10

  If not all of the bits are set to 0, the process uses the new function. The reference systems for translation and orientation may be set independently.

The meaning of the bits is explained in the table below 2-7.

Table 2-7 Bit assignment for SD 42650: CART_JOG_MODE (only one bit may be set)

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| Reserved | | | | | Translation in TCS | Translation in WCS | Translation in BCS |
| **Bit15** | **Bit14** | **Bit13** | **Bit12** | **Bit11** | **Bit10** | **Bit9** | **Bit8** |
| Reserved | | | | | Orientation in TCS | Orientation in WCS | Orientation in BCS |

**Combining reference systems**

The table below shows all the combination options for reference systems.

Table 2-8 Combination options for reference systems

| SD 42650: CART_JOG_MODE | | | | | | Reference system for | |
|---|---|---|---|---|---|---|---|
| Bit 10 | Bit 9 | Bit 8 | Bit 2 | Bit 1 | Bit 0 | Orientation | Translation |
| 0 | 0 | 0 | don't care | don't care | don't care | Standard | Standard |
| Standard | Standard | Standard | 0 | 0 | 0 | Standard | Standard |
| 0 | 0 | 1 | 0 | 0 | 1 | BCS | BCS |
| 0 | 0 | 1 | 0 | 1 | 0 | BCS | WCS |
| 0 | 0 | 1 | 1 | 0 | 0 | BCS | TCS |
| 0 | 1 | 0 | 0 | 0 | 1 | WCS | BCS |
| 0 | 1 | 0 | 0 | 1 | 0 | WCS | WCS |
| 0 | 1 | 0 | 1 | 0 | 0 | WCS | TCS |
| 1 | 0 | 0 | 0 | 0 | 1 | TCS | BCS |
| 1 | 0 | 0 | 0 | 1 | 0 | TCS | WCS |
| 1 | 0 | 0 | 1 | 0 | 0 | TCS | TCS |

## 2.11 Cartesian PTP travel

**Function**

This function makes it possible to approach a Cartesian position with a synchronized axis movement.

It is particularly useful in cases where, for example, the position of the joint is changed, causing the axis to move through a singularity.

When an axis passes through a singularity, the feed velocity would normally be reduced or the axis itself overloaded.

---

**Note**

The function can only be used meaningfully in conjunction with an active transformation. Furthermore, the “Cartesian PTP travel” function may only be used in conjunction with the G0 and G1 commands. Alarm 14144 “PTP travel not possible” is otherwise output.

---

**Activation**

The function is activated when the PTP command is programmed.
The function can be deactivated again with the CP command. Both these commands are contained in G group 49.

– PTP command:
  The programmed Cartesian position is approached with a synchronized axis motion (PTP=point-to-point)

– CP command:
  The programmed Cartesian point is approached with a path movement (default setting), (CP=continuous path)

**Power ON**

After power ON, traversing mode CP is automatically set for axis traversal with transformation. The initial setting can be altered to PTP in MD 20152: GCODE_RESET_VALUES[48].

**RESET**

MD 20152: GCODE_RESET_MODE[48] (group 49) defines which setting is active after RESET/end of parts program.

MD 20152: GCODE_RESET_MODE[48]

– MD=0:
  The setting depends on MD 20150:GCODE_RESET_VALUES[48]

– MD=1:
  Active setting remains valid

If the setting is selected with reference to MD 20150: GCODE_RESET_VALUES[48], the default setting PTP or CP can be activated with this machine data.

**Block search**

No change.

### 2.11.1 Programming of position

Generally speaking, a machine position is not uniquely defined solely by a position input with Cartesian coordinates and the orientation of the tool. Depending on the kinematics of the relevant machine, the joint may assume up to 8 different positions. These joint positions are specific to individual transformations.

**STAT address**

A Cartesian position must be convertible into a unique axis angle. For this reason, the position of the joints must be entered in the STAT address.

The STAT address contains a bit for every possible setting as a binary value. The meaning of these bits is determined by the relevant transformation.

As regards the transformations contained in the publication entitled "Handling of Transformation Package (TE4)", the bits are assigned to different joint positions, as shown in Fig. 2-28. See also Subsection 2.11.4

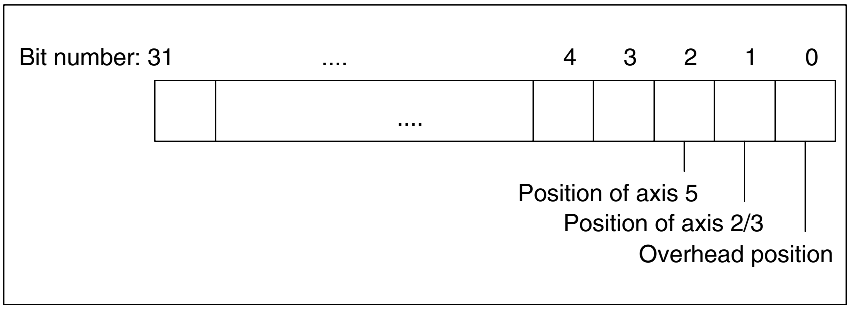



Fig. 2-28 Position bits for Handling Transformation Package

---

**Note**

It is only meaningful to program the STAT address for "Cartesian PTP travel", since changes in position are not normally possible while an axis is traversing with active transformation. The starting point position is applied as the destination point for traversal with the CP command.

---

## 2.11.2 Overlap areas of axis angles

**TU address**

In order to approach axis angles in excess of ± 180 degrees without ambiguity, the information must be programmed in the TU (turn) address. Address TU thus represents the sign of the axis angles, An axis angle of $|\theta| < 360$ degrees can therefore be specified uniquely.

Variable TU contains a bit, which indicates the traversing direction, for each axis included in the transformation.

– TU bit=0: 0 degrees $\leq \theta <$ 360 degrees

– TU bit=1: –360 degrees $< \theta <$ 0 degrees

The TU bit is set to 0 for linear axes.

In the case of axes with a traversing range > ± 360 degrees, the axis always moves across the shortest path, because the axis position cannot be specified uniquely by the TU information.

If no TU is programmed for a position, the axis always traverses via the shortest possible route.

## 2.11.3 PTP/CP switchover in JOG mode

In JOG mode, the transformation can be switched on and off via a PLC control signal. This control signal is active only in JOG mode and when a transformation has been activated via the program.

If the mode is switched back to AUTO, the state which was last active before switchover is made active again.

The “point-to-point traversal active” signal DBX317.6 shows which traversal type is active. By means of the “Activate point-to-point traversal” signal DBX29.4 the traversal type can be modified.

**Operating mode changeover**

The “Cartesian PTP travel” function can be used meaningfully only in the AUTO and MDA modes. The CP setting is automatically activated if the operating mode is switched to JOG. If the mode is then switched back to AUTO or MDA, the mode that was last active in either mode is made active again.

**REPOS**

The setting for “Cartesian PTP travel” is not altered during re-positioning. If PTP was set in the interruption block, then repositioning takes place in PTP. For a sloping axis “TRAANG”, only CP travel is active in REPOS mode.

## 2.11.4 Examples of ambiguities of position

The kinematics for a 6-axis joint have been used to illustrate the ambiguities caused by different joint positions.

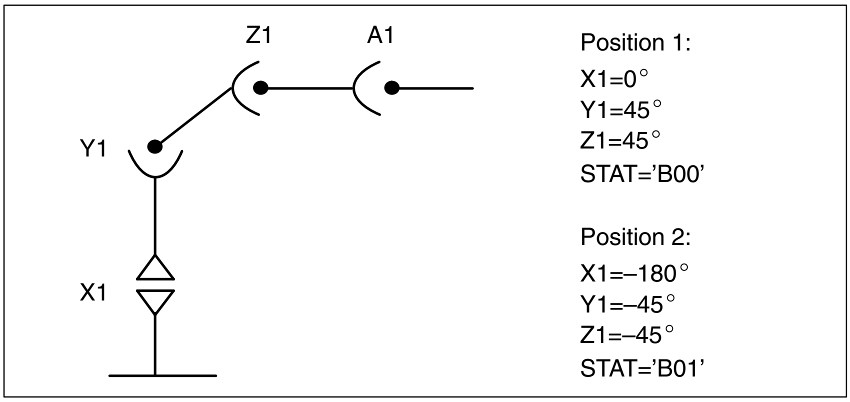



Fig. 2-29 Ambiguity in overhead area

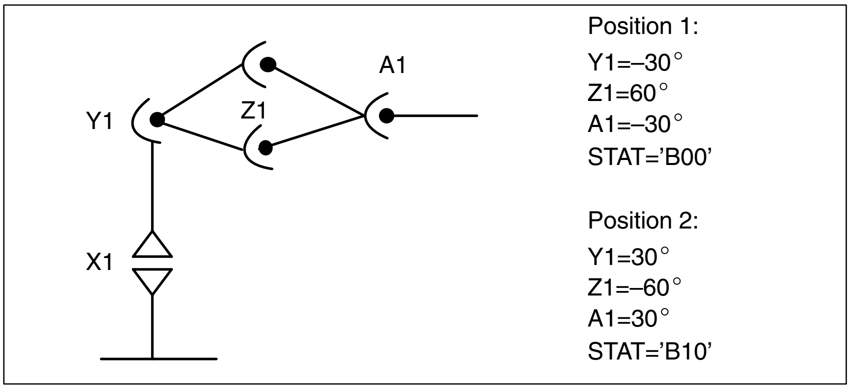



Fig. 2-30 Ambiguity of top or bottom elbow

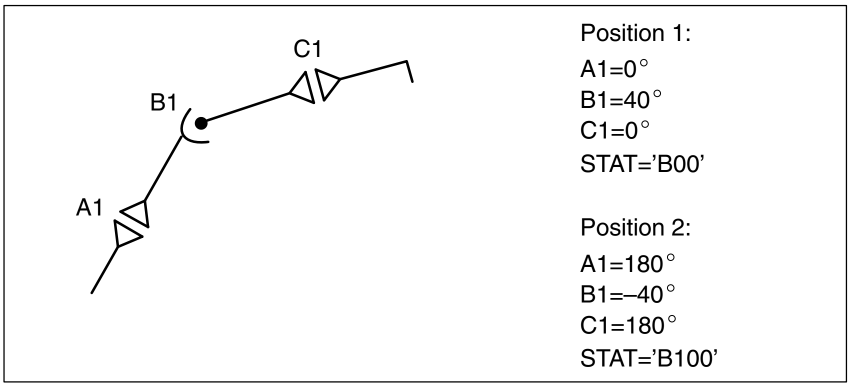



Fig. 2-31 Ambiguity of axis B1

## 2.11.5 Example of ambiguity in rotary axis position

The rotary axis position shown in Fig. 2-32 can be approached in negative or positive direction. The direction is programmed under address A1.

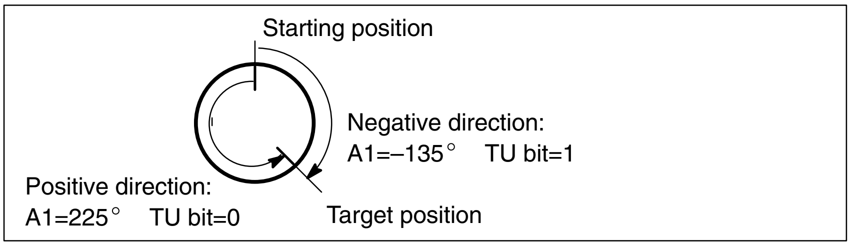



Fig. 2-32 Ambiguity in rotary axis position

■

# Notes

# Supplementary Conditions 3

## 3.1 Availability

**Function 5-axis machining package**

The function is an option and available with

- SINUMERIK 840D with NCU 572/573, with SW2 and higher.
- SINUMERIK 810D with CCU2, as of SW2 for handling the transformation package.

The function is not included in the export variant SINUMERIK 840DE/810DE.

**3-axis/4-axis transformation function**

This function is an option and available for

- SINUMERIK 840D with NCU 572/573, with SW 3.2 and higher.

The function is not included in the export variant SINUMERIK 840DE/810DE.

**Swiveled linear axis function**

This function is an option. It is available with SW 3.2 and higher.

**Universal milling head function**

This function is an option which can be partially implemented with SW 3.1 and higher. Only the transformation for the axis sequence CA' can be implemented in SW 3.1. In SW 3.2 and higher, all axis sequences shown in Table 2-4 are available.

**Generic 5-axis-transformation funtion**

This function is an option and available for

- SINUMERIK 840D with NCU 572/573, with SW 5.2 and higher
- SINUMERIK 810D with CCU3, SW 3.2 and higher.

**Online tool length offset**

This function requires the option Generic 5-axis transformation and is available for

- SINUMERIK 840D with NCU 572/573, with SW 6.4 and higher
- SINUMERIK 810D with CCU3, with SW 6.2 and higher.

**Number of transformations**

In software version 4.1 and higher, the total number of transformations available for each channel has been increased from 4 to 8.

SINUMERIK 810D with CCU2 V5.1 and higher offers transformations in two channels.

**Multi-axis interpolation function**

The function is not included in the export version SINUMERIK 840DE/810DE.

**Orientation axes function**

This function is included in the scope of the 5-axis transformation option and available in software version 5.3 and higher for

- SINUMERIK 810D with CCU2.
- SINUMERIK 840D with NCU572/573.

The function is not included in the export variant SINUMERIK 840DE/810DE.

**Cartesian PTP travel function**

This function is included in the scope of the 5-axis transformation option and available in software version 5.3 and higher for

- SINUMERIK 810D with CCU2.
- SINUMERIK 840D with NCU572/573.

The function is not included in the export variant SINUMERIK 840DE/810DE.

**Function Cartesian manual travel**

This function is available per default in

- SINUMERIK 810D with CCU3, SW 6.1 and higher.
- SINUMERIK 840Di with PCU50, SW 2.1 and higher

The function is included in the “Transformation package handling” option and is available with SW6.3 and higher for

- SINUMERIK 840D with NCU572/573.

■

# Data Descriptions (MD, SD) 4

## 4.1 General machine data

| | | | | | |
|---|---|---|---|---|---|
| **10620**<br>MD number | **EULER_ANGLE_NAME_TAB**<br>Name of Euler angle [GEOaxisNo.]: 0...2 | | | | |
| Default setting: A2, B2, C2 | | Min. input limit: – | | Max. input limit: – | |
| Changes effective after power ON | | | Protection level: 2/7 | | Unit: – |
| Data type: STRING | | | | Applies from SW 2.1 | |
| Special cases, errors... | • The entered name must not conflict with machine and geometry axis names/assignments.<br>• The entered name must not coincide with the channel axis names in the channel.<br>(MD 20080: AXCONF_CHANAX_NAME_TAB), names for directional vectors (MD 10640: DIR_VECTOR_NAME_TAB), names for intermediate point coordinates for CIP (MD 10660: INTERMEDIATE_POINT_NAME_TAB) and the names of interpolation parameters (MD 10650: IPO_PARAM_NAME_TAB).<br>• The entered name may not adopt the following reserved address characters:<br>– D Tool offset (D function) – E Reserved<br>– F Feedrate (F function) – G Preparatory function<br>– H Auxiliary function (H function) – L Subroutine call<br>– M Miscellaneous function (M function) – N Subblock<br>– P Subroutine number of passes – R Arithmetic parameter<br>– S Spindle speed (S function) – T Tool (T function)<br>• Vocabulary words (e.g. DEF, SPOS etc.) and predefined identifiers (e.g. ASPLINE, SOFT) are also illegal.<br>• An angle identifier consists of a valid address character (A, B, C, I, J, K, Q, U, V, W, X, Y, Z), followed by an optional numeric extension (1–99). | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **10630**<br>MD number | **NORMAL_VECTOR_NAME_TAB**<br>Name of normal vectors | | | | |
| Default setting: A4, B4, C4<br>Default: A5, B5, C5 | | Min. input limit: – | | Max. input limit: – | |
| Changes effective after power ON | | | Protection level: 2/7 | | Unit: – |
| Data type: STRING | | | | Applies from SW 4.1 | |
| Meaning: | List of identifiers of normal vector components at start and end of block.<br>The rules for axis identifiers described in MD 20080: AXCONF_CHANAX_NAME_TAB apply when chosing identifiers.<br>The identifiers must be selected such that there are no conflicts with other identifiers, for example, axes, Euler angle, normal vector, direction vector, intermediate coordinate. | | | | |
| Related to .... | Choice of possible axis identifiers as for MD 20080: AXCONF_CHANAX_NAME_TAB | | | | |

| 10640<br>MD number | DIR_VECTOR_NAME_TAB<br>Name of direction vectors | | | | |
|---|---|---|---|---|---|
| Default setting: A3, B3, C3 | | Min. input limit: – | | Max. input limit: – | |
| Changes effective after power ON | | | Protection level: 2/7 | | Unit: – |
| Data type: STRING | | | | Applies from SW 4.1 | |
| Meaning: | List of identifiers for direction vector components.<br>The rules for axis identifiers described in MD 20080: AXCONF_CHANAX_NAME_TAB apply when chosing identifiers.<br>The identifiers must be selected such that there are no conflicts with other identifiers, for example, axes, Euler angle, normal vector, direction vector, intermediate coordinate. | | | | |
| Related to .... | Choice of possible axis identifiers as for MD 20080: AXCONF_CHANAX_NAME_TAB | | | | |

| 10642<br>MD number | ROT_VECTOR_NAME_TAB<br>Name of rotation vectors | | | | |
|---|---|---|---|---|---|
| Default setting: A6, B6, C6 | | Min. input limit: – | | Max. input limit: – | |
| Changes effective after power ON | | | Protection level: 2/7 | | Unit: – |
| Data type: STRING | | | | Applies from SW 6.1 | |
| Meaning: | List of identifiers for rotational vector components in direction of taper.<br>The rules for axis identifiers described in MD 20080: AXCONF_CHANAX_NAME_TAB apply when chosing identifiers.<br>The identifiers must be selected such that there are no conflicts with other identifiers, for example, axes, Euler angle, normal vector, direction vector, intermediate coordinate. | | | | |
| Related to .... | Choice of possible axis identifiers as for MD 20080: AXCONF_CHANAX_NAME_TAB | | | | |

| 10644<br>MD number | INTER_VECTOR_NAME_TAB<br>Name of intermediate vector components | | | | |
|---|---|---|---|---|---|
| Default setting: A7, B7, C7 | | Min. input limit: – | | Max. input limit: – | |
| Changes effective after power ON | | | Protection level: 2/7 | | Unit: – |
| Data type: STRING | | | | Applies from SW 6.1 | |
| Meaning: | List of identifiers for intermediate vector components.<br>The rules for axis identifiers described in MD 20080: AXCONF_CHANAX_NAME_TAB apply when chosing identifiers.<br>The identifiers must be selected such that there are no conflicts with other identifiers, for example, axes, Euler angle, normal vector, direction vector, intermediate coordinate. | | | | |
| Related to .... | Choice of possible axis identifiers as for MD 20080: AXCONF_CHANAX_NAME_TAB | | | | |

| 10646<br>MD number | ORIENTATION_NAME_TAB<br>Identifier for programming a 2nd orientation path | | | | |
|---|---|---|---|---|---|
| Default setting: XH, YH, ZH | | Min. input limit: – | | Max. input limit: – | |
| Changes effective after power ON | | | Protection level: 2/7 | | Unit: – |
| Data type: STRING | | | | Applies from SW 6.1 | |
| Meaning: | Identifier list for programming the 2nd curve in space for tool optimization.<br>The rules for axis identifiers described in MD 20080: AXCONF_CHANAX_NAME_TAB apply when chosing identifiers.<br>The identifiers must be selected such that there are no conflicts with other identifiers, for example, axes, Euler angle, normal vector, direction vector, intermediate coordinate. | | | | |
| Related to .... | Choice of possible axis identifiers as for MD 20080: AXCONF_CHANAX_NAME_TAB | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **10648**<br>MD number | **NUTATION_ANGLE_NAME**<br>Name of the aperture angle | | | | |
| Default setting: NUT | | Min. input limit: – | | Max. input limit: – | |
| Changes effective after power ON | | | Protection level: 2/7 | | Unit: – |
| Data type: STRING | | | | Applies from SW 6.1 | |
| Meaning: | Identifier for the aperture angle for orientation interpolation.<br><br>The identifiers must be selected such that there are no conflicts with other identifiers, for example, axes, Euler angle, normal vector, direction vector, intermediate coordinate. | | | | |
| Related to .... | Choice of possible axis identifiers as for MD 20080: AXCONF_CHANAX_NAME_TAB | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **10670**<br>MD number | **STAT_NAME**<br>Name of position information | | | | |
| Default setting: STAT | | Min. input limit: – | | Max. input limit: – | |
| Changes effective after power ON | | | Protection level: 2/7 | | Unit: – |
| Data type: STRING | | | | Applies from SW 5.2 | |
| Meaning: | Identifier for position information<br>to resolve ambiguities for Cartesian PTP travel.<br><br>The identifiers must be selected such that there are no conflicts with other identifiers, for example, axes, Euler angle, normal vector, direction vector, intermediate coordinate. | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **10672**<br>MD number | **TU_NAME**<br>Name of position information of the axes | | | | |
| Default setting: TU | | Min. input limit: – | | Max. input limit: – | |
| Changes effective after power ON | | | Protection level: 2/7 | | Unit: – |
| Data type: STRING | | | | Applies from SW 5.2 | |
| Meaning: | Identifier for position information of axes<br>to resolve ambiguities for Cartesian PTP travel.<br><br>The identifiers must be selected such that there are no conflicts with other identifiers, for example, axes, Euler angle, normal vector, direction vector, intermediate coordinate. | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **10674**<br>MD number | **PO_WITHOUT_POLY**<br>Permissible programming for PO[ ] without POLY having to be active. | | | | |
| Default setting: FALSE | | Min. input limit: 0 | | Max. input limit: 1 | |
| Changes effective after power ON | | | Protection level: 2/7 | | Unit: – |
| Data type: BOOLEAN | | | | Applies from SW 5.3 | |
| Special cases, errors... | You can use the machine data to set the response of the control when programming polynomials with PO[...].<br><br>**MD 10674 = 0 (FALSE):**<br>Previous response, which is active during programming of PO[ ] without POLY.<br>An error message is notified.<br>**MD 10674 = 1 (TRUE):**<br>Programming of PO[...] is allowed without the G code POLY having to be active. POLY and POLYPATH( ) produce only the actual execution of the polynomial interpolation in this case. | | | | |

## 4.2 Channel-specific machine data

The following machine data are relevant for all transformations described in this Description of Functions. Afterwards, the specific machine data for swiveling linear axis and universal milling head are described.

<table>
<tr><td>21100<br>MD number</td><td colspan="3">ORIENTATION_IS_EULER<br>Angle definition for orientation programming</td></tr>
<tr><td>Default setting: 1</td><td>Min. input limit: 0</td><td colspan="2">Max. input limit: 1</td></tr>
<tr><td colspan="2">Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1)</td><td>Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1)</td><td>Unit: –</td></tr>
<tr><td colspan="2">Data type: BOOLEAN</td><td colspan="2">Applies from SW 2.1</td></tr>
<tr><td>Meaning:</td><td colspan="3">MD 21100 = 0 (FALSE):<br>The values programmed with A2, B2, C2 for orientation programming are interpreted as an RPY angle (in degrees).<br>The orientation vector is produced by rotating a vector in direction Z first through C2 about the Z axis, then through B2 about the new Y axis and finally through A2 about the new X axis. In contrast to Euler angle programming, all three values influence the orientation vector in this case.<br>MD 21100 = 1 (TRUE):<br>The values programmed with A2, B2, C2 for orientation programming are interpreted as Euler angle (in degrees).<br>The orientation vector is produced by rotating a vector in direction Z first through A2 about the Z axis, then through B2 about the new X axis and finally through C2 about the new Z axis. The value of C2 is therefore meaningless.</td></tr>
</table>

| | | | |
|---|---|---|---|
| **21108**<br>MD number | **POLE_ORI_MODE**<br>Behavior during large circle interpolation at pole position | | |
| Default setting: 0, 0, 0, 0, ... | Min. input limit: 0 | Max. input limit: 22 | |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher) | Protection level:<br>7 / 7 (SW 5.2 and higher) | Unit: – | |
| Data type: DWORD | Applies from SW 5.2 | | |
| Meaning: | Defines the treatment of orientation changes during large circle interpolation if the starting orientation is equal to the pole orientation or approximates to it and the end orientation of the block is outside the tolerance circle defined in MD TRAFO5_NON_POLE_LIMIT_1/2.<br>The position of the polar axis is arbitrary in the polar position. For the large circle interpolation, however, a specified orientation is required for this axis.<br>The MD is coded decimally.<br>The **unit digits**, define the behavior if the start orientation precisely matches the pole orientation,<br>the **ten digits**, the behavior if the start orientation of the block is outside the tolerance circle defined in MD TRAFO5_NON_POLE_LIMIT_1/2.<br><br>The **unit digits** can have the following values (effective if start orientation equals pole orientation):<br>**0**: Interpolation is executed as axis interpolation. The preset orientation path (large circle) is only maintained if the polar axis (randomly) has the correct position and the basic orientation is perpendicular to the 2nd rotary axis.<br>**1**: A block is inserted in front of the block in which the situation described above occurs. This block positions the polar axis such that large circle interpolation can be performed in the following block.<br>**2**: If the block in front of the block in which the situation described above occurs, contains a geometry axis movement but not an orientation movement, the necessary positioning movement of the polar axis is effected in addition in this previous block.<br>If one of the two conditions has not been fulfilled (block contains no geometry axis movement or block contains orientation movement), the polar axis movement is carried out in a block of its own (for behavior see 1).<br><br>The **ten digits** can have the following values (effective if the starting orientation is different from the pole orientation but is within the tolerance circle defined in TRAFO5_NON_POLE_LIMIT_1/2):<br>**00**: Interpolation is executed as axis interpolation. The preset orientation path (large circle) is only maintained if the polar axis (randomly) has the correct position and the basic orientation is perpendicular to the 2nd rotary axis.<br>**10**: A block is inserted in front of the block in which the situation described occurs. This block positions the two rotary axes at the point at which the programmed large circle interpolation intersects the tolerance circle defined in TRAFO5_NON_POLE_LIMIT_1/2. In the original block, movement is effected with large circle interpolation from this point onward.<br>**20**: If the block in front of the block in which the situation described above occurs, contains a geometry axis movement but not an orientation movement, the necessary positioning movements of the two rotary axes are effected in addition in this previous block. The residual movement in the original block is the same as that for value 10 of the MD. If one of the two conditions has not been fulfilled (block contains no geometry axis movement or block contains orientation movement), the polar axis movement is carried out in a block of its own (for behavior see 10).<br>The values of the unit and ten digit are added. | | |

| 24100, 24430<br>24200, 24440<br>24300, 24450<br>24400, 24460<br>MD number | **TRAFO_TYPE_1 TRAFO_TYPE_5**<br>**TRAFO_TYPE_2 TRAFO_TYPE_6**<br>**TRAFO_TYPE_3 TRAFO_TYPE_7**<br>**TRAFO_TYPE_4 TRAFO_TYPE_8**<br>Definition of transformation 1/2/3/4/5/6/7/8 in channel | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: – |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>Power ON (up to SW 5.1) | Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1) | Unit: – |
| Data type: DWORD | Applies from SW 1.1 | |
| Meaning: | This MD specifies for each channel which transformation is available in the channel as the first (MD 24100), second (MD 24200), third (MD 24300) or eighth (MD 24460) transformation. | |
| MD irrelevant for ...... | If no transformation is installed. | |
| Figure see Section 1 / 2 | Figures: 1-1, 1-2, 1-3, 2-1. | |
| Application example(s) | See Chapter 6. | |
| Special cases, errors, ...... | See Section 2.5 | |

For the implemented kinematics, the following tables show the values to be set in MD TRAFO_TYPE_n (n = 1 ... 8) as decimal numbers.

| **5-axis transformations** | | | | |
|---|---|---|---|---|
| 1. Rotary axis | 2. Rotary axis | Movable tool<br>**TRAFO_TYPE** | Movable workpiece<br>**TRAFO_TYPE** | Movable tool and workpiece<br>**TRAFO_TYPE** |
| A | B | 16 | 32 | 48 |
| A | C | | 33 | 49 |
| B | A | 18 | 34 | 50 |
| B | C | | 35 | 51 |
| C | A | 20 | | |
| C | B | 21 | | |
| General 5-axis transformations (SW 5.2 and higher) | | 24 | 40 | 56 |

| **3-axis and 4-axis transformations** | | | | |
|---|---|---|---|---|
| Rotary axis | Orientation plane | Movable tool<br>**TRAFO_TYPE** | Movable workpiece<br>**TRAFO_TYPE** | Tool orientation in zero position |
| A | Y – Z | 16 | | Z |
| B | X – Z | 18 | | Z |
| C | X – Y | 20 | | Y |
| C | X – Y | 21 | | X |
| A | Y – Z | | 32, 33 | Z |
| B | X – Z | | 34, 35 | Z |
| C | X – Y | | 36 | Z |

| Universal milling head | | | |
|---|---|---|---|
| 1. Rotary axis | 2. Rotary axis | Movable tool, **TRAFO_TYPE** | Tool orientation in zero position |
| A | B’ | 128 | X |
| A | B’ | 136 | Y |
| A | C’ | 129 | X |
| A | C’ | 145 | Z |
| B | A’ | 130 | X |
| B | A’ | 138 | Y |
| B | C’ | 139 | Y |
| B | C’ | 147 | Z |
| C | A’ | 132 | X |
| C | A’ | 148 | Z |
| C | B’ | 141 | Y |
| C | B’ | 149 | Z |

| Swiveled linear axis | | | |
|---|---|---|---|
| 1. Rotary axis | 2. Rotary axis | Swiveled linear axis | **TRAFO_TYPE** |
| A | B | Z | 64 |
| A | C | Y | 65 |
| B | A | Z | 66 |
| B | C | X | 67 |
| C | A | Y | 68 |
| C | B | X | 69 |

| Further transformations | |
|---|---|
| **Group of transformation** | **TRAFO_TYPE** |
| Transmit | 256 |
| Cylinder lateral surface | 512 |
| | 513 |
| Inclined axis | 1024 |
| Centerless grinding | 2048 |
| OEM transformation | from 4096 to 4098 |
| Linked transformation | 8192 |

| | | | | | |
|---|---|---|---|---|---|
| **24110, 24432**<br>**24210, 24442**<br>**24310, 24452**<br>**24410, 24462**<br>MD number | **TRAFO_AXES_IN_1[n] TRAFO_AXES_IN_5[n]**<br>**TRAFO_AXES_IN_2[n] TRAFO_AXES_IN_6[n]**<br>**TRAFO_AXES_IN_3[n] TRAFO_AXES_IN_7[n]**<br>**TRAFO_AXES_IN_4[n] TRAFO_AXES_IN_8[n]**<br>Axis assignment for transformation 1/2/3/4/5/6/7/8 [axis index]: 0 ... 7 | | | | |
| Default setting: 1,2,3,4,5,0,0,... | | Min. input limit: 0 | | Max. input limit: 10 [max. number of channels] | |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1) | | | Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1) | | Unit: – |
| Data type: Byte | | | Applies from SW 1.1 | | |
| Meaning: | This MD sets the axis assignment at the input of the first (TRAFO_AXES_IN_1), second (TRAFO_AXES_IN_2), third (TRAFO_AXES_IN_3) up to eighth transformation (TRAFO_AXES_IN_8).<br><br>The index entered at the nth position specifies which axis is imaged internally by the transformation onto axis n.<br>The numbers of the channel axes must be specified. This is important for multi-channel systems. | | | | |
| MD irrelevant for ...... | If no transformation is installed. | | | | |
| Application example(s) | The default for a machine with 6 axes is, for example, 1 2 3 4 5 6.<br>1 2 3 5 4 6 is then possible. The fourth axis is imaged on the fifth and the fifth imaged on the fourth axis. | | | | |
| Special cases, errors, ...... | Entry 0: Axis does not participate in the transformation. | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **24120, 24434**<br>**24220, 24444**<br>**24320, 24454**<br>**24420, 24464**<br>MD number | **TRAFO_GEOAX_ASSIGN_TAB_1[n] TRAFO_GEOAX_ASSIGN_TAB_5[n]**<br>**TRAFO_GEOAX_ASSIGN_TAB_2[n] TRAFO_GEOAX_ASSIGN_TAB_6[n]**<br>**TRAFO_GEOAX_ASSIGN_TAB_3[n] TRAFO_GEOAX_ASSIGN_TAB_7[n]**<br>**TRAFO_GEOAX_ASSIGN_TAB_4[n] TRAFO_GEOAX_ASSIGN_TAB_8[n]**<br>Assignment of geometry axes to channel axes with transformation 1/2/3/4/5/6/7/8 [geometry axis number]:0 ... 2. | | | | |
| Default setting: 1, 2, 3 | | Min. input limit: 0 | | Max. input limit: 8 | |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1) | | | Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1) | | Unit: – |
| Data type: Byte | | | Applies from SW 2.0 | | |
| Meaning: | The geometry axes are assigned to the channel axes for transformations in this table. | | | | |
| MD irrelevant for ...... | If no transformation is installed. | | | | |
| Related to .... | as for MD 20050: AXCONF_GEOAX_ASSIGN_TAB, but only when transformation is active | | | | |

| | | | | |
|---|---|---|---|---|
| **24500**<br>**24600**<br>MD number | **TRAFO5_PART_OFFSET_1[n]**<br>**TRAFO5_PART_OFFSET_2[n]**<br>Offset vector of 5-axis transformation 1/2 [axis number]: 0 ... 2 | | | |
| Default setting: 0.0 | Min. input limit: | | Max. input limit: | |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1) | | Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1) | | Unit: mm |
| Data type: DOUBLE | | | Applies from SW 1.1 | |
| Meaning: | This machine data defines an offset of the workpiece carrier for the first (MD: TRAFO5_PART_OFFSET_1) or second (MD: TRAFO5_PART_OFFSET_2) 5-axis transformation of a channel and has a specific meaning for the different machine types:<br>Machine type 1 (two-axis swivel head for tool):<br>Vector from machine reference point to zero point of workpiece table. This will generally be a zero vector if both coincide.<br>Machine type 2 (two-axis rotary table for workpiece):<br>Vector from second joint of workpiece rotary table to zero point of table.<br>Machine type 3 (single-axis rotary table for workpiece and single-axis swivel head for tool):<br>Vector from joint of workpiece table to zero point of table. | | | |
| MD irrelevant for ...... | If no transformation is installed. | | | |
| Figure see Subsection 2.1.3 | Example of machine type 2: See Fig. 2-2 | | | |

| | | | | |
|---|---|---|---|---|
| **24510**<br>**24610**<br>MD number | **TRAFO5_ROT_AX_OFFSET_1[n]**<br>**TRAFO5_ROT_AX_OFFSET_2[n]**<br>Position offset of rotary axis 1/2 for 5-axis transformation 1/2 [axis number]: 0 ... 1 | | | |
| Default setting: 0.0 | Min. input limit: | | Max. input limit: | |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1) | | Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1) | | Unit: Degrees |
| Data type: DOUBLE | | | Applies from SW 1.1 | |
| Meaning: | This machine data defines the angular offset of the first or second rotary axis in degrees for the first (MD: TRAFO5-ROT_AX_OFFSET_1) or the second 5-axis transformation (MD: TRAFO5_ROT_AX_OFFSET_2) of a channel. | | | |
| MD irrelevant for ...... | If no transformation is installed. | | | |

| | | | | | |
|---|---|---|---|---|---|
| **24520**<br>**24620**<br>MD number | **TRAFO5_ROT_SIGN_IS_PLUS_1[n]**<br>**TRAFO5_ROT_SIGN_IS_PLUS_2[n]**<br>Sign of rotary axis 1/2 for 5-axis transformation 1/2[axis number]: 0 ... 1 | | | | |
| Default setting: 1 | | Min. input limit: 0 | | Max. input limit: 1 | |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1) | | | Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1) | | Unit: – |
| Data type: BOOLEAN | | | | Applies from SW 1.1 | |
| Meaning: | This machine data defines the sign with which the two rotary axes are included in the first (MD: TRAFO5_ROT_SIGN_IS_PLUS_1) or<br>the second (MD: TRAFO5-ROT_SIGN_IS_PLUS_2) 5-axis transformation of a channel.<br>MD = 0 (FALSE): Sign is reversed.<br>MD = 1 (TRUE): Sign is not reversed and the traversing direction is defined according to AX_MOTION_DIR.<br>This machine data does not mean that the rotational direction of the rotary axis concerned is to be reversed, but specifies whether its motion is in the mathematically positive or negative direction when the axis is moving in the positive direction.<br>The result of a change to this data is not therefore a change in the rotational direction, but a change in the compensatory motion of the linear axes.<br>However, if a directional vector and thus, implicitly, a compensatory motion is specified, the result is a change in the rotational direction of the rotary axis concerned.<br>On a real machine, therefore, the machine data may be set to FALSE (or zero) only if the rotary axis is turning in an anti-clockwise direction when moving in a positive direction. | | | | |
| MD irrelevant for ...... | If no transformation is installed. | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **24530**<br>**24630**<br>MD number | **TRAFO5_NON_POLE_LIMIT_1**<br>**TRAFO5_NON_POLE_LIMIT_2**<br>Definition of pole range for 5-axis transformation 1/2 | | | | |
| Default setting: 2 | | Min. input limit: | | Max. input limit: | |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1) | | | Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1) | | Unit: Degrees |
| Data type: DOUBLE | | | | Applies from SW 1.1 | |
| Meaning: | This MD defines a limit angle for the fifth axis of the first (MD: TRAFO5_NON_POLE_LIMIT_1) or the second (MD: TRAFO5_NON_POLE_LIMIT_2) 5-axis transformation with the following characteristics: If the path runs past the pole at an angle lower than the value set here, it crosses through the pole.<br>With the 5-axis transformation, a coordinate system consisting of circles of longitude and latitude is spanned over a spherical surface by the two orientation axes of the tool.<br>If, as a result of orientation programming (i.e. the orientation vector is positioned on one plane), the path passes so close to the pole that the angle is less than the value defined in this MD, then a deviation from the specified interpolation is made such that the interpolation passes through the pole.<br>If this path modification results in a deviation which is greater than a tolerance defined by means of MD 24540/24640: TRAFO5_POLE_LIMIT, then alarm 14112 is output. | | | | |
| MD irrelevant for ...... | If no transformation is installed.<br>Also irrelevant for programming in machine coordinate system ORIMKS. | | | | |
| See Chapter 2 for Figure | Fig. 2-9 shows how this MD is used. | | | | |
| Related to .... | MD: TRAFO5_POLE_LIMIT_1 or _2 | | | | |

| | | | | |
|---|---|---|---|---|
| **24540**<br>**24640**<br>MD number | **TRAFO5_POLE_LIMIT_1**<br>**TRAFO5_POLE_LIMIT_2**<br>End angle tolerance with interpolation through pole for 5-axis transformation 1/2 | | | |
| Default setting: 2 | | Min. input limit: | | Max. input limit: |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1) | | | Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1) | Unit: Degrees |
| Data type: Double | | | Applies from SW 1.1 | |
| Meaning: | This MD defines an end angle tolerance for the fifth axis of the first (MD: TRAFO5_POLE_LIMIT_1) or the second (MD: TRAFO5_POLE_LIMIT_2) 5-axis transformation with the following characteristics:<br>With interpolation through the pole point, only the fifth axis moves; the fourth axis remains in its start position. If a movement is programmed which does not pass exactly through the pole point, but is to pass within the tolerance defined by TRAFO5_NON_POLE_LIMIT in the vicinity of the pole, a deviation is made from the specified path because the interpolation runs exactly through the pole point. As a result, the position at the end point of the fourth axis (pole axis) deviates from the programmed value.<br>This MD specifies the angle by which the pole axis may deviate from the programmed value with a 5-axis transformation if a switchover is made from the programmed interpolation to interpolation through the pole point. In the case of a greater deviation, an error message (alarm 14112) is output and the interpolation is not executed. | | | |
| MD irrelevant for ...... | If no transformation is installed.<br>Also irrelevant for programming in machine coordinate system ORIMKS. | | | |
| See Chapter 2 for Figure | Fig. 2-9 shows how this MD is used. | | | |
| Related to .... | MD: TRAFO5_NON_POLE_LIMIT_1 or _2 | | | |

| | | | | |
|---|---|---|---|---|
| **24550**<br>**24650**<br>MD number | **TRAFO5_BASE_TOOL_1[n]**<br>**TRAFO5_BASE_TOOL_2[n]**<br>Vector of base tool for activation of 5-axis transformation 1/2 [axis number]: 0 ... 2 | | | |
| Default setting: 0.0 | | Min. input limit: | | Max. input limit: |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1) | | | Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1) | Unit: mm |
| Data type: DOUBLE | | | Applies from SW 1.1 | |
| Meaning: | This MD specifies the vector of the base tool which takes effect when the first transformation (MD: TRAFO5_BASE_TOOL_1) or the second (MD: TRAFO5_BASE_TOOL_2) is activated when a length compensation is not selected.<br>Programmed length compensations have an additive effect with respect to the base tool. | | | |
| MD irrelevant for ...... | If no transformation is installed. | | | |

<table>
<tr><td colspan="2">24560<br>24660<br>MD number</td><td colspan="4">TRAFO5_JOINT_OFFSET_1[n]<br>TRAFO5_JOINT_OFFSET_2[n]<br>Vector of kinematic offset of 5-axis transformation 1/2 [axis number]: 0 ... 2</td></tr>
<tr><td colspan="2">Default setting: 0.0</td><td colspan="2">Min. input limit:</td><td colspan="2">Max. input limit:</td></tr>
<tr><td colspan="2">Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1)</td><td colspan="2">Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1)</td><td colspan="2">Unit: mm</td></tr>
<tr><td colspan="3">Data type: DOUBLE</td><td colspan="3">Applies from SW 1.1</td></tr>
<tr><td colspan="2">Meaning:</td><td colspan="4">This machine data defines the vector from the first to the second joint for the first (MD: TRAFO5_JOINT_OFFSET_1) or second (MD: TRAFO5_JOINT_OFFST_2) transformation of a channel and has a specific meaning for the different machine types:<br>Machine type 1 (two-axis swivel head for tool) and<br>machine type 2 (two-axis rotary table for tool):<br>Vector from first to second joint of tool rotary head or workpiece rotary table.<br>Machine type 3 (single-axis rotary table for workpiece and single-axis swivel head for tool):<br>Vector from machine reference point to joint of workpiece table.</td></tr>
<tr><td colspan="2">MD irrelevant for ......</td><td colspan="4">If no transformation is installed.The same applies for 3-axis and 4-axis transformation.</td></tr>
</table>

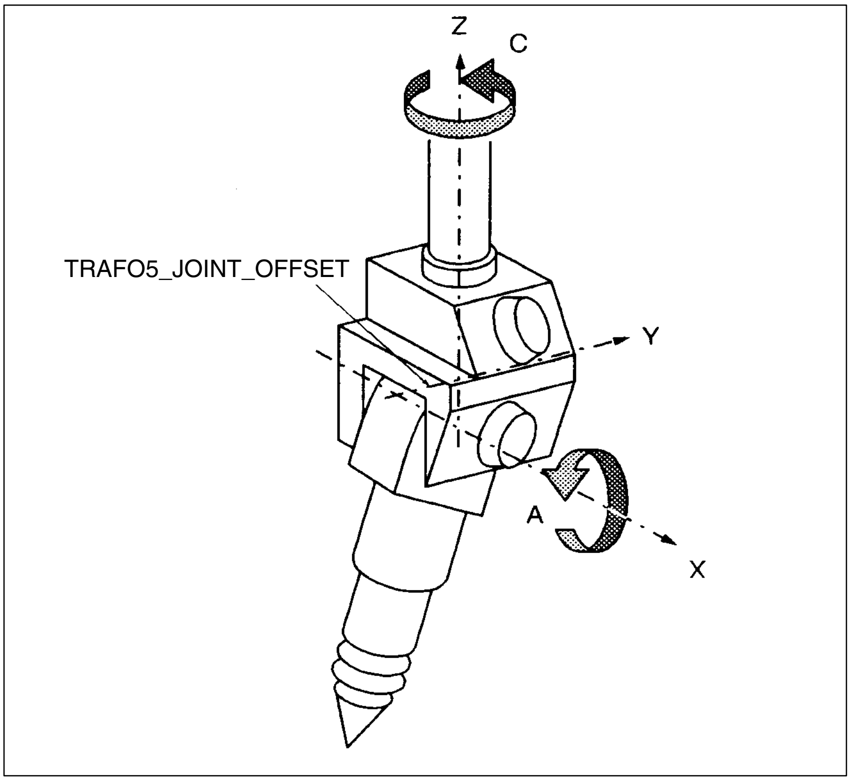



Fig. 4-1 Example of MD: TRAFO5_JOINT_OFFSET (joint offset for a 5-axis machine with two-axis swivel head for tools) (e.g. laser machining with machine type 1)

### 4.2.1 Channel-specific MD for swiveled linear axis

In addition to the machine data described in Section 4.2, the following machine data are required for the 5-axis transformation “swiveling linear axis”.

| | | | |
|---|---|---|---|
| **24562**<br>MD number | **TRAFO5_TOOL_ROT_AX_OFFSET_1**<br>Offset of focus of 1st 5-axis transformation with swiveled linear axis. | | |
| Default setting: 0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – | |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1) | Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1) | Unit: Degrees | |
| Data type: DOUBLE | Applies from SW 3.2 | | |
| Meaning: | In the case of 5-axis transformation with swiveled linear axis, the value indicates the offset of the rotary axis which swivels the linear axis with reference to machine zero for the 1st swiveled transformation. | | |
| MD irrelevant for ...... | Other 5-axis transformations | | |
| Diagram | 2-10 | | |
| Related to .... | 24662 | | |

| | | | |
|---|---|---|---|
| **24662**<br>MD number | **TRAFO5_TOOL_ROT_AX_OFFSET_2**<br>Offset of focus of 2nd 5-axis transformation with swiveled linear axis. | | |
| Default setting: 0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – | |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1) | Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1) | Unit: Degrees | |
| Data type: DOUBLE | Applies from SW 3.2 | | |
| Meaning: | In the case of 5-axis transformation with swiveled linear axis, the value indicates the offset of the rotary axis which swivels the linear axis with reference to machine zero for the 2nd swiveled transformation. | | |
| MD irrelevant for ...... | Other 5-axis transformations | | |
| Diagram | 2-10 | | |
| Related to .... | 24562 | | |

### 4.2.2 Channel-specific MD for universal milling head

| | | | |
|---|---|---|---|
| **24564**<br>MD number | **TRAFO5_NUTATOR_AX_ANGLE_1**<br>Angle of 2nd rotary axis and the corresponding axis in the rectangular coordinate system, 1st transform. | | |
| Default setting: 45 | Min. input limit: –89 | Max. input limit: 89 | |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1) | Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1) | Unit: Degrees | |
| Data type: DOUBLE | Applies from SW 3.1 | | |
| Meaning: | Angle between the second rotary axis and the corresponding axis in the rectangular coordinate system.<br>Valid for the first transformation of a channel. | | |
| MD irrelevant for ...... | Transformation type other than "universal milling head" | | |
| Application example(s) | 6.3 | | |
| Related to .... | TRAFO_TYPE_n | | |

| | | | |
|---|---|---|---|
| **24664**<br>MD number | **TRAFO5_NUTATOR_AX_ANGLE_2**<br>Angle of 2nd rotary axis and the corresponding axis in the rectangular coordinate system, 2nd transform. | | |
| Default setting: 45 | Min. input limit: –89 | Max. input limit: 89 | |
| Change effective after<br>NEWCONFIG (SW 5.2 and higher)<br>power ON (up to SW 5.1) | Protection level:<br>7 / 7 (SW 5.2 and higher)<br>2 / 7 (up to SW 5.1) | Unit: Degrees | |
| Data type: DOUBLE | Applies from SW 3.1 | | |
| Meaning: | Angle between the second rotary axis and the corresponding axis in the rectangular coordinate system.<br>Valid for the second transformation of a channel. | | |
| MD irrelevant for ...... | Transformation type other than "universal milling head" | | |
| Application example(s) | 6.3 | | |
| Related to .... | TRAFO_TYPE_n | | |

### 4.2.3 Channel-specific MD for orientation axes

| | | | | |
|---|---|---|---|---|
| **20621**<br>MD number | **HANDWH_ORIAX_MAX_INCR_SIZE**<br>Limitation of handwheel increment for orientation axes | | | |
| Default setting: 0.0 | | Min. input limit: 0.0 | | Max. input limit: – |
| Changes effective after RESET | | | Protection level: 2 / 7 | Unit: Degrees |
| Data type: DOUBLE | | | Applies from SW 5.3 | |
| Meaning: | Limitation of handwheel increment for orientation axes | | | |
| MD irrelevant for ...... | If no transformation is installed. | | | |

| | | | | |
|---|---|---|---|---|
| **20623**<br>MD number | **HANDWH_ORIAX_MAX_INCR_VSIZE**<br>Orientation velocity overlay | | | |
| Default setting: 0.1 | | Min. input limit: 0.0 | | Max. input limit: – |
| Changes effective after RESET | | | Protection level: 2 / 7 | Unit: rev/min |
| Data type: DOUBLE | | | Applies from SW 5.3 | |
| Meaning: | Orientation velocity overlay | | | |
| MD irrelevant for ...... | If no transformation is installed. | | | |

| | | | | |
|---|---|---|---|---|
| **21102**<br>MD number | **ORI_DEF_WITH_G_CODE**<br>Definition of orientation angles A2, B2, C2 | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 1 |
| Changes effective after power ON | | | Protection level: 2 / 7 | Unit: – |
| Data type: BYTE | | | Applies from SW 5.3 | |
| Meaning: | This machine data determines how orientation angles A2, B2, C2 are defined:<br>MD = 0 (FALSE): The angles are defined according to MD 21100 “ORIENTATION_IS_EULER”<br>MD = 1 (TRUE):<br>The angles are defined according to the G code (ORIEULER, ORIRPY, ORIVIRT1, ORIVIRT2) | | | |
| MD irrelevant for ...... | If no transformation is installed. | | | |

| | | | | |
|---|---|---|---|---|
| **21104**<br>MD number | **ORI_IPO_WITH_G_CODE**<br>Definition of interpolation type for orientation | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 1 |
| Changes effective after power ON | | | Protection level: 2 / 7 | Unit: – |
| Data type: BOOLEAN | | | Applies from SW 5.3 | |
| Meaning: | Definition of interpolation type for orientation<br>MD=0 (FALSE): The G codes ORIWKS and ORIMKS are the references<br>MD=1 (TRUE): The G codes ORIVECT and ORIAXIS are the references | | | |

| | |
|---|---|
| **21120**<br>**21130**<br>MD number | **ORIAX_TURN_TAB_1[n]**<br>**ORIAX_TURN_TAB_2[n]**<br>Assignment of rotations of orientation axes about the reference axes, definition 1 or definition 2 |
| Default setting: 1, 2, 3 | Min. input limit: 1 / Max. input limit: 3 |
| Changes effective after power ON | Protection level: 2 / 7 / Unit: – |
| Data type: Byte | Applies from SW 5.3 |
| Meaning: | Assignment of rotations of orientation axes about the reference axes, definition 1 or definition 2<br>1 = rotation about 1st reference axis (X)<br>2 = rotation about 2nd reference axis (Y)<br>3 = rotation about 3rd reference axis (Z) |
| MD irrelevant for ...... | If no transformation is installed. |
| Application example(s) | See Section 6.4 |

Assuming that the axes are mutually perpendicular, it is possible to obtain an orientation definition which corresponds to the orientation defined by the RPY angles or the Euler angles. As a result, there are 12 ways in which an orientation can be specified.
If another axis assignment is programmed, alarm “Configuring axes are incorrectly configured” is issued.

| **Definition analogous to RPY angles** | | | **Definition analogous to Euler angles** | | |
|---|---|---|---|---|---|
| 1st rotation | 2nd rotation | 3rd rotation | 1st rotation | 2nd rotation | 3rd rotation |
| X | Y | Z | X | Y | X |
| X | Z | Y | X | Z | X |
| Y | X | Z | Y | X | Y |
| Y | Z | X | Y | Z | Y |
| Z | X | Y | Z | X | Z |
| Z | Y | X | Z | Y | Z |

(light shading) corresponds to RPY angle NC (dark shading) corresponds to Euler angle NC

| | |
|---|---|
| **21150**<br>MD number | **JOG_VELO_RAPID_ORI[n]**<br>Rapid traverse in jog mode for orientation axes in the channel |
| Default setting: 10.0, 10.0, 10.0 | Min. input limit: 0.0 / Max. input limit: – |
| Changes effective after RESET | Protection level: 2 / 7 / Unit: rev/min |
| Data type: DOUBLE | Applies from SW 5.3 |
| Meaning: | Rapid traverse in jog mode for orientation axes in the channel |
| MD irrelevant for ...... | If no transformation is installed. |

| | |
|---|---|
| **21155**<br>MD number | **JOG_VELO_ORI[n]**<br>Orientation axis velocity in jog mode |
| Default setting: 2.0, 2.0, 2.0 | Min. input limit: 0.0 / Max. input limit: – |
| Changes effective after RESET | Protection level: 2 / 7 / Unit: rev/min |
| Data type: DOUBLE | Applies from SW 5.3 |
| Meaning: | Orientation axis velocity in jog mode |
| MD irrelevant for ...... | If no transformation is installed. |

| | | | | | |
|---|---|---|---|---|---|
| **21160**<br>MD number | **JOG_VELO_RAPID_GEO[n]**<br>Rapid traverse in jog mode for geometry axes in the channel | | | | |
| Default setting:<br>10000./60.,10000./60.,10000./60., | | Min. input limit: 0.0 | | Max. input limit: – | |
| Changes effective after RESET | | | Protection level: 2 / 7 | | Unit: mm/min |
| Data type: DOUBLE | | | | Applies from SW 5 | |
| Meaning: | Rapid traverse in jog mode for geometry axes in the channel | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **21165**<br>MD number | **JOG_VELO_GEO[n]**<br>Geometry axis velocity in jog mode | | | | |
| Default setting:<br>2000./60.,2000./60.,2000./60., | | Min. input limit: 0.0 | | Max. input limit: – | |
| Changes effective after RESET | | | Protection level: 2 / 7 | | Unit: mm/min |
| Data type: DOUBLE | | | | Applies from SW 5 | |
| Meaning: | Geometry axis velocity in jog mode | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **21170**<br>MD number | **ACCEL_ORI[n]**<br>Acceleration for orientation axes | | | | |
| Default setting:<br>{2.0, 2.0, 2.0}, ... | | Min. input limit: 0.0 | | Max. input limit: – | |
| Changes effective after RESET | | | Protection level: 2 / 7 | | Unit: $U/s^2$ |
| Data type: DOUBLE | | | | Applies from SW 5.3 | |
| Meaning: | The acceleration for the orientation axes can be set by means of this MD. | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **24580**<br>**24680**<br>MD number | **TRAFO5_TOOL_VECTOR_1**<br>**TRAFO5_TOOL_VECTOR_2**<br>Direction of tool vector for orientation transformation 1 or 2 | | | | |
| Default setting: 2,2,2,... | | Min. input limit: 0 | | Max. input limit: 2 | |
| Changes effective after power ON | | | Protection level: 2 / 7 | | Unit: – |
| Data type: Byte | | | | Applies from SW 5.1 | |
| Meaning: | This machine data defines the direction of the tool vector for orientation transformation 1 or 2:<br>0 = tool vector in X direction<br>1 = tool vector in Y direction<br>2 = tool vector in Z direction | | | | |
| MD irrelevant for ...... | If no transformation is installed. | | | | |
| Application example(s) | See Section 6.4 | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **24585**<br>**24685**<br>MD number | **TRAFO5_ORIAX_ASSIGN_TAB_1[n]**<br>**TRAFO5_ORIAX_ASSIGN_TAB_2[n]**<br>Assignment of orientation axes to channel axes | | | | |
| Default setting: 0, 0, 0 | | Min. input limit: 0 | | Max. input limit: 18 | |
| Changes effective after power ON | | | Protection level: 2 / 7 | | Unit: – |
| Data type: Byte | | | | Applies from SW 5.1 | |
| Meaning: | Assignment of orientation axes to channel axes for orientation transformation 1 or 2 | | | | |
| MD irrelevant for ...... | If no transformation is installed. | | | | |
| Application example(s) | See Section 6.4 | | | | |

## 4.2.4 MD and SD Cartesian manual travel (SW 6.3 and higher)

| **21106**<br>MD number | **CART_JOG_SYSTEM**<br>Coordinate system for Cartesian JOG | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 7 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | Applies from SW 6.3 | |
| Meaning: | This machine data has two different meanings. On the one hand, it is used to activate the function Cartesian manual travel. On the other, it can be used to specify the referencing systems for switchover.<br>The meanings of the individual bits is defined as follows:<br><br>The meanings of the individual bits is defined as follows:<br>Bit 0: Basic Coordinate System (BCS)<br>Bit 1 : Workpiece Coordinate System (WCS)<br>Bit 2 : Tool Coordinate System (TCS)<br><br>If no bit is set, setting data SD 42650: CART_JOG_MODE is not interpreted. Traversing is carried out as before in JOG mode.<br><br>SD 42650: CART_JOG_MODE can only be used to set the reference system for which the bits are set in MD 21106: CART_JOG_SYSTEM.<br>The HMI can use this machine data to decide which switchover options are offered for the individual coordinate systems. | |
| MD irrelevant for ...... | Transformation type not equal to “Transformation package handling” | |

| **42650**<br>MD number | **CART_JOG_MODE**<br>Coordinate system for Cartesian manual travel | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 0x0404 |
| Modification valid IMMEDIATELY | Protection level: 7 / 7 | Unit: – |
| Data type: DWORD | Applies from SW 6.3 | |
| Meaning: | This allows the reference coordinate system to be set for Cartesian manual travel.<br>Bits 0 to 7 are provided for selecting the coordinate system for the translation,<br>Bits 8 to 15 for selecting the coordinate system for the orientation.<br><br>If no bit is set, or only one bit either for the translation or for the orientation, the Cartesian manual travel is not active. This means that one bit must always be set for the translation and one bit set for the orientation. If more than one bit is set for the translation or for the orientation, the Cartesian manual travel is not active either.<br><br>The meanings of the individual bits is defined as follows:<br>Bit 0: Translation in the Basic Coordinate System (BCS)<br>Bit 1 : Translation in the Workpiece Coordinate System (WCS)<br>Bit 2 : Translation in the Tool Coordinate System (TCS)<br>Bit 3 : reserved<br>Bit 4 : reserved<br>Bit 5 : reserved<br>Bit 6 : reserved<br>Bit 7 : reserved<br>Bit 8 : Orientation in the Basic Coordinate System (BCS)<br>Bit 9 : Orientation in the Workpiece Coordinate System (WCS)<br>Bit 10 : Orientation in the Tool Coordinate System (TCS)<br>Bit 11 : reserved<br>Bit 12 : reserved<br>Bit 13 : reserved<br>Bit 14 : reserved<br>Bit 15 : reserved | |
| MD irrelevant for ...... | Transformation type not equal to “Transformation package handling” | |

### 4.2.5 Channel-specific MD for Cartesian point to point travel

| | | | | |
|---|---|---|---|---|
| **20150**<br>MD number | **GCODE_RESET_VALUES[n]** 0 up to max. no. of G codes-1<br>Initial setting of G group 49 | | | |
| Default setting: – | Min. input limit: – | | Max. input limit: – | |
| Changes effective after RESET | | Protection level: 2 / 7 | | Unit: – |
| Data type: Byte | | Applies from SW 5 | | |
| Meaning: | This MD defines whether PTP or CP is active after a RESET/end of parts program.<br>MD=1: CP is active after RESET<br>MD=2: PTP is active after RESET | | | |
| Related to .... | MD $MC_GCODE_RESET_MODE[48] | | | |

| | | | | |
|---|---|---|---|---|
| **20152**<br>MD number | **GCODE_RESET_MODE[n]** Max. no. of G codes-1<br>Setting and response after reset and end of parts program in G group 49 | | | |
| Default setting: – | Min. input limit: – | | Max. input limit: – | |
| Changes effective after RESET | | Protection level: 2 / 7 | | Unit: – |
| Data type: Byte | | Applies from SW 5 | | |
| Meaning: | This MD can be programmed to influence the setting after RESET and end of parts program.<br>MD=0:<br>A separate value is programmed for each entry in MD GCODE_RESET_VALUES (i.e. for each G group) to determine whether the setting is to be made according to MD GCODE_RESET_VALUES in the event of a RESET/end of parts program.<br>MD=1:<br>The current setting remains valid after a RESET/end of parts program. | | | |
| Related to .... | MD GCODE_RESET_VALUES[48] | | | |

### 4.2.6 Machine data for generic 5-axis transformation

| | | | | |
|---|---|---|---|---|
| **21180**<br>MD number | **ROT_AX_SWL_CHECK_MODE**<br>Check software limits for orientation axes | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 2 |
| Modification valid IMMEDIATELY | | | Protection level: 7 / 7 | Unit: – |
| Data type: DWORD | | | Applies from SW 6.1 | |
| Meaning: | This machine data is only evaluated for generic 5-axis transformation. It determines how the motion of the rotary axes is modified when the direction is programmed if the system detects during block preparation that the programmed path would trigger a violation of the software limits.<br><br>The machine data can have the following three values:<br><br>0: No path modification takes place. If a movement across the shortest path is not possible, alarm 10620 or 10720 (SW–LIMITSWITSCH) is output.<br><br>1: If the initial orientation path violates the limits of the orientation axes, the system attempts to alter the end point in order to enable a movement. The system then initially attempts to use the second solution. (Two solutions virtually always result from the orientation ==> axis angle conversion). If this solution also violates the axis limits, the system attempts to find an acceptable solution by modifying both rotary axes in both solutions by multiples of 360 degrees. The end position modifications described above are only performed if the axis interpolation of the rotary axis is active.<br><br>0: The monitoring response and resulting modifications of the rotary axis positions are the same as for value 1 of the machine data. However, modifications are also accepted if vector interpolation (large circle interpolation, conical interpolation, etc.) is active. If a modification of the rotary axis positions is necessary in such a case, the system switches to axis interpolation. The originally programmed orientation path is then usually no longer followed. | | | |

| | | | | |
|---|---|---|---|---|
| **24570**<br>MD number | **TRAFO5_AXIS1_1**<br>Vector for the first rotary axis and the first orientation transformation | | | |
| Default setting: – | | Min. input limit: – | | Max. input limit: – |
| Change effective after NEWCONFIG | | | Protection level: 7 / 7 | Unit: – |
| Data type: DOUBLE | | | Applies from SW 5.2 | |
| Meaning: | Specifies the vector which describes the direction of the first rotary axis when a general 5-axis transformation is the first orientation transformation in the system (TRAFO_TYPE_* = 24, 40, 56).<br>Any value, but $\neq 0$<br>Example: The same axis (in the direction of the 2nd geometry axis, i.e. generally Y) is described by both (0,1,0) and (0, 7.21, 0). Geometry axis, i.e. described in rule Y). | | | |
| Related to .... | 24572, 24670, 24672 | | | |

| | | | | |
|---|---|---|---|---|
| **24572**<br>MD number | **TRAFO5_AXIS2_1**<br>Vector for the first rotary axis and the first orientation transformation | | | |
| Default setting: – | | Min. input limit: – | | Max. input limit: – |
| Change effective after NEWCONFIG | | | Protection level: 7 / 7 | Unit: – |
| Data type: DOUBLE | | | Applies from SW 5.2 | |
| Meaning: | Same as **TRAFO5_AXIS1_1**, but for the second axis | | | |
| Related to .... | 24570, 24670, 24672 | | | |

| | | | | | |
|---|---|---|---|---|---|
| **24574**<br>MD number | **TRAFO5_BASE_ORIENT_1**<br>Basic orientation for the first orientation transformation | | | | |
| Default setting: – | | Min. input limit: – | | Max. input limit: – | |
| Change effective after NEWCONFIG | | | Protection level: 7 / 7 | | Unit: – |
| Data type: DOUBLE | | | | Applies from SW 5.2 | |
| Meaning: | Describes the basic orientation applied as the basis for the transformation when the general 5-axis transformation is the first orientation in the system (TRAFO_TYPE_* = 24, 40, 56) in cases where the basic orientation has not been defined through activation of a transformation or by means of a tool.<br>The value of the vector is freely selectable, but must not equal zero. | | | | |
| Related to .... | 24674 | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **24670**<br>MD number | **TRAFO5_AXIS1_2**<br>Vector for the first rotary axis and the second orientation transformation | | | | |
| Default setting: – | | Min. input limit: – | | Max. input limit: – | |
| Change effective after NEWCONFIG | | | Protection level: 7 / 7 | | Unit: – |
| Data type: DOUBLE | | | | Applies from SW 5.2 | |
| Meaning: | Same as **TRAFO5_AXIS1_1**, but for the second orientation transformation in the channel | | | | |
| Related to .... | 24570, 24572, 24672 | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **24672**<br>MD number | **TRAFO5_AXIS2_2**<br>Vector for the second rotary axis and the second orientation transformation | | | | |
| Default setting: – | | Min. input limit: – | | Max. input limit: – | |
| Change effective after NEWCONFIG | | | Protection level: 7 / 7 | | Unit: – |
| Data type: DOUBLE | | | | Applies from SW 5.2 | |
| Meaning: | Same as **TRAFO5_AXIS2_1**, but for the second orientation transformation in the channel | | | | |
| Related to .... | 24570, 24572, 24670 | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **24674**<br>MD number | **TRAFO5_BASE_ORIENT_2**<br>Basic orientation for the second orientation transformation | | | | |
| Default setting: – | | Min. input limit: – | | Max. input limit: – | |
| Change effective after NEWCONFIG | | | Protection level: 7 / 7 | | Unit: – |
| Data type: DOUBLE | | | | Applies from SW 5.2 | |
| Meaning: | Same as **TRAFO5_BASE_ORIENT_1**, but for the second orientation transformation in the system | | | | |
| Related to .... | 24574 | | | | |

## 4.2.7 MD and SD online tool length offset (SW 6.4)

| | | | | | |
|---|---|---|---|---|---|
| **21190**<br>MD number | **TOFF_MODE**<br>Effect of online offset in tool direction | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: – | |
| Changes effective after RESET | | | Protection level: 2 / 7 | | Unit: 0F HEX |
| Data type: BYTE | | | | Applies from SW 6.4 | |
| Meaning: | This machine data is used to set the effect of online offset in the tool direction via $AA_TOFF[ ].<br><br>Bit 0: Response of $AA_TOFF on RESET<br>Bit 0 = 0 $AA_TOFF is deselected on RESET<br>Bit 0 = 1 $AA_TOFF is retained after RESET<br>Bit 1: Effect of value assignment to 1st component of $AA_TOFF[ ]<br>Bit 1 = 0 absolute value<br>Bit 1 = 1 incremental value (integrator)<br>Bit 2: Effect of value assignment to 2nd component of $AA_TOFF[ ]<br>Bit 2 = 0 absolute value<br>Bit 2 = 1 incremental value (integrator)<br>Bit 3: Effect of value assignment to 3rd component of $AA_TOFF[ ]<br>Bit 1 = 0 absolute value<br>Bit 1 = 1 incremental value (integrator) | | | | |
| Related to .... | MD 21194: TOFF_VELO velocity of online offset in tool direction<br>MD 21196: TOFF_ACCEL acceleration of online offset in tool direction | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **21194**<br>MD number | **TOFF_VELO**<br>Velocity of online offset in tool direction | | | | |
| Default setting: 0 | | Min. input limit: ≥ 0 | | Max. input limit: – | |
| Change effective after NEWCONFIG | | | Protection level: 2 / 7 | | Unit: mm/min |
| Data type: DOUBLE | | | | Applies from SW 6.4 | |
| Meaning: | Velocity of online offset in tool direction [mm/min] via $AA_TOFF | | | | |
| Related to .... | MD 21190: TOFF_MODE effect of online offset in tool direction<br>MD 21196: TOFF_ACCEL acceleration of online offset in tool direction | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **21196**<br>MD number | **TOFF_ACCEL**<br>Acceleration of online offset in tool direction | | | | |
| Default setting: 100 | | Min. input limit: 0,001 | | Max. input limit: – | |
| Change effective after NEWCONFIG | | | Protection level: 2 / 7 | | Unit: $m/s^2$ |
| Data type: DOUBLE | | | | Applies from SW 6.4 | |
| Meaning: | Acceleration for online offset in tool direction [$m/s^2$] via $AA_TOFF | | | | |
| Related to .... | MD 21190: TOFF_MODE effect of online offset in tool direction<br>MD 21194: TOFF_ACCEL Velocity for online offset in tool direction | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **42970**<br>MD number | **TOFF_LIMIT**<br>**U**pper limit for offset value $AA_TOFF | | | | |
| Default setting: 100000000.0 | | Min. input limit: ≥ 0 | | Max. input limit: 100000000.0 | |
| Modification valid IMMEDIATELY | | | Protection level: 2 / 7 | | Unit: mm/inches |
| Data type: DOUBLE | | | | Applies from SW 6.4 | |
| Meaning: | Upper limit of offset value, which can be specified using synchronous actions via $AA_TOFF.<br>This limit applies to the absolute effective offset value through $AA_TOFF. System variable $AA_TOFF_LIMIT can be used to query whether the offset value is close to the limit. | | | | |

### 4.2.8 MD and SD compression of orientation (SW 6.3)

| | | | | | |
|---|---|---|---|---|---|
| **20482**<br>MD number | **COMPRESSOR_MODE**<br>Mode of the compressor | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 3 | |
| Changes effective after NEW CONF | | | Protection level: | | Unit: – |
| Data type: BYTE | | | | Applies from SW 6.3 | |
| Meaning: | This MD can be used to set the compressor mode.<br>The following options are available:<br><br>0: The tolerances defined by the axis-specific MD 33100: COMPRESS_POS_TOL are maintained on all axes (geometry and orientation axes) for the compressor.<br><br>1: For the compressor, the contour tolerance specified by setting data SD 42475: COMPRESS_CONTUR_TOL is active for the geometry axes. For the orientation axes, the axis-specific tolerances from MD 33100: COMPRESS_POS_TOL are active.<br><br>2: For the compressor, the axis-specific tolerances for the geometry axes from MD 33100: COMPRESS_POS_TOL are active. The orientation movement is compressed, allowing for the maximum angle displacement defined in setting data SD 42476: COMPRESS_ORI_TOL and/or SD 42477: COMPRESS_ORI_ROT_TOL.<br><br>3: For the compressor, the contour tolerance specified by setting data SD 42475: COMPRESS_CONTUR_TOL for the geometry axes and the maximum angular displacement specified by SD 42476: COMPRESS_ORI_TOL and/or SD 42477: COMPRESS_ORI_ROT_TOL for the orientation axes is active. | | | | |
| Related to .... | MD 33100: COMPRESS_POS_TOL, SD 42475: COMPRESS_CONTUR_TOL,<br>SD 42476: COMPRESS_ORI_TOL, SD 42477: COMPRESS_ORI_ROT_TOL | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **42475**<br>MD number | **COMPRESS_CONTOUR_TOL**<br>Max. contour deviation for compressor | | | | |
| Default setting: 0.05 | | Min. input limit: 0.000001 | | Max. input limit: 999999 | |
| Modification valid IMMEDIATELY | | | Protection level: | | Unit: mm |
| Data type: DOUBLE | | | | Applies from SW 6.3 | |
| Meaning: | The maximum tolerance for the contour is defined by this setting data for the compressor. | | | | |
| Related to .... | MD 20482: COMPRESSOR_MODE | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **42476**<br>MD number | **COMPRESS_ORI_TOL**<br>Max. angular displacement of tool orientation for the compressor | | | | |
| Default setting: 0.05 | | Min. input limit: 0.000001 | | Max. input limit: 90 | |
| Modification valid IMMEDIATELY | | | Protection level: | | Unit: Degrees |
| Data type: DOUBLE | | | | Applies from SW 6.3 | |
| Meaning: | This setting data is used to define the maximum tolerance for tool orientation for the compressor. The data determines the maximum legal angular displacement of the tool orientation.<br>The data is only effective if an orientation transformation is active. | | | | |
| Related to .... | MD 20482: COMPRESSOR_MODE | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **42477**<br>MD number | **COMPRESS_ORI_ROT_TOL**<br>Max. angular displacement of tool rotation for the compressor | | | | |
| Default setting: 0.05 | | Min. input limit: 0.000001 | | Max. input limit: 90 | |
| Modification valid IMMEDIATELY | | | Protection level: | | Unit: Degrees |
| Data type: DOUBLE | | | | Applies from SW 6.3 | |
| Meaning: | This setting data is used to define the maximum tolerance for rotation of the tool orientation for the compressor. The data determines the maximum legal angular displacement of the tool rotation.<br>The data is only effective if an orientation transformation is active.<br>A rotation of the tool orientation is only possible on 6-axis machines. | | | | |
| Related to .... | MD 20482: COMPRESSOR_MODE | | | | |

# 4.3 System variable

System variable $P_TOOLO is available up to SW 6.4. This variable indicates the end orientation of the block determined at the time of run-in.

**SW 6.4 and higher**

The following channel/specific system variables are provided :

| | |
|---|---|
| $AC_TOOLO_ACT[i], i = 1, 2, 3 | ith component of the vector of the current setpoint orientation |
| $AC_TOOLO_END[i], i = 1, 2, 3 | ith component of the vector of the end orientation of the current block |
| $AC_TOOLO_DIFF | Remaining angle in degrees, i.e. this is the angle between the vectors $AC_TOOLO_END[i] and $AC_TOOLO_ACT[i]. |
| $VC_TOOLO[i], i = 1, 2, 3 | ith component of the vector of the actual orientation |
| $VC_TOOLO_DIFF | Angle in degrees between setpoint and actual orientation |
| $VC_TOOLO_STAT | Status variable for actual orientation |

The components of the vectors $AC_TOOLO_ACT[i], $AC_TOOLO_END[i] and $VC_TOOLO[i] of he orientation are normalized such that the orientation vector has the value 1.

These system variables can be read by parts programs and in synchronous actions. Write access is not permitted.

Status variable $VC_TOOLO_STAT shows whether the calculation for actual orientation can be performed. The following values are possible:

0 : Actual orientation can be calculated
–1 : Actual orientation cannot be calculated, since currently active transformation cannot calculate these values in real time.

The actual value of the tool optimization is not provided by all transformations in real time.

**Online tool length offset**

| | |
|---|---|
| $AA_TOFF[Geo axis] | Position offset in the Tool Coordinate System (TCS) |
| $AA_TOFF_VAL[Geo axis] | Integrated position offset in (TCS) |
| $AA_TOFF_LIMIT[Geo axis] | Query whether tool length offset value lies close to limit. |
| $AA_TOFF_LIMIT [ ] = 0: | Offset not close to limit |
| $AA_TOFF_LIMIT [ ] = 1: | Offset in positive direction reached |
| $AA_TOFF_LIMIT [ ] = –1: | Offset in negative direction reached |
| $AA_TOFF_PREP_DIFF [ ] | Large difference between currently active value of $AA_TOFF[ ] and the value prepared as current motion block motion block. |

**References:** /PGA/, Programming Guide Production Planning, List of system variables

■

# Notes

# Signal Descriptions 5

## 5.1 Channel-specific signals

| DB21–30<br>DBB232<br><br>Data block | **Number of active G function of G function group 25 (tool orientation reference)**<br><br>Signal(s) from channel (NCK → PLC) | |
|---|---|---|
| Edge evaluation: | Signal(s) updated: | Signal(s) valid from SW 2 |
| Signal state 1 or signal transition 0 ——> 1 | **ORIWKS:** The tool orientation is implemented in a workpiece coordinate system and is thus not dependent on the machine kinematics. This is the default setting for SW1.1. | |
| Signal state 0 or signal transition 1——> 0 | **ORIMKS:** The tool orientation is implemented in a machine coordinate system and is thus dependent on the machine kinematics. This is the default setting with SW2.1 and higher. | |

| DB21–30<br>DBX 33.6<br><br>Data block | **Transformation active**<br><br>Signal(s) from channel (NCK → PLC) | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW 1.1 |
| Signal state 1 or signal transition 0 ——> 1 | Active transformation | |
| Signal state 0 or signal transition 1——> 0 | Transformation not (no longer) active | |
| Signal irrelevant for ...... | No transformation used | |
| Further references | /PA/, Programming Guide Fundamentals | |

| DB21 – DB30<br>DBX317.6<br><br>Data block | **PTP traversal active**<br><br>Signal(s) from channel (NCK -> PLC) |
|---|---|
| Edge evaluation: yes | Signal(s) updated: |
| Signal state 1<br>(or signal transition 0 –> 1) | PTP travel active |
| Signal state 0<br>(or signal transition 1 –> 0) | CP travel active |
| Signal irrelevant for ... | No handling transformations active |
| Further references | FB Special Functions, F2 |

| DB21 – DB30<br>DBX29.4<br><br>Data block | **Activate PTP traversal**<br><br><br>Signal(s) to channel (PLC–> NCK) |
|---|---|
| Edge evaluation: yes | Signal(s) updated: |
| Signal state 1<br>(or signal transition 0 –> 1) | Activate PTP travel |
| Signal state 0<br>(or signal transition 1 –> 0) | Activate CP travel |
| Signal irrelevant for ... | No handling transformations active |
| Further references | FB Special Functions, F2 |

| DB21 – DB30<br>DBX318.2<br><br>Data block | **TOFF active**<br><br><br>Signal(s) to channel (NCK–> PLC) |
|---|---|
| Edge evaluation: yes | Signal(s) updated: |
| Signal state 1<br>(or signal transition 0 –> 1) | Activate online tool length offset |
| Signal state 0<br>(or signal transition 1 –> 0) | Reset online tool length offset |
| Signal irrelevant for ... | If the "Generic 5-axis transformation" option is not available and no handling transformations are active. |
| Further references | FB Special Functions, F2 |

| DB21 – DB30<br>DBX318.3<br><br>Data block | **TOFF motion active**<br><br><br>Signal(s) to channel (NCK–> PLC) |
|---|---|
| Edge evaluation: yes | Signal(s) updated: |
| Signal state 1<br>(or signal transition 0 –> 1) | Activate offset motion |
| Signal state 0<br>(or signal transition 1 –> 0) | Deactivate offset motion |
| Signal irrelevant for ... | If the "Generic 5-axis transformation" option is not available and no handling transformations are active. |
| Further references | FB Special Functions, F2 |

# Examples 6

## 6.1 Example of a 5-axis transformation

```
CHANDATA(1)

$MA_IS_ROT_AX[AX5] = TRUE
$MA_SPIND_ASSIGN_TO_MACHAX[AX5] = 0
$MA_ROT_IS_MODULO[AX5] = 0

;————————————————————————————————
; General 5–axis transformation
;
; Kinematics:     1. Rotary axis is parallel to Z
;                 2. Rotary axis is parallel to X
;                    Movable tool
;————————————————————————————————


$MC_TRAFO_TYPE_1 = 20

$MC_ORIENTATION_IS_EULER = TRUE

$MC_TRAFO_AXES_IN_1[0] = 1
$MC_TRAFO_AXES_IN_1[1] = 2
$MC_TRAFO_AXES_IN_1[2] = 3
$MC_TRAFO_AXES_IN_1[3] = 4
$MC_TRAFO_AXES_IN_1[4] = 5

$MC_TRAFO_GEOAX_ASSIGN_TAB_1[0]=1
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[1]=2
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[2]=3


$MC_TRAFO5_PART_OFFSET_1[0] = 0
$MC_TRAFO5_PART_OFFSET_1[1] = 0
$MC_TRAFO5_PART_OFFSET_1[2] = 0
$MC_TRAFO5_ROT_AX_OFFSET_1[0] = 0
$MC_TRAFO5_ROT_AX_OFFSET_1[1] = 0

$MC_TRAFO5_ROT_SIGN_IS_PLUS_1[0] = TRUE
$MC_TRAFO5_ROT_SIGN_IS_PLUS_1[1] = TRUE

$MC_TRAFO5_NON_POLE_LIMIT_1 = 2.0

$MC_TRAFO5_POLE_LIMIT_1 = 2.0
```

```
$MC_TRAFO5_BASE_TOOL_1[0] = 0.0
$MC_TRAFO5_BASE_TOOL_1[1] = 0.0
$MC_TRAFO5_BASE_TOOL_1[2] = 5.0

$MC_TRAFO5_JOINT_OFFSET_1[0] = 0.0
$MC_TRAFO5_JOINT_OFFSET_1[1] = 0.0
$MC_TRAFO5_JOINT_OFFSET_1[2] = 0.0

CHANDATA(1)
M17
```

```
;————————————————————————————————————————————————
; Program example for general 5-axis transformation
;————————————————————————————————————————————————

; Definition of tool T1
$TC_DP1[1,1] =   10            ; Type
$TC_DP2[1,1] =   0
$TC_DP3[1,1] =   20.           ; z Length compensation vector G17
$TC_DP4[1,1] =   0.            ; y
$TC_DP5[1,1] =   0.            ; x
$TC_DP6[1,1] =   0.            ; Radius
$TC_DP7[1,1] =   0
$TC_DP8[1,1] =   0
$TC_DP9[1,1] =   0
$TC_DP10[1,1] =  0
$TC_DP11[1,1] =  0
$TC_DP12[1,1] =  0

;——————————Approach initial position ————————————————————
N100 G1 x1 y0 z0 a0 b0 F20000 G90 G64 T1 D1 G17 ADIS=.5 ADISPOS=3

;==========================================================
; Orientation vector programming
;==========================================================
N110 TRAORI(1)
N120 ORIWKS
N130 G1 G90
N140 a3 = 0 b3 =   0 c3 = 1 x0
N150 a3 = 0 b3 = –1 c3 = 0
N160 a3 = 1 b3 =   0 c3 = 0
N170 a3 = 1 b3 =   0 c3 = 1
N180 a3 = 0 b3 =   1 c3 = 0
N190 a3 = 0 b3 =   0 c3 = 1

;==========================================================
; Euler angle programming
;==========================================================
N200 ORIMKS
N210 G1 G90
N220 a2 =   0 b2 =  0  x0
N230 a2 =   0 b2 =  90
N240 a2 = 90 b2 =  90
N250 a2 = 90 b2 =  45
N260 a2 =   0 b2 =–90
N270 a2 =   0 b2 =  0

;==========================================================
; Axis programming
;==========================================================
N300 a0 b0 x0
N310 a45
N320 b30
```

```
;=============================================================
; TOFRAME
;=============================================================

N400 G0 a90 b90 x0 G90
N410 TOFRAME
N420 z5
N430 x3 y5
N440 G0 a0 b0 x1 y0 z0 G90

N500 TRAFOOF
m30
```

# 6.2 Example of a 3-axis and 4-axis transformation

## 6.2.1 Example of a 3-axis transformation

Example: The 3-axis transformation can be configured as follows for the machine shown in the schematic diagram in Fig. 1-1:

```
$MC_TRAFO_TYPE_n = 18

$MC_TRAFO_GEOAX_ASSIGN_TAB_n[0] =  1   ; Assignment of channel axes to geometry axes
$MC_TRAFO_GEOAX_ASSIGN_TAB_n[1] =  0
$MC_TRAFO_GEOAX_ASSIGN_TAB_n[2] =  3

$MC_TRAFO_AXES_IN_n[0] = 1             ; x axis is channel axis 1
$MC_TRAFO_AXES_IN_n[1] = 0             ; y axis is not used
$MC_TRAFO_AXES_IN_n[2] = 3             ; z axis is channel axis 3

$MC_TRAFO_AXES_IN_n[4] = 0             ; There is no 2nd rotary axis
```

## 6.2.2 Example of a 4-axis transformation

Example: The 4-axis transformation can be configured as follows for a machine identical to the one illustrated in Fig. 1-2, but with an additional axis (Y):

```
$MC_TRAFO_TYPE_n = 18

$MC_TRAFO_GEOAX_ASSIGN_TAB_n[0] =  1
$MC_TRAFO_GEOAX_ASSIGN_TAB_n[1] =  2
$MC_TRAFO_GEOAX_ASSIGN_TAB_n[2] =  3

$MC_TRAFO_AXES_IN_n[0] = 1             ; x axis is channel axis 1
$MC_TRAFO_AXES_IN_n[1] = 2             ; y axis is channel axis 2
$MC_TRAFO_AXES_IN_n[2] = 3             ; z axis is channel axis 3

$MC_TRAFO_AXES_IN_n[4] = 0             ; There is no 2nd rotary axis
```

### 6.2.3 Set of machine data and parts program (extract)

Machine data for a 3-axis and 4-axis transformation

```
CHANDATA(1)

$MC_AXCONF_MACHAX_USED[0] = 1          ; Machine axes used
$MC_AXCONF_MACHAX_USED[1] = 2
$MC_AXCONF_MACHAX_USED[2] = 3
$MC_AXCONF_MACHAX_USED[3] = 4          ; With 4-axis transformation only
$MA_IS_ROT_AX[AX4] = 1

; 3-axis transformation for moved tool and orientation in the XY plane

$MC_TRAFO_TYPE_1 = 20

$MC_TRAFO_GEOAX_ASSIGN_TAB_1[0] = 1
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[1] = 2
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[2] = 0

$MC_TRAFO_AXES_IN_1[0] = 1
$MC_TRAFO_AXES_IN_1[1] = 2
$MC_TRAFO_AXES_IN_1[2] = 0             ; No 3rd translatory axis available
$MC_TRAFO_AXES_IN_1[3] = 4             ; Rotary axis
$MC_TRAFO_AXES_IN_1[4] = 0             ; No 2nd rotary axis, i.e. 3-axis transformation

; 4-axis transformation for moved workpiece and orientation in the XZ plane

$MC_TRAFO_TYPE_1 = 34

$MC_TRAFO_GEOAX_ASSIGN_TAB_2[0] = 1
$MC_TRAFO_GEOAX_ASSIGN_TAB_2[1] = 2
$MC_TRAFO_GEOAX_ASSIGN_TAB_2[2] = 3

$MC_TRAFO_AXES_IN_2[0] = 1
$MC_TRAFO_AXES_IN_2[1] = 2
$MC_TRAFO_AXES_IN_2[2] = 3             ; 3rd translatory axis available
$MC_TRAFO_AXES_IN_2[3] = 4             ; Rotary axis
$MC_TRAFO_AXES_IN_2[4] = 0             ; No 2nd rotary axis, i.e. 4-axis transformation

CHANDATA(1)
M17                                    ; End of machine data
```

**Parts program** (extract)

```
N10 $TC_DP1[1,1] = 10
N20 $TC_DP2[1,1] = 20
N30 $TC_DP3[1,1] = 1.0
N40 $TC_DP4[1,1] = 0.0
N50 $TC_DP5[1,1] = 0.0

N60 G0 x0 y0 z0 a0 b0 c0 F10000 G90 T0 D0
N70 TRAORI(1)                          ; Switch on 3-axis transformation
N80 a30                                ; Axis programming, rotation 30 degrees
N90  a3=–0.5 b3=0.866025 c3=0.0        ; Progr. direction vector
N100 TRAFOOF()                         ; End of 3-axis transformation
N110 TRAORI(2)                         ; 2nd transformation defined in MD (4-axis)
N120 a45
N130 M30
```

## 6.3 Example of a universal milling head

**General**

The following two subsections show the main steps which need to be taken in order to activate a transformation for the universal milling head.

**Machine data**

```
; Machine kinematics CA' with orientation of tool in position zero towards Z
$MC_TRAFO_TYPE_1 = 148

$MC_TRAFO_GEOAX_ASSIGN_TAB_1[0] = 1
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[1] = 2
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[2] = 3

; Angle of 2nd rotary axes
$MC_TRAFO5_NUTATOR_AX_ANGLE_1 = 45
```

**Program**

```
; Definition of tool T1
$TC_DP1[1,1] = 120;          Type
$TC_DP2[1,1] = 0;
$TC_DP3[1,1] = 20;           Z length compensation vector G17
$TC_DP4[1,1] = 8.;           Y
$TC_DP5[1,1] = 5.;           X

TRAORI(1);                   Activation of transformation
ORIMKS;                      Reference of orientation to MCS
G0 X1 Y0 Z0 A0 B0 F20000 G90 G64 T1 D1 G17

; Programming of directional vector
G1 G90
a3 = 0 b3 = 1 c3 = 0

; Programming in Euler angles
G1 G90
a2 = 0 b2 = 0 X0

; Programming of rotary axis motion
G1 X10 Y5 Z20 A90 C90

m30
```

**References:** /PA/, Programming Guide

## 6.4 Example for orientation axes (SW 5.3 and higher)

**Example 1**:
3 orientation axes for the 1st orientation transformation for kinematics with 6 transformed axes. The axis must rotate first

- about the Z axis, then
- about the Y axis and finally again
- about the Z axis.

The tool vector must point in the X direction.

CHANDATA(1)

```
$MC_TRAFO5_TOOL_VECTOR_1=0                ;Tool vector in X direction
$MC_TRAFO5_ORIAX_ASSIGN_TAB_1[0]=4        ;Channel index 1st orient.
                                           axis
$MC_TRAFO5_ORIAX_ASSIGN_TAB_1[1]=5        ;Channel index 2nd orient.
                                           axis
$MC_TRAFO5_ORIAX_ASSIGN_TAB_1[2]=6        ;Channel index 3rd orient.
                                           axis
$MC_ORIAX_TURN_TAB_1[0]=3                 ;Z direction
$MC_ORIAX_TURN_TAB_1[1]=2                 ;Y direction
$MC_ORIAX_TURN_TAB_1[2]=3                 ;Z direction
```

CHANDATA(1)
M17

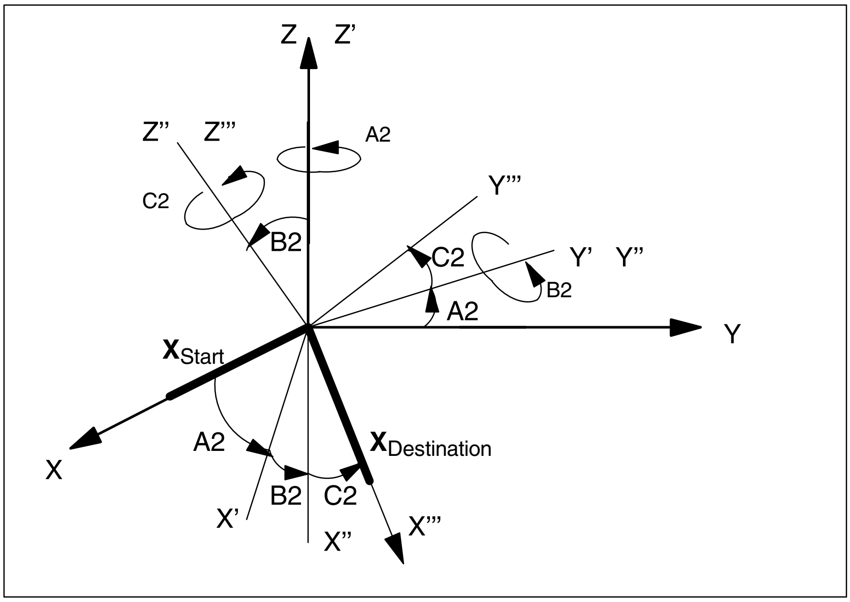



Fig. 6-1 3 orientation axes for the 1st orientation transformation for kinematics with 6 transformed axes

**Example 2**:
3 orientation axes for the 2nd orientation transformation for kinematics with 5 transformed axes. The axis must rotate first

- about the X axis, then
- about the Y axis and finally
- about the Z axis.

The tool vector must point in the Z direction.

CHANDATA(1)

| | |
|---|---|
| $MC_TRAFO5_TOOL_VECTOR_2=2 | ;Tool vector in Z direction |
| $MC_TRAFO5_ORIAX_ASSIGN_TAB_1[0]=4 | ;Channel index 1st orient. axis |
| $MC_TRAFO5_ORIAX_ASSIGN_TAB_1[1]=5 | ;Channel index 2nd orient. axis |
| $MC_TRAFO5_ORIAX_ASSIGN_TAB_1[2]=0 | ;Channel index 3rd orient. axis |
| $MC_ORIAX_TURN_TAB_1[0]=1 | ;X direction |
| $MC_ORIAX_TURN_TAB_1[1]=2 | ;Y direction |
| $MC_ORIAX_TURN_TAB_1[2]=3 | ;Z direction |

CHANDATA(1)
M17

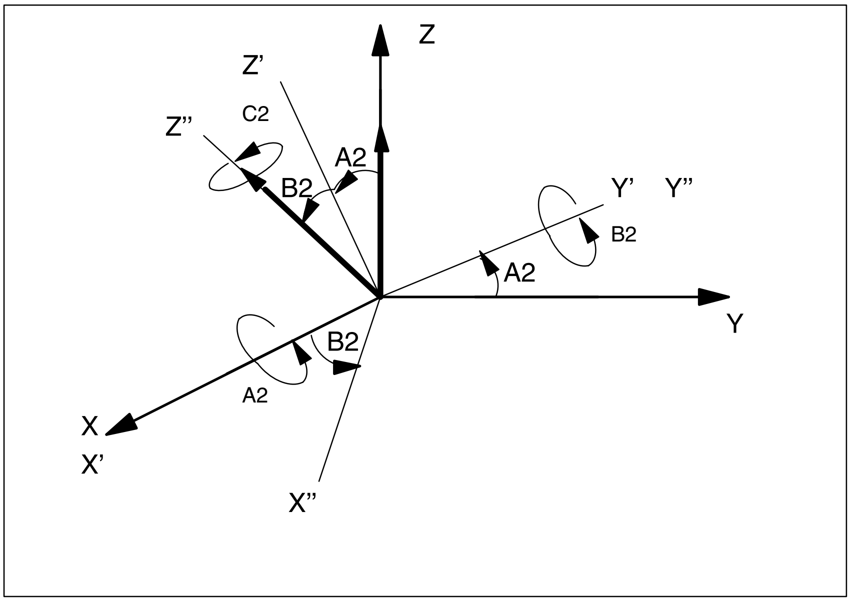



Fig. 6-2 3 orientation axes for the 2nd orientation transformation for kinematics with 5 transformed axes

The rotation through angle C2 about the Z” axis is omitted in this case, because the tool vector orientation can be determined solely from angles A2 and B2 and no further degree of freedom is available on the machine.

**References:** /PGA/, Programming Guide Advanced

# 6.5 Examples for orientation vectors (SW 5.3 and higher)

## 6.5.1 Example for polynomial interpretation of orientation vectors

**Orientation vector in Z–X plane**

The orientation vector is programmed directly in the examples below. The movements of the rotary axes that result depend on the particular kinematics of the machine.

```
N10 TRAORI
N20 POLY                        ; Polynomial interpolation is enabled.
N30 A3=0 B3=0 C3=1              ; Orientation in +Z direction (start vector)
N40 A3=1 B3=0 C3=0              ; Orientation in +X direction (end vector)
```

In N40, the orientation vector is rotated in the Z–X plane that is spanned from the start and end vector. Here, the PHI angle is interpolated in a line in this plane between the values 0 and 90 degrees (large circle interpolation).

The additional specification of the polynomials for the two angle PHI and PSI means that the interpolated orientation vector can lies anywhere between the start and end vector.

**PHI angle using polynomial PHI**

In contrast to the example above, the PHI angle is interpolated using the polynomial $PHI(u) = (90–10)u + 10*u^2$ between the values 0 and 90 degrees.

The PSI angle is not equal to zero and is interpolated as the polynomial $PSI(u) = –10*u + 10*u^2$.

The maximum “tilt” of the orientation vector from the plane between the start and end vector is obtained in the middle of the block (u = 1/2).

```
N10 TRAORI
N20 POLY                        ; Polynomial interpolation is enabled.
N30 A3=0 B3=0 C3=1              ; Orientation in +Z direction (start vector)
N40 A3=1 B3=0 C3=0    PO[PHI]=(10) PO[PSI]=(10)    ; in +X direction (end vector)
```

### 6.5.2 Example for rotations of orientation vector (SW 6.1 and higher)

**Rotations with angle of rotation THETA**

In the following example, the angle of rotation is interpolated in linear fashion from starting value 0 degrees to end value 90 degrees. The angle of rotation changes according to a parabola or a rotation can be executed without an orientation change. The tool orientation is rotated from the Y direction to the X direction.

```
N10 TRAORI                              ; Activate orientation transformation
N20 G1 X0 Y0 Z0 F5000                   ;
                                        ; Tool orientation
N30 A3=0 B3=0 C3=1 THETA=0              ; in Z direction with angle of rotation 0
N40 A3=1 B3=0 C3=0 THETA=90             ; in X direction and rotation through 90 degrees
N50 A3=0 B3=1 C3=0 PO[THT]=(180,90)     ; in Y direction and rotation to 180 degrees
N60 A3=0 B3=1 C3=0 THETA=IC(–90)        ; Remain constant and rotation to 90 degrees.
N70 ORIROTT                             ; Angle of rotation relative to the change in the
                                          orientation.
N80 A3=1 B3=0 C3=0 THETA=30             ; Rotation vector at an angle of 30 degrees to
                                          X–Y plane.
```

N40 Linear interpolation of angle of rotation from starting value 0 degrees to end value 90 degrees.

N50 The angle of rotation changes from 90 degrees to 180 degrees according to parabola $\theta(u) = 90 + 90u^2$.

N60 A rotation can also be programmed without an orientation change taking place.

N80 The tool orientation is rotated from the Y direction to the X direction. The orientation change takes place in the X–Y plane and the rotation vector describes an angle of 30 degrees to this plane.

## 6.6 Example of generic 5-axis transformation (SW 5.2 and higher)

The following example is based on a machine with rotatable tool on which the first rotary axis is a C axis and the second a B axis (CB kinematics, see Fig. LEERER MERKER). The basic orientation defined in the machine data is the bisecting line between the X and Z axes.

The relevant machine data are as follows:

```
CHANDATA (1)
$MC_TRAFO_TYPE_1 = 24                  ; General 5-axis transformation;
                                       ; Rotatable tool
$MC_TRAFO5_AXIS1_1[0] = 0.0
$MC_TRAFO5_AXIS1_1[1] = 0.0
$MC_TRAFO5_AXIS1_1[2] = 1.0            ; 1st rotary axis is parallel to Z.
$MC_TRAFO5_AXIS2_1[0] = 0.0
$MC_TRAFO5_AXIS2_1[1] = 1.0
$MC_TRAFO5_AXIS2_1[2] = 0.0            ; 2nd rotary axis is parallel to Y.
$MC_TRAFO5_BASE_ORIENT_1[0] = 1.0
$MC_TRAFO5_BASE_ORIENT_1[1] = 0.0
$MC_TRAFO5_BASE_ORIENT_1[2] = 1.0
M30
```

**Example program:**

```
N10  $TC_DP1[1,1 ] = 120      ; End mill
N20  $TC_DP3[1,1 ] = 0        ; Length compensation vector
N30
N40                           ; Definition of toolholder
N50  $TC_CARR7[1]  = 1        ; Component of 1st rotary axis in X direction
N60  $TC_CARR11[1] = 1        ; Component of 2nd rotary axis in Y direction
N70  $TC_CARR13[1] = –45      ; Angle of rotation of 1st axis
N80  $TC_CARR14[1] = 0        ; Angle of rotation of 2nd axis
N90
N100 X0 Y0 Z0 B0 C0 F10000 ORIWKS G17
N110 TRAORI()                 ; Selection of transf. basic orientation from
                              ; machine data
N120 C3=1                     ; Set orientat. parallel to Z ⇒ B–45 C0
N130 T1 D1                    ; Basic orientation is now parallel to Z
N140 C3=1                     ; Set orientat. parallel to Z ⇒ B0 C0
N150 G19                      ; Basic orientation is now parallel to X
N160 C3=1                     ; Set orientat. parallel to Z ⇒ B–90 C0
N170 G17 TCARR=1 TCOABS       ; Basic orientation is now bisecting line Y–Z
N180 A3=1                     ; Set orientat. par. to X ⇒ B–90 C–135
N190 B3=1 C3=1                ; Orientat. par. to basic orientation ⇒ B0 C0
N200 TRAORI(,2.0, 3.0, 6.0)   ; Transfer basic orientat. in call
N210 A3=2 B3=3 C3=6           ; Orientat. par. to basic orientation ⇒ B0 C0
N220 TOFRAME                  ; Z axis points in direction of orientation
N230 G91 Z7                   ; Travel 7mm in new Z direction ⇒ X2 Y3 Z6
N240 C3=1                     ; Orientat. par. to new Z axis ⇒ B0 C0
N250 M30
```

### 6.6.1 Example for modification of rotary axis motion (SW 6.1 and higher)

The machine is a 5-axis machine of machine type 1 (two-axis inclinable head with CA kinematics) on which both rotary axes rotate the tool (transformation type 24). The first rotary axis is a modulo axis parallel to Z (C axis); the second rotary axis is parallel to Y (B axis) and has a traversing range from –5 degrees to +185 degrees.

To allow modification at any time, machine data
MD 21180: ROT_AX_SWL_CHECK_MODE contains the value 2.

```
N10 X0 Y0 Z0 B0 C0
N20 TRAORI( )                ; Basic orientation, 5-axis transformation
N30 B–1 C10                  ; Rotary axis positions B–1 and C10
N40 A3=–1 C3=1 ORIWKS        ; Large circle interpolation in WCS
N50 M30
```

At the start of block N40 in the example program, the machine is positioned at rotary axis positions B–1 C10. The programmed end orientation can be achieved with either of the axis positions B–45 C0 (1st solution) or B45 C180 (2nd solution).

The first solution is selected initially, because it is nearest to the starting orientation and, unlike the second solution, can be achieved using large circle interpolation (ORIWKS). However, this position **cannot** be reached because of the axis limits of the B axis.

The second solution is therefore used instead, i.e. the end position is B45 C180. The end orientation is achieved by axis interpolation. The programmed orientation path cannot be followed.

## 6.7 Compressor example for orientation (SW 6.3 and higher)

**Task**

In the example program below, a circle approached by a polygon definition is compressed. The tool orientation moves synchronously across the outside of a taper. Although the sequence of programmed orientation changes is unsteady, the compressor generates a smooth orientation movement.

```
DEF INT ANZAHL = 60
DEF REAL RADIUS = 20
DEF INT COUNTER
DEF REAL WINKEL
N10 G1 X0 Y0 F5000 G64

$SC_COMPRESS_CONTUR_TOL = 0.05          ; Maximum deviation
                                        ; of contour 0.05mm
$SC_COMPRESS_ORI_TOL = 5                ; Maximum deviation
                                        ; of orientation 5 degrees

TRAORI
COMPCURV
              ; The movement describes a circle generated from polygons.
              ; The orientation moves across a taper about the
              ; Z axis with an angle of aperture of 45 degrees.

N100 X0 Y0 A3=0 B3=–1 C3=1
N110 FOR COUNTER = 0 TO ANZAHL
N120 WINKEL = 360 * COUNTER/ANZAHL
N130 X=RADIUS*cos(WINKEL) Y=RADIUS*sin(WINKEL)
     A3=sin(WINKEL) B3=–cos(WINKEL) C3=1
N140 ENDFOR
...
```

■

# Data Fields, Lists 7

## 7.1 Interface signals

| DB number | Bit, byte | Name | Ref. |
|---|---|---|---|
| Channel-specific | | | |
| 21–30 | 33.6 | Transformation active | K1 |
| 21–30 | 232 | Number of active G function of G function group 25 | |
| 21–30 | 317.6 | PTP traversal active | TE4 |
| 21–30 | 29.4 | Activate PTP traversal | TE4 |
| 21–30 | 318.2 | Activate online tool length offset | |
| 21–30 | 318.3 | Activate correction movement | |

## 7.2 Setting data

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| General ($SD_ ...) | | | |
| 41110 | JOG_SET_VELO | Geometry axes | |
| 41130 | JOG_ROT_AX_SET_VELO | Orientation axes | |
| Channel-specific ($SC_...) | | | |
| 42475 | COMPRESS_CONTOUR_TOL | Max. contour deviation for compressor | |
| 42476 | COMPRESS_ORI_TOL | Max. angular displacement of tool orientation for the compressor | |
| 42477 | COMPRESS_ORI_ROT_TOL | Max. angular displacement of tool rotation for the compressor | |
| 42650 | CART_JOG_MODE | Coordinate system for Cartesian manual travel (SW 6.3 and higher) | |
| 42970 | TOFF_LIMIT | Upper limit for offset value $AA_TOFF (SW 6.4 and higher) | |

## 7.3 Machine data

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| General ($MN_ ...) | | | |
| 10620 | EULER_ANGLE_NAME_TAB | Name of Euler angles or names of orientation axes | |
| 10630 | NORMAL_VECTOR_NAME_TAB | Name of normal vectors (SW 4.1 and higher) | |
| 10640 | DIR_VECTOR_NAME_TAB | Name of direction vectors (SW 4.1 and higher) | |
| 10642 | ROT_VECTOR_NAME_TAB | Name of rotating vectors (SW 6.1 and higher) | |
| 10644 | INTER_VECTOR_NAME_TAB | Name of intermediate vector component | |
| 10646 | ORIENTATION_NAME_TAB | Name for programming a second orientation path (SW 6.1 and higher) | |
| 10648 | NUTATION_ANGLE_NAME | Name of orientation angle (SW 6.1 and higher) | |
| 10670 | STAT_NAME | Name of position information (SW 5.3 and higher) | |
| 10672 | TU_NAME | Name of position information of axes | |
| 10674 | PO_WITHOUT_POLY | Allows Programming of PO[ ] without having to activate POLY (SW 5.3 and higher) | |

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| Channel-specific ($MC_ ...) | | | |
| 20150 | GCODE_RESET_VALUES[n] | Initial setting of G groups | |
| 20152 | GCODE_RESET_MODE[n] | Setting after RESET/end of parts program | |
| 20482 | COMPRESS_MODE | Compressor mode (SW 6.3 and higher) | |
| 20621 | HANDWH_ORIAX_MAX_INCR_SIZE | Limitation of handwheel increment | |
| 20623 | HANDWH_ORIAX_MAX_INCR_VSIZE | Orientation velocity overlay | |
| 21100 | ORIENTATION_IS_EULER | Angle definition for orientation programming | |
| 21102 | ORI_DEF_WITH_G_CODE | Definition of orientation angles A2, B2, C2 | |
| 21104 | ORI_IPO_WITH_G_CODE | Definition of interpolation type for orientation | |
| 21106 | CART_JOG_SYSTEM | Coordinate system for Cartesian JOG (SW 6.3 and higher) | |
| 21108 | POLE_ORI_MODE | Behavior during large circle interpolation at pole position | |
| 21120 | ORIAX_TURN_TAB_1[n] | Assignment of rotation of orientation axes about the reference axes, definition 1 [n = 0..2] | |
| 21130 | ORIAX_TURN_TAB_2[n] | Assignment of rotation of orientation axes about the reference axes, definition 2 [n = 0..2] | |
| 21150 | JOG_VELO_RAPID_ORI[n] | Rapid traverse in jog mode for orientation axes in the channel [n = 0..2] | |
| 21155 | JOG_VELO_ORI[n] | Orientation axis velocity in jog mode [n = 0..2] | |
| 21160 | JOG_VELO_RAPID_GEO[n] | Rapid traverse in jog mode for geometry axes in the channel [n = 0..2] | |

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| 21165 | JOG_VELO_GEO[n] | Geometry axis velocity in jog mode [n = 0..2] | |
| 21170 | ACCEL_ORI[n] | Acceleration for orientation axes [n = 0..2] | |
| 21180 | ROT_AX_SWL_CHECK_MODE | Check software limits for orientation axes (SW 6.1 and higher) | |
| 21190 | TOFF_MODE | Effect of online offset in tool direction (SW 6.4 and higher) | |
| 21194 | TOFF_VELO | Velocity of online offset in tool direction (SW 6.4 and higher) | |
| 21196 | TOFF_ACCEL | Acceleration of online offset in tool direction (SW 6.4 and higher) | |
| 24100 | TRAFO_TYPE_1 | Definition of transformation 1 in channel | |
| 24110 | TRAFO_AXES_IN_1[n] | Axis assignment for transformation 1 [axis index] | |
| 24120 | TRAFO_GEOAX_ASSIGN_TAB_1[n] | Assignment geometry axis to channel axis for transformation 1 [geometry no.] | |
| 24200 | TRAFO_TYPE_2 | Definition of transformation 2 in channel | |
| 24210 | TRAFO_AXES_IN_2[n] | Axis assignment for transformation 2 [axis index] | |
| 24220 | TRAFO_GEOAX_ASSIGN_TAB_2[n] | Assignment geometry axis to channel axis for transformation 2 [geometry no.] | |
| 24300 | TRAFO_TYPE_3 | Definition of transformation 3 in channel | |
| 24310 | TRAFO_AXES_IN_3[n] | Axis assignment for transformation 3 [axis index] | |
| 24320 | TRAFO_GEOAX_ASSIGN_TAB_3[n] | Assignment geometry axis to channel axis for transformation 3 [geometry no.] | |
| 24400 | TRAFO_TYPE_4 | Definition of transformation 4 in channel | |
| 24410 | TRAFO_AXES_IN_4[n] | Axis assignment for transformation 4 [axis index] | |
| 24420 | TRAFO_GEOAX_ASSIGN_TAB_4[n] | Assignment geometry axis to channel axis for transformation 4 [geometry no.] | |
| 24430 | TRAFO_TYPE_5 | Definition of transformation 5 in channel | |
| 24432 | TRAFO_AXES_IN_5[n] | Axis assignment for transformation 5 [axis index] | |
| 24434 | TRAFO_GEOAX_ASSIGN_TAB_5[n] | Assignment geometry axis to channel axis for transformation 5 [geometry no.] | |
| 24440 | TRAFO_TYPE_6 | Definition of transformation 6 in channel | |
| 24442 | TRAFO_AXES_IN_6[n] | Axis assignment for transformation 6 [axis index] | |
| 24444 | TRAFO_GEOAX_ASSIGN_TAB_6[n] | Assignment geometry axis to channel axis for transformation 6 [geometry no.] | |
| 24450 | TRAFO_TYPE_7 | Definition of transformation 7 in channel | |
| 24452 | TRAFO_AXES_IN_7[n] | Axis assignment for transformation 7 [axis index] | |
| 24454 | TRAFO_GEOAX_ASSIGN_TAB_7[n] | Assignment geometry axis to channel axis for transformation 7 [geometry no.] | |
| 24460 | TRAFO_TYPE_8 | Definition of transformation 8 in channel | |
| 24462 | TRAFO_AXES_IN_8[n] | Axis assignment for transformation 8 [axis index] | |

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| 24464 | TRAFO_GEOAX_ASSIGN_TAB_8[n] | Assignment geometry axis to channel axis for transformation 8 [geometry no.] | |
| 24500 | TRAFO5_PART_OFFSET_1[n] | Offset vector of 5-axis transformation 1 [n = 0.. 2] | |
| 24510 | TRAFO5_ROT_AX_OFFSET_1[n] | Position offset of rotary axis 1/2 for 5-axis transformation 1 [axis no.] | |
| 24520 | TRAFO5_ROT_SIGN_IS_PLUS_1[n] | Sign of rotary axis 1/2 for 5-axis transformation 1 [axis no.] | |
| 24530 | TRAFO5_NON_POLE_LIMIT_1 | Definition of pole range for 5-axis transformation 1 | |
| 24540 | TRAFO5_POLE_LIMIT_1 | End angle tolerance with interpolation through pole for 5-axis transformation 1 | |
| 24550 | TRAFO5_BASE_TOOL_1[n] | Vector of base tool for activation of 5-axis transformation 1 [n = 0.. 2] | |
| 24560 | TRAFO5_JOINT_OFFSET_1[n] | Vector of kinematic offset of 5-axis transformation 1 [n = 0.. 2] | |
| 24562 | TRAFO5_TOOL_ROT_AX_OFFSET_1 | Offset of the swivel point of the 1st 5-axis transformation with swiveled linear axis. | |
| 24564 | TRAFO5_NUTATOR_AX_ANGLE_1 | Angle of 2nd rotary axis for the universal milling head | |
| 24570 | TRAFO5_AXIS1_1 | Vector for the first rotary axis and the first orientation transformation (SW 5.2 and higher) | |
| 24572 | TRAFO5_AXIS2_1 | Vector for the second rotary axis and the first orientation transformation (SW 5.2 and higher) | |
| 24574 | TRAFO5_BASE_ORIENT_1 | Basic orientation for the first orientation transformation (SW 5.2 and higher) | |
| 24580 | TRAFO5_TOOL_VECTOR_1 | Direction of tool vector for orientation transformation 1 | |
| 24585 | TRAFO5_ORIAX_ASSIGN_TAB_1[n] | Assignment of orientation axes to channel axes for orientation transformation 1 [n = 0.. 2] | |
| 24600 | TRAFO5_PART_OFFSET_2[n] | Offset vector of 5-axis transformation 2 [n = 0.. 2] | |
| 24610 | TRAFO5_ROT_AX_OFFSET_2[n] | Position offset of rotary axis 1/2 for 5-axis transformation 2 [axis no.] | |
| 24620 | TRAFO5_ROT_SIGN_IS_PLUS_2[n] | Sign of rotary axis 1/2 for 5-axis transformation 2 [axis no.] | |
| 24630 | TRAFO5_NON_POLE_LIMIT_2 | Definition of pole range for 5-axis transformation 2 | |
| 24640 | TRAFO5_POLE_LIMIT_2 | End angle tolerance with interpolation through pole for 5-axis transformation 2 | |
| 24650 | TRAFO5_BASE_TOOL_2[n] | Vector of base tool for activation of 5-axis transformation 2 [n = 0.. 2] | |
| 24660 | TRAFO5_JOINT_OFFSET_2[n] | Vector of kinematic offset of 5-axis transformation 2 [n = 0.. 2] | |
| 24662 | TRAFO5_TOOL_ROT_AX_OFFSET_2 | Offset of the swivel point of the 2nd 5-axis transformation with swiveled linear axis. | |
| 24664 | TRAFO5_NUTATOR_AX_ANGLE_2 | Angle of 2nd rotary axis for the universal milling head | |
| 24670 | TRAFO5_AXIS1_2 | Vector for the first rotary axis and the second orientation transformation (SW 5.2 and higher) | |

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| 24672 | TRAFO5_AXIS2_2 | Vector for the second rotary axis and the second orientation transformation (SW 5.2 and higher) | |
| 24674 | TRAFO5_BASE_ORIENT_2 | Basic orientation for the second orientation transformation (SW 5.2 and higher) | |
| 24680 | TRAFO5_TOOL_VECTOR_2 | Direction of tool vector for orientation transformation 2 | |
| 24685 | TRAFO5_ORIAX_ASSIGN_TAB_2[n] | Assignment of orientation axes to channel axes for orientation transformation 2 [n = 0.. 2] | |

## 7.4 Alarms

A detailed description of the alarms which may occur is given in

**References:** /DA/, Diagnostics Guide

or in the online help of systems with MMC 101/102/103.

■

# SINUMERIK 840D/840Di
# Description of Functions Special Functions (Part 3)

# Gantry Axes (G1)



■

# Notes

# 1 Brief Description

**Gantry axes**

The "Gantry axes" function (see Section 3) allows two or more mechanically coupled machine axes to be traversed simultaneously with no mechanical offset.

With regards to operation and programming, the axes defined in the **gantry grouping** are treated as if they were **one** machine axis (called "master" axis).

While the gantry axes are traversing, the control continuously monitors the position actual values of the coupled axes to check whether the difference is still within the specified tolerance range. When the actual position values of the synchronized axes deviate too much from that of the master axis, the control automatically shuts down all axes in the gantry grouping to prevent any damage to the machine.

The purpose of the "Gantry axes" function is to control and monitor machine axes which are rigidly coupled in this way.

**Application**

Two feed drives are required to traverse the gantry on large gantry-type milling machines, i.e. one drive with its own position measuring system on each side. Owing to the mechanical forced coupling, both drives must be operated in absolute synchronism to prevent canting of mechanical components.

**Configurations**

A total of three gantry groupings can be defined. One gantry grouping consists of a master axis and up to two synchronized axes.

■

## Notes

# Detailed Description 2

## 2.1 "Gantry axes" function

**Application**

On large gantry-type milling machines, various axis units (e.g. gantry or crossbeam; see Fig. 2-1) are moved in each case by a number of drives which are mutually independent. Each drive has its own measuring system and thus constitutes a complete axis system. When these mechanically rigidly coupled axes are traversed, both drives must be operated **in absolute synchronism** in order to prevent canting of mechanical components (resulting in power/torque transmission).

The purpose of the "Gantry axes" function (see Chapter 3) is to control and monitor machine axes which are rigidly coupled in this way.

**Terms**

The following terms are frequently used in this functional description:

| | |
|---|---|
| Gantry axes: | Gantry axes comprise at least one pair of axes, the master axis and the slave axis. Since these are mechanically coupled, they must always be traversed simultaneously by the NC. The difference between the actual positions of the axes is monitored continuously. The axes in a gantry grouping are either all linear axes or all rotary axes. |
| Gantry axis grouping: | The gantry axis grouping defines which synchronized axes are controlled by which master axis based on machine data settings. The master and synchronized axes cannot be traversed separately. |
| Master axis: | The master axis is the gantry axis which actually exists from the point of the view of the operator and programmer and can be controlled accordingly in the same way as a normal NC axis. The axis name of the master axis identifies all axes in the gantry axis grouping. |
| Synchronized axis: | The synchronized axis is the gantry axis of which the setpoint position is always derived from the traversing motion of the master axis. It therefore moves in exact synchronism with the master axis. From the point of view of the programmer and operator, the synchronized axis "does not exist". |

**Axis definition**

Axial MD 37100: GANTRY_AXIS_TYPE must be set to define

- whether the axis belongs to a **gantry grouping** and, if yes, to which
- whether the axis is defined as a **master axis** or **synchronized axis** within this grouping.

A total of up to 3 gantry groupings can be defined.

Each gantry grouping consists of **one** master axis and **one or two** synchronized axes.

**Conditions for a gantry grouping**

- A gantry grouping must not contain a spindle.
- A synchronized axis must not be a concurrent POS axis.
- A synchronized axis must not be addressed by a transformation.
- A synchronized axis must not be the slave axis in another type of axis coupling.
- A synchronized axis must not be defined as the master axis in another axis coupling.

---

**Note**

Each axis in the gantry grouping must be set such that it can take over the function of master axis at any time, i.e. matching velocity, acceleration and dynamic response settings.

---

The control performs a plausibility check on the axis definition.

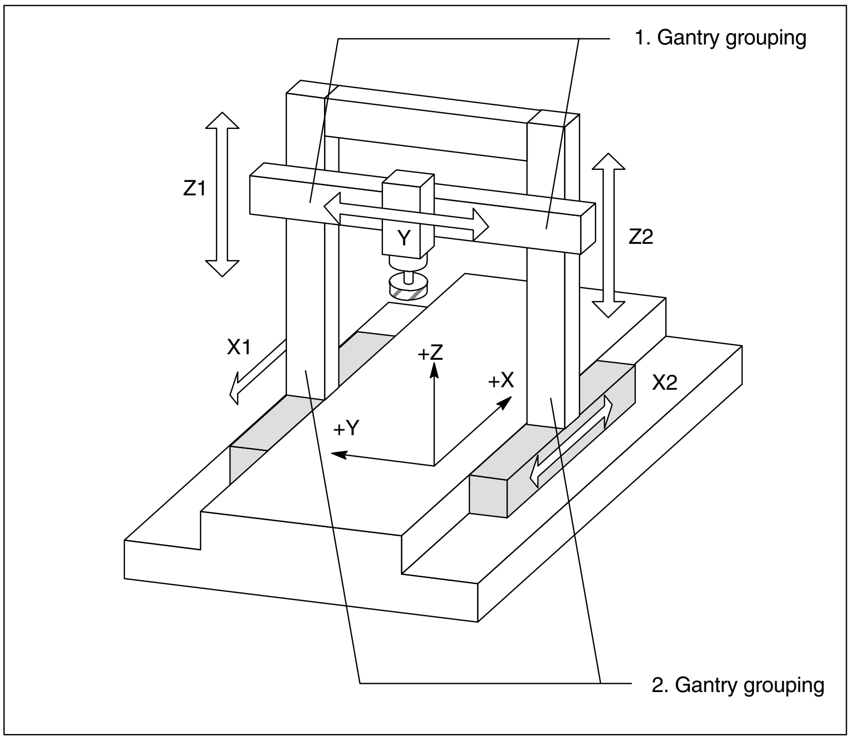



Fig. 2-1 Example: Gantry-type milling machine with 2 gantry groupings

**Functional units**

The "Gantry axes" function can be subdivided into the following functional units:

1. Setpoint generation of synchronized axis
2. Monitoring of actual value difference
3. Referencing and synchronization of master axis and synchronized axes.

**Setpoint generation of synchronized axis**

From the point of view of the operator, all coupled gantry axes are traversed as if only one axis, i.e. the master axis, were programmed in the NC. Analogously, only the master axis is programmed in the parts program. The commands and traverse requests from the operator, the PLC interface or via the parts program therefore apply in equal measure to all axes in the gantry grouping.

When the "Gantry axes" function is active, the synchronized axis setpoint is generated directly from the setpoint of the master axis in all operating modes.

---

**Note**

The dynamic control response of the master and synchronized axes must be set identically.

---

**Monitoring the actual value difference**

The position actual values of the master and synchronized axes are continuously compared with one another in the interpolation clock cycle and monitored to check that they are still within the permissible tolerance range.

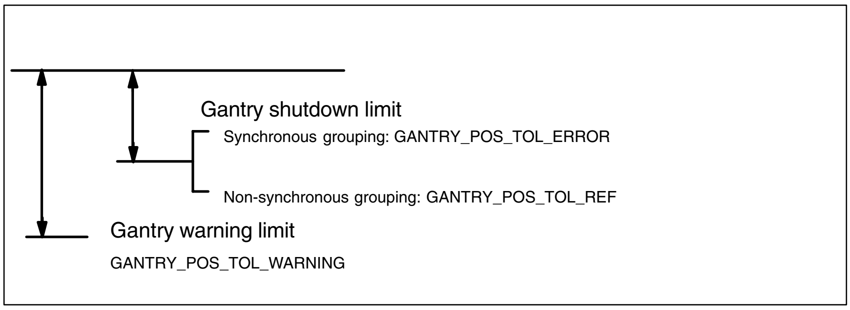
Machine data can be set to specify the following limit values for alarm output and termination of the traversing motion for specific axes:

- **Gantry warning limit:**
  If the position actual value difference exceeds the gantry warning limit (MD:37110: GANTRY_POS_TOL_WARNING), then the warning "Warning limit exceeded" is output to the operator. Furthermore, the interface signal (IS) "Gantry warning limit exceeded" (DB31, ... ; DBX101.3) is output to the PLC. When the actual value difference drops below the warning limit again, the alarm and IS are automatically canceled.

  No message is output if 0 is set for MD 37110: GANTRY_POS_TOL_WARNING.

- **Gantry shutdown limit**:
  When the maximum permissible position actual value deviation for the machine (MD 37120: GANTRY_POS_TOL_ERROR) is exceeded, alarm 10653 "Error limit exceeded" is output and the gantry axes stopped immediately along the braking ramp to prevent any damage to the mechanical components of the machine. The value in MD 37120: GANTRY_POS_TOL_ERROR is applied when the gantry grouping is synchronized. The alarm must be acknowledged with RESET. In addition, IS "Gantry trip limit exceeded" (DB31, ... ; DBX101.2) is output to the PLC.

  If the gantry axis grouping is not yet synchronized, the setting for the gantry trip limit is derived from MD 37130: GANTRY_POS_TOL_REF (gantry trip limit for referencing).




Gantry shutdown limit exceeded is also activated if the gantry grouping is jammed (no servo enable, gantry grouping in "Hold" state).

The monitoring functions are deactivated while the grouping is operating in "Follow-up" mode.

**Referencing and synchronization of gantry axes**

As the example “Gantry-type milling machine” shows (see Fig. 2-1), the forced coupling between gantry axes must remain in tact in all operating modes as well as immediately after power ON. In cases where an incremental measuring system is being used for the master or the synchronized axis, the reference point must be approached while maintaining the axis coupling immediately the machine is switched on.

After every axis in the grouping has approached its reference point, any misalignment which may exist between the axes must be eliminated (so-called “gantry synchronization process”). Once all axes are synchronized, IS “Gantry grouping is synchronized” (DB31, ... ; DBX101.5) is sent to the PLC.

The operational sequence for referencing and synchronizing gantry axes is described in detail in Section 2.2.

**Closed-loop control**

The dynamic control response settings of the coupled gantry axes must be identical (see Section 2.3). This ensures that the master and synchronized axes traverse in positional synchronism even during acceleration and braking in normal operation.

Load effects are compensated by the appropriate 611D drive of the master or synchronized axis.

**Response to disturbances**

When a disturbance occurs which causes shutdown of one gantry axis owing, for example, to cancellation of the controller enabling signal (example: EMERGENCY STOP), all other coupled gantry axes are also shut down.

**Separation of forced coupling**

In certain situations (e.g. one gantry axis is no longer referenced owing to an encoder failure), it may be necessary to correct or reduce the misalignment between the gantry axes prior to referencing. To do this, it must be possible to traverse the master or the synchronized axis **manually in the uncoupled state**.

The forced coupling between the gantry axes can be separated by means of MD 37140: GANTRY_BREAK_UP=1 (separate gantry grouping) followed by a RESET. The gantry axes can then be traversed separately by hand; the monitoring of the warning and trip limits is not operative in this state.

**Caution**

**If the gantry axes remain mechanically coupled, there is a risk of damage to the machine when the master or synchronized axes are traversed in this operating state!**

## 2.2 Referencing and synchronization of “gantry axes”

### 2.2.1 Introduction

**Misalignment after starting**

Immediately after the machine is switched on, the master and synchronized axes may not be ideally positioned in relation to one another (e.g. misalignment of a gantry). Generally speaking, this misalignment is relatively small so that the gantry axes can still be referenced.

In special cases (e.g. gantry axes were stopped owing to a disturbance, power failure or EMERGENCY STOP), the dimensional offset must be checked for permissible tolerance values and a compensatory motion executed if necessary before the axes are traversed.

To execute this compensatory motion, the gantry grouping must first be separated by means of MD 37140: GANTRY_BREAK_UP.

**Gantry synchronization process**

All gantry axes must first be referenced and then synchronized after the control system is switched on. During gantry synchronization, all gantry axes approach **the reference position of the gantry grouping in the decoupled state**. The reference position of the gantry grouping for referencing the gantry axes corresponds to the **reference position of the master axis** (MD 34100: REFP_SET_POS) or otherwise the **current actual position of the master axis**.

These operations for referencing and synchronizing the gantry axes are executed automatically in accordance with a special flowchart.

**Referencing operation**

The flowchart for the referencing of gantry axes which use an incremental measuring system is as follows:

Section 1: **Referencing of master axis**
Axis-specific referencing of the gantry axes is started by means of IS “Traversing key plus/minus” (DB31, ...; DBX4.7/4.6) of the master axis from the PLC user program when machine function REF is active.

The master axis approaches the reference point (operational sequence as for reference point approach (see **References:** /FB/, R1 “Reference Point Approach”). The appropriate synchronized axes traverse in synchronism with the master.

IS “Referenced/synchronized” of the master axis is output to indicate that the reference point has been reached.

Section 2: **Referencing of synchronized axes**
As soon as the master axis has approached its reference point, the synchronized axis is **automatically** referenced (corresponding to reference point approach (see **References:** /FB/, R1 “Reference Point Approach”). The dependency between the master axis and synchronized axis is reversed in the control for this phase so that the master now traverses in synchronism with the synchronized axis.

IS “Referenced/synchronized” of the synchronized axis is output to indicate that the reference point has been reached. The gantry axis dependency then reverts to its previous status.

If a further synchronized axis is defined in the grouping, then this is also referenced in the way described above.

Section 3: Gantry synchronization
Once all axes in the gantry grouping have been referenced, they must be synchronized with the defined reference position. The actual position of each gantry axis is first compared to the defined reference position of the master axis.

The next step in the operating sequence depends on the difference calculated between the actual values of the master and synchronized axes:

a) Difference is **lower** than gantry warning limit (MD 37110: GANTRY_POS_TOL_WARNING):

   The gantry synchronization process is started **automatically**. The message “Synchronization in progress gantry grouping x” is output during this process.

   All gantry axes traverse **in the decoupled state** at the velocity set in MD 34040: REFP_VELO_SEARCH_MARKER to the position value defined for the master axis in MD 34100: REFP_SET_POS.

   If the master axis uses absolute or distance-coded encoders, the gantry axes traverse (according to setting in MD 34330: REFP_STOP_AT_ABS_MARKER) either to the current actual position of the master axis or to the reference point. For this operation, the axes traverse at the same velocity as set for reference point approach (MD 34070: REFP_VELO_POS (reference point approach velocity).

   As soon as all gantry axes have reached their target position (ideal position), IS “Gantry grouping is synchronized” is set to “1” followed by re-activation of the gantry axis coupling. The position actual value of all axes in the gantry grouping must now be identical. The gantry synchronization process is now complete.

b) Difference is **higher** than the gantry warning limit for at least one synchronized axis

   IS “Gantry synchronization read to start” is set to “1” and the message “Wait for synchronization start of gantry grouping x” is output. The gantry synchronization process is not started automatically in this case, but must be started explicitly by the operator or from the PLC user program. The process is initiated by IS “Start gantry synchronization” on the master axis. The signal is set on the master axis. The operational sequence is then the same as that described above.

The following flowchart gives a graphic illustration of the referencing and synchronization processes.

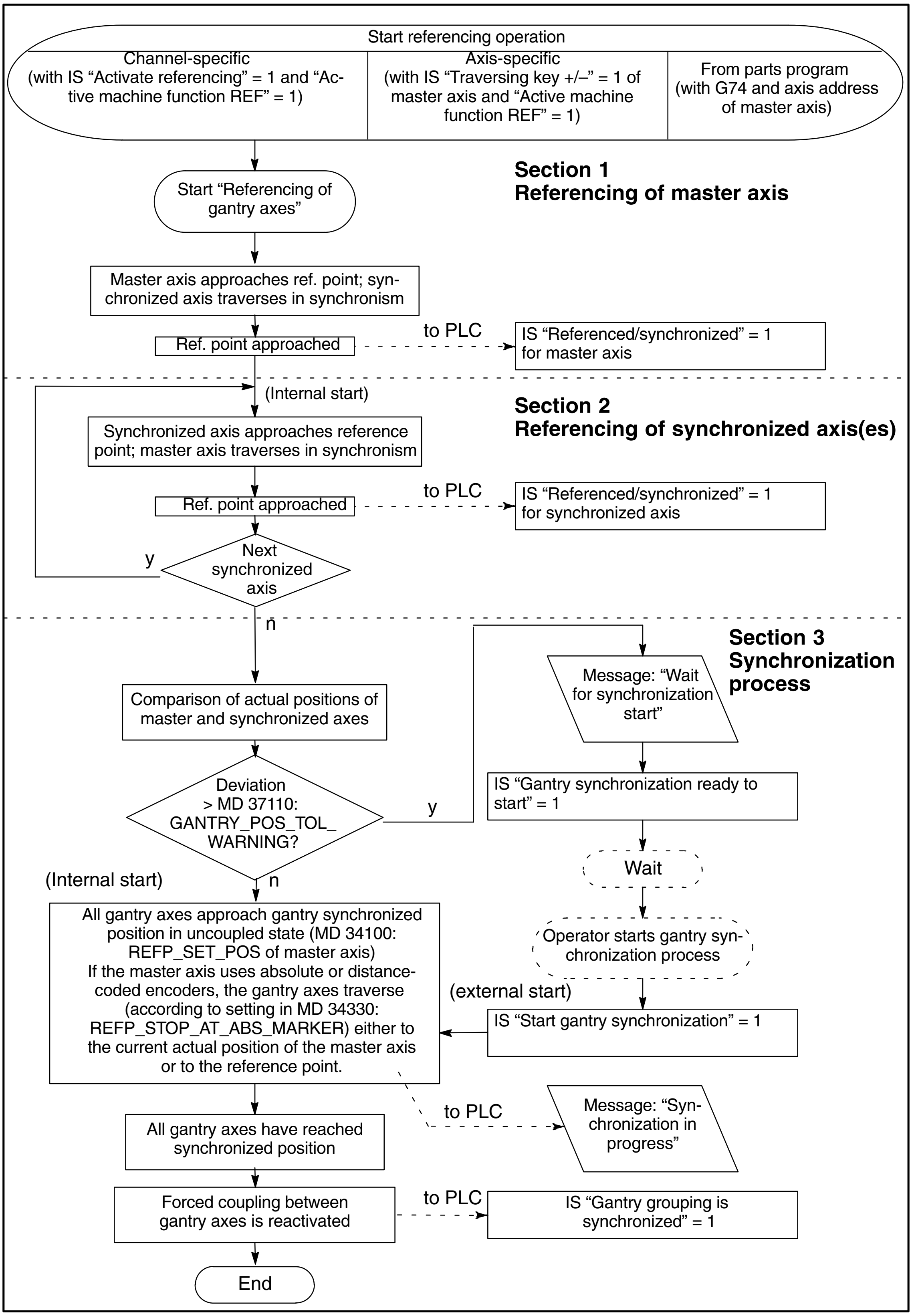



Fig. 2-2 Flowchart for referencing and synchronization of gantry axes

**Synchronization run**

A synchronization run must always be performed

- after the reference point approach of all axes included in a grouping
- if the axes become de-synchronized (see below).

**Interruption of process**

If the referencing process described above is interrupted as a result of disturbances or a RESET, proceed as follows:

1. **Abort within Section 1 or 2:**
   Approach reference point again with master axis (see Section 1)
2. **Abort within Section 3:**
   In cases where the gantry axes are not yet referenced (IS “Referenced/synchronized” = 1), the gantry synchronization process can be started again with IS “Synchronize gantry grouping”.

**Restarting gantry synchronization process**

Synchronization of the gantry axes can be started by means of IS “Start gantry synchronization” under the following conditions only:

- Machine function JOG/REF must be active (IS “Active machine function REF” (DB11, DBX5.2)
- IS (DB 31, ... DBX 101.5) “Gantry grouping is synchronized” = 0
- All axes in the group are within the tolerance or NST (DB 31, ... DBX 101.4) “Gantry synchronization ready to start” = 1
- No axis is being referenced in the relevant NC channel (IS “Referencing active” DB21–30, DBX33.0 = 0).

If the gantry synchronization process is **not started from the referencing process** by means of IS “Start gantry synchronization”, then the reference position (MD 34100: REFP_SET_POS) is not specified as the target position for the synchronized axes, but instead **the current actual position of the master axis**, which is approached by the axes in the uncoupled state.

---

**Note**

You can use IS DB31, ... DBX29.5 to interlock automatic synchronization for the leading axis. This always makes sense if no axis enabling signal has yet been issued for the axes. In this case, the synchronization process should also be started explicitly with IS DB31, ... DBX 29.4 = 1.

---

**Loss of synchronization**

The synchronization of the gantry grouping is lost (IS “Gantry grouping is synchronized” → 0) if

- the gantry axes were in “Follow-up” mode
- the reference position of a gantry axis is lost, e.g. during “Parking” (no measuring system active)
- one gantry axis is re-referenced (IS “referenced/synchronized” changes to 0)
- the gantry grouping was invalidated (MD 37140: GANTRY_BREAK_UP)

In cases where the gantry grouping has lost synchronization during operation as the result of a disturbance, then the gantry synchronization process can be restarted directly by means of IS “Start gantry synchronization) (condition: IS “Referenced/synchronized” = 1 for all axes in the grouping). In this case, the synchronized axes approach the current actual position of the master axis in the decoupled state.
(SW 5.3 and higher) If an Emergency Stop command is issued and then canceled again for a moving gantry grouping, and the two axes have drifted apart by less than the standstill tolerance of the slave axes, then the grouping is automatically re-synchronized. It is no longer necessary to switch to REFP mode. Automatic synchronization can be disabled via IS DBxx.DB29.5 of the slave axis.

**Selection of reference point**

To ensure that the shortest possible paths are traversed when the gantry axes are referenced, the reference point values of the master and synchronized axes in MD 34100: REFP_SET_POS should be identical. Allowance for deviations in distance between the zero mark and the reference point must be made for specific axes via MD 34080: REFP_MOVE_DIST and MD 34090: REFP_MOVE_DIST_CORR.

In the course of the referencing process, the reference point value of the master axis is specified as the target position for all axes in the grouping for the synchronization compensatory motion. This position is then approached without axis coupling. If the master axis uses absolute or distance-coded encoders, the gantry axes traverse (according to setting in MD 34330: REFP_STOP_AT_ABS_MARKER) either to the current actual position of the master axis or to the reference point.

If only one reference cam is used for the master and synchronized axes, then this must be taken into account in the PLC user program.

## 2.2.2 Automatic synchronization

Automatic synchronization can take place:

- In referencing mode (see 2.2.1).
- In other modes (SW 6 and higher) as described below.

If a gantry grouping is switched to follow-up mode, monitoring of the actual values between the master and synchronized axes is disabled. The grouping is no longer synchronized as a result. NST “Gantry grouping is synchronized” (from leading axis, DB31, ... DBX101.5) is set to 0, independent of the positions of the axes.

If the gantry grouping is switched from follow-up mode to position control mode, axis synchronism is automatically restored provided the actual-value monitor does not detect a difference between the positions of the master and synchronized axes greater than the setting in MD 36030: STANDSTILL_POS_TOL. In this case, a new setpoint is specified for the synchronized axis (axes) without interpolation. The positional difference detected earlier is then corrected by the position controller. The correction causes only the synchronized axis (axes) to move.

The motional sequence of the synchronized axis (axes) is analogous to the situation in which the grouping switches from the “Hold” state to position control mode. In this case, the position specified by the position controller before the grouping is halted is set again on condition that the zero speed monitor has not activated alarm 25040 (with follow-up as alarm reaction) in the meantime.

The same tolerance window is used for this mode of automatic synchronization as for the zero speed monitoring function: MD 36030: STANDSTILL_POS_TOL, with MD 36012: STOP_LIMIT_FACTOR applied as a function of parameter set.

---

**Note**

The signal IS “No automatic synchronization run” DB31, ... DBX29.5 blocks automatic synchronization in all modes, except for homing mode. If the automatic synchronization is to be activated here, specify IS “No automatic synchronization run” DB31, ... DBX29.5 = 0. Then switch one of the axes in the gantry group from follow-up mode to position controlled mode. This is achieved with:

IS “Follow-up mode” DB31, ... DBX1.4 = 1 and
NST “Delete distance to go/spindle reset” DB31, ... DBX2.1 = 1 signal transition from 1 to 0 to 1”.

---

### 2.2.3 Special features

**2nd position measuring systems per gantry axis**

Different types of position measuring system can be mounted on the gantry axes of a grouping. Furthermore, each gantry axis is capable of processing two position measuring systems, it being possible to switch over from one system to the other at any time (IS “Position measuring system 1/2” (DB31 ... DBX1.5 and 1.6)).

The maximum tolerance for position actual value switchover (MD 36500: ENC_CHANGE_TOL) should be set to a lower value than the gantry warning limit.

The two position measuring systems must, however, have been referenced beforehand. The relevant measuring system must be selected before referencing is initiated. The operational sequence is then the same as that described above.

**Channel-specific referencing**

Gantry axes can also be referenced on a channel-specific basis by means of IS “Activate referencing” (DB21–28, DBX1.0). The value of axial MD 34110: REFP_CYCLE_NR of the master axis is used as the axis sequence for channel-specific referencing. After the reference point of the master axis has been reached, the synchronized axes are referenced first as described above.

**Referencing from parts program with G74**

The referencing and synchronization process for gantry axes can also be initiated from the parts program by means of command G74. In this case, only the axis name of the master axis may be programmed. The operational sequence is analogous to that described for axis-specific referencing.

**Position measuring system with distance-coded reference marks**

To avoid the necessity of traversing large distances to approach the reference point, it is possible to use a position measuring system with distance-coded reference marks as the sole or the second measuring system. In this way the measuring system is referenced after traversal of a short path (e.g. 20 mm). The procedure for referencing the gantry axes is the same as that described for normal incremental measuring systems (**References:** /FB/, R1 “Reference Point Approach”).

**or absolute encoder**

During the course of the synchronization compensatory motion, all axes in the gantry axis grouping traverse to the reference point value of the master axis defined in MD 34100: REFP_SET_POS in the decoupled state. If the master axis uses absolute or distance-coded encoders, the gantry axes traverse (according to setting in MD 34330: REFP_STOP_AT_ABS_MARKER) either to the current actual position of the master axis or to the reference point.

**Activation of axis compensations**

Compensation functions can be activated for both the master axis and the synchronized axes. Compensation values are applied separately for each individual gantry axis. These values must therefore be defined and entered for the master axis and the synchronized axes during start-up.

The compensations do not become operative internally in the control until the axis is referenced or the gantry grouping synchronized. In this case, the following applies:

| Compensation type | Takes effect when | PLC interface signal |
|---|---|---|
| Backlash compensation | Axis is referenced | “Referenced/Synchronized” |
| Leadscrew error compensation | Axis is referenced | “Referenced/Synchronized” |
| Sag compensation | Gantry grouping is synchronized | “Gantry grouping is synchronized” |
| Temperature compensation | Gantry grouping is synchronized | “Gantry grouping is synchronized” |

If a movement by the synchronized axis (axes) is caused by an active compensation, a travel command is displayed for the synchronized axis (axes) independently of the master axis.

**Monitoring functions effective**

Analogous to normal NC axes, the following monitoring functions do not take effect for gantry axes until the reference point is reached (IS “Referenced/synchronized”):

- Working area limitations
- Software limit switches
- Protection zones.

The axial machine data values are used as monitoring limit values for the synchronized axes as well.

**Multi-channel block search in SW 6.1 and higher**

The cross-channel block search in Program Test mode (SERUPRO ”**Se**rch **R**un by **Pro**gram Test”) can be used to simulate the traversal of gantry axis groupings in SW 6.2 and higher.

---

**Note**

For further information about the SERUPRO multi-channel block search please refer to:

**References** /FB/, K1, ”Mode Group, Channel, Program Operation”
2.4 Program testing

---

## 2.3 Start-up of “gantry axes”

**General**

Owing to the forced coupling which is normally present between master and synchronized gantry axes, the gantry axis grouping must be started up as if it were an axis unit. For this reason, the axial machine data for the master and synchronized axes must always be defined and entered jointly.

**References**: /IAD/, SINUMERIK 840D Installation and Start-Up Guide

**Special points** to be noted with regard to starting up gantry axes are described below.

**Axis traversing direction**

As part of the start-up procedure, a check must be made to ensure that the direction of rotation of the motor corresponds to the desired traversing direction of the axis (correct by means of axial MD 32100: AX_MOTION_DIR (traversing direction)).

**Activation of the axis grouping**

The following must be defined for the gantry axis in MD 37100: GANTRY_AXIS_TYPE:

- To which gantry grouping (1, 2 or 3) the axis must be assigned.
- Whether it is to act as the master axis (single-decade MD value only) or as a synchronized axis.

It must be noted for start-up purposes that the axes in a gantry grouping are all declared as linear axes or all declared as rotary axes (MD 30300: IS_ROT_AX).

Table 2-1 Examples of definition of the gantry axis grouping:

| MD: GANTRY_AXIS_TYPE | Gantry axis | Gantry grouping |
|---|---|---|
| 0 | None | – |
| **1** | Master axis | 1 |
| **11** | Synchronized axis | 1 |
| **2** | Master axis | 2 |
| **12** | Synchronized axis | 2 |
| **3** | Master axis | 3 |
| **13** | Synchronized axis | 3 |

**Entering gantry trip limits**

For the purposes of monitoring the position actual value deviation between the synchronized axis and the actual position of the master axis, the trip limit values (MD: 37120 GANTRY_POS_TOL_ERROR or MD 37130: GANTRY_POS_TOL_REF) must be entered for the master axis and for the synchronized axis in accordance with the machine manufacturer’s data.

---

**Note**

The control must then be switched off and then on again because the gantry axis definition and the trip limit values only take effect after power ON.

---

### Response to set-point changes and disturbances

Since the digital 611D drives respond well to disturbances and setpoint changes, there is no need for a compensatory control between the gantry axes. However, the gantry axes can only operate in exact synchronism if the parameters for the control circuits of the master and synchronized axes are set to the **same dynamic** response value.

To ensure the best possible synchronism, the master axis and synchronized axis must be capable of the **same dynamic response** to **setpoint changes**. The axial control loops (position, speed and current controllers) should each be set to the **optimum** value so that disturbances can be eliminated as quickly and efficiently as possible. The **dynamic response adaptation** function in the setpoint branch is provided to allow differing dynamic responses of axes to be matched without loss of control quality.

The following control parameters must be set to the optimum axial value for both the master axis and the synchronized axis:

- Servo gain (MD 32200: POSCTRL_GAIN)
- Feedforward control parameters

  MD 32620: FFW_MODE
  MD 32610: VELO_FFW_WEIGHT
  MD 32650: AX_INERTIA
  MD 32800: EQUIV_CURRCTRL_TIME
  MD 32810: EQUIV_SPEEDCTRL_TIME

  **References**: /FB/, K3 "Compensations"

The following control parameters must be set to the same value for the master axis and synchronized axis:

- Fine interpolator type (MD 33000: FIPO_TYPE)
- Axial jerk limitation

  MD 32400: AX_JERK_ENABLE
  MD 32410: AX_JERK_TIME
  MD 32420: JOG_AND_POS_JERK_ENABLE
  MD 32430: JOG_AND_POS_MAX_JERK

  **References**: /FB/, G2 "Velocities, Setpoint/Actual Value Systems, Closed-Loop Control"

### Dynamic response matching

The master axis and the coupled axis must be capable of the same dynamic response to setpoint changes. The "same" dynamic response means: The following errors are equal in magnitude when the axes are operating at the same speed.

The dynamic response adaptation function in the setpoint branch makes it possible to obtain an excellent match in the response to setpoint changes between axes which have different dynamic characteristics (control loops). The difference in equivalent time constants between the dynamically "weakest" axis and the other axis in each case must be specified as the dynamic response adaptation time constant.

**Example**

When the speed feedforward control is active, the dynamic response is primarily determined by the equivalent time constant of the "slowest" speed control loop.

Master axis MD 32810: EQUIV_SPEEDCTRL_TIME [n] = 5ms
Synchronized axis MD 32810: EQUIV_SPEEDCTRL_TIME [n] = 3ms

→ Time constant of dynamic response adaptation for synchronized axis:
MD 32910: DYN_MATCH_TIME [n] = 5ms – 3ms = 2ms

The dynamic response adaptation function must be activated axially by means of MD 32900: DYN_MATCH_ENABLE.

**Check of dynamic response adaptation:**
The following errors of the master and synchronized axes must be equal in magnitude when the axes are operating at the same speed!

For the purpose of fine tuning, it may be necessary to adjust servo gain factors or feedforward control parameters slightly to achieve an optimum result.

**Referencing of gantry axes**

The positions of the reference points of the master and synchronized axes must first be set to almost identical values.

To ensure that the synchronization compensatory motion of the gantry axes is not automatically started, the gantry warning limit (MD 37100: GANTRY_POS_TOL_WARNING) must be set to 0 prior to referencing on initial start-up. This will prevent the output of a warning message during the traversing motion.

In cases where an excessively high additional torque is acting on the drives due to misalignment between the master and synchronized axes, the gantry grouping must be aligned before the axes are traversed.
The gantry axes must then be referenced as described in Section 2.2 and
**References:** /FB/, R1 "Reference Point Approach".

After the master and synchronized axes have been referenced, the difference between them must be measured (comparison of position actual value indication in "Service axes" display of "Diagnosis" operating area). This difference must be applied as the reference point offset (MD34080: REFP_MOVE_DIST and MD 34090: REFP_MOVE_DIST_CORR).

The differences in distance between the zero mark and reference point must also be calculated for each gantry axis and adjusted in MD 34080: REFP_MOVE_DIST and MD 34090: REFP_MOVE_DIST_CORR in such a way that the position actual values of the master and synchronized axes are identical after execution of the compensatory motion.

**Synchronization of gantry axes**

The gantry synchronization process must be activated with IS "Start gantry synchronization" (see Section 2.2). Once the axes have been synchronized (IS "Gantry grouping is synchronized" = 1), the dimensional offset between the master and synchronized axes must be checked to ensure that it equals 0. Corrections may need to be made in the above-mentioned machine data.

**Input of gantry warning limit**

Once the reference point values for the master and synchronized axes have been set optimally so that the gantry axes are perfectly aligned with one another after synchronization, the warning limit values for all axes must be entered in MD 37110: GANTRY_POS_TOL_WARNING.

To do this, the value must be increased incrementally until the value is just below the alarm (limit exceeded) response limit. It is particularly important to check the acceleration phases.

This limit value also determines the position deviation value at which gantry synchronization is automatically started in the control.

**Calculation and activation of compensations**

In cases where the gantry axes require compensation (backlash, sag, temperature or leadscrew error), the compensation values for the master axis **and** the synchronized axis must be calculated and entered in the appropriate parameters or tables.

**References:** /FB/, K3 “Compensations”

**Function generator/ measuring function**

**Up to and including SW 3.1**
The function generator and measuring function must only be activated for the **master axis** in all software versions up to and including SW 3.1.
The **synchronized axis** automatically follows (by being coupled to the actual value of the leading axis). If the zero speed control responds on the synchronized axis, increase the size of the monitoring window temporarily.

**Caution**

Activation of the function generator and measuring function on the synchronized axis or master and synchronized axis simultaneously is not prevented by an internal monitor in software versions up to and including SW 3.1 but if used incorrectly, can cause damage to the machine.

**SW 3.2 and higher**
In SW 3.2 and higher, activation of the function generator and measuring function on the synchronized axis is canceled with an error message.
If it becomes necessary to activate the synchronized axis (e.g. to perform measurements on the machine), the master axis and synchronized axis must be exchanged temporarily.

Special cases

If **individual** axes have to be activated, the gantry groups must be temporarily canceled. As the second axis no longer travels in synchronism with the first axis, the activated axis must not be allowed to traverse beyond the positional tolerance.

If the gantry grouping is canceled, the following points must be noted:

- Always activate the traversing range limits and set them to the lowest possible values (position tolerance)
- Synchronize the gantry grouping first if possible and then execute a POWER-ON-RESET **without** referencing the axes again. This ensures that the traversing range limits always refer to the same position (i.e. that which was valid on power ON).
- Avoid using the step-change function. Position step changes are only permissible if they stay within the permitted tolerance.
- Always use an offset of 0 for the function generator and measuring function in contrast to the recommendations for normal axes.
- Set the amplitudes for function generator and measuring function to such low values that the activated axis traverses a shorter distance than the position tolerance allows. Always activate the traversing range limits as a check (see above).

**References**: /FBA/, DD2 "Speed control loop"

---

**Note**

As a supplement to the more general description given here of features of start-up and dynamic control response of drives, a complete example of a concrete constellation defined on the basis of its machine data can be found in Chapter 6.

---

### Start-up support for gantry groupings

**SW 5.1 and higher**

The start-up functions "Function generator" and "Measurement" are parameterized via PI services, as in earlier SW. All parameterized axes commence traversing when the NC Start key on the MCP panel is pressed in JOG mode.

A window is displayed in the "Measuring function and function generator in gantry grouping" operator interface. Two amplitude values, each with an offset and bandwidth, must be entered in this window. The first amplitude value applies to the measuring axis and the second to the other coupled axes.

## 2.4 PLC interface signals for "gantry axes"

**Special IS for gantry axes**

The special PLC interface signals of the coupled gantry axes are taken via the axial PLC interface of the master or synchronized axes. Table 2-2 below shows all special gantry-PLC interface signals and indicates whether the IS is evaluated on the master or the synchronized axis.

Table 2-2 Assignment of gantry-PLC interface signals to master and synchronized axes

| PLC interface signal | PLC ↔ NCK | DB31, ... ; DBX ... | Master axis | Synchronized axis |
|---|---|---|---|---|
| Start gantry synchronization | → | 29.4 | X | |
| No automatic synchronization process | → | 29.5 | X | |
| Gantry axis | ← | 101.7 | 1 | 1 |
| Gantry master axis | ← | 101.6 | 1 | 0 |
| Gantry grouping is synchronized | ← | 101.5 | X | |
| Gantry synchronization ready to start | ← | 101.4 | X | |
| Gantry warning limit exceeded | ← | 101.3 | | X |
| Gantry trip limit exceeded | ← | 101.2 | | X |

**Effect of the axial IS on gantry axes**

a) **Axial interface signals from PLC to axis (PLC –> NCK)**

The axial interface signals from the PLC to the axis are always referred to all gantry axes in the grouping. In this case, all gantry axes (master and synchronized axis) have equal priority.

If, for example, the IS "Controller enable" (DB31, ... ; DBX2.1) of the master axis is set to "0", then all axes in the gantry grouping are shut down at the same instant.

Table 2-3 shows the effect of individual interface signals (from PLC to axis) on gantry axes:

Table 2-3 Effect of interface signals from PLC to axis on master and synchronized axes

| PLC interface signal | DB31, ... ; DBX ... | Effect on | |
|---|---|---|---|
| | | Master axis | Synchronized axis |
| Axis/spindle disable | 1.3 | On all axes in gantry grouping | No effect |
| Position measuring system 1/2 | 1.4 and 1.5 | Axial 1) | Axial 1) |
| Controller enable | 2.1 | On all axes in gantry grouping 2) | |
| Delete distance to go (axial) | 2.2 | Axial | No effect |
| Clamping in progress | 2.3 | Axial | Axial |
| Reference point value 1–4 | 2.4–2.7 | Axial | Axial |
| Feed stop | 4.4 | On all axes in gantry grouping | |
| Hardware limit switch plus/minus | 12.0 and 12.1 | Axial alarm: Brake request on all axes in gantry grouping | |
| 2nd software limit switch plus/minus | 12.2 and 12.3 | Axial | Axial |

Table 2-3 Effect of interface signals from PLC to axis on master and synchronized axes

| PLC interface signal | DB31, ... ; DBX ... | Effect on | |
|---|---|---|---|
| | | Master axis | Synchronized axis |
| Ramp-function generator fast stop (RFGFS) | 20.1 | On all axes in gantry grouping | |
| Select drive parameter set | 21.0 – 21.2 | Axial | Axial |
| Pulse enable | 21.7 | Axial | Axial |

1. IS “Position measuring system 1/2” (DB31, ... ; DBX1.5 and 1.6)
   The signal for switchover between position measuring systems 1 and 2 applies individually for each gantry axis. However, deactivation of both position measuring systems (called parked position) applies as a common signal for all gantry axes.
2. IS “Controller enable” (DB31, ... ; DBX2.1)
   When the controller enabling signal of one gantry axis is canceled, then all axes in the gantry grouping are shut down simultaneously. The method by which shutdown is implemented (e.g. with fast stop) is identical for all gantry axes.

Depending on the IS “Follow-up mode” (DB31, ... ; DBX1.4), either the “Follow-up” state (IS of one gantry axis = 1) or the “Stop” state (IS of all gantry axes = 0) is activated for all gantry axes.

b) **Axial interface signals from axis to PLC (NCK –> PLC)**

Each of the axial, axis-to-PLC interface signals for the synchronized axis and the master axis is always set on an axis-specific basis and output to the PLC.

**Example:**
IS “Referenced/synchronized 1/2” (DB31, ... ; DBX60.4 or 60.5)

**Exception:**
When the master axis is traversing, IS “Traversing command plus or minus” (DB31, ... ; DBX64.6 and 64.7) is also set for the synchronized axis.

## 2.5 Miscellaneous points regarding “gantry axes”

**Manual traverse**

It is not possible to traverse a synchronized axis directly by hand in JOG mode. Traverse commands entered via the traversing keys of the synchronized axis are ignored internally in the control. Rotation of the handwheel for the synchronized axis has no effect either.

**Handwheel override**

An overriding motion by means of the handwheel can only be applied to the master axis in coupled axis mode. In this case, the synchronized axes traverse in synchronism with the master.

**DRF offset**

A DRF offset can only be applied to the master axis. In this case, the synchronized axes traverse in synchronism with the master.

**Programming in parts program**

Only the master axis of a gantry axis grouping may be programmed in the parts program. An alarm is generated when a synchronized axis is programmed.

**PLC or command axes**

Only the master axis of the gantry grouping can be traversed from the PLC by means of FC18 or as a command axis via synchronous actions.
**References:** /FB/, P3, “Basic PLC Program”
/FBSY/, Synchronized Actions

**PRESET**

The PRESET function can only be applied to the master axis. All axes in the gantry grouping are reevaluated internally in the control when PRESET is activated. The gantry axes then lose their reference and synchronization (IS “Gantry grouping synchronized” = “0”).

**Axis release**

All axes in the gantry grouping are released automatically in response to a RELEASE command (master axis).

**Display data**

The position actual value display shows the actual values of both the master axis and the synchronized axes. The same applies to the service display values in the “Diagnosis” operating area.

**SW limit switch**

The SW limit switch monitor is processed for the master axis only. If the master axis crosses the limit switch, all axes in the gantry grouping are braked to a standstill.

**Differences between “gantry axes” and “coupled axes” functions**

The main differences between the gantry axes and coupled axes functions are listed below:

- The axis coupling between the gantry axes must always be active. It is therefore not possible to separate the axis coupling between “Gantry axes” by means of the parts program. In contrast, the coupled axis grouping can be separated by means of the parts program and the axes then traversed individually.
- With “Gantry axes”, the difference between the position actual values of the master and synchronized axes is continuously monitored and the traversing motion terminated in response to illegal deviations. No such monitoring takes place with the “Coupled axes” function.
- Gantry axes must remain coupled even during referencing. For this reason, special procedures are applied for the reference point approach of gantry axes. In contrast, “Coupled axes” are referenced as individual axes.
- To allow “Gantry axes” to traverse without a mechanical offset, the dynamic control response settings of the synchronized axes and the master axis must be identical. In contrast, the “Coupled axes” function permits axes with different dynamic control response characteristics to be coupled.

**References:** /FB/, M3 “Coupled Axes”

■

# Notes

# Supplementary Conditions 3

**Availability of “gantry axes” function**

This function is an option and available for

- SINUMERIK 840D with NCU 572/573, with SW2 and higher.
- SINUMERIK 810D with CCU 2, SW 2 and higher.
- SINUMERIK 810D with CCU1, SW 3.2 and higher.

■

# Notes

# Data Descriptions (MD, SD) 4

## 4.1 Axis-specific machine data

<table>
<tr><td colspan="2">37100<br>MD number</td><td colspan="4">GANTRY_AXIS_TYPE<br>Gantry axis definition</td></tr>
<tr><td colspan="2">Default setting: 0</td><td colspan="2">Min. input limit: 0</td><td colspan="2">Max. input limit: 13</td></tr>
<tr><td colspan="3">Changes effective after power ON</td><td colspan="2">Protection level: 2/4</td><td>Unit: –</td></tr>
<tr><td colspan="4">Data type: BYTE</td><td colspan="2">Applies from SW 2.1</td></tr>
<tr><td>Meaning:</td><td colspan="5">The following must be defined with the axial MD: GANTRY_AXIS_TYPE in the form of a two-decade value specification:<br>• Whether the axis belongs to a gantry grouping and if so, to which (1st decade)<br>• and whether the axis is declared within the grouping as a master axis (MD value has only 1 decade) or as a synchronized axis (2nd decade is set to value 1)<br><br>| 2nd decade | 1st decade |<br><br>Gantry axis type<br>(no entry): Masteraxis<br>1: Synchronized axis<br><br>Gantry axis grouping<br>0: No gantry axis<br>1: Gantry axis grouping 1<br>2: Gantry axis grouping 2<br>3: Gantry axis grouping 3<br><br>A maximum total of 3 gantry axis groupings can be defined.<br><br>A gantry axis grouping has a master axis and at least one synchronized axis (a maximum of two synchronized axes is possible).<br>Declaration conditions:<br>• All axes in a gantry axis grouping must be declared either as linear or as rotary axes (MD 30300: IS_ROT_AX).<br>• A spindle may not be declared within a gantry axis grouping.<br>• A synchronized axis may not be declared either as a geometry axis or as a “concurrent positioning axis”.<br>• A synchronized axis may not be declared as the master axis of another gantry grouping.<br><br>The gantry axis definition is subjected to a plausibility check internally in the control; if it is incorrectly parameterized, alarm 10650 “Incorrect gantry machine data” or 10651 “Gantry unit undefined” is output.</td></tr>
<tr><td>MD irrelevant for ......</td><td colspan="5">SINUMERIK FM-NC; SINUMERIK 840D with NCU 571</td></tr>
<tr><td>Application example(s)</td><td colspan="5">0: No gantry axis<br>1: Axis is master axis in gantry grouping 1<br>11: Axis is synchronized axis in gantry grouping 1<br>2: Axis is master axis in gantry grouping 2<br>12: Axis is synchronized axis in gantry grouping 2<br>3: Axis is master axis in gantry grouping 3<br>13: Axis is synchronized axis in gantry grouping 3</td></tr>
<tr><td>Special cases, errors, ......</td><td colspan="5">Alarm 10650 “Incorrect gantry machine data” and 10651 “Gantry unit undefined” in response to incorrect gantry axis definition.</td></tr>
<tr><td>Related to ....</td><td colspan="5">MD 37110: GANTRY_POS_TOL_WARNING Gantry warning limit<br>MD 37120: GANTRY_POS_TOL_ERROR Gantry trip limit<br>MD 37130: GANTRY_POS_TOL_REF Gantry trip limit for referencing</td></tr>
</table>

| | | | | | |
|---|---|---|---|---|---|
| **37110**<br>MD number | **GANTRY_POS_TOL_WARNING**<br>Gantry warning limit | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: plus | |
| Changes effective after RESET | | | Protection level: 2/4 | | Unit:<br>Linear axis: mm<br>Rotary axis: Degrees |
| Data type: DOUBLE | | | | Applies from SW 2.1 | |
| Meaning: | Value > 0<br>With gantry axes, the difference between the position actual values of the master and synchronized axes is constantly monitored.<br>The MD: GANTRY_POS_TOL_WARNING is used to define a limit value for the position actual value difference; when the limit is exceeded, warning 10652 “Warning limit exceeded” is output. However, the gantry axes are not shut down internally in the control. The warning threshold must therefore be selected such that the machine can withstand the position actual value deviation between the gantry axes without sustaining mechanical damage.<br>Furthermore, IS “Gantry warning limit exceeded” (DB31, ... ; DBX101.3) to the PLC is set to “1”. The PLC user program can thus initiate the necessary measures (e.g. program interruption at block end) when the warning limit is exceeded.<br>As soon as the current position actual value difference has dropped below the limit again, the message is canceled and interface signal “Gantry warning limit exceeded” reset.<br>Effect of gantry warning limit on gantry synchronization process:<br>The position actual value difference between the master and synchronized axes is determined during gantry synchronization. If the deviation is lower than the warning limit, the synchronizing motion of the gantry axes is automatically started internally in the control.<br>The synchronizing motion must otherwise be initiated via the PLC interface (IS “Start gantry synchronization process”).<br>Value = 0<br>Setting MD: GANTRY_POS_TOL_WARNING to 0 deactivates the monitoring for violation of the warning limit.<br>Gantry synchronization is not initiated internally in the control. | | | | |
| MD irrelevant for ...... | SINUMERIK FM-NC; SINUMERIK 840D with NCU 571 | | | | |
| Special cases, errors, ...... | Alarm 10652 “Warning limit exceeded” in response to violation of gantry warning limit. | | | | |
| Related to .... | MD 37100: GANTRY_AXIS_TYPE<br>MD 37120: GANTRY_POS_TOL_ERROR<br>MD 37130: GANTRY_POS_TOL_REF<br>IS “Gantry warning limit exceeded”<br>IS ”Start gantry synchronization run” | | | Gantry axis definition<br>Gantry trip limit<br>Gantry trip limit for referencing<br>(DB31, ... ; DBX101.3)<br>(DB31, ... ; DBX29.4) | |

| | | | | | |
|---|---|---|---|---|---|
| **37120**<br>MD number | **GANTRY_POS_TOL_ERROR**<br>Gantry trip limit | | | | |
| Default setting: 0.0 | | Min. input limit: 0 | | Max. input limit: plus | |
| Changes effective after power ON | | | Protection level: 2/4 | | Unit:<br>Linear axis: mm<br>Rotary axis: Degrees |
| Data type: DOUBLE | | | | Applies from SW 2.1 | |

| 37120<br>MD number | GANTRY_POS_TOL_ERROR<br>Gantry trip limit |
|---|---|
| Meaning: | With gantry axes, the difference between the position actual values of the master and synchronized axes is constantly monitored. The MD: GANTRY_POS_TOL_ERROR. Monitoring for violation of this limit value takes place only if the gantry axis grouping is already **synchronized** (IS "Gantry grouping is synchronized" = 1); otherwise the value set in MD 37130: GANTRY_POS_TOL_REF is used.<br>When the limit value is exceeded, alarm 10653 "Error limit exceeded" is output. The gantry axes are immediately shut down internally in the control to prevent any damage to the machine.<br>In addition, IS "Gantry trip limit exceeded" to the PLC is set to "1". |
| MD irrelevant for ...... | SINUMERIK FM-NC; SINUMERIK 840D with NCU 571 |
| Special cases, errors, ...... | Alarm 10653 "Error limit exceeded" in response to violation of gantry trip limit. |
| Related to .... | MD 37100: GANTRY_AXIS_TYPE — Gantry axis definition<br>MD 37110: GANTRY_POS_TOL_WARNING — Gantry warning limit<br>MD 37130: GANTRY_POS_TOL_REF — Gantry trip limit for referencing<br>IS "Gantry grouping is synchronized" — (DB31, ... ; DBX101.5)<br>IS "Gantry trip limit exceeded" — (DB31, ... ; DBX101.2) |

| 37130<br>MD number | GANTRY_POS_TOL_REF<br>Gantry trip limit when referencing | |
|---|---|---|
| Default setting: 0.0 | Min. input limit: 0 | Max. input limit: plus |
| Changes effective after power ON | Protection level: 2/4 | Unit:<br>Linear axis: mm<br>Rotary axis: Degrees |
| Data type: DOUBLE | Valid from SW 2.1 | |
| Meaning: | With gantry axes, the difference between the position actual values of the master and synchronized axes are continuously monitored. The MD: GANTRY_POS_TOL_REF the maximum permissible deviation in position actual values between the synchronized axis and the master axis that is monitored if the gantry axis grouping is **not yet synchronized** (IS "Gantry grouping is synchronized" = "0") must be set.<br>When the limit value is exceeded, alarm 10653 "Error limit exceeded" is output. The gantry axes are **immediately shut down** in the control to prevent any damage to the machine.<br>In addition, IS "Gantry trip limit exceeded" to the PLC is set to "1". | |
| MD irrelevant for ...... | SINUMERIK FM-NC; SINUMERIK 840D with NCU 571 | |
| Special cases, errors, ...... | Alarm 10653 "Error limit exceeded" in response to violation of gantry trip limit. | |
| Related to .... | MD 37100: GANTRY_AXIS_TYPE — Gantry axis definition<br>MD 37110: GANTRY_POS_TOL_WARNING — Gantry warning limit<br>MD 37120: GANTRY_POS_TOL_ERROR — Gantry trip limit<br>IS "Gantry grouping is synchronized" — (DB31, ... ; DBX101.5)<br>IS "Gantry trip limit exceeded" — (DB31, ... ; DBX101.2) | |

| 37140<br>MD number | GANTRY_BREAK_UP<br>Invalidate gantry axis grouping | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 1 |
| Changes effective after RESET | Protection level: 2/4 | Unit: – |
| Data type: BOOLEAN | Applies from SW 2.1 | |

| **37140**<br>MD number | **GANTRY_BREAK_UP**<br>Invalidate gantry axis grouping | |
|---|---|---|
| Meaning: | **GANTRY_BREAK_UP = “0”**<br>The forced coupling of the gantry axis grouping remains valid. Monitoring of violation of the gantry warning or trip limit is effective.<br>**GANTRY_BREAK_UP = “1”**<br>This **cancels the forced coupling of the gantry grouping**, thus allowing all gantry axes in this grouping to be traversed individually by hand. The monitoring for violation of the gantry warning or trip limit is deactivated. IS “Gantry grouping is synchronized” is set to “0”.<br><br>**Caution:**<br>**In cases where the gantry axes are still mechanically coupled, the machine may sustain damage in this operating state when the master or synchronized axis is traversed!**<br>**The gantry axes cannot be referenced individually.** | |
| MD irrelevant for ...... | SINUMERIK FM-NC; SINUMERIK 840D with NCU 571 | |
| Related to .... | MD 37100: GANTRY_AXIS_TYPE | Gantry axis definition |
| | MD 37110: GANTRY_POS_TOL_WARNING | Gantry warning limit |
| | MD 37130: GANTRY_POS_TOL_REF | Gantry trip limit for referencing |
| | IS “Gantry grouping is synchronized” | (DB31, ... ; DBX101.5) |
| | IS ”Gantry trip limit exceeded” | (DB31, ... ; DBX101.2) |

■

# Signal Descriptions 5

## 5.1 Signals to axis/spindle

| | | | |
|---|---|---|---|
| **DB31, ... ;**<br>**DBX29.4**<br>Data block | **Start gantry synchronization**<br><br>Signal(s) to NC (PLC → NC) | | |
| Edge evaluation: no | | Signal(s) updated: Cyclically | Signal(s) valid from SW 2.1 |
| Signal state 1 or signal transition 0 ——> 1 | Request from PLC user program to synchronize the master axis with the assigned synchronized axes (MD 37100: GANTRY_AXIS_TYPE) (i.e. all gantry axes approach the reference position of the gantry grouping in the decoupled state).<br><br>Synchronization of the gantry axes can be started only under the following conditions:<br>• Machine function REF must be active (IS “Active machine function REF” = “1”)<br>• IS “Gantry grouping is synchronized” = “0”<br>• IS “Gantry synchronization ready to start” = “1”<br>• No axis is being referenced in the appropriate NC channel (IS “Referencing active” = “0”) | | |
| Signal state 0 or signal transition 1 ——> 0 | The PLC user program can then, for example, reset the interface signal to signal state “0” on completion of gantry synchronization (IS “Gantry grouping is synchronized” = “1”).<br><br>If the IS is left continuously in state “1”, the gantry synchronization process would be started automatically as soon as the above mentioned conditions are fulfilled. | | |
| Signal irrelevant for ...... | Gantry synchronized axis | | |
| Application example(s) | If the deviation between the position actual values and the reference position is greater than the gantry warning threshold after referencing, then automatic gantry synchronization is not started and IS “Gantry synchronization ready to start” is set to “1”.<br><br>Synchronization of the gantry axes can be started by the user or the PLC user program with IS “Start gantry synchronization”. | | |
| Related to .... | IS “Gantry grouping is synchronized”<br>IS “Gantry synchronization run ready to start”<br>IS “Active machine function REF”<br>IS “Referencing active” | (DB31, **...** ; DBX101.5)<br>(DB31, **...** ; DBX101.4)<br>(DB11, DBX5.2)<br>(DB21–30, DBX33.0) | |

| | | |
|---|---|---|
| **DB31, ... ;**<br>**DBX29.5**<br>Data block | **Start automatic synchronization**<br><br>Signal(s) to NC (PLC → NC) | |
| Edge evaluation: no | | Signal(s) valid from<br>SW 5 |
| Signal state 1 or signal transition 0 ––> 1 | No automatic synchronization process | |
| Signal state 1 or signal transition 1 ––> 0 | The automatic synchronization process is active | |
| Signal irrelevant for ...... | Gantry synchronized axis | |
| Application example(s) | The automatic synchronization process can be locked with a VDI signal to the axial PLC –> NC interface of the master axis. This always makes sense when the axes are not activated per default. In this case, the synchronization process should also be started explicitly. | |

## 5.2 Signals from axis/spindle

| | | |
|---|---|---|
| **DB31, ... ;**<br>**DBX101.2**<br>Data block | **Gantry trip limit exceeded**<br><br>Signal(s) to PLC (NC → PLC) | |
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW 2.1 |
| Signal state 1 or signal transition 0 ——> 1 | The difference between the position actual values of the master and synchronized axes has exceeded the maximum permissible limit value. The axes in the gantry grouping are shut down internally in the control. Alarm 10653 “Error limit exceeded” is output additionally.<br>The monitored limit value is derived from the following machine data:<br>• MD 37120: GANTRY_POS_TOL_ERROR if gantry grouping is synchronized·<br>• MD 37120: GANTRY_POS_TOL_REF if gantry grouping is not yet synchronized.<br>**Note:** IS “Gantry trip limit exceeded” is output to the PLC via the PLC interface of the synchronized axis. | |
| Signal state 0 or signal transition 1 ——> 0 | The difference between the position actual values of the master and synchronized axes is still within the permissible tolerance range. | |
| Signal irrelevant for ...... | Gantry master axis | |
| Related to .... | MD 37120: GANTRY_POS_TOL_ERROR Gantry trip limit<br>MD 37130: GANTRY_POS_TOL_REF Gantry trip limit for referencing<br>IS “Gantry grouping is synchronized” (DB31, **... ;** DBX101.5) | |

| | | |
|---|---|---|
| **DB31, ... ;**<br>**DBX101.3**<br>Data block | **Gantry warning limit exceeded**<br><br>Signal(s) to PLC (NC → PLC) | |
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW 2.1 |
| Signal state 1 or signal transition 0 ——> 1 | The difference in the position actual values of the master and synchronized axes has exceeded the limit value defined with MD 37110: GANTRY_POS_TOL_WARNING.<br>Message “Warning limit exceeded” is output additionally.<br>**Note:** IS “Gantry warning limit exceeded” is output to the PLC via the PLC interface of the synchronized axis. | |
| Signal state 0 or signal transition 1 ——> 0 | The difference between the position actual values of master and synchronized axes is less than the limit value defined with MD 37110: GANTRY_POS_TOL_WARNING. | |
| Signal irrelevant for ...... | Gantry master axis | |
| Application example(s) | When the gantry warning limit is exceeded, the necessary measures (e.g. program interruption at block end) can be initiated by the PLC user program. | |
| Special cases, errors, ...... | If MD 37110: GANTRY_POS_TOL_WARNING is set to zero, monitoring of the warning limit is deactivated. | |
| Related to .... | MD 37110: GANTRY_POS_TOL_WARNING Gantry warning limit | |

| | | |
|---|---|---|
| **DB31, ... ;**<br>**DBX101.4**<br>Data block | **Gantry synchronization ready to start**<br><br>Signal(s) to PLC (NC → PLC) | |
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW 2.1 |
| Signal state 1 or signal transition 0 ——> 1 | After gantry axis referencing, the monitoring function has detected that the position actual value deviation between the master and synchronized axes is greater than the gantry warning limit (MD: GANTRY_POS_TOL_WARNING). It is therefore not possible to start the **automatic** synchronization compensatory motion of the gantry axes internally in the control.<br>The compensatory motion must be started by the user or the PLC user program (IS “Start gantry synchronization”). The signal is processed for the gantry master axis only. | |

| DB31, ... ;<br>**DBX101.4**<br>Data block | **Gantry synchronization ready to start**<br><br>Signal(s) to PLC (NC → PLC) | |
|---|---|---|
| Signal state 0 or signal transition 1 ——> 0 | After the synchronization compensatory motion has been started by the PLC user program (IS "Start gantry synchronization" = "1"). | |
| Signal irrelevant for ...... | Gantry synchronized axis | |
| Related to .... | MD 37110: GANTRY_POS_TOL_WARNING<br>IS "Start gantry synchronization"<br>IS "Referenced/Synchronized 1/2" | Gantry warning limit<br>(DB31, **...** ; DBX29.4)<br>(DB31, **...** ; DBX60.4 and 60.5) |

| DB31, ... ;<br>**DBX101.5**<br>Data block | **Gantry grouping is synchronized**<br><br>Signal(s) to PLC (NC → PLC) | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW 2.1 |
| Signal state 1 or signal transition 0 ——> 1 | The gantry axis grouping defined with MD 37100: GANTRY_AXIS_TYPE is synchronized. Any existing misalignment between the master and synchronized axes (e.g. after start-up of the machine) is eliminated by the gantry axis synchronization process (see Section 2.3). The synchronization process is initiated either automatically after referencing of the gantry axes or via the PLC user program (IS "Start gantry synchronization").<br>The compensation values for temperature and sag do not become effective internally in the control until the gantry grouping is synchronized.<br><br>**Note:** IS "Gantry grouping is synchronized" is output to the PLC via the PLC interface of the master axis. | |
| Signal state 0 or signal transition 1 ——> 0 | The gantry axis grouping defined with MD 37100: GANTRY_AXIS_TYPE is not synchronized which means that the positions of the master and synchronized axes may not be ideally aligned (e.g. gantry misalignment).<br>Workpiece machining with a non-synchronized gantry axis grouping will result in impaired machining accuracy or mechanical damage to the machine.<br><br>The gantry grouping becomes desynchronized if·<br>• the gantry axes were in "Follow-up" mode.<br>• the reference position of a gantry axis is no longer valid or the axis is referenced again (IS "Referenced/synchronized").<br>• the gantry grouping has been invalidated (via MD: GANTRY_BREAK_UP). | |
| Signal irrelevant for ...... | Gantry synchronized axis | |
| Application example(s) | Machining should be enabled only if the gantry axes are already synchronized. This can be implemented in the PLC user program by combining NC Start with IS "Gantry grouping is synchronized". | |
| Related to .... | IS "Start gantry synchronization"<br>IS "Referenced/Synchronized 1/2"<br>MD 37140: GANTRY_BREAK_UP | (DB31, **...** ; DBX29.4)<br>(DB31, **...** ; DBX60.4 and 60.5)<br>Invalidate gantry axis grouping |

| DB31, ... ;<br>**DBX101.6**<br>Data block | **Gantry master axis**<br><br>Signal(s) to PLC (NC → PLC) | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW 2.1 |
| Signal state 1 or signal transition 0 ——> 1 | The axis is defined as the **master axis** within a gantry axis grouping (see MD 37100: GANTRY_AXIS_TYPE).<br><br>**Note:** The following interface signals are evaluated or output to the PLC via the PLC interface of the gantry master axis:<br>• IS "Start gantry synchronization" (DB31, **...** ; DBX29.4)<br>• IS "Gantry grouping is synchronized" (DB31, **...** ; DBX101.5) | |

| | |
|---|---|
| **DB31, ... ;**<br>**DBX101.6**<br>Data block | **Gantry master axis**<br><br>Signal(s) to PLC (NC → PLC) |
| Signal state 0 or signal transition 1 ––> 0 | The axis is defined as the **synchronized axis** within a gantry axis grouping (see MD 37100: GANTRY_AXIS_TYPE).<br>It is not possible to traverse a synchronized axis directly by hand (in JOG mode) or to program it in a parts program.<br>**Note:** The following interface signals are output to the PLC via the PLC interface of the gantry synchronized axis:<br>• IS "Gantry warning limit exceeded" (DB31, **... ;** DBX101.3)·<br>• IS "Gantry trip limit exceeded" (DB31, **... ;** DBX101.2)<br>The NCK does not evaluate individual axial PLC interface signals for the synchronized axis (see Table 2-3) |
| Related to .... | MD 37100: GANTRY_AXIS_TYPE Gantry axis definition<br>IS "Gantry axis" (DB31, **... ;** DBX101.7) |

| | | | |
|---|---|---|---|
| **DB31, ... ;**<br>**DBX101.7**<br>Data block | **Gantry axis**<br><br>Signal(s) to PLC (NC → PLC) | | |
| Edge evaluation: no | | Signal(s) updated: Cyclically | Signal(s) valid from SW 2.1 |
| Signal state 1 or signal transition 0 ––> 1 | The axis is defined as a gantry axis within a gantry axis grouping (see MD 37100: GANTRY_AXIS_TYPE).<br>On the basis of IS "Gantry master axis", the PLC user program can detect whether the axis is declared as a master or synchronized axis. | | |
| Signal state 0 or signal transition 1 ––> 0 | The axis is not defined as a gantry axis (see MD: GANTRY_AXIS_TYPE). | | |
| Related to .... | MD 37100: GANTRY_AXIS_TYPE Gantry axis definition<br>IS "Gantry master axis" (DB31, **... ;** DBX101.6) | | |

■

# Notes

# Example 6

## 6.1 Creating a gantry grouping

**Introduction**

The gantry grouping, the referring of its axes, the orientation of possible offsets and, finally, the synchronization of the axes involved are complicated procedures. The individual steps involved in the process are explained below by an example constellation.

**Constellation**

| | |
|---|---|
| Machine axis 1 = gantry master axis | Incremental measuring system |
| Machine axis 3 = gantry synchronized axis | Incremental measuring system |

**Machine data**

The following machine data describe the original values at the beginning of the procedure. Individual settings must be corrected or added later according to the information below.

Gantry machine data

Axis 1

| | |
|---|---|
| MD 37100: GANTRY_AXIS_TYPE | = 1 |
| MD 37110: GANTRY_POS_TOL_WARNING | = 0 |
| MD 37120: GANTRY_POS_TOL_ERROR | = e.g. 1mm |
| MD 37130: GANTRY_POS_TOL_REF | = e.g. 100mm (max. skew) |
| MD 37140: GANTRY_BREAK_UP | = 0 |

Axis 3

| | |
|---|---|
| MD 37100: GANTRY_AXIS_TYPE | = 11 |
| MD 37110: GANTRY_POS_TOL_WARNING | = 0 |
| MD 37120: GANTRY_POS_TOL_ERROR | = e.g. 1mm |
| MD 37130: GANTRY_POS_TOL_REF | = e.g. 100mm (max. skew) |
| MD 37140: GANTRY_BREAK_UP | = 0 |

Reference point machine data (for first encoder in each case)

Axis 1

| | |
|---|---|
| MD 34000: REFP_CAM_IS_ACTIVE | = TRUE |
| MD 34010: REFP_CAM_DIR_IS_MINUS | = e.g. FALSE |
| MD 34020: REFP_VELO_SEARCH_CAM | = |
| MD 34030: REFP_MAX_CAM_DIST | = equals max. travel path |
| MD 34040: REFP_VELO_SEARCH_MARKER | = |
| MD 34050: REFP_SEARCH_MARKER_REVERSE | = e.g. FALSE |
| MD 34060: REFP_MAX_MARKER_DIST | = difference betw. cam edge and 0 mark |
| MD 34070: REFP_VELO_POS | = |
| MD 34080: REFP_MOVE_DIST | = 0 |
| MD 34090: REFP_MOVE_DIST_CORR | = 0 |

MD 34092: REFP_CAM_SHIFT = 0
MD 34100: REFP_SET_POS = 0
MD 34200: ENC_REFP_MODE = 1

The reference point machine data (for the first encoder) of axis 3 must be specified analogously.

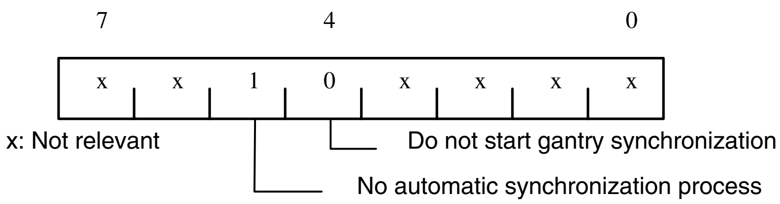
## 6.2 Setting of NCK PLC interface

**Introduction**

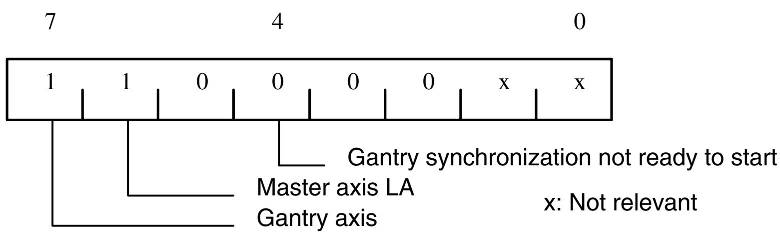
An automatic synchronization process during axis referencing must be disabled initially so as to prevent any damage to grouping axes that are misaligned.

**Disabling of automatic synchronization**

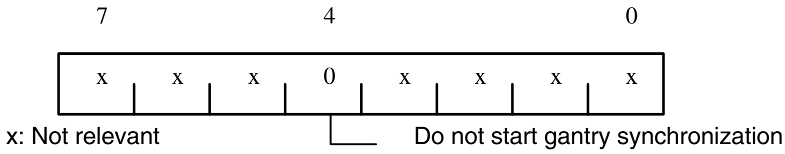
The user PLC routine sets:
DB31, ... ; DBX 29.4 = 0
DB31, ... ; DBX 29.5 = 1
for the axis data block of axis 1



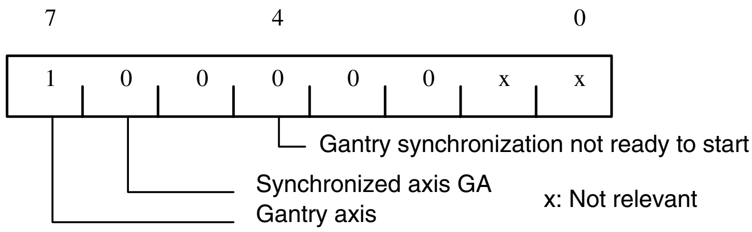

The NCK sets the following as a confirmation in the axis block of axis 1:
DB31, ... ; DBB101:




The user PLC routines sets:
DB31, ... ; DBX 29.4 = 0
for the axis data block of axis 3




The NCK sets the following as a confirmation in the axis block of axis 3:
DB31, ... ; DBB101:



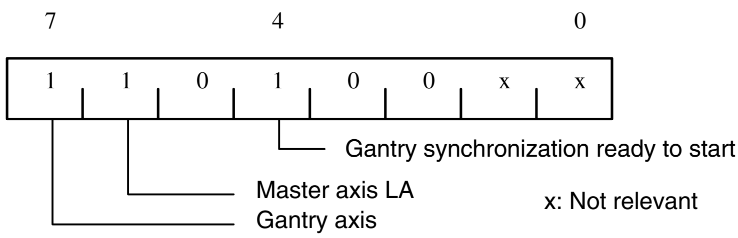

## 6.3 Commencing start-up

**Referencing**

The following steps must be taken:

1. Select “REF“ operating mode
2. Start referencing for axis 1 (master axis)
3. Wait until message “10654 Channel 1 Waiting for synchronization start” appears.

At this point in time, the NCK has prepared axis 1 for synchronization and signals this state via IS DB31, ..., DBB101 with:




4. RESET
5. Read off values in machine coordinate system:
   E.g. X = 0.941
   Y = 0.000
   XF = 0.000

6. Enter the X value of master axis 1 with inverted sign in MD 34090: REFP_MOVE_DIST_CORR of slave axis 3:

   REFP_MOVE_DIST_CORR = – 0.941

---

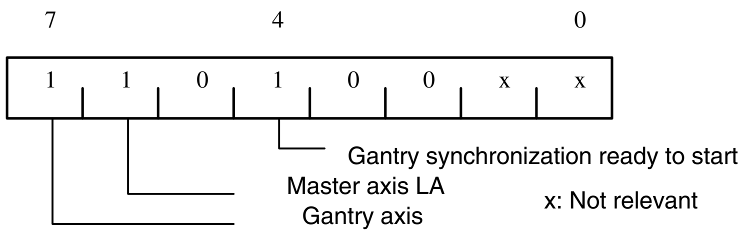
**Note**

This MD is effective after power ON. To avoid having to perform a power ON now, the value can also be entered in MD 34080 REFP_MOVE_DIST. The MD is then valid after a RESET.

---

7. Start referencing again for axis 1 (master axis) with the modified machine data
8. Wait until message “10654 Channel 1 Waiting for synchronization start” appears.

At this point in time, the NCK has prepared axis 1 for synchronization and signals this state via IS DB31, ... ; DBB101 with:




9. Examine actual positions of machine. Case A or B might apply:

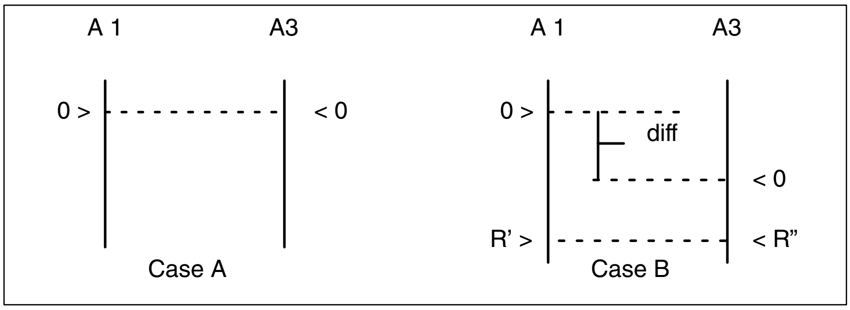



Fig. 6-1 Possible results after referencing of axis 1 (master axis)

If Case A applies, the synchronization process can be started immediately. See Step 10.
If Case B applies, the offset "diff" must be calculated and taken into account:
a) Measurement of diff
b) The positional difference can be traversed in JOG mode thanks to two reference points R' and R" positioned at right angles on the machine base (on right in diagram). The offset diff can then be read off as a deviation from the positional display.
Offset diff must be entered in MD 34100: REFP_SET_POS of axis 3 (synchronized axis). Continue with Step 1 (see above).

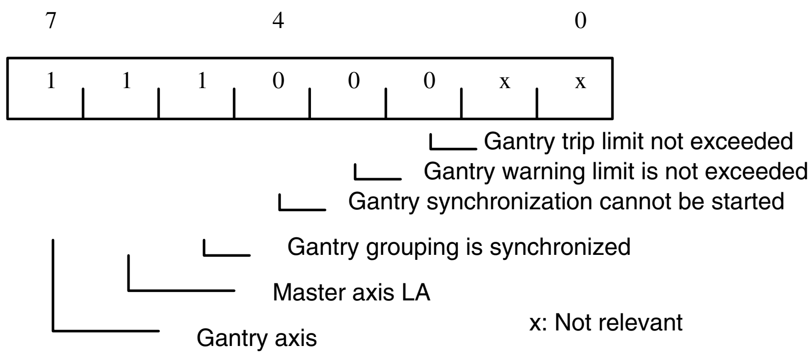
10. Start gantry synchronization. The PLC sets:
 DB31, ... ; DBX 29.4= 1

## 6.4 Setting warning and trip limits

As soon as the gantry grouping has been set and synchronized, machine data
MD 37110: GANTRY_POS_TOL_WARNING and
MD 37120: GANTRY_POS_TOL_ERROR
have still to be set to appropriate values.

**Procedure**

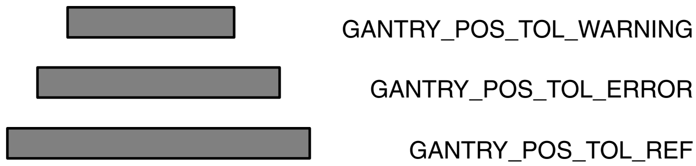
– Set MD 37120: GANTRY_POS_TOL_ERROR to a high value for all axes initially.
– Enter a very low value in MD 37110: GANTRY_POS_TOL_WARNING.If you now subject the axes to a high dynamic load, self-resetting alarm: "10652 Channel %1 axis %2 Gantry warning limit exceeded" should be output repeatedly.
– Now increase the setting in MD 37110: GANTRY_POS_TOL_WARNING until the alarm no longer appears. The interface indicates the status specified below. (This must occur within a window that is reasonable for production purposes).If the monitoring function is only activated very sporadically, it is possible to program an edge trigger flag in the user PLC program.




– Enter the value obtained for the warning limit + a small safety margin in MD 37120: GANTRY_POS_TOL_ERROR.

**Error limit values**

The ratios between the values stored in machine data
MD 37110: GANTRY_POS_TOL_WARNING
MD 37120: GANTRY_POS_TOL_ERROR
MD 37130: GANTRY_POS_TOL_REF
should be as follows at the end of the adjustment process:




The system expects the error windows to have the proportions shown above. If GANTRY_POS_TOL_ERROR and GANTRY_POS_TOL_REF are smaller than GANTRY_POS_TOL_WARNING, this monitoring is not active!
GANTRY_POS_TOL_WARNING = 0 can be set to disable the monitoring functions for the warning limit.
GANTRY_POS_TOL_REF must be set to allow for the maximum possible leadscrew error compensation value.

---

**Note**

The same procedure must be followed when starting up a gantry grouping in which the coupled axes are driven by **linear motors** and associated measuring systems.

---

**Note**

The error limits entered in MD 37110: GANTRY_POS_TOL_WARNING and MD 37120: GANTRY_POS_TOL_ERROR are considered as additional tolerance values of the actual-value difference of the master and following axis if the IS “Gantry is synchronous” is not present (e.g. to be resynchronized after canceling alarms without gantry).

---

# Data Fields, Lists 7

## 7.1 Interface Signals

| DB number | Bit, byte | Name | Ref. |
|---|---|---|---|
| **General** | | | |
| **11–14** | **5.2** | **Active machine function REF** | **R1** |
| **Channel-specific** | | | |
| 21–28 | 33.0 | Referencing active | **R1** |
| **Axis/spindle-specific** | | | |
| 31, ... ; | 60.4, 60.5 | Referenced/synchronized 1, referenced/synchronized 2 | **R1** |
| 31, ... ; | 29.4 | Start gantry synchronization | |
| 31, ... ; | 29.5 | No automatic synchronization process | |
| 31, ... ; | 101.2 | Gantry trip limit exceeded | |
| 31, ... ; | 101.3 | Gantry warning limit exceeded | |
| 31, ... ; | 101.4 | Gantry synchronization ready to start | |
| 31, ... ; | 101.5 | Gantry grouping is synchronized | |
| 31, ... ; | 101.6 | Gantry master axis | |
| 31, ... ; | 101.7 | Gantry axis | |

## 7.2 Machine data

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| **Axis/spindle-specific($MA_ ...)** | | | |
| 30300 | IS_ROT_AX | Rotary axis | R2 |
| 32200 | POSCTRL_GAIN | Servo gain factor | G2 |
| 32400 | AX_JERK_ENABLE | Axial jerk limitation | B2 |
| 32410 | AX_JERK_TIME | Time constant for axis jerk filter | B2 |
| 32420 | JOG_AND_POS_JERK_ENABLE | Basic setting of axis jerk limitation | B2 |
| 32430 | JOG_AND_POS_MAX_JERK | Axial jerk | B2 |
| 32610 | VELO_FFW_WEIGHT | Feedforward control factor for speed feedforward control | K3 |
| 32620 | FFW_MODE | Feedforward control mode | K3 |
| 32650 | AX_INERTIA | Moment of inertia for torque feedforward control | K3 |
| 32800 | EQUIV_CURRCTRL_TIME | Equivalent time constant, current control loop for feedforward control | K3 |
| 32810 | EQUIV_SPEEDCTRL_TIME | Equivalent time constant, speed control loop for feedforward control | K3 |
| 32910 | DYN_MATCH_ENABLE | Dynamic response adaptation | G2 |
| 32910 | DYN_MATCH_TIME | Time constant for dynamic response adaptation | G2 |
| 33000 | FIPO_TYPE | Fine interpolator type | G2 |
| 34040 | REFP_VELO_SEARCH_MARKER | Creep velocity | R1 |
| 34070 | REFP_VELO_POS | Reference point approach velocity | R1 |
| 34080 | REFP_MOVE_DIST | Reference point approach distance | R1 |
| 34090 | REFP_MOVE_DIST_CORR | Reference point offset | R1 |
| 34100 | REFP_SET_POS | Reference point value | R1 |
| 34110 | REFP_CYCLE_NR | Axis sequence for channel-specific referencing | R1 |
| 34330 | REFP_STOP_AT_ABS_MARKER | Distance-coded linear measurement system without target point | R1 |
| 36012 | STOP_LIMIT_FACTOR | Exact stop coarse/fine factor and zero speed | B1 |
| 36030 | STANDSTILL_POS_TOL | Zero speed tolerance | A3 |
| 36500 | ENC_CHANGE_TOL | Maximum tolerance for position actual value switchover | G2 |
| 37100 | GANTRY_AXIS_TYPE | Gantry axis definition | |
| 37110 | GANTRY_POS_TOL_WARNING | Gantry warning limit | |
| 37120 | GANTRY_POS_TOL_ERROR | Gantry trip limit | |
| 37130 | GANTRY_POS_TOL_REF | Gantry trip limit for referencing | |
| 37140 | GANTRY_BREAK_UP | Invalidate gantry axis grouping | |

## 7.3 Alarms

A detailed description of the alarms which may occur is given in
**References:** /DA/, Diagnostics Guide
or in the online help in systems with MMC 101/102/103.

■

# SINUMERIK 840D/840Di Description of Functions Special Functions (Part 3)

# Cycle Times (G3)



■

## Notes

# 1 Brief Description

**810D and 840D without PROFIBUS**

The SINUMERIK 840D to NCU 572 and 810D with CCU derive both the position control cycle and the interpolator cycle (IPO cycle) from the system clock cycle.

**840Di and 840D with PROFIBUS**

The SINUMERIK 840Di derives the position control cycle and the interpolator cycle from the system clock cycle. The system clock cycle is defined by the PROFIBUS DP cycle for a SINUMERIK 840Di, which is set in the SIMATIC S7 project loaded in the PLC.

With Software release SW 6.4 and an NCU 573.2 or higher it is possible to operate a PROFIBUS DP via the hardware module “Link module”. In this case, the SINUMERIK 840D has two PROFIBUS DP interfaces and similar restrictions apply as for SINUMERIK 840Di.

The following description explains the interrelations and the machine data of these cycle times.

■

## Notes

# Detailed Description 2

## 2.1 General cycle times

**Definition**

The system clock cycle, position control cycle and interpolator cycle are defined in the following machine data.

| | |
|---|---|
| MD 10050: SYSCLOCK_CYCLE_TIME | System clock cycle |
| MD 10060: POSCTRL_SYSCLOCK_TIME_RATIO | Factor for position control cycle |
| MD 10070: IPO_SYSCLOCK_TIME_RATIO | Factor for interpolator cycle |

MD 10050: SYSCLOCK_CYCLE_TIME sets the system clock cycle for the system software in seconds. The other cycles are multiples of the system clock cycle.

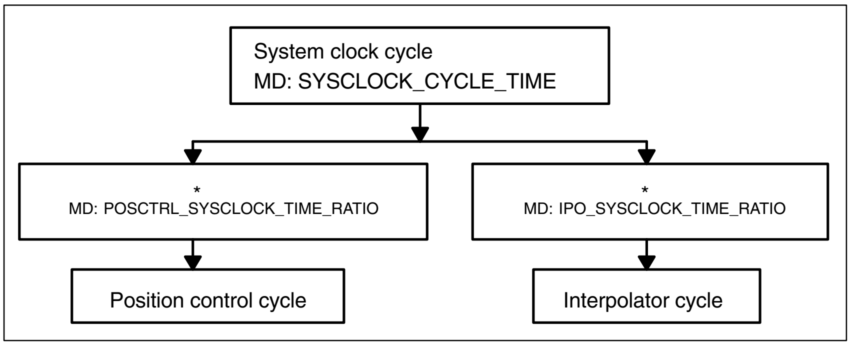



Fig. 2-1 Derivation of cycle times

---

**Note**

**On MD 10050 SYSCLOCK_CYCLE_TIME**
The system clock cycle is entered in s.

**On MD 10060 POSCTRL_SYSCLOCK_TIME_RATIO**
The position control cycle is entered as a factor of the system clock cycle.

**On MD 10070 IPO_SYSCLOCK_TIME_RATIO**
The interpolator cycle is entered as a factor of the system clock cycle.

---

**Cycle times defaults**

The default settings ensure that a maximum configuration of the system can power up reliably. The cycle times, e.g. for the **NCU 573**, can generally be set to lower values.
The cycle time default settings are as follows:

| Cycle | 810D CCU | 840D NCU 571 | 840D NCU 572 | 840D NCU 573 | Setting via MD |
|---|---|---|---|---|---|
| System clock cycle | 2.5ms | 6ms | 4ms | 4* / 8#ms | SYSCLOCK_CYCLE_TIME |
| Position control cycle | 2.5ms | 6ms | 4ms | 4* / 8#ms | POSCTRL_SYSCLOCK_TIME_RATIO |
| Interpolator cycle | 10ms | 18ms | 12ms | 12* / 40#ms | IPO_SYSCLOCK_TIME_RATIO |

* With 2 channels and 12 axes
\# With > 2 channels

**General example of cycle setting**

The machine data for the cycle times are assigned as follows:

| If MD ... = ... | then the ... = ... |
|---|---|
| SYSCLOCK_CYCLE_TIME = 0.002 | System clock cycle = 2ms |
| POSCTRL_SYSCLOCK_TIME_RATIO = 1 | Position control cycle = 2 ms (1· 2ms) |
| IPO_SYSCLOCK_TIME_RATIO = 3 | Interpolator cycle = 6 ms (3· 2ms) |

## 2.2 SINUMERIK 810D and 840D without PROFIBUS DP

**Interpolator cycle**

The interpolator cycle defines the cycle time in which the setpoint interface to the position controllers is updated. The interpolator cycle is important for two reasons in normal processing:

- The product of velocities and interpolator cycles defines the geometry resolution of the interpolated contour. A long interpolator cycle causes a large path error along curved contours. This error is, however, reduced in the ratio interpolator / position control cycle by cubic fine interpolation MD 33000: FIPO_TYPE.
- The interpolator cycle determines the possible resolution of the velocity profiles. It must be adapted to the dynamics of the drives so that the machine axes traverse and accelerate evenly (i.e. position control cycle time </= interpolator cycle << acceleration time constant).

**Position control cycle**

The position control cycle is the time which it takes for the control to calculate the actual value and transfer a new speed setpoint to the speed controller.

**Block cycle time**

The block cycle time is the sum of the block change time and block preparation time. It is at least as long as the cycle time for sending the position setpoints to the servos – in normal operation therefore as long as the interpolator cycle.

The block cycle time is a common form of measurement used to judge whether the control is suitable for traversing contours defined in points (frequent problem with 3 and 5-axis milling). It determines the maximum possible velocity at which a defined point pattern can be traversed (max. feedrate = average distance between points/block cycle time).

**Setting the IPO cycle and position control cycle**

The interpolator and position control cycles are set in integer multiples of the system clock cycle in the following machine data:

$$\text{POSCTRL\_SYSCLOK\_TIME\_RATIO} = \frac{\text{Position control cycle}}{\text{System clock cycle}}$$

$$\text{IPO\_SYSCLOK\_ TIME\_RATIO} = \frac{\text{Interpolator cycle}}{\text{System clock cycle}}$$

The smallest possible position control and interpolator cycle should be aimed for.

Apart from special applications in which machine data MD 10060: POSCTRL_SYSCLOCK_TIME_RATIO is set greater than 1, the position control cycle corresponds to the system clock cycle.

The ratio of interpolator to position control cycle must be an integer value and greater than or equal to 1. If this is not the case, the value is corrected automatically and alarm 4102, “IPO cycle increased to [ ] ms” is output.

$$\frac{\text{Interpolator cycle}}{\text{Position control cycle}} \geq 1$$

---

**Note**

As of software version SW 6.4 and an NCU 573.2 or higher, it is also possi-bleoperate SINUMERK 840D with the PROFIBUS DP. For more information about configuring the clock cycles, please refer to Section 2.3.

---

## 2.3 SINUMERIK 840Di and 840D with PROFIBUS DP

For more information about SINUMERK 840D and 840Di with PROFIBUS DP, please refer to: **References:** /FB1/, K4 Communication

### 2.3.1 Description of a DP cycle

**Actual values**

At time $T_I$, the current actual values are read from all equidistant drives (DP slaves). In the next DP cycle, the actual values are transferred to the DP master in the time $T_{DX}$.

**Positon controller**

The NC position controller is started at the time $T_{M,}$ with $T_M > T_{DX,}$ and computes the new speed setpoints on the basis of the transferred actual positions.

**Setpoints**

At the start of the next DP cycle, the speed setpoints are transferred from the DP master to the DP slaves (drives) in the time $T_{DX}$.

At time $T_O$, the speed setpoints are taken as new specified values for all drive controllers.

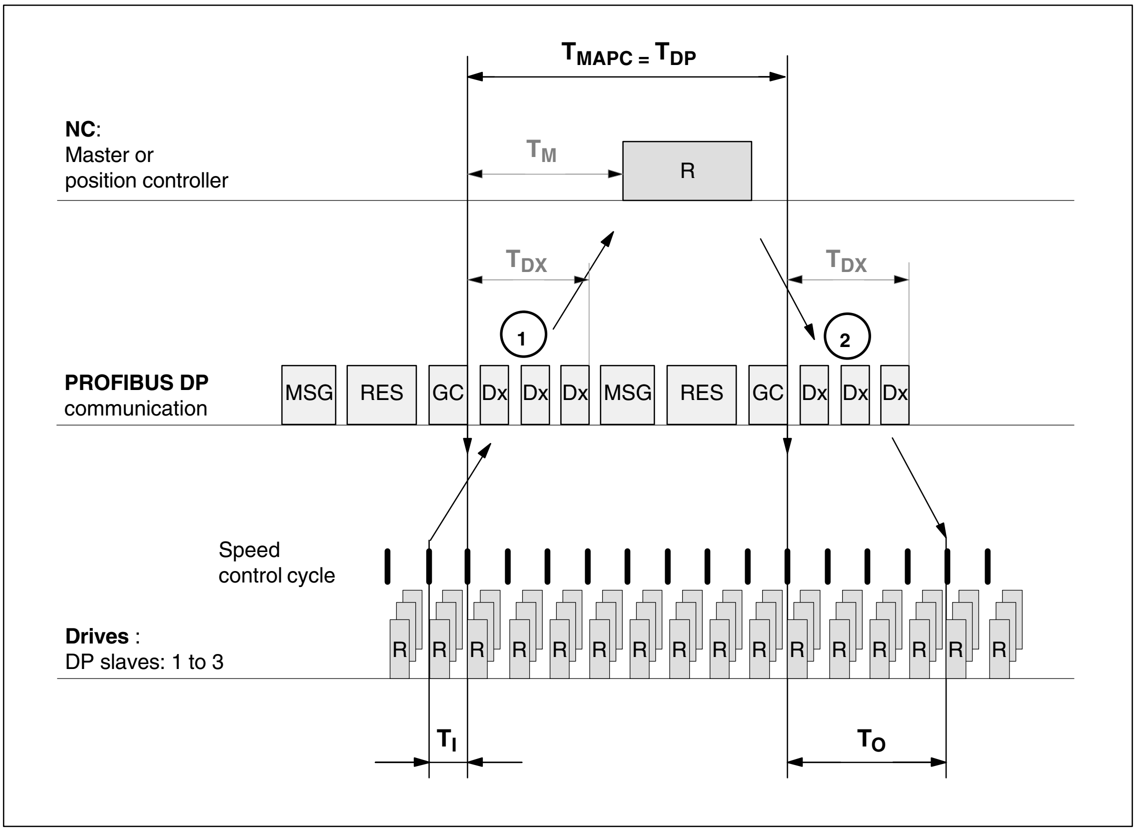



Fig. 2-2 Optimized DP cycle with 3 DP slave with a SIMODRIVE 611 universal

Key to Fig. 2-2:

| | |
|---|---|
| $T_{MAPC}$ | Master Application Cycle: NC position control cycle<br>the following always applies for SINUMERIK 840Di: $T_{MAPC} = T_{DP}$ |
| $T_{DP}$ | DP cycle time: DP cycle time |
| $T_{DX}$ | Data Exchange Time: total transfer times for all DP slaves |
| $T_M$ | Master Time: offset of the start time for NC position control |
| $T_I$ | Input Time: Time of actual value recording |
| $T_O$ | Output Time: Time of setpoint transfer |
| GC | Global control telegramm (broadcast message) for cyclic synchronization of the equidistance between DP master and DP slaves |
| R | Computation time for speed position control |
| Dx | Exchange of user data between DP master and DP slaves |
| MSG | Acyclic services (e.g. DP/V1, pass token) |
| RES | Reserve: “active pause” until equidistant cycle finishes |
| (1) | The actual values for the current DP cycles / position control cycle are transferred from the DP slave drives to the NC position controller |
| (2) | The setpoints computed by the NC position controller are transferred to the DP slave drives |

### 2.3.2 Clock cycles and position control cycle offset

**Cycle times**

The NC derives the cycle times, system clock cycle, position control cycle and interpolator cycle from the equidistant PROFIBUS DP cycle set in the SIMATIC S7 project during configuration of the PROFIBUS.

System clock cycle

The system clock cycle is set to the fixed ratio 1:1 with respect to the PROFIBUS DP cycle. It cannot be changed.

- MD10050: SYSCLOCK_CYCLE_TIME (system basic time)

Position control cycle

The position control cycle is set to the fixed ratio 1:1 with respect to the system clock cycle. It cannot be changed.

- MD10061: POSCTRL_CYCLE_TIME (position control cycle)

Interpolation cycle

The interpolator cycle may be chosen freely as a whole multiple of the position control cycle.

- MD10070: IPO_SYSCLOCK_TIME_RATIO (factor for interpolator cycle)

**Position control cycle**

The offset for the position control cycle ($T_M$) is set independently of the conditions described below within a PROFIBUS DP/system cycle and independently of the cyclic communication with the DP slave.

- MD10062 POSCTRL_CYCLE_DELAY (position control cycle offset)

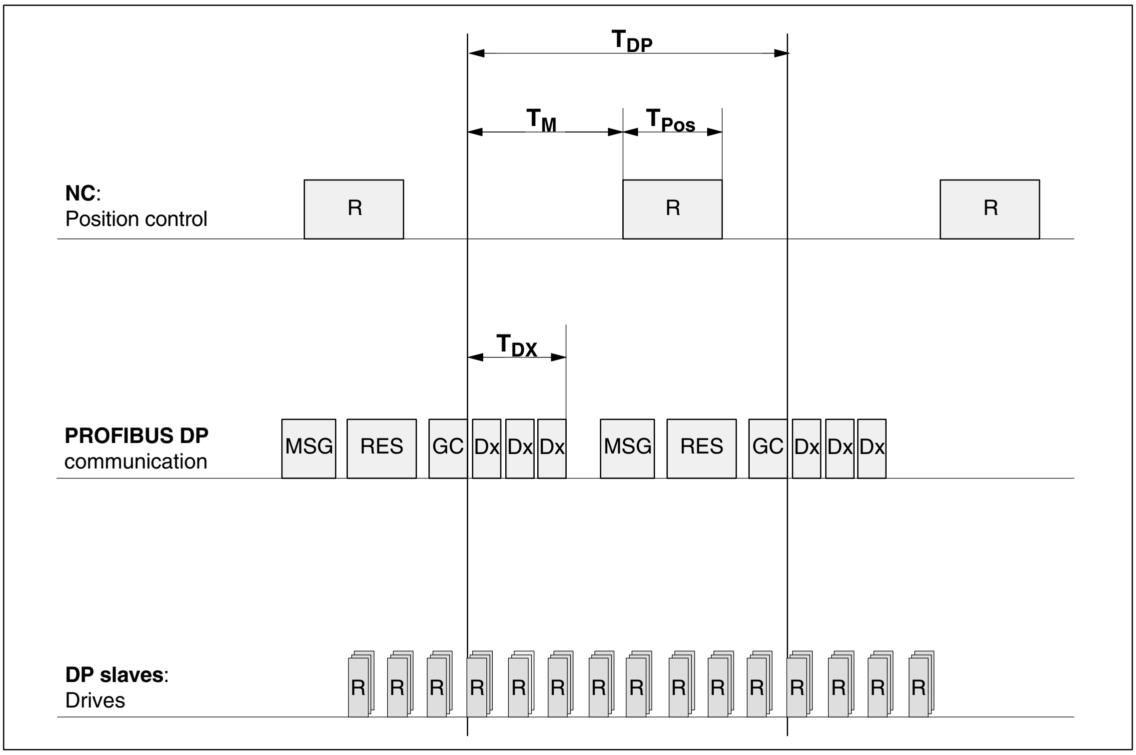



Fig. 2-3 Position control cycle offset compared to PROFIBUS DP cycle

Key to Fig. 2-3:

| | |
|---|---|
| $T_{Pos}$ | Computation time required by position controller |
| $T_{DP}$ | DP cycle time: DP cycle time |
| $T_{DX}$ | Data exchange Time: total transfer times for all DP slaves |
| $T_M$ | Master Time: offset of the start time for NCposition control |
| GC | Global Control: Broadcast message for cyclic synchronization of the equidistance between DP master and DP slaves |
| R | Computation time |
| Dx | Exchange of user data between DP master and DP slaves |
| MSG | Acyclic services (e.g. DP/V1, pass token) |
| RES | Reserve: “active pause” until equidistant cycle finishes |

**Conditions and recommendations for MD 10062**

MD10062 POSCTRL_CYCLE_DELAY (position control cycle offset)
The offset for the position control cycle ($T_M$) must be set such that the following conditions are satisfied within a PROFIBUS DP/system cycle:

– The cyclic communication with the DP slaves (drives) must be completed before the position controller is started.
  Condition: $T_M > T_{DX}$

– The position controller must be completed before the PROFIBUS DP/system cycle comes to an end.
  Condition: $T_M + T_{Pos} < T_{DP}$

The following setting is recommended as approximate value for the position control cycle offset:

$T_M = T_{DP} – 3*Tmax_{position\ controller}$

– $T_{DP}$
  Position control cycle or PROFIBUS DP cycle

– $Tmax_{position\ controller}$

  Note for HMI Advanced:
  The maximum time for the position controller is displayed in the dialog: **NC load** under **Menu area switchover** > **Diagnostics** > **Service displays** > **System resources**.

**Error reaction**

- Alarm: “380005 PROFIBUS DP: Bus access conflict, type t, counter z

**Cause of errors/error handling**

- t = 1
  The position control cycle offset chosen is too small. The cyclic PROFIBUS DP communication with the drives was not completed before the position controller started.

  Increase the position control cycle offset.

- t = 2
  The position control cycle offset selected is too large. Cyclic PROFIBUS DP communication with the drives started before the position controller had finished. The position controller requires more computation time than is available in the PROFIBUS DP cycle.

  – Decrease the position control cycle offset

  or

  – Increase the PROFIBUS DP cycle.
    The PROFIBUS DP cycle must be set in the SIMATIC S7 project.

**MD 10059**

MD10059: PROFIBUS_ALARM_MARKER (PROFIBUS alarm marker)

**Alarm requests in the event of a conflict during start-up**

- In this machine data, alarm requests on the PROFIBUS level are stored even after reboot.

  If a conflict occurs during start-up between the machine data

  – MD 10050: SYSCLOCK_CYCLE_TIME (system clock cycle)

  – MD 10060: POSCTRL_SYSCLOCK_TIME_RATIO (factor for position control cycle)

– MD 10070: IPO_SYSCLOCK_TIME_RATIO (factor for interpolator cycle)

and the

– data found in PROFIBUS-SDB,

the machine data are adapted according to this SDB and an appropriate alarm set during next start-up. These alarm request are stored here.

**Machine data 840D and 840Di**

For system-internal reasons, a distinction is made between the configured default and maximum values in SINUMERIK 840D with PROFIBUS DP compared to SINUMERK 840Di.

| Machine data | 840D Standard / max | 840Di Standard / max |
|---|---|---|
| MD 10050: SYSCLOCK_CYCLE_TIME | 4ms / 31ms | 2ms / 8ms |
| MD 10060: POSCTRL_SYSCLOCK_TIME_RATIO | 1 / 31 | 1 / 1 |
| MD 10062: POSCTRL_CYCLE_DELAY | 3ms / 8ms | 1.5ms / 8ms |
| MD 10070: IPO_SYSCLOCK_TIME_RATIO | 3 for 2-channel*<br>5 for 10-channel | Standard 4 |
| MD 10080: SYSCLOCK_SAMPL_TIME_RATIO | 4 for 2-channel*<br>5 for 10-channel | No meaning |

* Default values

**Special points**

The following special points must be observed for cycle-specific machine data:

- MD10050: SYSCLOCK_CYCLE_TIME (system clock cycle)
  The machine data is used only for display purposes. The system cycle is always identical to the equidistant PROFIBUS DP cycle.
- MD 10060 POSCTRL_SYSCLOCK_TIME_RATIO (factor for position control cycle)
  The factor for position control is set permanently to 1 only for SINUMERK 840Di and cannot be changed.

  With SINUMERIK 840D, the factor for the position control cycle can still be set in MD 10060 POSCTRL_SYSCLOCK_TIME_RATIO for reasons of compatibility with the drive bus for SIMODRIVE 611 digital.

**Caution**

If you change the cycle times, check the behavior of the drive in all operating modes before you finish commissioning.

**Note**

The smaller the cycle times (PROFIBUS DP cycle) chosen, the greater the degree of control for the drive and the better the surface quality on the workpiece.

■

# Supplementary Conditions 3

None

# Data Descriptions (MD, SD) 4

## 4.1 General machine data

| | | | | | |
|---|---|---|---|---|---|
| **10050**<br>MD number | **SYSCLOCK_CYCLE_TIME**<br>System clock cycle | | | | |
| Default setting: 0.004 for 840D<br>Default setting: 0.002 for 840Di | | Min. input limit: 0.000125 | | Max. input limit: 0.032 for 840D<br>max. input limit: 0.008 for 840Di | |
| Changes effective after power ON | | | Protection level: 2 / 7 | | Unit: s |
| Data type: DOUBLE | | | | Applies from SW 1.1, as of SW 6.4 PROFIBUS | |
| Meaning: | Clock cycle of the system software, i.e. the unit of time after which time interrupts are triggered in the operating system.<br>The interpolator and position control cycle/IPO settings are multiples of the system clock cycle. The system clock cycle corresponds to the position control cycle when MD 10060: POSCTRL_SYSCLOCK_TIME_RATIO is not set higher than 1.<br><br>If a digital drive is used, the system basic clock cycle and MD 10060:<br>POSCTRL_SYSCLOCK_TIME_RATIO must be set (otherwise a drive alarm is issued) so that the position control cycle is no longer than 16ms.<br>Automatic offsets during start–up can cause a shift in the set value for MD 10050: SYSCLOCK_CYCLE_TIME.<br><br>For systems with PROFIBUS DP, this corresponds to MD 10050:<br>SYSCLOCK_CYCLE_TIME: PROFIBUS DP cycle time. This time is read from the PROFIBUS/SDB project file during start–up and written to this machine data.<br>This machine data is can only be modified via the project file. | | | | |
| Special cases, errors, ...... | **Note:**<br>The system clock cycle is incremented in multiples of MD 10080: SYSCLOCK_SAMPL_TIME_RATIO of units of the measured value sampling clock cycle. During system start–up, the entered value is automatically rounded to a multiple of the incrementing.<br>Details:<br>After a power OFF/ON, discrete timer division ratios can produce a value that is not an integer of the input value.<br>E.g.: Input = 0.005s<br>after power OFF/ON = 0.00499840<br>or<br>input = 0.006s<br>after power OFF/ON = 0.0060032 | | | | |
| Related to .... | MD 10060: POSCTRL_SYSCLOCK_TIME_RATIO,<br>MD 10080: SYSCLOCK_SAMPL_TIME_RATIO | | | | |

| | | | | |
|---|---|---|---|---|
| **10059**<br>MD number | **PROFIBUS_ALARM_MARKER**<br>Profibus alarm marker (internal only) | | | |
| Profibus Adpt.: 0.0 | | Min. input limit: – | | Max. input limit: – |
| Changes effective after power ON | | | Protection level: 0 / 0 | Unit: – |
| Data type: BYTE | | | Applies from SW 5.2 | |
| Meaning: | PROFIBUS alarm marker:<br>In this machine data, alarm requests on the PROFIBUS level are stored even after reboot.<br>If a conflict is found between the machine data<br>MD 10050 SYSCLOCK_CYCLE_TIME, MD 10060 POSCTRL_SYSCLOCK_TIME_RATIO, MD 10070 IPO_SYSCLOCK_TIME_RATIO and the data in SDB1000, the machine data are adapted according to SDB1000 and an appropriate alarm set during next start–up.<br>These alarm requests are stored here. | | | |
| Related to .... | MD 10050: SYSCLOCK_CYCLE_TIME, MD 10060 POSCTRL_SYSCLOCK_TIME_RATIO | | | |

| | | | | |
|---|---|---|---|---|
| **10060**<br>MD number | **POSCTRL_SYSCLOCK_TIME_RATIO**<br>Factor for position control cycle | | | |
| Default setting: 1 | | Min. input limit: 1 | | Max. input limit: 31 for 840D<br>max. input limit: 1 for 840Di |
| Changes effective after power ON | | | Protection level: 2 / 7 | Unit:<br>Factor × MD 10050 |
| Data type: DWORD | | | Applies from SW 1.1 | |
| Meaning: | The position control cycle is entered as a multiple of the time units of the system clock cycle of MD 10050: SYSCLOCK_CYCLE_TIME.<br>The **normal setting is 1**. The position control cycle then corresponds to the system clock cycle.<br>Settings which are > 1 use up more computing time for the processing of additional time interrupts by the operating system and should therefore only be used if the system has to execute a task which needs to run faster than the position control cycle.<br>If a digital drive is being used, the value set for the position control cycle can change because of an automatic correction on start-up. This is accompanied by alarm 4101 “Position control cycle for digital drive reduced to [ ] ms”. The position control cycle may not be set longer than 16ms when using a digital drive.<br>For systems with PROFIBUS DP connection, machine data MD 10060 indicates the ratio for the PROFIBUS DP cycle and the position control cycle. | | | |
| Related to .... | MD 10050: SYSCLOCK_CYCLE_TIME, MD 10080 SYSCLOCK_SAMPL_TIME_RATIO | | | |

| | | | | |
|---|---|---|---|---|
| **10061**<br>MD number | **POSCTRL_CYCLE_TIME**<br>Position control cycle | | | |
| Default setting: 0.0 | | Min. input limit: – | | Max. input limit: – |
| Changes effective after power ON | | | Protection level: 0 / 7 | Unit: – |
| Data type: DOUBLE | | | Applies from SW 5 | |
| Meaning: | Position control cycle: Display of the position control cycle time (cannot be modified !).<br>Formed internally from the machine data SYSCLOCK_CYCLE_TIME and POSCTRL_SYSCLOCK_TIME_RATIO. | | | |
| Related to .... | MD 10050: SYSCLOCK_CYCLE_TIME, MD 10060 POSCTRL_SYSCLOCK_TIME_RATIO | | | |

| | | | | |
|---|---|---|---|---|
| **10062**<br>MD number | **POSCTRL_CYCLE_DELAY**<br>Position control cycle shift | | | |
| Default setting: 0.003 for 840D<br>Default setting: 0.0015 for 840Di | | Min. input limit: 0.000<br>Profibus Adpt.: 0.000 | | Max. input limit: 0.008 for 840D<br>max. input limit: 0.008 for 840Di |
| Changes effective after power ON | | | Protection level: 2 / 7 | Unit: s |
| Data type: DOUBLE | | | Applies from SW 5 | |
| Meaning: | NCK position control cycle time offset compared to PROFIBUS DP cycle | | | |
| Related to .... | MD 10050: SYSCLOCK_CYCLE_TIME, MD 10080: SYSCLOCK_SAMPL_TIME_RATIO | | | |

| | | | | | |
|---|---|---|---|---|---|
| **10070**<br>MD number | **IPO_SYSCLOCK_TIME_RATIO**<br>Factor for interpolator cycle | | | | |
| Default setting: 4 for 840Di<br>3 for 840D 2-channel<br>5 for 840D 10-channel | | Min. input limit: 1 | | Max. input limit: 100 | |
| Changes effective after power ON | | | Protection level: 2 / 7 | | Unit:<br>Factor × MD 10050 |
| Data type: DWORD | | | | Applies from SW 1.1 | |
| Meaning: | The interpolator cycle is entered as a multiple of the time units of the system clock cycle MD 10050: SYSCLOCK_CYCLE_TIME.<br>Only whole multiples of the position control cycle may be set (set via MD 10060: POSCTRL_SYSCLOCK_TIME_RATIO). Values that are not an integer multiple of the position control cycle are automatically increased to the next integer multiple of the position control cycle before they become active (on next power up). This is accompanied by the alarm 4102 "IPO cycle increased to [ ]ms".<br>The values set in the NCU-link group must be identical for all linked NCUs. This additional requirement is omitted if the NCU-link option is present with a different interpolator cycle. | | | | |
| Related to .... | MD 10050: SYSCLOCK_CYCLE_TIME, MD 10060 POSCTRL_SYSCLOCK_TIME_RATIO | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **10071**<br>MD number | **IPO_CYCLE_TIME**<br>Interpolator cycle | | | | |
| Default setting: 0.0 | | Min. input limit: *** | | Max. input limit: *** | |
| Changes effective after power ON | | | Protection level: 0 / 7 | | Unit: s |
| Data type: DOUBLE | | | | Applies from SW 5 | |
| Meaning: | Display of the interpolator cycle time (cannot be modified!).<br>If produced internally from MD 10050: SYSCLOCK_CYCLE_TIME and MD 10070: IPO_SYSCLOCK_TIME_RATIO. | | | | |
| Related to .... | MD 10050: SYSCLOCK_CYCLE_TIME, MD 10060 POSCTRL_SYSCLOCK_TIME_RATIO | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **10080**<br>MD number | **SYSCLOCK_SAMPL_TIME_RATIO**<br>Division factor of position control cycle for actual value acquisition | | | | | |
| Default setting: –<br>No meaning for 840Di<br>4 for 840D 2-channel<br>5 for 840D 10-channel | | Min. input limit: 1 | | Max. input limit: 31 | | |
| Changes effective after power ON | | | Protection level: 0 / 0 | | Unit: – | |
| Data type: DWORD | | | | Applies from SW1, as of SW 6.4 PROFIBUS | | |
| Meaning: | This machine data sets the division factor of a cycle divider which is hardware located between the measured value sampler and the interrupt controller.<br><br>Values of MD 10080: SYSCLOCK_SAMPL_TIME_RATIO > 1<br>may only be set if the conditions described in the special cases are satisfied.<br><br>produce a larger increment, in which the time for the system clock cycle can also be set (see MD 10050: SYSCLOCK_CYCLE_TIME).<br><br>If a digital drive controller and systems with PROFIBU-DP connection are used the division factor is set automatically. | | | | | |
| Special cases, errors, ...... | 1st Using an analog speed interface<br>When using the conventional drive interface, the division can be set according to the following criteria:<br>It is beneficial for the control to keep the deadtime between reading the current actual axis position and outputting the associated setpoints as short as possible.<br><br>The following applies when setting MD 10080: SYSCLOCK_SAMPL_TIME_RATIO to values > 1:<br>The delay time of the position control output can be set to fractions of the position control clock cycle.<br><br>A multiple triggering of the I/O hardware during the position control cycle can be achieved.<br>This procedure is **not** recommended on account of the resulting interrupt load on the operating system.<br><br>2nd Using a digital drive controller<br>If a digital drive controller is used the division factor is set automatically.<br><br>The sampling clock cycle is thereby set to<br>1, 2, 3 multiples of 0.000625s SINUMERIK 810D<br>Standard case:<br>1 time 0.000625s SINUMERIK 810D new design<br>1, 2, 3 ... 8 multiples of 0.000125s SINUMERIK 840D<br><br>The SIMODRIVE 611 digital drive can synchronize its own clock generation to these values.<br><br>3rd Systems with PROFIBUS DP connection<br>When systems with PROFIBUS DP are used, the division factor is set automatically. | | | | | |
| Related to .... | MD 10050: SYSCLOCK_CYCLE_TIME, MD 10060 POSCTRL_SYSCLOCK_TIME_RATIO | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **11250**<br>MD number | **PROFIBUS_SHUTDOWN_TYPE**<br>PROFIBUS shutdown handling | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 2 | |
| Changes effective after power ON | | | Protection level: 2 / 7 | | Unit: – |
| Data type: BYTE | | | | Applies from SW 6.3 | |
| Meaning: | Handling the PROFIBUS DP on shutdown of the NCK (NCK reset)<br><br>Value 0: The bus is deactivated directly from within cyclic mode, without pre-warning<br><br>Value 1: In the case of NCK shutdown, the PROFIBUS DP is first brought to the CLEAR status for at least 20 clock cycles and then deactivated. If this is not possible on account of the hardware, the procedure is as for the value 2.<br><br>Value 2: In the case of NCK shutdown, the PROFIBUS DP is first brought to the as status for at least 20 clock cycles in which all drives are sent a zero as Control Word 1 and Control Word 2 (pseudo clear). The bus itself remains operational. | | | | |
| Related to .... | – | | | | |

## 4.2 Axis-specific machine data

| | | | | | |
|---|---|---|---|---|---|
| **37600**<br>MD number | **PROFIBUS_ACTVAL_LEAD_TIME**<br>Actual value recording time (PROFIBUS Ti) | | | | |
| Default setting: 0.000125 | | Min. input limit: 0.0 | | Max. input limit: 0.032 | |
| Changes effective after power ON | | | Protection level: 0 / 0 | | Unit: s |
| Data type: DOUBLE | | | | Applies from SW 6.1 | |
| Meaning: | Date and setting of the setpoint transfer time (Ti) of the PROFIBUS encoder<br>Unit: Seconds, default is thus 125µs<br>(this is also the default that Step 7 set for a SIMODRIVE 611 universal) | | | | |
| Special cases, errors, ...... | **Caution:**<br>The actual Ti value is read directly from the PROFIBUS configuration if possible.<br>In this case, the value of the machine data is converted to a **display** machine data machine data according to its commenting. Its value is automatically set to Ti. | | | | |
| Related to .... | – | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **37602**<br>MD number | **PROFIBUS_OUTVAL_DELAY_TIME**<br>Setpoint delay time (PROFIBUS To) | | | | |
| Default setting: 0.003 | | Min. input limit: 0.0 | | Max. input limit: 0.032 | |
| Changes effective after power ON | | | Protection level: 0 / 0 | | Unit: s |
| Data type: DOUBLE | | | | Applies from SW 6.3 | |
| Meaning: | Date and setting of the setpoint transfer time (To) of the PROFIBUS encoder<br>Unit: Seconds, default is thus 3ms | | | | |
| Special cases, errors, ...... | **Caution:**<br>The actual To value is read directly from the PROFIBUS configuration where possible.<br>In this case, the value of the machine data is converted to a **display** machine data machine data according to its commenting. Its value is automatically set to To. | | | | |
| Related to .... | – | | | | |

■

# 5 Signal Descriptions

None

# 6 Example

None

# 7 Data Fields, Lists

## 7.1 Machine data

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| **General ($MN_ ...)** | | | |
| 10050 | SYSCLOCK_CYCLE_TIME | System clock cycle | |
| 10059 | PPOFIBUS_ALARM_MARKER | PTOFIBUS alarm marker (internal only) | |
| 10060 | POSCTRL_SYSCLOCK_TIME_RATIO | Factor for position control cycle | |
| 10061 | POSCTRL_CYCLE_TIME | Position control cycle (SW 5 and higher) | |
| 10062 | POSCTRL_CYCLE_DALAY | Position control cycle offset (SW 5 and higher) | |
| 10070 | IPO_SYSCLOCK_TIME_RATIO | Factor for interpolator cycle | |
| 10071 | IPO_CYCLE_TIME | Interpolation cycle (SW 5 and higher) | |
| 10080 | SYSCLOCK_SAMPL_TIME_RATIO | Division factor of the position control cycle for actual value acquisition | |
| 11250 | PPOFIBUS_SHUTDOWN_TYPE | PROFIBUS DP shutdown handling (SW 6.3 and higher) | |
| **Axis/spindle-specific ($MA_ ...)** | | | |
| 33000 | FIPO_TYPE | Fine interpolator type | G2 |
| 37600 | PPOFIBUS_ACTVAL_LEAD_TIME | Actual value acquisition (Profibus Ti) SW 6.1 and higher | |
| 37602 | PPOFIBUS_OUTVAL_DELAY_TIME | Setpoint delay time (Profibus To) SW 6.3 and higher | |

## 7.2 Alarms

A detailed description of the alarms which may occur is given in
**References:** /DA/, Diagnostics Guide or in the online help in systems with HMI Embedded/MM 102/HMI Advanced.

■

# SINUMERIK 840D/840Di/810D Description of Functions Special Functions (Part 3)

# Contour Tunnel Monitoring (K6)



■

# Notes

# Brief Description 1

## 1.1 Contour tunnel monitoring

**Definition**

The absolute movement of the tool tip in space is monitored. The function is channel-specific (see Chapter 3).

**Model**

A round tunnel with a definable diameter is defined around the programmed path of a machining operation. Axis movements are stopped as an option if the path deviation of the tool tip is greater than the defined tunnel as the result of axis errors.

**Response**

Movement is stopped as soon as the deviation is detected, although at least one interpolator cycle elapses before the system responds.

- An alarm is triggered when the tunnel is violated and the axes continue to traverse or
- violation of the tunnel triggers an alarm and the axis movements are decelerated.

**Deceleration methods**

When the monitoring tunnel is violated, deceleration can be performed either

- according to a braking ramp or
- with speed setpoint zero and follow-up mode.

**Application**

The function can be used for 2D and 3D paths. With 2D paths, the monitoring surface is defined by lines parallel to the programmed path.
The monitoring protection zone is determined by 2 or 3 **geometry axes**.

**Other axes**

Monitoring of synchronous axes, positioning axes, etc. that are not geometry axes is performed directly on the machine axis plane with the “Contour monitoring” function already implemented in SW 1.0.

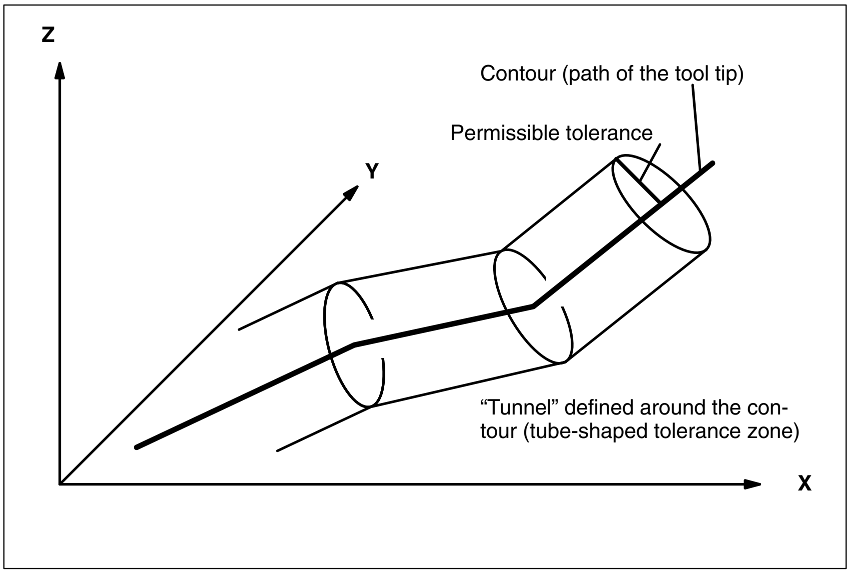



Fig. 1-1 Position of the contour tunnel around the programmed path

Fig. 1-1 is a diagram of the monitoring area shown by way of a simple example. As long as the calculated actual position of the tool tip remains inside the sketched tunnel the motion is continued in the normal way. If the calculated actual position violates the tunnel, an alarm is triggered (in the default setting) and the axes are stopped with a ramp. This response to the violation of the tunnel can be disabled by setting a machine data (alarm triggered but movement continued) or intensified (rapid stop).

**Analysis**

The calculated distance between the programmed path and the actual values can be routed to an analog output to analyze the progression of the contour errors during normal operation (quality control).

## 1.2 Programmable contour accuracy

**Alternative**

As an alternative to the function described in Section 1.1, i.e. monitoring of the machining accuracy and stopping machining if excessive deviations occur, another function is offered as from software version 3.2. With this function, the selected accuracy is always achieved with the path velocity being reduced if necessary. For details of this function, please see Section 2.2.

■

## Notes

# Detailed Description 2

## 2.1 Contour tunnel monitoring

**Aim of the monitoring function**

The aim of the monitoring function is to stop the movement of the axes if axis deviation causes the distance between the tool tip (actual value) and the programmed path (setpoint) to exceed a defined value (tunnel radius).

**Tunnel size**

The radius of the contour tunnel being monitored around the programmed path must be defined to implement the monitoring function. The value is defined in MD: $MC_CONTOUR_TUNNEL_TOL.

If the MD is set to 0.0, monitoring is not performed. The value of the MD is transferred to the control for new configurations.

**Setting the deceleration method**

The deceleration response to the monitoring can be set in MD: $MC_CONTOUR_TUNNEL_REACTION.

One of the following three values must be set:

0: Display alarm, continue machining

1: Deceleration according to the deceleration ramps of the axes (default setting)

2: Rapid stop (speed setpoint zero is set)

**Encoder switchover**

Switching between two encoder systems usually causes a sudden change in the actual position of the tool tip. This change resulting from encoder switchover must not be so large as to cause the tool tip to violate the monitoring tunnel. The radius set in MD: $MC_CONTOUR_TUNNEL_TOL must be greater than the permissible tolerance for the actual-value encoder switchover in MD: $MA_ENC_CHANGE_TOL.

**Activation**

The monitoring function is only activated if

– the contour tunnel monitoring function is set,

– $MC_CONTOUR_TUNNEL_TOL is greater than 0.0 and

– at least two geometry axes have been defined.

**Stop**

The monitoring function is stopped by setting MD: $MC_CONTOUR_TUNNEL_TOL to a value of 0.0.

**Analysis output**

In MD: $MC_CONTOUR_ASSIGN_FASTOUT you can define whether and, if so, to which fast analog output the deviation values of the actual value of the tool tip from the programmed path are to be routed (precision monitoring). The machine data can be set to the following values:

– 0: no output (default setting)

– 1: output to output 1

– 2: output to output 2
etc.

– 8:.....output to output 8

**Equivalence:**
The tunnel radius stored in MD: $MC_CONTOUR_TUNNEL_TOL corresponds to a voltage of 10V at the output.

## 2.2 Programmable contour accuracy

**Initial situation**

There is always a velocity-dependent difference between setpoint and actual position when an axis is traversed without feedforward control. This lag results in inaccurate curved contours.

**References:** /PA/, Programming Guide: Fundamentals

**Function**

The function “Programmable contour accuracy” permits the user to specify a maximum error for the contour in the NC program. The control calculates the KV factor (servo gain factor) for the axes concerned and limits the maximum path velocity so that the contour error resulting from the lag does not exceed the value specified. The Look Ahead function then guarantees that the velocity necessary for maintaining the required contour accuracy is not exceeded at any point along the path.

**Application**

The function guarantees a defined contour accuracy in situations where feedforward control cannot or must not be used.

**Positioning axes**

The function does not affect the velocities of positioning axes.

**Active feedforward control**

The function is also operative in conjunction with active feedforward control if the machine data MC_CPREC_WITH_FFW has the value TRUE. With active feedforward control, the reduction of the path velocity is calculated on the basis of the effective KV factor with feedforward control.

**Minimum feed**

In order to avoid burn marks, the feed is limited only to the minimum value stored in the setting data $SC_MINFEED.

**Activation**

The function can be switched on and off by means of the modal G codes CPRECON and CPRECOF (**C**ontour **Prec**ision **ON**/**OFF**).

The contour accuracy setting is entered in the new setting data $SC_CONTPREC. Changes to the setting data become valid during preprocessing.

**RESET/ end of program**

The response set for the G code group 39 in the machine data

– $MC_RESET_MODE_MASK

– $MC_START_MODE_MASK

becomes effective; i.e. nothing special applies to programmable contour accuracy.

**References:** /FB/, K2, “Workpiece-Related Actual-Value System”

■

## Notes

# Supplementary Conditions 3

**Availability of “Contour tunnel monitoring” function**

This function is an option and available as follows:

- SINUMERIK 840D with NCU 572/573, with SW2 and higher

**Availability of “Programmable contour accuracy” function**

This function is available in SW 3.2 and higher for basic versions of the SINUMERIK 810D, FM-NC, 840D systems.

# Data Descriptions (MD, SD) 4

## 4.1 Channel-specific machine data

| | | | |
|---|---|---|---|
| **21050**<br>MD number | **CONTOUR_TUNNEL_TOL**<br>Response threshold for contour tunnel monitoring | | |
| Default setting: 0.0 | Min. input limit: 0.0 | Max. input limit: *** | |
| Changes effective after NEW CONF | Protection level: 2/4 | Unit: mm | |
| Data type: DOUBLE | Applies from SW 2.0 | | |
| Meaning: | Response threshold for contour tunnel monitoring Specifies the radius of the “tunnel” which is laid around the tool tip path.<br><br>If three geometry axes are defined, then the tunnel can be imagined as a tube with the tool tip path running through the middle of it.<br>If only two geometry axes are defined, then the tube is flattened in the plane of the two geometry axes. | | |
| MD irrelevant for ...... | Contour tunnel monitoring option not available | | |
| Related to .... | CONTOUR_TUNNEL_REACTION, CONTOUR_ASSIGN_FASTOUT, ENC_CHANGE_TOL | | |

| | | | | |
|---|---|---|---|---|
| **21060**<br>MD number | **CONTOUR_TUNNEL_REACTION**<br>Reaction to response of contour tunnel monitoring | | | |
| Default setting: 1 | Min. input limit: 0 | | Max. input limit: 2 | |
| Changes effective after power ON | | Protection level: 2/4 | | Unit: – |
| Data type: BYTE | | Applies from SW 2.0 | | |
| Meaning: | Reaction to activation of alarm<br>0: Display alarm only, continue machining<br>1: Ramp stop<br>2: Rapid stop | | | |
| MD irrelevant for ...... | Contour tunnel monitoring option not available | | | |
| Related to .... | CONTOUR_TUNNEL_TOL, CONTOUR_ASSIGN_FASTOUT | | | |

| | | | | |
|---|---|---|---|---|
| **21070**<br>MD number | **CONTOUR_ASSIGN_FASTOUT**<br>Assignment of an analog output for contour error output | | | |
| Default setting: 0 | Min. input limit: 1 | | Max. input limit: 8 | |
| Changes effective after power ON | | Protection level: 2/4 | | Unit: – |
| Data type: BYTE | | Applies from SW 2.0 | | |
| Meaning: | Assignment of an analog output to which the calculated contour error can be output.<br>0: No output<br>1: Output to output 1<br>2: Output to output 2<br>etc.<br>8: Output to output 8<br>An error which violates the response threshold $MC_CONTOUR_TUNNEL_TOL appears at the output as 10V voltage.<br>An automatic check is performed for multiple assignment of the same output to other signals. | | | |
| MD irrelevant for ...... | Contour tunnel monitoring option not available | | | |
| Related to .... | CONTOUR_TUNNEL_TOL, CONTOUR_TUNNEL_REACTION | | | |

| | | | | |
|---|---|---|---|---|
| **20470**<br>MD number | **CPREC_WITH_FFW**<br>Progr. contour accuracy | | | |
| Default setting: 0 | Min. input limit: 0 | | Max. input limit: 1 | |
| Changes effective after POWER ON | | Protection level: 2 / 7 | | Unit: – |
| Data type: BOOLEAN | | Applies from SW: 3.2 | | |
| Meaning: | This machine data is for defining the response of the programmable function CPRECON in conjunction with feedforward control.<br>FALSE: The function CPRECON is inoperative with simultaneously active feedforward control.<br>TRUE: CPRECON is also operative with feedforward control. | | | |
| Related to .... | $SC_CONTPREC, $SC_MINFEED | | | |

# 4.2 Channel-specific setting data

| | | | | | |
|---|---|---|---|---|---|
| **42450**<br>MD number | **CONTPREC**<br>Contour accuracy | | | | |
| Default setting: 0.1 | | Min. input limit: 0.000001 | | Max. input limit: 999999 | |
| Changes effective after POWER ON | | | Protection level: 7 / 7 | | Unit: mm |
| Data type: DOUBLE | | | | Applies from SW: 3.2 | |
| Meaning: | Contour accuracy. With this setting data you can specify which accuracy is to be applied for the path of the geometry axes on curved contours. The smaller the value and the smaller the servo gain factor of the geometry axes, the higher the reduction in path feed on curved contours. | | | | |
| Related to .... | $MC_CPREC_WITH_FFW, $SC_MINFEED | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **42460**<br>MD number | **MINFEED**<br>Minimum path feed with CPRECON | | | | |
| Default setting: 1.0 | | Min. input limit: 0.000001 | | Max. input limit: 999999 | |
| Changes effective after POWER ON | | | Protection level: 7 / 7 | | Unit: mm/min |
| Data type: DOUBLE | | | | Applies from SW: 3.2 | |
| Meaning: | Minimum path feed with active “contour accuracy” function. The feed is not reduced to below this value, unless a lower F value has been programmed or the value is not permitted by the axis dynamics. | | | | |
| Related to .... | $MC_CPREC_WITH_FFW, $SC_CONTPREC | | | | |

■

## Notes

# 5 Signal Descriptions

None

■

# 6 Example

## 6.1 Programmable contour accuracy

Extract

```
N10 X0 Y0 G0
N20 CPRECON                 ; Switch on the contour accuracy defined by MD
N30 F10000 G1 G64 X100      ; Machine at 10 m/min in continuous-path mode
N40 G3 Y20 J10              ; Automatic limitation of feedrate in circle block
N50  G1 X0                  ; Feedrate again 10 m/min
...
N100 CPRECOF                ; Switch off the programmed contour accuracy
N110 G0 ...
```

■

# Notes

# 7 Data Fields, Lists

## 7.1 Machine data

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| **Channel-specific ($MC_ ...)** | | | |
| 20470 | CPREC_WITH_FFW | Programmable contour accuracy | |
| 21050 | CONTOUR_TUNNEL_TOL | Response threshold for contour tunnel monitoring | |
| 21060 | CONTOUR_TUNNEL_REACTION | Reaction to response of contour tunnel monitoring | |
| 21070 | CONTOUR_ASSIGN_FASTOUT | Assignment of an analog output for output of the contour error | |
| **Axis/spindle-specific ($MA_ ...)** | | | |
| 36500 | ENC_CHANGE_TOL | Maximum tolerance for position actual value switchover | G2 |
| **Channel-specific setting data ($SC_ ...)** | | | |
| 42450 | CONTPREC | Contour accuracy | |
| 42460 | MINFEED | Minimum path feedrate for CPRECON | |

## 7.2 Alarms

A detailed explanation of the alarms which may occur is given in

**References:** /DA/, Diagnostics Guide

or the online help for systems with MMC 101/102/103.

■

# Notes

# SINUMERIK 840D/840Di/810D
# Description of Functions Special Functions (Part 3)

# Axis Couplings and ESR

**Former title: "Coupled motion and master value coupling" (M3)**

■

# Notes

# Brief Description 1

**Note**

Since the range of functions covered by M3 has now been extended, the description has been given a more appropriate and general title.

## Axis couplings and ESR

### Former title: “Coupled motion and master value coupling” (M3)

**ESR** stands for Extended Stop and Retract.

## 1.1 Coupled axes

With the “Coupled axes” function (see Section 2.1), it is possible to define any axis as a leading axis and to assign any number of coupled axes to it.

The coupled axis/axes is/are moved simultaneously with the leading axis as a function of the coupling factor.

It is also possible to program an independent motion for the coupled axis/axes.

The master and coupled axes are defined and the coupled axis grouping is activated/deactivated in the NC parts program or by synchronous action.

The “Coupled axes” function can also be used in manual operation.

## 1.2 Curve tables

Using the function “Curve tables” (see Section 2.2) you can define the complex sequence of motions of an axis in a curve table.

Any axis can be defined as a leading axis and a following axis traversed according to a curve table.

The command variable in these motion sequences is an abstract master value which is generated by the control or derived from an external variable (e.g. position of an axis).

The master and following axes are defined and activated/deactivated with the curve table in the NC parts program or by synchronous action.

The curve tables in SRAM remain valid after the end of a parts program or power down.

For SW 6.3 and higher, curve tables can also be stored in DRAM. They must be loaded again to DRAM after a power ON.

Axis groups must be re/activated independently of the storage location of the curve tables after power ON.

## 1.3 Master value coupling

With the “Master value coupling” function (see Section 2.3) it is possible to process short programs cyclically with close coupling of the axes to one another and a master value that is either generated by the control or input from an external source.

The master value can be derived from a conveyor belt or a line shaft.

Either an axis or path master value coupling can be used which can be activated and deactivated in the NC parts program or via a synchronous action.

The coupling with the master value is defined using a curve table.

## 1.4 Electronic gear EG (SW 5 and higher)

**SW 5**

With the “Electronic gear” function (see Section 2.4) it is possible to control the motion of one **following axis** as the function of up to five **leading axes**. The relationship between each leading axis and the following axis is defined by the coupling factor. The following axis motion components derived in this manner from the individual leading axis motion components have an additive effect. The coupling factors can be referred to one of the following quantities:

- Actual values of leading axis.
- Setpoints of leading axis.

An electronic gear grouping can be defined, switched on, switched off and deleted from the parts program.

When an EG axis grouping is activated, it is possible to synchronize the following axes in relation to a defined position.

Application examples:

– Machine tools for gear cutting.

– Gear trains for production machines.

**SW 6**

As an extension of the functions of SW 5, SW 6 also allows the implementation of non-linear relationships between the leading axes and the following axis via curve tables (see Sections 1.2 and 2.2).

Electronic gears can be cascaded, i.e. the following axis of an electronic gear can be the leading axis for a further electronic gear.

For details of this extension, please see Section 2.4.

**SW 6.4**

An additional function for synchronizing the following axis permits the following selection:

– Approach next division (tooth gap) time-optimized
– Approach next division (tooth gap) path-optimized
– Approach in positive direction of axis rotation, absolute
– Approach in negative direction of axis rotation, absolute
– Traverse time-optimized with respect to programmed synchronized position
– Traverse path-optimized with respect to programmed synchronized position

## 1.5 Extended stop and retract: ESR (SW 5 and higher)

**SW 5**

The “Extended stop and retract” function ESR allows the operator to choose how to react to definable error sources, reducing the risk of causing damage to the workpiece.

- **Stop**: As far as possible, all axes participating in the electronic coupling are stopped (this concerns the axes in the machining plane).
- **Retract** (SW 5.2 and higher): The tool/workpiece pair engaged in action is separated as quickly as possible by retracting the tool from the gear wheel gap.
- **Generator operation:** In the event of a power failure, the power for the retract operation can be provided in the DC link (the DC link voltage is maintained through additional back-up capacitors or drive operating in generator mode).

**SW 6**

As expansion of the stop and retract measures implemented in SW 5, **NC-prompted** stop and retract operations are available from SW 6. With NC-prompted operations, for example, an axis group of an electronic gear grouping remains in synchronism during stopping for a specified period to prevent risks to:

- Man
- Material (workpiece and expensive hobbing cutter for example)

as far as possible.

■

# Notes

# Detailed Description 2

## 2.1 Coupled axes

### 2.1.1 General functionality

**Dependent coupled axis**

The “Coupled axes” function allows any axis of the control to be declared as a “master axis” and any desired number of “coupled axes” to be assigned to the “leading axis”.

The “master” and “coupled” axes then form a coupled axis grouping.

If the leading axis is programmed in a block of the NC parts program, all following axes in the axis grouping traverse the paths derived from the leading axis via coupling factors.

A total of 2 leading axes may be assigned to each coupled axis.

**Independent coupled axis**

A coupled axis can be programmed with the full range of available motion commands (G0, G1, G2, G3,...). The coupled axis not only traverses the independently defined paths, but also those derived from its leading axes on the basis of coupling factors.

**Position of a coupled axis**

The position of a coupled axis at any given time corresponds to the sum of the dependent motion (motion of leading axis allowing for coupling factor) and the independent motion (i.e. the motion programmed for the axis concerned).

**Coupled axis as leading axis**

A coupled axis can also act as the leading axis for other coupled axes. In this way, it is possible to create a range of different coupled axis groupings.

**Axis types**

A coupled axis grouping can consist of any desired combinations of linear and rotary axes.

In this case, it is possible to define a “simulated” axis to act as the leading axis.

**Coordinate system**

Coupled axis motion is always executed in the base coordinate system (BCS).

**Synchronous action**

Coupled axis motion can be switched on and off by means of a synchronous action. Following activation of the coupled motion, the following axis is first accelerated to the new set speed. The position of the leading axis after synchronization of the speed is taken as the start position for coupled motion.

**Activating, deactivating**

The coupled axis activated per synchronous action can only be deactivated again per synchronous action.

The coupled axis activated per parts program can only be deactivated again by means of the parts program.

**Application**

1. **Traversal of an axis by means of a simulated axis**
   The leading axis is a “simulated” axis and the coupled axis a “real” axis. In this way, the real axis can be traversed as a function of the coupling factor.
2. **Two-side machining**
   The following configuration applies for this application:

- Coupled axis grouping: Leading axis “Y” ⇔ coupled axis “V”
- Coupled axis grouping: Leading axis “Z” ⇔ coupled axis “W”

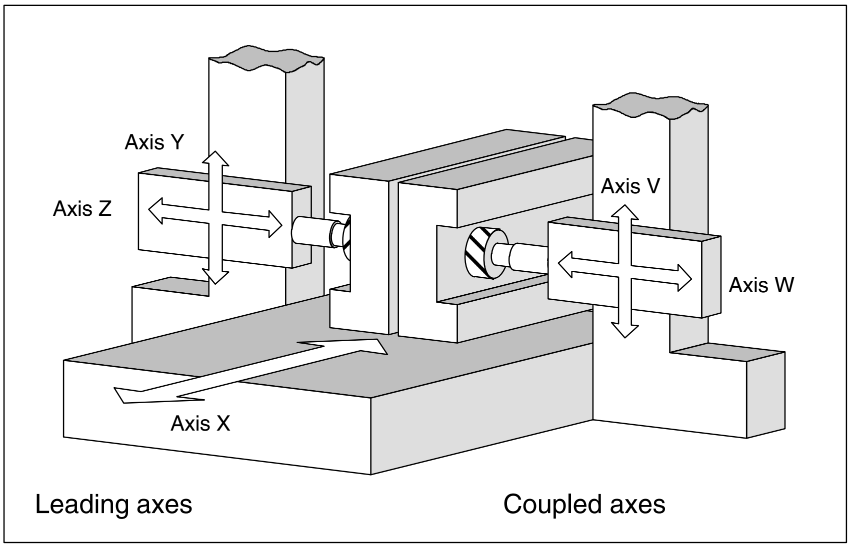



Fig. 2-1 Two-side machining

### 2.1.2 Programming a coupled axis grouping

**Definition and activation**

A coupled axis grouping is defined and activated simultaneously with the modal command:

**TRAILON**(coupled axis, leading axis, coupling factor)

Parameters

Coupled axis, leading axis:
The corresponding axis identifier must be specified here.

Coupling factor:
The coupling factor specifies the desired relationship between the paths of the coupled axis and the leading axis.

$$\text{Coupling factor} = \frac{\text{Paths of the coupled axis}}{\text{Paths of the leading axis}}$$

---

**Note**

- If a coupling factor is not programmed, then coupling factor 1 automatically applies.
- The factor is entered as a fraction with decimal point (of type REAL). The input of a negative value causes the master and coupled axes to traverse in opposition.
- A coupled axis may not be activated in more than two coupled axis groupings at the same time.
- The number of coupled axis groupings which may be simultaneously activated is limited only by the maximum possible number of combinations of axes on the machine.
- Depending on the setting of MD 20110: RESET_MODE_MASK, bit 8
  MD 20112: START_MODE_MASK, bit 8
  all active coupled axis groupings are canceled after RESET or NC Start from the RESET state.

---

**Deactivation**

The coupling of **one** leading axis is deactivated with the command:

**TRAILOF**(coupled axis, leading axis)

In this case, a preprocessing stop is generated.

---

**Note**

An active coupling can also be deactivated from the PLC by means of an ASUB.

**References:** /FB/, P3, “Basic PLC Program”

---

Examples

Example 1: Example of an NC parts program for the axis constellation shown in Fig. 2-1:

```
       :
TRAILON (Y,V,1)      ;Activate 1st coupled axis grouping
TRAILON (Z,W,–1)     ;Activate 2nd coupled axis grouping
G0 Z10               ;Infeed Z and W axes in oppositeaxial directions
G0 Y20               ;Infeed Y and V axes in the same axial direction
G1 Y22 V25           ;Overlay a dependent and an independent motion of
                     ;coupled axis "V"
       :
TRAILOF (Y,V)        ;Deactivate 1st coupled axis grouping
TRAILOF (Z,W)        ;Deactivate 2nd coupled axis grouping
       :
```

Example 2: With coupled axes, two motions are added together: the motion of the following axis which is dependent on the leading axis, and the independent motion which is based on the programming of the following axis. The dependent part is treated like a coordinate offset such that it is possible to set a defined behavior for the following axis.

| | |
|---|---|
| N01 G90 G0 X100 U100 | |
| N02 TRAILON(X,U,1) | ;Activate coupled-axis grouping |
| N03 G1 F2000 X200 | :Dependent motion of U, $U_{pos}$=200, $U_{Trail}$=100 |
| N04 U201 | ;Independent motion, $U_{pos}$=U201+$U_{Trail}$=301 |
| N05 X250 | ;Dependent motion of U,<br>;$U_{Trail}$=$U_{Trail}$(100)+50=150, $U_{pos}$=351 |
| N06 G91 U100 | ;Independent motion, $U_{pos}$(351)+U100=451 |
| N07 G90 X0 | :Dependent motion of U,<br>;$U_{pos}$=$U_{pos}$(451)–$U_{Trail}$(250)=201 |
| N10 TRAILOF(X,U) | |

## 2.1.3 Behavior in AUTOMATIC, MDA, JOG modes

**Effectiveness**

An activated coupled axis grouping is functional in the AUTOMATIC, MDA and JOG modes.

**Manual operation**

When a coupled axis grouping is activated, traversal of the leading axis (e.g. in rapid traverse or incremental feed mode INC1 ... INC10000) causes the coupled axis to traverse simultaneously as a function of the coupling factor.

**Reference point approach**

A reference point approach does not separate coupled axis groupings.

**Deletion of distance-to-go**

When deletion of distance-to-go is executed for a leading axis, all axes in the associated, activated coupled axis groupings are shut down.

Deletion of distance-to-go of a coupled axis only shuts down independent movements of this axis.

**Basic setting after power-up**

The grouping status after power-up is dependent on the setting in MD 20112: STAR_MODE_MASK, i.e.:

- Coupled axis groupings are retained (bit 8=0)
- Coupled axis groupings are invalidated (bit 8=1)

**Basic setting after RESET/parts program end**

The grouping status after RESET/parts program end is dependent on the setting in MD 20110: STAR_MODE_MASK, i.e.:

- Coupled axis groupings are invalidated on RESET/parts program end (bit 8=0)
- Coupled axis groupings remain active after RESET/parts program end and parts program start (bit 8=1)

## 2.1.4 Effectiveness of PLC interface signals

**Independent coupled axis**

All relevant channel-specific and axis-specific interface signals of the coupled axis are effective for the independent motion of this axis, for example
Activate DRF (DB21–28, DBX0.3),

Feed override (DB31–48, DBX0.0 – 0.7)
Axis inhibit (DB31–48, DBX1.3),
Controller enable (DB31–48, DBX2.1),
Activate handwheel (DB31–48, DBX4.0 – 4.2),
Feed stop (DB31–48, DBX4.3), etc.

⇒ For the purpose of independent motion, for example,

- the velocity can be altered via the feed override and
- a DRF offset can be input with the handwheel in the AUTOMATIC and MDA modes.

**Dependent coupled axis**

With respect to the motion of a coupled axis which is dependent on the leading axis, only the coupled axis interface signals which effect termination of the motion (e.g. axis-specific feed stop, axis inhibit, controller enable, etc.) are effective.

**Leading axis**

When a coupled axis grouping is active, the interface signals (IS) of the leading axis are applied to the appropriate coupled axis via the axis coupling, i.e.

- a position offset or feed control action of the leading axis is applied via the coupling factor to effect an appropriate position offset or feed control action in the coupled axis.
- shutdown of the leading axis as the result of an IS (e.g. axis-specific feed stop, axis inhibit, controller enable, etc.) causes shutdown of the appropriate coupled axis.

**Position measuring system 1/2 (DB31, ... DBX1.5/1.6)**

Switchover of the position measuring system for the master and coupled axes is not inhibited for an active coupled axis grouping. The coupling is not invalidated on measuring system switchover.

**Recommendation:** Switch the measuring system over when the coupling is deactivated.

### 2.1.5 Special characteristics of function

**Control dynamics**

Depending on the application in question, it may be advisable to match the position controller parameter settings (e.g. servo gain factor) of the master and coupled axes in a coupled axis grouping. It may be necessary to activate other parameter sets for coupled motion.

**Acceleration and velocity limits**

The acceleration and velocity limits of the axes in the coupling are determined by the “weakest” axis in the coupled axis grouping.

**Note**

The following applies to coupled axis activated with synchronous action:
If the following axis does not have a sufficient dynamic response (e.g. controller parameters) to follow the leading axis, an alarm is issued.

**Multiple couplings**

If the coupling is activated for a coupled axis grouping which is already active, then the activation process is ignored and an appropriate alarm generated.

**Status of coupling**

The coupling status of an axis can be interrogated with the system variable

**$AA_COUP_ACT**[axis identifier]

in the NC parts program.

| Coding of $AA_COUP_ACT | Meaning |
|---|---|
| 0 | No coupling active |
| 1, 2, 3 | Tangential follow-up |
| 4 | Synchronous spindle coupling |
| 8 | Coupled motion active |
| 16 | Master value coupling |
| 32 | Following axis of electronic gear |

**Note**

Only one of the 4 coupling modes may be active at any given time.

**Actual value display**

The display of the actual position and distance to go is updated for all axes in a coupled axis grouping.
In this case, the distance to go display of the coupled axes refers to the total path of the motion.

## 2.2 Curve tables

### 2.2.1 General functionality

**Curve table**

A functional relation between a command variable "Master value" and an abstract following value is described in the curve table.

A following variable can be assigned uniquely to each master value within a defined master value range.

**Curve segment**

The functional relation can be subdivided into separate sections of the master value axis, called curve segments.

The relation between the master value and following value is generally described by a polynomial up to the third order.

From **SW 6**, polynomials up to the **5th degree** are permissible. See also

**References:** /PGA/, Programming Guide "Advanced".

**Tool radius offset SW 6.3**

As of SW 6.3, curve tables are available in which it is possible to specify the Tool radius offset in the table definition even if polynomial blocks or blocks with no motion for an axis, or jumps for the following axis, occur in the curve table (G41/G42/G40 in the table definition).

**Tool radius offset**

Tool radius offset (G41/G42) may be programmed with restriction within a curve table up to SW 6.2. The contour to be corrected must consist only of linear and circular blocks and the circular blocks must have a circular plane that corresponds to that of the offset plane.

For more complex contours (splines, polynomials), the tool radius offset is available in the table of curves **SW 6.3 and higher**.

The equidistant curve (tool center point path of the tool radius offset) of a curve consisting of polynomials can no longer be displayed exactly using polynomials. The associated curve tables must be approximated stepwise using polynomials in this case. This means that the number of segments of the curve tables no longer matches the number of programmed segments. The number of segments required for the curve table is defined by the bend of the curve. The larger the bend for the programmed curve, the more segments are required for the curve table.

On account of the tool radius correction for curve tables, more memory may be required. The selection option for the memory type means that this need not produce a shortage of SRAM memory, however.

The tool radius correction can produce segments for which the following axis or leading axis have no movement. Whereas a missing movement in the following axis does not cause a problem, the same situation for the leading axis must be solved by defining the behavior for handling discontinuities.

Machine data MD 20900: CTAB_ENABLE_NO_LEADMOTION can be set to specify whether or not a curve table is set in this case. The following options are available:

0:
No curve tables that contain a jump in the following axis are procduced. Alarm 10949 is output and the program processing is aborted.

1:
Curve tables with a jump in the following axis can be generated. If a segment contains a jump in the following axis, Alarm 10955 is output but the program processing is continued.

2:
Curve tables with a jump in the following can be created without an alarm being output.

---

**Note**

In the case of a curve table that contains segments without leading axis movement (this means that the following axis jumps at this point), the following axis can only make a jump within its dynamic limits (max. velocity and max. acceleration). This means that there is always a deviation from the programed curve.

---

**Defining the memory type for curve tables**

As of SW 3, it is possible to specify the memory type to be used for creating he curve table when defining the curve table. The following options are available:

SRAM

DRAM

Table definitions for memory type SRAM are still available after power up of the control. Curve tables for DRAM must be re–defined after power up of the control. Additional machine data have been provided for reserving space in DRAM:

MD 20905: CTAB_DEFAULT_MEMORY_TYPE defines the memory type if no memory type has been specified in the table definition (see above).

Storing curve tables in a variety of memory types entails an optional specification of the memory type in delete calls for curve tables. (see above).

## 2.2.2 Programming a curve table

**Definition**

The following modal language commands act on curve tables:
(explanations for the parameters are given at the end of the list of functions)

- Beginning of definition of curve table:

  **CTABDEF**(following axis, leading axis, n, applim, memType)

- End of definition of curve table:

  **CTABEND**()

- Delete curve table(s):

  **CTABDEL**(n) ; Curve table n
  **CTABDEL**(n, m) ; [n < m], several in number range
  ; Deletion is performed in SRAM and DRAM.

  **CTABDEL**(n, m, memType) ; Delete from specified memory type:

  The curve tables with the numbers in the specified range are deleted from the specified memory type. The others are retained.

  Delete all tables in a particular memory type:
  **CTABDEL**(, , "DRAM") ; All in DRAM or
  **CTABDEL**(, , "SRAM") ; All in SRAM:

  **CTABDEL**() ; All, independent of memory type

- Read the following value for a master value

  **CTAB**(master value, n, degrees, [following axis, leading axis])

- Read the master value for a following value

  **CTABINV**(following value, approx. master value, n, degrees, [following axis, leading axis])

- Read start value (following axis) of a table segment

  **CTABSSV**(leading value, n, degrees, [following axis, leading axis])

- Read end value (following axis) of a table segment

  **CTABSEV**(leading value, n, degrees, [following axis, leading axis])

- The following functions allow you to enable or inhibit deletion and overwriting of parts programs:

- **Set lock** against deletion and overwriting.
  General form: **CTABLOCK**(n, m, memType)
  Applications of the forms:
  Curve table with number n
  **CTABLOCK**(n)

  Curve tables in the number range from n to m
  **CTABLOCK**(n, m)

  All curve tables irrespective of the memory type
  **CTABLOCK**()

All curve tables of the specified memory type
**CTABLOCK**(, , memType)

- **Cancel locking** against deletion and overwriting.

  CTABUNLOCK releases the tables locked with CTABLOCK. Tables which are involved in an active coupling remain locked, i.e. they cannot be deleted. However, the CTABLOCK command is canceled, i.e. the table can be deleted as soon as the coupling is deactivated. It is not necessary to call CTABUNLOCK again.

  General form: **CTABUNLOCK**(n, m, memType)
  Applications of the form:
  Curve table with number n
  **CTABUNLOCK**(n)

  Curve tables in the number range from n to m
  **CTABUNLOCK**(n, m)

  All curve tables irrespective of memory type
  **CTABUNLOCK**()

  All curve tables of the specified memory type
  **CTABUNLOCK**(, , memType)

**Parameters**

- Following axis:
  Identifier of axis via which the following axis is programmed in the definition.
- Leading axis:
  Identifier of axis via which the leading axis is programmed.
- n, m
  Numbers for curve tables.

  The curve table numbers can be freely assigned. They are used exclusively to uniquely identify a curve table.

  In order to delete a curve table area using the command CTABDEL(n, m), m must be greater than n.
- p
  Entry location (in memory area memType)
- applim:
  Behavior at the curve table edges.
  – 0 non-periodic (table is processed only once, even for rotary axes)
  – 1 periodic, modulo (the modulo value corresponds to the LA table values)
  – 2 periodic, modulo (LA and FA are periodic)
- Master value
  Position value for which a following value is to be determined.
- Following value
  Position value for which a master value is to be calculated.
- aproxmastervalue
  Position value that can be used to determine a unique master value in the case of an ambiguous reversing function of the curve table.
- grad
  Parameter in which the pitch of the table function is returned.
- memType

  Optional parameter for specifying the memory type to be used to create the curve table.
  Possible values:
  “SRAM” Curve table is created in static memory
  “DRAM” Curve table is created in dynamic memory.

**Restrictions**

The following restrictions apply when programming:

- The NC block must not generate a preprocessing stop.
- No jumps must occur in the leading axis motion.
- Any block that contains a travel instruction for the following axis must also include a movement for the leading axis.
- The direction of motion of the leading axis must not reverse at any point in the rule of motion, i.e. the position of the leading axis must always be unique within the sequence of motions.
  The programmed contour must not run perpendicular to the leading axis.
- Axis names from gantry axis groups cannot be used to define a table (only leading axis are possible).

**Restrictions as of SW 6.3**

Depending on the setting in MD 20900: CTAB_ENABLE_NO_LEADMOTION, jumps in the following axis may be tolerated when a movement is missing in the leading axis. The usual restrictions stated in the previous section still apply.

**Axis assignment**

Does not take effect until coupling is activated with curve table.

---

**Note**

The dynamic limit values of the motion commands for a curve table are not checked until activation or interpolation.

---

**Starting value**

The first motion command in the definition of a curve table defines the starting value for the master and following value.

All instructions that cause a preprocessing stop must be skipped.

**Example 1**

No tool radius correction, no memory type

```
N100 CTABDEF(AX2, AX1, 3,0)    ;Beginning of definition of non–periodic
                               ;curve table number 3
N110 AX1=0 AX2=0               ;1st travel command determines the
                               ;starting value
                               ;Master value: 0, following value:0
N110 AX1=20 AX2=0              ;1st curve segment:
                               ;Master value: 0...20, following value:
                               ;starting value ...0
N120 AX1=100 AX2=6             :2nd curve segment:
                               ;Master value: 20...100, following value:
0...6
N130 AX1=150 AX2=6             ;3rd curve segment:
                               ;Master value: 100...150, following value 6
N130 AX1=180 AX2=0             ;4th curve segment:
                               ;Master value: 150...180, following value:
6...0
N200 CTABEND                   ;End of definition, the curve table is
                               ;generated in its internal form.
                               ;Preprocessing reorganizes to state at
                               ;beginning of N100
```

**Example 2**

Example of a curve table with active tool radius correction:

Prior to definition of a curve table with CTABDEF(), the tool radius correction must not be active; otherwise, Alarm 10942 is output. This means that the tool radius correction **must be activated within** the definition of the curve table. Similarly, it must be deactivated again before the end of the curve table definition using CTABEND.

```
                 ...
N10 CTABDEF(Y, X, 1, 0)          ;Beginning of definition of non–periodic
                                 ;Periodic curve table number 1
N20 X0 Y0
N30 G41 X10 Y0                   ; TR correction ON
N40 X20 Y20
N50 X40 Y0
N60 X60 Y20
N70 X80 Y0
N80 G40 X90 Y0                   ; TR correction OFF
N90 CTABEND
                 ....
```

The tool radius correction is activated in block N30; this causes the approach movement for radius correction to be made in this block. Similarly, the approach movement for deactivation of the radius correction is made in block N80.

---

**Note**

The value pairs between CTABDEF and CTABEND must be specified for precisely the axis identifiers that have been programmed in CTABDEF as the leading axis and following axis identifiers. In the case of programming errors, alarms or incorrect contours may be generated.

---

**Reading table positions**

With the program commands CTAB and CTABINV you can read off the following value for a master value (CAB) from the parts program and from synchronous actions, or the reversal of the curve table, i.e. read off the master value for a following value. With the aid of the pitch value, you can calculate the speed of the following axis or leading axis at any position in the table.

**Reading segment positions**

The segment positions of a curve table for the value for the following axis can be read using the calls:

CTABSSV

CTABSEV

The language commands CTABSSV and CTABSEV generally provide the start and end values of the internal segments of the curve tables for the following axis. These values only agree with the programmed values of the curve tables if the programmed segments can be converted 1:1 to the internal segments of the curve table. This is always ensured if only G1 blocks or axis polynomials are used to defined the curve tables and no other functions are active.

The programmed segments may be modified when converted to the internal segments of the curve tables in the following cases:

1. Circles or involutes are programmed
2. Chamfer or rounding is active (CHF, RND)
3. Smoothing with G643 is active
4. Compressor is active (COMPON, COMPCURV, COMPCAD)
5. Tool radius correction is active for polynomial interpolation.

In these cases, the language commands CTABSSV and CTABSEV may not be sued to query the start and end points of the programmed segments. An example for reading the start and end values of the segment is given in:

**References:** PGA, Programming Guide, Production Planning

**Optional parameters**

The functions CTAB, CTABINV, CTABSSV and CTABSEV have optional parameters for the leading and following axes. If one of these parameters is programmed, the master value and following value are modified using the scaling factors of the relevant axes.

This is particularly important if axes have been configured with different length units (inch/metric). If no optional parameters are programmed, the master value and following value are treated as path positions in the conversion from external to internal representation. This means that the values are multiplied according to the configured resolution (decimal places) and the remaining decimal places truncated.

**CTABINV**

When using the inversion function for the curve tables (CTABINV), you must note that the following value mapped to the leading value may not be unique.

Within a curve table, the following value can assume the same value for any number of master value positions. In order to resolve this ambiguity, the program command CTABINV requires a further parameter, in addition to the following value, which it uses to select the 'correct' master value. CTABINV always returns the master value that is closest to this auxiliary parameter. This auxiliary value can, for example, be the master value from the previous interpolator cycle.

---

**Note**

Although the auxiliary parameter permits calculation of a unique result for the reversal function of the curve table, it should be noted that numerical inaccuracies may give rise to contours, which can cause the reversal function to produce results that deviate from those that would be obtained in a calculation where the accuracy is unrestricted.

---

**Activation**

Coupling a real axis to a curve table:

**LEADON** (following axis, leading axis, n)     n=number of curve table

The coupling between an existing channel axis and master value can be established via a curve table

- in a parts program or
- in the definition of a synchronous action.

**Multiple use**

A curve table can be used several times in a single parts program to couple different channel axes.

**Deactivation**

**LEADOF** (following axis, leading axis)

You can deactivate the function both in the program and using synchronous actions.

**Example**

```
N1000    LEADON(A, X, 3)    ;axis A follows the master value X according to
                            ;the rules of motion defined in curve table
                            ;number 3.
N1010    LEADOF(A, X)       ;the coupling of axis A to its master value
                            ;is canceled.
...
N10010   CTABDEL(3)         ;curve table number 3 is deleted,
                            ;it is no longer available for
                            ;activation of a coupling.
```

**Using memory**

The memory available in SRAM and DRAM (from SW 6.4) for the curve tables is limited.

**SRAM**

MD 18400: MM_NUM_CURVE_TABS specifies the number of curve tables that can be stored in SRAM.

18402: MM_NUM_CURVE_SEGMENTS specifies the number of curve table segments that can be stored in SRAM

18404: MM_NUM_CURVE_POLYNOMS specifies the number of curve table polynomials that can be stored in SRAM.

**DRAM**

MD 18406: MM_NUM_CURVE_TABS_DRAM specifies the number of curve tables that can be stored in DRAM.

MD 18408: MM_NUM_CURVE_SEGNENTS_DRAM specifies the number of curve table segments that can be stored in DRAM.

MD 18410: MM_NUM_CURVE_POLYNOMS_DRAM specifies the number of curve table polynomials that can be stored in DRAM.

**Preliminary table**

When a **new curve table is created**, a temporary curve table is set up first in the memory and then extended block by block. On completion (CTABEND), the table is checked for consistency. The temporary table is converted to a table than can be used in a coupling only if it is found to be consistent.

**Insufficient memory**

If there is insufficient memory available to create a new curve table, the temporary table is deleted again as soon as the appropriate alarm is activated.

If there is insufficient memory, one or more tables that is/are no longer required can be deleted with CTABDEL or, alternatively, the memory re-configured via the appropriate MD.

**Same table number**

A new curve table may have the same number as an **existing table**. The new curve table then overwrites the existing table with the same number. This is done only if the new curve table does not contain any errors. If an error is detected in the new table, the old table is not overwritten.

If the user wishes to have the option of overwriting an existing curve table without deleting it first, then he will need to dimension the table memory such that there is always **enough extra memory to accommodate the table to be overwritten**.

**Overwriting curve tables**

Curve tables that are not active in a master value coupling may be overwritten.

**Deleting curve tables**

Curve tables that are not active in a leading value coupling may be deleted.

**Transformation**

Transformations may not be programmed in curve tables. RAANG is an exception. If TRAANG is programmed, the motion rule programmed in the basic coordinate system is transformed to the associated machine coordinate system. In this way, it is possible to program a curve table as Cartesian coordinates for a machine with inclined linear axes.

The condition which stipulates that “the direction of motion of the leading axis must not reverse at any point of the motion rules” must then be satisfied in the machine coordinate system. It must be noted that this condition in the basic coordinate system does not have the same meaning as in the machine coordinate system, since the contour tangents are changed by the transformation.

**Generating curve tables (example)**

with linear blocks:

```
%_N_TAB_1_NOTPERI_MPF
;$PATH=/_N_WKS_DIR/_N_CURVETABLES_WPD
;Def.TAB1 0–100mm Kue1/1 non-perio.
N10 CTABDEF(YGEO,XGEO,1,0) ;FA=Y LA=X Curve No.=1 Non–perio
N1000 XGEO=0 YGEO=0            ; start values
N1010 XGEO=100 YGEO=100
CTABEND
M30
```

with polynomial blocks:

```
%_N_TAB_1_NOTPERI_MPF
;$PATH=/_N_WKS_DIR/_N_CURVETABLES_WPD
;Def.TAB1 0–100mm Kue1/1 non-perio.
N10 CTABDEF(Y,X,1,0) ;FA=Y LA=X Curve No.=1 non-perio
N16 G1  X0.000 Y0.000
N17  POLY PO[X]=(31.734,0.352,–0.412) PO[Y]=(3.200,2.383,0.401)
N18  PO[X]=(49.711,–0.297,0.169) PO[Y]=(7.457,1.202,–0.643)
N19  PO[X]=(105.941,1.961,–0.938) PO[Y]=(11.708,–6.820,–1.718)
N20  PO[X]=(132.644,–0.196,–0.053) PO[Y]=(6.815,–2.743,0.724)
N21  PO[X]=(147.754,–0.116,0.103) PO[Y]=(3.359,–0.188,0.277)
N22  PO[X]=(174.441,0.578,–0.206) PO[Y]=(0.123,1.925,0.188)
N23  PO[X]=(185.598,–0.007,0.005) PO[Y]=(–0.123,0.430,–0.287)
N24  PO[X]=(212.285,0.040,–0.206) PO[Y]=(–3.362,–2.491,0.190)
N25  PO[X]=(227.395,–0.193,0.103) PO[Y]=(–6.818,–0.641,0.276)
N26  PO[X]=(254.098,0.355,–0.053) PO[Y]=(–11.710,0.573,0.723)
N26  PO[X]=(254.098,0.355,–0.053) PO[Y]=(–11.710,0.573,0.723)
N27  PO[X]=(310.324,0.852,–0.937) PO[Y]=(–7.454,11.975,–1.720)
N28  PO[X]=(328.299,–0.209,0.169) PO[Y]=(–3.197,0.726,–0.643)
N29  PO[X]=(360.031,0.885,–0.413) PO[Y]=(0.000,–3.588,0.403)
CTABEND
N30 M30
```

## 2.2.3 Behavior in AUTOMATIC, MDA, JOG modes

**Effectiveness**

An activated curve table is functional in the AUTOMATIC, MDA and JOG modes.

**Basic setting after power-up**

No curve tables are active after power-up.

### 2.2.4 Effectiveness of PLC interface signals

**Dependent Following axis**

With respect to the motion of a following axis which is dependent on the leading axis, only the following axis interface signals which effect termination of the motion (e.g. axis-specific feed stop, axis inhibit, controller enable, etc.) are effective.

**Leading axis**

When a coupled axis grouping is active, the interface signals (IS) of the leading axis are applied to the appropriate following axis via the axis coupling, i.e.

- A feed control of the leading axis causes a corresponding feed control of the following axis.
- Shutdown of the leading axis as the result of an IS (e.g. axis-specific feed stop, axis inhibit, controller enable, etc.) causes shutdown of the appropriate following axis.

**Position measuring system 1/2 (DB31, ... DBX1.5/1.6)**

Switchover of the position measuring system for the leading and following axes is not inhibited for an active coupled axis grouping. The coupling is not invalidated on measuring system switchover.

**Recommendation:** Switch the measuring system over when the coupling is deactivated.

### 2.2.5 Diagnostics and optimization of resource utilization

The following functions permit **parts programs** to obtain current

- information about the assignment of resources for curve tables, table segments and polynomials

One result of the diagnostic functions is that resources still available can be used **dynamically** with the functions in 2.2.2 without necessarily having to increase the memory requirement. The description of the parameters in 2.2.2 also applies to the following functions.

**a) Curve tables**

- Determine total number of defined tables.
  The definition applies to all memory types (see also CTABNOMEM)
  **CTABNO()**
- Determine number of tables defined in SRAM or DRAM memory.
  **CTABNOMEM**(memType**)**
  If memType is not specified, the memory type set in
  MD 22905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  >= 0 Number of defined tables
  –2 Invalid memory type

- Determine number of curve tables still possible in memType.
  **CTABFNO(**memType**)**
  If memType is not specified, the memory type set in
  MD 220905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  >= 0 Number of possible tables
  –2 Invalid memory type

- Determine number of pth table in the optionally specified memory type
  **CTABID**(p, memType)
  If memType is not specified, the memory type in
  MD 22905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  Table number or
  alarm for invalid p or memType

- Query **blocking status**
  Table n
  **CTABISLOCK**(n)
  Result:
  > 0 Table is blocked
  Reason for blocking:
  1 by CTABLOCK()
  2 by active coupling
  3 by CTABLOCK() and active coupling
  = 0 Table is not blocked
  – 1 Table doe snot exist

- Check whether curve table **exists**
  **CTABEXISTS**(n)
  Result:
  1 Table exists
  0 Table does not exist

- Determine **memory type** of a curve table
  **CTABMEMTYP**(n)
  Result:
  0 Table in SRAM
  1 Table in DRAM
  –1 Table does not exist

- Determine whether table is defined as **periodic**
  **CTABPERIOD**(n)
  Result:
  0 Table is not periodic
  1 Table is periodic in the leading axis
  2 Table is periodic in the leading and following axes
  –1 Table does not exist

**b) Curve table segments**

- Determine number of curve table segments **used** in memory memType.
  **CTABSEG**(memType)
  If memType is not specified, the memory type set in
  MD 22905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  >= 0 Number of curve segments
  –2 Invalid memory type

- Determine number of curve table segments used for a particular curve table
  **CTABSEGID**(n)
  Result:
  >= 0 Number of curve segments
  –1 Curve table with number n does not exist.

- Determine number of**free** curve segments of memory type
  **CTABFSEG**(memType)
  If memType is not specified, the memory type set in
  MD 22905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  >= 0 Number of curve segments still free
  –2 Invalid memory type

- Determine **maximum** number of curve segments of memory type
  **CTABMSEG**(memType)
  If memType is not specified, the memory type set in
  MD 22905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  >= 0 Maximum number of possible curve segments
  –2 Invalid memory type

**c) Polynomials**

- Determine number of polynomials **used** for memory type
  **CTABPOL**(memType)
  If memType is not specified, the memory type set in
  MD 22905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  >= 0 Number of polynomials used in memory type
  –2 Invalid memory type
- Determine number of curve polynomials used by a curve table
  **CTABPOLID**(n)
  Result:
  >=0 Number of curve polynomials used
  –1 Curve table with number n does not exist
- Determine number of polynomials **free** for memory type
  **CTABFPOL**(memType)
  If memType is not specified, the memory type set in
  MD 220905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  >= 0 Number of curve polynomials still free
  –2 invalid memory type
- Determine number **maximum** possible number of polynomials for memory type **CTABMPOL**(memType)
  If memType is not specified, the memory type set in
  MD 220905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  >= 0 Maximum possible number of curve polynomials
  –2 Invalid memory type

# 2.3 Master value coupling

## 2.3.1 General functionality

**Introduction**

Master value couplings are divided into axis and path master value couplings. In both cases, the axis and path positions are defined by the control on the basis of master values (e.g. positions of another axis).

### Axis master value coupling

**Function**

The axis master value coupling is an axis coupling with motion rules that are represented internally as a one-dimensional real function, a curve table (see Subsection 2.2.1).

**Master value object**

Is the input variable for the curve table.

The following can be defined as the position of the master value object:

- The axis actual position (actual value measured by encoder) or
- the setpoint (calculated by the interpolator).
- If the leading axis is interpolated by the same NCU, the setpoint value coupling produces a better following response than is possible for actual-value coupling (in the same IPO cycle).
- If the leading axis is not interpolated by the same NCU, the interpolator that is implemented in the NCU for this particular leading axis can be used for master value simulation. To do this set MD 30132: IS_VIRTUAL_AX=1.

---

**Note**

Setpoint coupling is the default for the master value object.

---

**Virtual leading axis/simulated master value**

When switching over to master value coupling, the simulation can be programmed with the last actual value read, whereas the path of the actual value is generally outside the control of the NCU.

If, for master value simulation, i.e. depending on MD 30132: IS_VIRTUAL_AX=1, the master value object is switched from actual-value coupling to setpoint value coupling and a traversing command issued for the leading axis in the same interpolator cycle, the interpolator for the axis is initialized by the NCK such that the master value produces a constant path in the first derivation.

- Separation of IPO and Servo
- Actual values of the axis are recorded
- Setpoint values are produced by IPO but not passed on to the servo motor

MD 30130: CTRLOUT_TYPE[n] defines the setpoint value output of the axis type.

0: Simulated axis
1: Standard real axis
2: Stepper motor FM-NC
3: Not assigned
4: Path setpoint coupling (virtual axis) for external axes

**Offset and scaling**

The setpoint value for the following axis can be shifted and scaled.

The setting data below are used for this:

- SD 43102: LEAD_OFFSET_IN_POS
- SD 43104: LEAD_SCALE_IN_POS
- SD 43106: LEAD_OFFSET_OUT_POS
- SD 43108: LEAD_SCALE_OUT_POS

If (x) is a periodic curve table and this is interpreted as vibration, the offset and scaling can also be interpreted as follows:

- SD 43102: LEAD_OFFSET_IN_POS[Y]
  offsets the phase of the vibration
- SD 43104: LEAD_SCALE_IN_POS[Y]

- SD 43106: LEAD_SCALE_OUT_POS[Y]
  affects the amplitude
- SD 43108: LEAD_OFFSET_OUT_POS[Y]
  offsets the center point of the vibration

If the coupling is activated and synchronous, the new set position is approached as soon as values are written to these setting data.

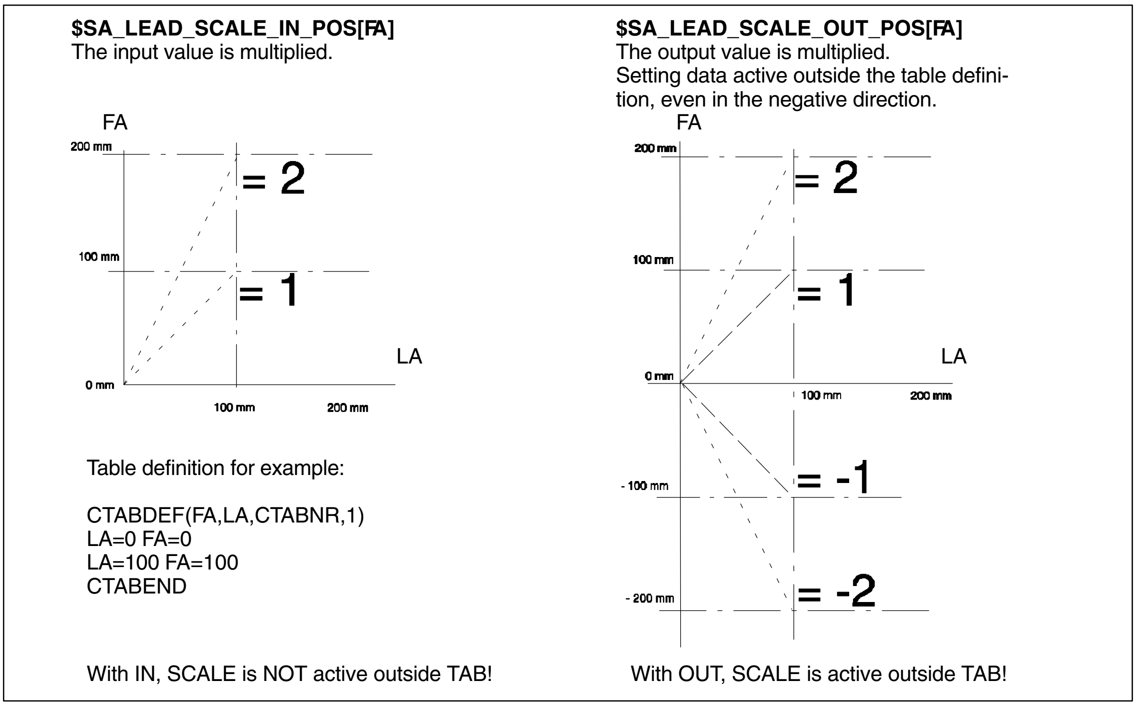


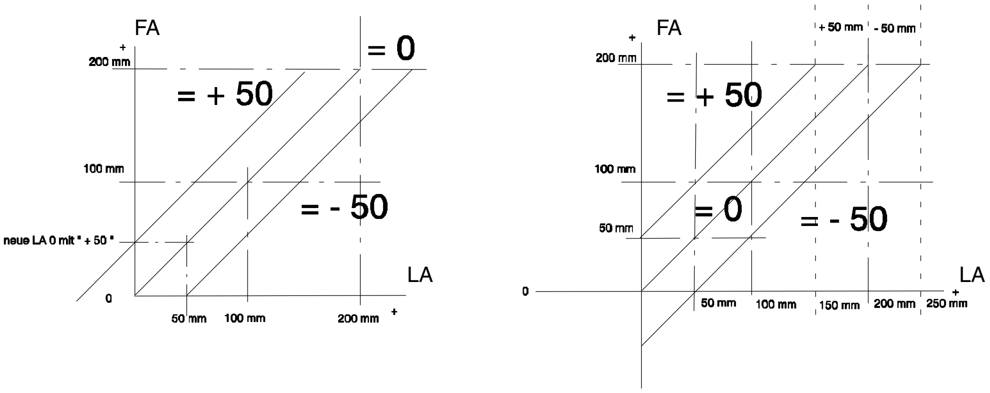

Fig. 2-2 Master value coupling offset and scaling (multiplied)

**$SA_LEAD_OFFSET_IN_POS[FA]**

The calculated position of the master value is offset by the increment specified.

**$SA_LEAD_OFFSET_OUT_POS[FA]**

The calculated position of the following axis is offset by the specified increments. If the TAB is offset beyond the FA LEADON position, the link cannot be activated.
Setting data active outside the table definition, even in the negative direction.




With IN, OFFSET is not active outside TAB!

With OUT, OFFSET active outside TAB!

Fig. 2-3 Master value coupling offset and scaling (with increment offset)

### Reaction to stop

All following axes coupled to the master value react to channel stop and MODE GROUP stop.

Following axes coupled to the master value react to a stop due to end of program (M30, M02) if they have not been activated by static synchronous actions (IDS=...). (Note MD 20110: RESET_MODE_MASK: MD 20112: START_MODE_MASK)

Leading axis and following axis must always interpolate in the same channel. A following axis in a different channel cannot be coupled (axis exchange).

START and mode change enable a following axis in the master value coupling that has been stopped.

RESET also enables a stopped following axis in the master value coupling. If enabling by RESET is not desired, or if it is dangerous (e.g. because the following axis is coupled to an external master value not controlled by the NC), MD 20110: RESET_MODE_MASK should be programmed so that the master value couplings are deactivated on RESET (2001H, i.e. set bit 13 to 1).

### Interface to axis exchange

A following axis that is coupled to a leading axis receives its setpoint values from the tables of curves. **Overlaid programming of this axis is not possible in the parts program.** Thus, the following axis coupled by the master value is removed from the channel in the same way as for axis exchange. This is carried out automatically when the coupling is activated in the parts program.

If the coupling is to be activated with synchronous actions, it must be prepared beforehand with **RELEASE**, otherwise the alarm 16777 “Channel %1 block %2 master value coupling: Following axis %3 no longer available for leading axis %4” is issued.

After a master value coupling has been deactivated, the former following axis can be programmed again in the parts program.

**Spindles in the master value coupling**

A spindle can only be used as the master-value-coupled following axis if it has been switched to axis mode beforehand. The machine data parameter block of the axis drive then applies.

Example (activate from synchronous action):

```
SPOS=0
B=IC(0xb)          ; switch spindle to axis mode
RELEASE(Y)         ; enable for synchronous action
ID=1 WHEN ($AA_IM[X]<–50) DO LEADON(B,X,2)
                   ;Y is coupled to X using curve table no. 2
```

## 2.3.2 Programming a master value coupling

**Definition and activation**

A master value coupling is defined and activated simultaneously with the modal command for:

- Axis master value coupling

  **LEADON**(FA, LA, CTABn)

  – FA=following axis, as GEO axis name, channel or machine axis name (X,Y,Z,...).

  – LA=leading axis, as GEO axis name, channel or machine axis name (X,Y,Z,...). Software axis possible (MD 30130 : CTRLOUT_TYPE=0)

  – CTABn=number of the curve table 1 to 999

  Example: LEADON(Y,X,1) FA=Y, LA=X, curve number=1

  No reference point is necessary to activate the coupling (KOP). A defined FA cannot be traversed in JOG mode (not even if the interface signal synchronism FINE/COARSE is not yet present). An activated coupling must first be deactivated with LEADOF before it can be activated again with LEADON, or note MD 20112: START_MODE_MASK/MD 20110: RESET_MODE_MASK

  Error message: 16792 “Too many couplings for axis/spindle”

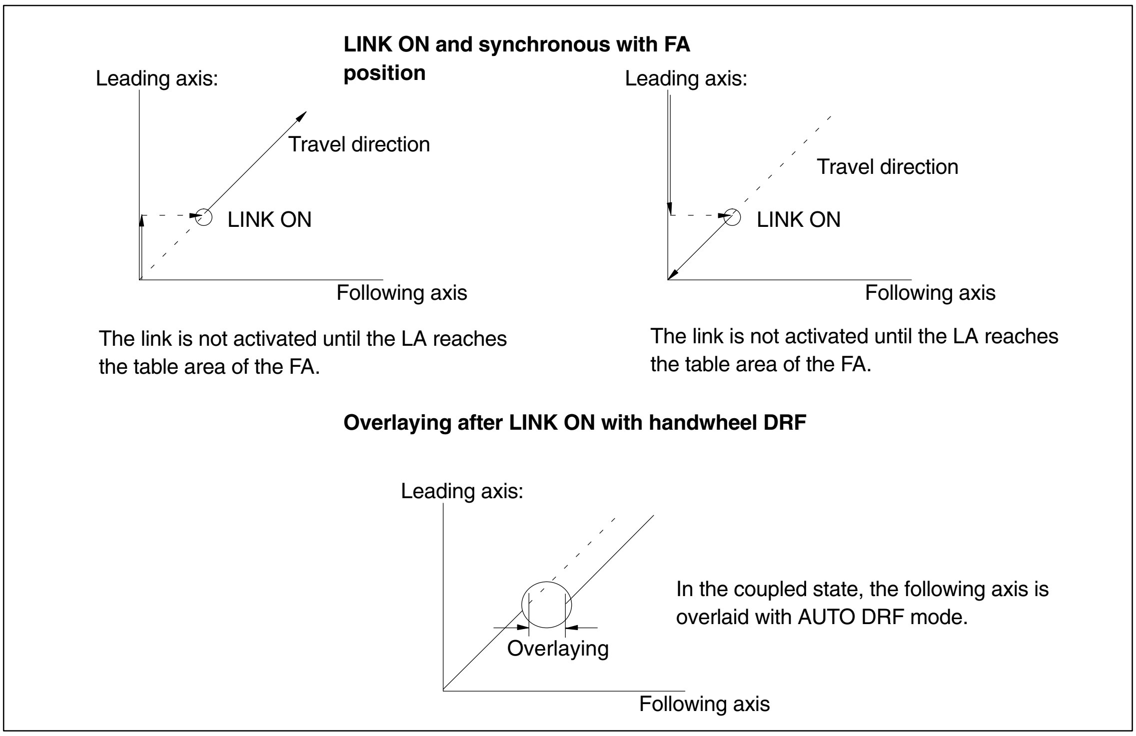



Fig. 2-4 Activating the master value coupling

**Deactivation**

A master value coupling is deactivated with the model command for:

- Axis master value coupling

  **LEADOF**(FA, LA)

  – FA=following axis, as GEO axis name, channel or machine axis name (X,Y,Z,...).

  – LA=leading axis, as GEO axis name, channel or machine axis name (X,Y,Z,...). Software axis possible (MD 30130 : CTRLOUT_TYPE=0)

  Example: LEADOF(Y,X) FA=Y, LA=X

  When the axis master value coupling is deactivated the following axis becomes the command axis and a stop command is generated implicitly for the following axis. The stop command can be overwritten by another command with a synchronous action.

---

**Note**

Activating and deactivating the axis master value coupling with LEADON, LEADOF is permissible both in the parts program and in synchronous actions.

**References:** /FB/, S5, “Synchronous actions”

---

**Type of coupling**

Setting data SD 43100: LEAD_TYPE[LA] specifies the coupling type. Switchover between actual and setpoint value coupling is possible at any time, preferably in the idle phase.

LA: Leading axis as GEO axis name, channel axis name or machine axis name (X,Y,Z,...)

0: Actual-value coupling (this type of coupling must be used for external Leading axis)

1: Setpoint value coupling (default)

2: Simulated master value (note that the virtual axis is not evaluated for FA)

**Readable system variables of the master value**

The system variables of the master value can be read from the parts program and from synchronous actions:

- $AA_LEAD_V[ax] ;Velocity of the leading axis
- $AA_LEAD_P[ax] ;Position of the leading axis
- $AA_LEAD_P_TURN ;Master value position: the part that is deducted in the modulo reaction. The actual (non modulo-reduced) position of the master value is $AA_LEAD_P_TURN+$AA_LEAD_P.

**Readable and writable master value variables**

The velocities and positions of simulated master values (when $SA_LEAD_TYPE[ax]=2) can be written in and read from the parts program and synchronous actions:

- $AA_LEAD_SV[ax] ;simulated master value velocity per IPO cycle
- $AA_LEAD_SP[ax] ;simulated position in MCS

**Detecting synchronism**

System variable $AA_SYNC[ax] that can be read from the parts program and synchronous action indicates whether and how following axis FA is synchronized:

0: Not synchronized

1: Synchronism coarse (according to MD 37200: COUPLE_POS_TOL_COARSE)

3: Synchronism fine (according to MD 37210: COUPLE_POS_TOL_FINE)

The information from system variable $AA_SYNC[ax] corresponds to the assigned VDI signals:

NST “Synchronism fine” DB 31, ... DBX98.0 und

NST “Synchronism coarse” DB 31, ... DBX98.1

---

**Note**

If the following axis is not enabled for travel it is stopped and is no longer synchronous.

---

### 2.3.3 Response in AUTOMATIC, MDA, JOG

**Effectiveness**

A master value coupling is active depending on the settings in the parts program and in the machine data MD 20110: RESET_MODE_MASK and MD 20112: START_MODE_MASK.

**Manual operation**

When an axis master value coupling is activated, the traversing movement of the leading axis (e.g. in rapid traverse or incremental dimension INC1 ... INC10000) causes a following axis to traverse as a function of the definition in the curve table.

**Referencing**

A following axis coupled with a leading axis is to be referenced prior to activation of the coupling. The following axis cannot be referenced when the coupling is activated.

**Deletion of distance-to-go**

When deletion of distance-to-go is performed for a leading axis, all axes in the associated, activated master value coupling are shut down.

**Basic setting after power ON**

No master value couplings are active after power ON (options with ASUB).

**Behavior after NC start/RESET**

The behavior after NC start/RESET is determined by the setting in MD 20110: RESET_MODE_MASK (bit 13) and MD 20112: START_MODE_MASK (bit 13):

- MD 20110: RESET_MODE_MASK=2001H && MD 20112: START_MODE_MASK=0H
  ==> Master value coupling remains valid after RESET and START
- MD 20110: RESET_MODE_MASK=2001H && MD 20112: START_MODE_MASK=2000H
  ==> Master value coupling remains valid after RESET and is cancelled on START.
  Master value coupling activated per IDS=... remains valid however.
- MD 20110: RESET_MODE_MASK=1H
  ==> Master value coupling is cancelled with RESET, independently of MD 20112: START_MODE_MASK
  Master value coupling activated per IDS=... is deactivated only via operator panel front reset and remains valid after program end/reset (M30, M02).
- MD 20110: RESET_MODE_MASK=0H
  ==> Master value coupling remains valid beyond RESET and is invalidated on START, independently of MD 20112: START_MODE_MASK.
  Master value coupling activated per IDS=... remains valid however.

**References:** /FB/, K2, “Coordinate System, Axis Types, Axis Configurations, ...”

**Activating, deactivating**

Master value couplings activated per synchronous action (IDS=...) are

- not deactivated during program start, irrespective of the value for MD 20110: RESET_MODE_MASK and MD 20112: START_MODE_MASK.
- not deactivated during program end reset (M30, M02), irrespective of the value for MD 20110: RESET_MODE_MASK.

### 2.3.4 Effectiveness of PLC interface signals

**Leading axis**

When a master value coupling is active, the interface signals (IS) of the leading axis are applied to the appropriate following axis via the axis coupling, i.e.

- a feed control action of the leading axis is applied via the master value coupling to effect an appropriate feed control action in the following axis.
- Shutdown of the leading axis as the result of an IS (e.g. axis-specific feed stop, axis inhibit, controller enable, etc.) causes shutdown of the appropriate following axis.

**Position measuring system 1/2 (DB31, ... DBX1.5/1.6)**

Switchover of the position measuring system for the leading and following axes is not inhibited for an active coupled axis grouping. The coupling is not invalidated on measuring system switchover.

**Recommendation:** Switch the measuring system over when the coupling is deactivated.

### 2.3.5 Special characteristics of axis master value coupling function

**Dynamic response of control system**

Depending on the application in question, it may be advisable to match the position controller parameter settings (e.g. servo gain factor) of the leading axis and coupled axis in an axis grouping. It may be necessary to activate other parameter sets for the following axis. The dynamics of the following axis should be the same or better than those of the leading axis.

**Status of coupling**

See Subsection 2.1.5.

**Actual-value display**

The display of the actual value is updated for all axes of an axis grouping (only real axes) coupled via a master value.

**Interpolation**

When the movement defined in the curve table is interpolated, an axis position and axis speed are calculated for a master value and its speed.

**Archiving**

The curve tables generated by the definition of motion sequences are stored in the battery-backed memory.

The curve tables are not lost when the control is switched off.

These functions have no effect on cyclic machines because they are performed without operator actions. Nor does it make sense to perform automatic (re-)positioning via the NC with external master values.

# 2.4 Electronic gear EG (SW 5 and higher)

**Function**

The “Electronic gear” function can be used to interpolate the motion of a **following axis FA** as a function of up to five **leading axes LA**. The relationship between each leading axis and the following axis is defined by a coupling factor. The following axis motion components derived in this manner from the individual leading axis motion components have an additive effect.

$FA_{set} = SynPosFA + (LA_1 - SynPosLA_1)*KF_1 + ... + (LA_5 - SynPosLA_5)*KF_5$

where:

| | |
|---|---|
| SynPosFA, $SynPosLA_i$ | From call EGONSYN (see below) |
| $FA_{set}$ | Part setpoint of following axis |
| $LA_i$ | Setpoint or actual value of leading axis i (dependent on coupling type (see below) |
| $KF_i$ | Coupling factor of leading axis i (see below) |

All paths are referred to the basic coordinate system **BCS**.

When an EG axis grouping is activated, it is possible to synchronize the leading axes and following axis in relation to a defined starting position.

An EG axis grouping can be

- defined,
- activated,
- deactivated,
- deleted

from the parts program.

**Expansions from SW 6**

As of Software Release 6, the influence of each of the 5 leading axes can be specified through a **curve table** (see Section 2.2) as an alternative to a gear ratio (KF=numerator/denominator).

It is thus possible for each curve (except for the special case of a straight line) for the leading axis to influence the following axis in a **non-linear** manner. The function can only be used with EGONSYN.

**As of SW 6.3,**
the function EG wit curve tables can be activated with EGON.

**As of SW 6.4**
The function EGONSYNE is available for approaching the synchronous position of the following axis with a specified approach mode.

For special applications, it can be expedient to configure the position controller as a **PI controller**.

---

**Caution**

Knowledge of the control technology and measurements with servo trace are an absolute prerequisite for using this function.

---

**References:** See /IAD/. Installation and Start-Up Guide
/FB/, G2, Vecolities, Setpoint/Actual Value Systems, Control

**Coupling type**

The following axis motion can be derived from either of the following:

- Setpoints of leading axes
- Actual values of leading axes

The reference is set in the definition call for the EG axis grouping:

EGDEF (For more details, see Subsection 2.4.2).

**Coupling factor**

The coupling factor must be programmed for each leading axis in the grouping. It is defined on the basis of numerators/denominators.
The coupling factor values “numerator” and “denominator” are specified for each leading axis in the activation calls

EGON
EGONSYN
EGONSYNE

(For more details, see Subsection 2.4.3).

**Number of EG axis groupings**

Several EG axis groupings can be defined at the same time. The maximum possible number of EG axis groupings is set in

MD 11660: NUM_EG.

The maximum permissible number of EG axis groupings is 31.

---

**Note**

The option must be enabled.

---

**EG cascading SW 6**

Electronic gears can be switched in series. This means: the following axis of an EG can take the place of a leading axis of a series-connected EG. Chapter 6 contains a detailed example of this.

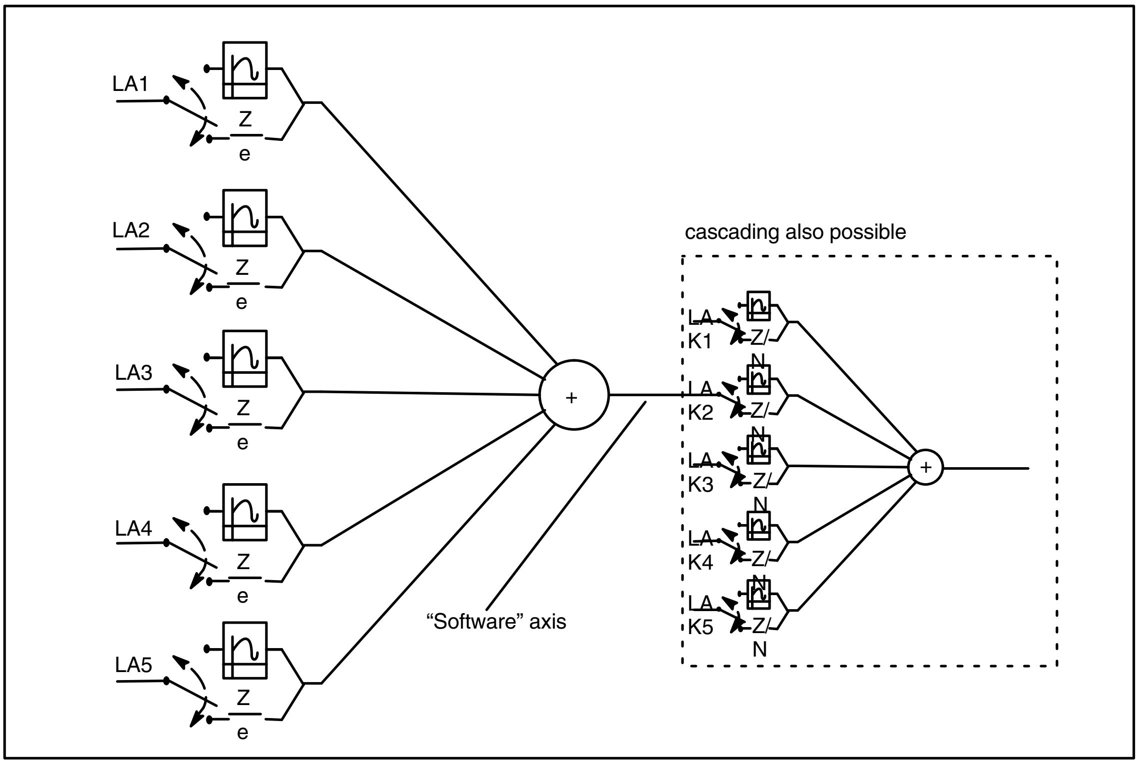



Fig. 2-5 Block diagram of an electronic gear

**Synchronized positions**

To start up the EG axis grouping, you can first request an approach to defined positions for the following axis.
Synchronized positions are programmed with

EGONSYN (see below for details).

EGONSYNE (extended EGONSYN call).

**Synchronization**

If a gear is started with EGON(), EGONSYN() or EGONSYNE() see below, the actual position of the following axis is only identical to the setpoint position defined by the rule of motionn of the gear specified by the positions of the leading axes at this time if the parts program developer makes sure that it is. The control then uses the motion of the following axis to ensure that the setpoint and actual positions of the following axes correspond as quickly as possible it the leading axes are moved further. This procedure is called synchronization. After synchronization of the following axis, the term **synchronous** gearing is used.

**Activation response**

An electronic gear can be activated in two different ways:

1. On the basis of the axis positions that have been reached up to now in the course of processing the command to activate the EG axis group is issued without specifying the synchronizing positions for each individual axis. EGON see 2.4.3.
2. The command to activate the EG axis group specifies the synchronized positions for each axis. From the point in time when these positions are reached, the EG should be synchronized. EGONSYN see Subsection 2.4.3.
3. The command to activate the EG axis group specifies the synchronized positions and approach mode for each axis. From the point in time when these positions are reached, the EG should be synchronized. EGONSYNE see 2.4.3.

**Synchronization with EGON**

With EGON(), no specifications are made for the positions at which the following axis is to be synchronized. The control activates the EG and issues the signal "Synchronized position reached".

**Synchronization for EGONSYN**

With EGONSYN(), the positions of the leading axes and the synchronizing position for the following axis is specified by the command.

The control then traverses the following axis with just the right acceleration and speed to the specified synchronization position so that the following axis is in position with the leading axes at its synchronization position.

**Synchronizationfor EGONSYNE**

With EGONSYNE(), the positions of the leading axes and the synchronizing position for the following axis is specified by the command.

The control moves the following axis to the synchronized position according to the program approach mode.

**Synchronism monitoring**

The synchronism of the gear is monitored in each interpolator cycle on the basis of the actual values of the slave and leading axes. For this purpose, the actual values of the axes are computed according to the rule of motion of the coupling. The **deviation in synchronism** is the difference between the actual value of the following axis and the value calculated from the leading axis actual values according to the rule of motion. The deviation in synchronism called be polled from the parts program. See below.

**Changes in the deviation in synchronism**

The mass inertia of the axis systems during acceleration can cause dynamic fluctuations in the deviation in synchronism. The deviation in synchronism is checked continuously and the tolerance values in the machine data used to produce interface signals.

The deviation in synchronism is compared with machine data
MD 37200: COUPLE_POS_TOL_COARSE,
MD 37210: COUPLE_POS_TOL_FINE.

Depending on the result of this comparison, the following signals are set:
IS "Synchronous run, fine" DB 31, ... DBX 98.0
IS "Synchronous run, coarse" DB 31, ... DBX 98.1

**Difference > ..TOL_COARSE**

As long as the difference in the synchronous run is greater than MD 37200: COUPLE_POS_TOL_COARSE, the gear is not synchronized and neither the IS "Synchronization coarse" DB 31, ... DBX 98.1 nor IS "Synchronization fine" DB 31, ... DBX 98.0 are present at the interface. Instead, the signal IS "Synchronization in progress" DB 31, ... DBX 99.4 is displayed.

**Difference < ..TOL_COARSE**

If the difference in the synchronous run is less than MD 37200: COUPLE_POS_TOL_COARSE, IS "Synchronous run, coarse" " DB 31, ... DBX 98.1 is present at the interface an IS "Synchronization in progress" DB 31, ... DBX 99.4 is deleted.

**Difference < ..TOL_FINE**

If the difference in the synchronous run is less than MD 37210: COUPLE_POS_TOL_FINE, IS "Synchronous run, fine" DB 31, ... DBX 98.0 is present at the interface.

**Deviation in synchronism for EG cascades SW 6**

Deviation in synchronism for EG cascades is the deviation of the actual position of the following axis from setpoint position that results fro the rule of motion for the real axes involved.

Example:

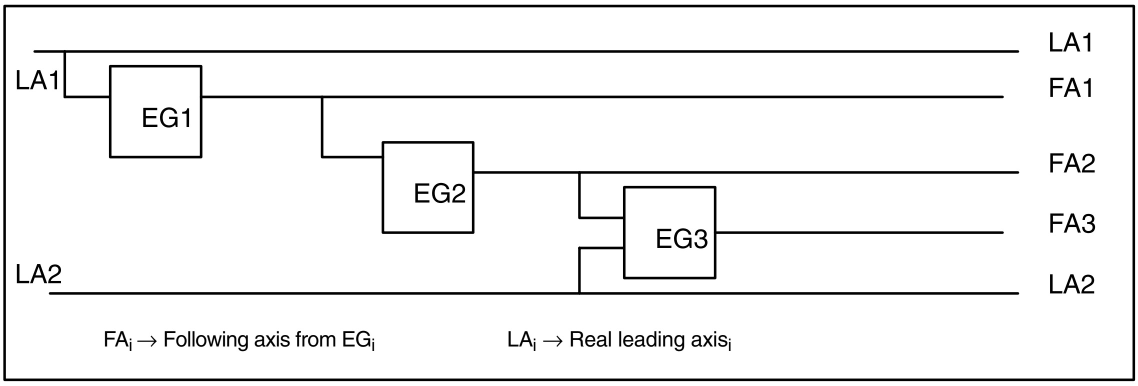



Fig. 2-6 Three-level EG cascade

The deviation in synchronism of following axis FA3 in the example below is determined by the by the value of following axis $FA3_{Act}$ and the value of leading axis $FA2_{Act}$ and $LA2_{Act}$, but not by $LA1_{Act}$ and $FA1_{Act}$ according to the definition given.

If FA2 is not a real axis, the actual value $FA2_{Act}$ is not available. In this case, the **setpoint** of the axis derived solely from the leading axis value $FA1_{Act}$ must be used instead of the actual value of the setpoint of the axis.

**Other signals**

If an EGON(), EGONSYN() or EGONSYNE() block is encountered in the main run, the signal "Coupling active is set for the following axis. If the following axis is only overlaid, the signals "Coupling active" and "Axis override" are set. If EGON(), EGONSYN() or EGONSYNE() is active and the following axis is also overlaid, the signals "Coupling active" and "Axis override" are also set.

IS “Following spindle active” DB31, ... DBX 99.1 Coupling active,

IS “Overlaid movement” DB31, ... DBX98.4 Axis is overlaid,

IS “Enable overlay for following axis” DB31, ... DBX26.4

In the case of the commands EGON() and EGONSYNE(), the signal “Enable following axis override” must be present for the gear to synchronize to the specified synchronization position for the following axis. If it is not present, alarm 16771 “Override movement not enabled” is issued. If the signal is present, the following axis travels to the synchronized position with the calculated acceleration and at the speed set for the approach mode.

**Further monitoring signals**

With machine data MD 37550: EG_VEL_WARNING,
it is possible to specify a % block of the velocities and accelerations in
MD 32000: MAX_AX_VELO
MD 32300: MAX_AX_ACCEL

in relation to the following axis; the interface signals:
NST “Velocity warning threshold” DB 31, ... DBX 98.5
NST “Acceleration warning threshold” DB 31, ... DBX 98.6
are produced with this value. The monitoring signals can be used as trigger criteria for emergency retraction. See 2.6.5.

A maximum drift value for automatic compensation is activated in machine data MD 37560: EG_ACC_TOL see Chapter 4
a % block can be defined relative to
MD 32300: MA_MAX_AX_ACCEL
of the following axis; if the value is exceeded, the signal
NST “Axis accelerated” DB 31, ... DBX 99.3
is produced.

**Scanning the synchronism deviation value**

The result of the synchronism deviation calculation can be read as an amount in the parts program with system variable $VA_EG_SYNCDIFF. The relevant value with sign is available from SW 6.4 in the system variables $VA_EG_SYNCDIFF_S. It means:

– Negative value (for leading axis and following axis in positive direction of motion):
  The following axis stays behind the computed setpoint position.

– Positive value (for leading axis in positive direction of motion):
  The following axis is in front of its computed setpoint position (overshoot).

The amount of the synchronization difference with sign corresponds to the system variables without sign from $VA_EG_SYNCDIFF.

$VA_EG_SYNCDIFF[ax] = ABS($VA_EG_SYNCDIFF_S[ax])

**Block change mode**

When an EG axis grouping is activated, it is possible to specify the conditions under which a parts program block change is to be executed:

The conditions are specified by means of string parameters with the following meanings:
“NOC” Change block immediately
“FINE” Change block in response to “Fine synchronism” signal
“COARSE” Change block in response to “Coarse synchronism” signal
“IPOSTOP” Change block in response to “setpoint-related synchronism” signal

---

**Note**

When programmed in activation calls EGON, EGONSYN, EGONSYNE, each of the above strings can be abbreviated to the first two characters.

---

If no block change has been defined for the EG axis group and none is currently specified, “FINE” applies.

## 2.4.1 Overview of EG features in SW 6 (summary)

**EG**

An EG has:

a) max. 5 leading axes

b) 1 following axis

c) max. 5 associated curve tables or

d) max. 5 associated coupling factors (Z/N) or

e) combination of curve tables and coupling factors for max. 5 leading axes

**Following axis**

A following axis can:

a) identify the EG uniquely

b) be the leading axis of a different EG (cascading)

c) not simultaneously be the leading axis of the same EG (no feedback)

d) not be a command axis

**Leading axis**

A leading axis can:

a) be used once in the same EG

b) can be used as leading axis in several EGs

c) be a PLC axis

d) be a command axis

**Leading and following axis**

The following are permissible as leading and following axes:

| real | simulated |
|---|---|
| Linear axis | |
| rotary axis | |
| Modulo-corrected rotary axis | |

**Type of coupling**

For each leading axis, the EG may refer to:

a) the actual value or

b) the setpoint

**Reference system**

The calculations are made in the basic coordinate system BCS.

**Synchronized actions**

Synchronized actions (see **Reference:** /FBSY/) are not supported.

**Block search**

EG commands are ignored in the case of block search.

**Mode change**

In the case of a mode change,

a) the EG status and

b) the EG configuration are retained

**RESET**

For RESET:

a) the EG status and

b) the EG configuration are retained

**End of part program**

On end of a parts program,

a) the EG status and

b) the EG configuration are retained

**Warm start and cold start**

In the case of a warm start per MMC/HMI operation and cold start (power OFF/ power ON)

a) the EG status is **not** and

b) the EG configuration are **not** retained.

**Violated synchronism conditions**

If the synchronism conditions are violated, all axes are stopped. I(n this case, their positions checked by the control up to the stop. Extended stop and retract (ESR) may be active in this situation see Section 2.6.

**Power-up conditions of EG**

The EG may be powered up:

a) at the current axis positions (EGON) or

b) at the synchronized positions to be specified (EGONSYN).

c) at synchronous positions to be specified with details of an approach mode (EGONSYNE)

**Block change behavior**

In the EG activation commands (EGON, EGONSYN, EGONSYNE), you can specify for which condition (with respect to synchronism) the next block of the parts program is to be processed. Options:

a) NOC No condition.

b) FINE Sum of the difference between the setpoint and actual positions of all axes is less than MD 37210: COUPLE_POS_TOL_FINE

c) COARSE Sum of the difference between the setpoint and actual positions of al axes is less than MD 37200: COUPLE_POS_TOL_COARSE

d) IPOSTOP When the specified end positions of the axes is reached.

### 2.4.2 Defining an EG axis grouping

**Note**

The following definition commands and switching instruction commands relating to the electronic gear must be programmed
**on their own in a block** in a parts program.
All commands for the electronic gear, except for activation commands
– EGON
– EGONSYN
– EGONSYNE trigger a **preprocessing stop**.

**Definition and activation**

The definition described below and the activation are separate processes. An activation is not possible unless it has been defined previously.

**Definition of an EG axis grouping**

An EG axis grouping is defined through the input of the following axis and at least one leading axis (up to five masters are allowed), each with the relevant coupling type:

**EGDEF**(following axis, leading axis1, coupling type1, leading axis2, coupling type 2,...)

The coupling type does not need to be the same for all leading axes and must be programmed separately for each individual master.

Coupling type: Evaluate actual value of leading axis: 0
Evaluate setpoint of leading axis: 1

The coupling factors are preset to zero when the EG axis grouping is defined. As such, the grouping has no effect on the following axis until it is activated. (S. EGON, EGONSYN, EGONSYNE).

Preconditions for defining an EG axis grouping:
No existing axis coupling may already be defined for the following axis.
(If one does already exist, it must be deleted beforehand with EGDEL.)
EGDEF triggers a preprocessing stop with error message.

**EGDEF in SW 6**

Gear definition with EGDEF should also be used unaltered when one or more leading axes affect the following axis via a curve table in systems with SW 6 and higher.

### 2.4.3 Activating an EG axis grouping

**Without synchronization**

The EG axis grouping is switched on **selectively without synchronization** with:

**EGON**(FA, block change mode, LA1, Z1, N1, LA2 , Z2, N2,..LA5, Z5, N5.)

The coupling is activated immediately.

with:

| | |
|---|---|
| FA | Following axis |

According to the block change mode, the program advances to the next block:

| | |
|---|---|
| "NOC" | Block change takes place immediately |
| "FINE" | Block change takes place with "Synchronization fine" |
| "COARSE" | Block change takes place with "Synchronization coarse" |
| "IPOSTOP" | Block change takes place with synchronization on the setpoint side |
| $LA_i$ | Axis identifier of leading axis i |
| $Z_i$ | Numerator for coupling factor of leading axis i |
| $N_i$ | Denominator for coupling factor of leading axis i |

Only the leading axes previously specified with the EGDEF command may be programmed in the activation line. At least one following axis must be programmed.

The positions of the leading axes and following axis at the instant the grouping is switched on are stored as "Synchronized positions". The "Synchronized positions" can be read with the system variable $AA_EG_SYN.

**With synchronization**

The EG axis grouping is switched on **selectively with synchronization** with:

**1st EGONSYN**

**EGONSYN**(FA, block change mode, SynPosFA, $LA_i$, SynPos$LA_i$, Z_$LA_i$, N_$LA_i$)

with:

| | |
|---|---|
| FA | Following axis |

Block change mode:

| | |
|---|---|
| "NOC" | Block change takes place immediately |
| "FINE" | Block change takes place with "Synchronization fine" |
| "COARSE" | Block change takes place with "Synchronization coarse" |
| "IPOSTOP" | Block change takes place with synchronization on the setpoint side |
| SynPosFA | Synchronized position of following axis |
| $LA_i$: | Axis identifier of leading axis i |
| SynPosLAi: | Synchronized position of leading axis i |
| Z i: | Numerator for coupling factor of leading axis i |
| N i: | Denominator for coupling factor of leading axis i |

---

**Note**

The parameters indexed with i must be programmed for at least one leading axis, but for no more than five.

---

Only leading axes previously specified with the EGDEF command may be programmed in the activation line.

Through the programmed “Synchronized positions” for the following axis (SynPosFA) and for the leading axes (SynPosLA), positions are defined for which the axis grouping is interpreted as *synchronous*. If the electronic gear is not in the synchronized state when the grouping is switched on, the **following axis** traverses to its defined synchronized position.

If the axis grouping includes modulo axes, their position values are reduced in the modulo, thereby ensuring that they approach the fastest possible synchronized position (so-called *relative synchronization*: e.g. the next tooth gap after “centering”).

If IS “Enable following axis overlay” interface signal DB30(+axis number), ... DBX 26.4 is not specified for the following axis, the synchronized position is not approached. Instead the program is stopped at the EGONSYN block and the self-clearing alarm 16771 is issued until the above mentioned signal is set.

## 2nd EGONSYNE

**EGONSYNE**(FA, block change mode, SynPosFA, approach mode, $LA_i$, $SynPosLA_i$, $Z\_LA_i$, $N\_LA_i$)

with:

FA Following axis

Block change mode:

“NOC” Block change takes place immediately
“FINE” Block change takes place with “Synchronization fine”
“COARSE” Block change takes place with “Synchronization coarse”
“IPOSTOP” Block change takes place with synchronization on the setpoint side

SynPosFA Synchronized position of following axis

Approach mode:

“NTGT” NextToothGapTime optimized
next gap is approached time-optimized
(default value active if nothing specified)

“NTGP” NextToothGapPath optimized
next gap approach path-optimized

“ACN” AbsolutCoordinateNegative, Absolute Dimension,
Rotary axis traverses in negative direction of axis rotation

“ACP” AbsulteCoordinatePositiv, Absolute Dimension
Rotary axis traverses in positive direction of axis rotation

“DCT” DirectCoordinateTime optimized,Absolute Dimension,
Rotary axis traverses time-optimized in relation to programmed synchronous position

“DCP” DirectCoordinatePath optimized, Absolute Dimension
Rotary axis traverses path-optimized in relation to programmed synchronous position

$LA_i$: Axis identifier of leading axis i
SynPosLAi: Synchronized position of leading axis i
Z i: Numerator for coupling factor of leading axis i
N i: Denominator for coupling factor of leading axis i

---

**Note**

The parameters indexed with i must be programmed for at least one leading axis, but for no more than five.

---

The function is active only for modulo following axes that are coupled to modulo leading axes.

**Tooth gap**

The tooth gap is defined as 360 degrees * Zi / Ni

Example:

EGONSYNE(A, "FINE", FASysPos, "Traversing mode", B, 0, 2, 10)

Tooth gap: 360*2/10 = 72 (degrees)

**Approach response with FA at standstill**

In this case, the traversing modes time-optimized and path-optimized are identical.

The table below shows the target positions and traversed paths with direction marker (in brackets) for the particular approach modes:

| Programmed synchronous position FaSysPos | Position of the following axis before EGONSYNE | Traversing mode NTGT/ NTGP | Traversing mode DCT/ DCP | Traversing mode ACP | Traversing mode ACN |
|---|---|---|---|---|---|
| 110 | 150 | 182 (+32) | 110 (–40) | 110 (+320) | 110 (–40) |
| 110 | 350 | 326 (–24) | 110 (+120) | 110 (+120) | 110 (–240) |
| 130 | 0 | 346 (–14) | 130 (+130) | 130 (+130) | 130 (–230) |
| 130 | 30 | 58 (+28) | 130 (+100) | 130 (+100) | 130 (–260) |
| 130 | 190 | 202 (+12) | 130 (–60) | 130 (+300) | 130 (–60) |
| 190 | 0 | 334 (–26) | 190 (–170) | 190 (+190) | 190 (–170) |
| 230 | 0 | 14 (+14) | 230 (–130) | 230 (+230) | 230 (–130) |

**Approach response for moving FA**

The following axis moves at almost maximum speed in the positive direction when the coupling is activated by EGONSYNE. The programmed synchronous position of the following axis is 110, the current position 150. This produces the two alternative synchronous positions 110 and 182 (see table above).

In the case of traversing mode NTGP (path-optimized), synchronous position 182 is selected independent of the current velocity. This has the shortest distance from the current position of the following axis. Traversing mode NTGT (time-optimized) considers the current velocity of the following axis and produces a deceleration on account of the limit for the maximum axis velocity to reach synchronism in the shortest possible time (see Figure).

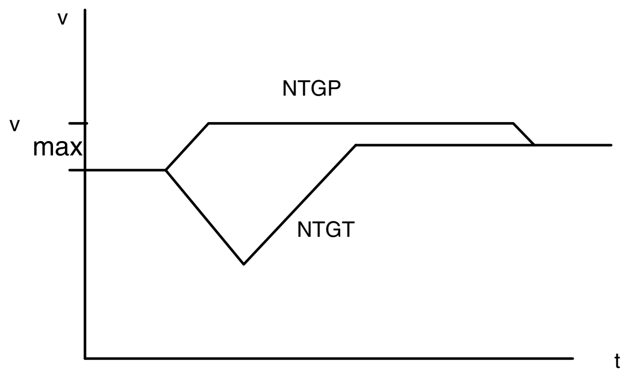



Fig. 2-7 Reaching the next tooth gap, FA path-optimized (top) vs. time-optimized (bottom)

### Sample notations

EGONSYNE(A, “FINE”, 110, “NTGT”, B, 0, 2, 10)
couple A to B, synchronous position A = 110, B = 0, coupling factor 2/10, approach mode = NTGT

EGONSYNE(A, “FINE”, 110, “DCT”, B, 0, 2, 10)
couple A to B, synchronous position A = 110, B = 0, coupling factor 2/10, approach mode = DCT

EGONSYNE(A, “FINE”, 110, “NTGT”, B, 0, 2, 10, Y, 15, 1, 3)
couple A to B, and Y, synchronous position A = 110, B = 0, Y = 15, coupling factor for B = 2/10, coupling factor for Y = 1/3, approach mode = NTGT

### Synchr. SW 6 and higher

The syntax specified above applies with the following different meanings. If a **curve table** is used for one of the :

$N_i$ the numerator of the coupling factor for linear coupling must be set to 0 (nominator 0 would be illegal for linear coupling) nominator zero tells the control that

$Z_i$ should be taken as the number of the curve table to be used. The curve table with the specified number must already be defined at power up in accordance with Section 2.2.

$LA_i$ The leading axis specified corresponds to the one specified Coupling via coupling factor (linear coupling).

## 2.4.4 Deactivating an EG axis grouping

**Variant 1**

The following methods can be used to deactivate an active EG axis grouping.

**EGOFS**(following axis)

The electronic gear is deactivated. The following axis is braked to a standstill.
This call triggers a preprocessing stop.

**Variant 2**

The following parameterization of the command makes it possible to **selectively** control the influence of individual leading axes on the motion of the following axis.

**EGOFS**(following axis, leading axis 1, ... leading axis 5)

---

**Note**

At least one following axis must be specified.

---

The influence of the specified leading axes on the slave is selectively inhibited.
This call triggers a preprocessing stop.

If the call still includes active leading axes, then the slave continues to operate under their influence. If the influence of all leading axes is excluded by this method, then the following axis is braked to a standstill.

**As of SW 6**, there is no axis movement if the command EGONSYN is deactivated selectively. If the following axis is to re-synchronize as in software versions up to 5.3, you must program EGONSYN.

---

**Note**

**Up to SW 5.3**, the following axis approaches the new synchronous position without the signal "Following axis overlay" DB31, ... DBX26.4 if the command EGONSYN is deactivated selectively.

---

**Variant 3**

**EGOFC**(following spindle)

The electronic gear is deactivated. The following spindle continues to traverse at the speed/velocity that applied at the instant of deactivation.
This call triggers a preprocessing stop.

---

**Note**

The call is available for following **spindles** only.
With EGOFC it is necessary to program a spindle identifier.

---

### 2.4.5 Deleting an EG axis grouping

An EG axis grouping must be switched off as described in Subsection 2.4.4 before its definition can be deleted.

**EGDEL**(following axis)

The defined coupling of the axis grouping is deleted.
It is then possible to define further new axis groupings with EGDEF until the maximum permissible number of simultaneously active axis groupings has been reached.

This call triggers a preprocessing stop.

### 2.4.6 Interaction between revolutional feedrate (G95) and electronic gear (SW 5.2 and higher)

The FPR() command can be used in SW 5 and higher to specify the following axis of an electronic gear as the axis which determines the revolutional feedrate. Please note the following with respect to this command:

- The feedrate is determined by the setpoint velocity of the following axis of the electronic gear.
- The setpoint velocity is calculated from the speeds of the leading spindles and modulo axes (which are not path axes) and from their associated coupling factors.
- Velocity components from other leading axes and overlaid motions of the following axis are not taken into account.

**References:** /V1/, Feeds

### 2.4.7 Response to power ON, operating mode change, RESET, block search

**No** coupling is active after power ON.

The status of active couplings is not affected by RESET or operating mode switchover.

During block searches, commands for switching, deleting and defining the electronic gear are not executed or collected, but skipped.

For further details of specials states, please refer to Subsection 2.4.1.

For an example of how to use the EG function for gear hobbing, please refer to Section 6.2.

The variant extended to include non-linear coupling via curve tables (SW 6) is described in Subsection 6.2.2

## 2.4.8 System variables for electronic gear

**Application**

The following system variables can be used in the parts program to scan the current states of an EG axis grouping and initiate appropriate reactions if necessary:

Table 2-1 System variables, R means: Read access possible

| Name | Type | Access | | Preproces. stop | | Meaning, value | Cond. Index |
|---|---|---|---|---|---|---|---|
| | | Parts prog. | Sync act. | Parts prog. | Sync act. | | |
| **$AA_EG_ TYPE[a,b]** (from SW 5.2) | INT | R | | R | | Type of coupling:<br>0: Actual value coupling<br>1: Setpoint coupling | Axis identifier<br>a: Following axis<br>b: Leading axis: |
| **$AA_EG_ NUMERA[a,b]** (from SW 5.2)<br>———<br>**(SW 6 and higher)** | REAL | R | | R | | Numerator of coupl. factor KF<br>KF = numerator/denominator<br>Default: 0<br>——<br>Number of curve table when $AA_EG_DENOM[a,b] is 0. | Axis identifier<br>a: Following axis<br>b: Leading axis: |
| **$AA_EG_ DENOM[a,b]** (from SW 5.2)<br>——<br>**(SW 6 and higher)** | REAL | R | | R | | Denominator of coupl. fact. KF<br>KF = numerator/denominator<br>Default: 1<br>Denominator must be positive.<br>——<br>Denominator is 0 if, instead of the numerator $AA_EG_NUMERA[a,b], the number of a curve table is specified. | Axis identifier<br>a: Following axis<br>b: Leading axis: |
| **$AA_EG_ SYN[a,b]** (from SW 5.2) | REAL | R | | R | | Synchronized position for specified leading axis<br>Default: 0 | Axis identifier<br>a: Following axis<br>b: Leading axis: |
| **$AA_EG_ SYNFA[a]** (from SW 5.2) | REAL | R | | R | | Synchronized position for specified following axis<br>Default: 0 | Axis identifier<br>a: Following axis |
| **$AA_EG_BC[a]** | STRING | R | | R | | Block change criterion for EG activation calls: EGON, EGONSYN:<br>"NOC" Immediate<br>"FINE" Synchronism fine<br>"COARSE" Synchronism coarse<br>"IPOSTOP" Setpoint-related synchronism | Axis identifier<br>a: Following axis |
| **$AA_EG_ NUM_LA[a]** | INT | R | | R | | Number of leading axes defined with EGDEF. 0 if no axis has been defined as a following axis with EGDEF. | Axis identifier<br>a: Following axis |
| **$AA_EG_ AX[n,a]** | AXIS | R | | R | | Axis identifier of leading axis whose index n has been specified. | Axis identifier<br>n: Index of leading axis in EG grouping 0 ... 4<br>a: Following axis |

Table 2-1 System variables, R means: Read access possible

| Name | Type | Access | | Preproces. stop | | Meaning, value | Cond. Index |
|---|---|---|---|---|---|---|---|
| | | Parts prog. | Sync act. | Parts prog. | Sync act. | | |
| **$AA_EG_ ACTIVE[a,b]** (from SW 5.2) | BOOL | R | | R | | Determine the operational state of a leading axis:<br>0: Deactivated<br>1: Activated | Axis identifier<br>a: Following axis<br>b: Leading axis: |
| **$VA_EG_ SYNCDIFF[a]** | REAL | R | R | R | | Actual value of synchronism deviation. The comparison with MD $MA_COUPLE_POS_TOL _COARSE and _FINE supplies interface signals. | Axis identifier<br>a: Following axis |

# 2.5 Dynamic response of following axis

## 2.5.1 Function

**Couplings**

The sections above (function descriptions for S3 and T3) describe axis couplings in which a following axis is moved depending on one or more leading axes/spindles.

Table 2-2 Programming of axis couplings is possible in

| coupling | in parts program | in synchronized actions |
|---|---|---|
| Tangential tracking | x | |
| Coupled axes | x | x |
| Master value coupling | x | x |
| Electronic gears | x | |

Machine data can be used to specify general limits for the following axis:

MD 32000: MAX_AX_VELO max. axis velocity

MD 32300: MAX_AX_ACCEL max. axis acceleration

**SW 6.4 and higher**

As of this software version, the dynamics limits specified above may be reduced or increased using commands from the **parts program** or from the **synchronous actions**.
Furthermore, it is also possible to configure the positions controller as a PI controller.

**Caution**

This option can only be used in conjunction with servo trace and with the appropriate technical knowledge of the control.

**References:** See /IAD/. Installation and Start–Up Guide
/FB/, G2, Vecolities, Setpoint/Actual Value Systems, Control

The share **in percent** that is considered is specified by the language command:

VELOLIMA[FA]

ACCLIMA[FA]

where FA is the following axis. Both an increase (100 < values $\leq$ 200) and a reduction (1 $\leq$ value < 100) is possible. Values outside the valid range (1 $\leq$ value $\leq$ 200) are rejected and Alarm 14811 issued.

The relevant limits are then:

| MD | Command | relevant limits for |
|---|---|---|
| 32000:<br>MAX_AX_VELO[Ax] | VELOLIMA[Ax] | axial velocity of the FA:<br>MAX_AX_VELO[A] * VELOLIMA[A] |
| 32300:<br>MAX_AX_ACCEL[Ax] | ACCLIMA[Ax] | axial acceleration of FA:<br>MAX_AX_ACCEL[A]*ACCLIMA[A] |

**Power ON**

The values for VELOLIMA and ACCLIMA are initialized to 100%.

**Mode change**

The dynamic offsets remain valid on transition from AUTO => JOG.

**RESET/program end**

The dynamic values for reset, i.e. the values of VELOLIMA and ACCLIMA are set channel-specific via MD 22410: F_VALUES_ACTIVE_AFTER_RESET.

In MD 22410: F_VALUES_ACTIVE_AFTER_RESET=FALSE, the values for VELOLIMA(FA) and ACCLIMA(FA) are set to 100%.

In MD 22410: F_VALUES_ACTIVE_AFTER_RESET=TRUE, the values set last are retained. This response also applies for dynamic offsets that were set by static synchronous actions.

If this is not the case despite the setting F_VALUES_ACTIVE_AFTER_RESET=FALSE, the dynamic offset must be applied again per IDS synchronous action or permanent writing.

**References:** /FBSY/, Synchronous actions

## 2.5.2 Examples

**Electronic gear**

Axis 4 is coupled via a electronic gear coupling to X. The acceleration capability of the following axis is limited to 70% of the maximum acceleration. The maximum permissible velocity is limited to 50% of the maximum velocity. After switching on, the maximum permissible velocity is set to 100% again,

```
......
N120 ACCLIMA[AX4]=70
N130 VELOLIMA[AX4]=50          ; Reduced velocity
.....
N150 EGON(AX4, "FINE", X, 1, 2)
.....
N200 VELOLIMA[AX4]=100         ; Full velocity
.....
```

**Master value coupling**

Axis 4 is coupled to X via leading value coupling. The acceleration capacity of the following axis is limited to 80% of the maximum acceleration.

```
......
N120 ACCLIMA[AX4]=80           ; 80 %
N130 LEADON(AX4, X, 2)         ; Activate coupling
.....
```

**Leading value coupling with synchronous action**

Axis 4 is coupled to X via leading value coupling. The acceleration response is limited to 80% as of position 100 per static synchronous action

```
......
N120 IDS=2 WHENEVER $AA_IM[AX4] > 100 DO ACCLIMA[AX4]=80
N130 LEADON(AX4, X, 2)
.....
```

## 2.5.3 System variables

For axis types Geometry axis, channel axis, machine axis and spindle, the following readable system variables are available in parts programs and synchronous actions:

**In run–in**

Table 2-3 System variables, programmable following axis dynamics

| Identifier | Data type | Meaning | Unit |
|---|---|---|---|
| $PA_ACCLIMA[n] | REAL | Acceleration offset set with ACCLIMA[Ax] | % |
| $PA_VELOLIMA[n] | REAL | Velocity offset set with VELOLIMA[Ax] | % |

**In the main run**

Reading the main run variable implicitly triggers a stop preprocessor.

| Identifier | Data type | Meaning | Unit |
|---|---|---|---|
| $AA_ACCLIMA[n] | REAL | Acceleration offset set with ACCLIMA[Ax] | % |
| $AA_VELOLIMA[n] | REAL | Velocity offset set with VELOLIMA[Ax] | % |

## 2.6 Extended stop/retract: ESR (SW 5 and higher)

The “Extended stop and retract” function ESR allows the operator to choose how to react to definable error sources, reducing the risk of causing damage to the workpiece and protecting man and machine against injury or damage.

**Introduction**

In contrast to conventional milling/turning/grinding, gear teeth machining has an “electronic coupling” among several axes, which **cannot** be supported or replaced by a relevant “mechanical coupling”; nevertheless, crucial for fault–free manufacture of the workpiece to be produced. Unlike a simple rotating milling tool, for example, which does not remove any more stock without an axis feedforward movement, the friction between the gear cutting (non–machined part) teeth and the gear hobbing (worm–shaped tool) is such that the teeth of the gear hobber would ’eat their way into’ the material if the electronic coupling failed.

Especially in the case of gear cutting (gear hobbing, generating grinding, gear shaping) expensive tools and workpieces are in use and must not be destroyed in the event of such a fault. Better protection for man and machine is also provided, if flaying cuttings are avoided from the outset.

**Solution concept**

The hazard conditions in the control are checked cyclically (*sources* of disruption) and linked (synchronized actions). Actions are triggered when reasons for initiating a separation of the tool and the workpiece are detected under the supplementary conditions for temporary upholding of the axis coupling in the electronic gear. These actions can be one or more of the ESR reactions described below.

**ESR reactions**

In SW % and lower, “Extended stop and retract” provides the following partial reactions:

- “Extended stop”
  (independent drive or externally driven) is a defined, delayed stop.
- “Retract”
  (independent drive or externally driven) means “escape” from the machining plane to a safe retraction position. This is to avoid a potential collision between tool and workpiece.
- “Generator operation”
  (independent drive) Generator operation is possible in the event that the DC link power is insufficient for safe retraction. As a separate drive operating mode, it provides the necessary power to the drive DC link for carrying out an orderly “Stop” and “Retract” in the event of a power or similar failure.

**In addition, for SW 6 and higher**

- "Extended stop" **(NC–prompted)** is a defined, "delayed" and "contour–friendly" stop.
- "Retract" **(NC–prompted)** means a "retreat" from the machining plane to a safe retracted position. This is intended to prevent any danger of collision between the tool and the workpiece. Gear cutting, for example, means a "Retraction from tooth gaps just machined".

All reactions are independent of each other and can be used as gearing in conjunction with other production. In this way, it is possible to configure retractions and temporary continuation of axis couplings so that they can be executed in parallel prior to standstill. In this case, a further axis in generator mode can maintain the DC link voltage.

**Interplay of NC–prompted reactions with ...**

NC–prompted reactions are triggered via channel–specific system variable $A**C**_ESR_TRIGGER (not to be mistaken for nc–global system variables for drive–independent retraction $A**N**_ESR_TRIGGER).

$A**C**_ESR_TRIGGER enables a smooth interpolatory *stop* on the path or contour.
The NC–prompted *Retraction* is performed in synchronism by the retraction axes in the **channel**. To prevent confusion and operator errors, retraction axes must always be assigned to exactly **one** NC channel and may not be switched among the channels.

For NC–prompted *stop*, path interpolation is also processed further in the same way as an electronic coupling via a definable period (MD 21380: ESR_DELAY_TIME1), although an alarm, for example, with motion stop is present. Then it is decelerated on the path as for NC–STOP.

For NC–prompted retraction, **LIFTFAST/LFPOS** is used in the same way as for thread cutting.
In order to perform retraction outside AUTOMATIC mode as well, triggering of this function is linked to the system variable $AC_ESR_TRIGGER. Retraction initiated per $AC_ESR_TRIGGER is locked to prevent multiple retractions.

---

**Note**

For the "Gearing" technology all reactions must go hand in hand: For example, the electronic coupling should be maintained for a certain time with constant motion before being stopped. In parallel, a retraction axis is operating to disconnect the machining action and another axis in generator operation supplies the necessary power in the DC link in the event of a power failure (provided appropriately configured).

The NCK does not trigger an **EMERGENCY STOP** autonomously. Since the EMERGENCY STOP signal is feed from the PLC to the NCK, the PLC able to actively trigger an ESR is required (e.g. per $A_DBB) and delay forwarding of the actual EMERGENCY STOP request to the NCK by the relevant time.

---

### 2.6.1 Reactions external to the control

Sending the requisite switching signals to the digital outputs (system variable $A_OUT) in the IPO cycle is called a reaction that is “external to the control“. For example, a hydraulic retraction axis can be connected to this type of digital output. The machine manufacturer or start-up engineer is responsible for defining further reactions.

### 2.6.2 Independent drive reactions

Independent drive reactions are defined axially, that is, if activated each drive processes its stop and retract request independently. An interpolatory or path-aligned coupling of axes at stop or retract is **not** available (only for control management).
A reference to the axes is timed.

During and after execution of independent drive reactions, the NC enables/ travel commands no longer have an effect on the respective drive, therefore, it is necessary to perform a power OFF/power ON. Alarm 26110: “Independent drive stop/retract triggered” indicates this.

**Note:**
If retraction is not triggered in drive 611D, no alarm 26110 is issued.

---

**!**

**Important**

Independent drive stop and retract are “automatically” triggered (as with communication failure).

These drive-end reactions are therefore **cross-channel functions**. This means that if independent drive stop and retract is triggered in one channel, the drives of another channel also produce the independent drive stop/retract reactions configured (and just enabled) for them.

---

**Independent drive generator operation ESR_REACTION = 10**

The generator operation is

- configured (configuration: MD 37500: ESR_REACTION=10; the configuration must be defined in the axis-specific machine data of the appropriate axis),
- enabled ($AA_ESR_ENABLE) and
- activated: in the drive according to the settings in the drive machine data with DC link undervoltage.

**Independent drive retract ESR_REACTION = 11**

Independent drive retract is

- configured, (MD 37500: ESR_REACTION=11; time specification and retract velocity are set in MD; see “Example: Using the independent drive reaction” in Section 6.3).
- enabled: system variable $AA_ESR_ENABLE
- started: system variable $AN_ESR_TRIGGER.

**Independent drive stop ESR_REACTION = 12**

Independent drive stop is

- configured (configuration: MD 37500: ESR_REACTION=12);
- enabled ($AA_ESR_ENABLE) and
- started: system variable $AN_ESR_TRIGGER.

---

**Note**

For independent drive reactions the behavior can be determined individually for each axis.

---

**Example**

For an example of how the independent drive reaction can be used, please refer to Section 6.3.

## 2.6.3 NC-controlled extended stop (SW 6 and higher)

**Response**

The schedule for extended stop is defined by both machine data MD 21380: ESR_DELAY_TIME1 and MD 21381: ESR_DELAY_TIME2.
For the duration of the period in MD 21380: ESR_DELAY_TIME1 interpolates the axis unhindered as programmed. After expiry of the period in MD 21380: ESR_DELAY_TIME1, interpolatory braking (ramp stop) is initiated.
The maximum time for interpolatory braking is then the time in MD 21381: ESR_DELAY_TIME2; after this period, rapid deceleration with subsequent correction is initiated.

This schedule applies if MD 37500: ESR_REACTION > 20 applies for at least one of the axes applied by the NCU. If this condition is not satisfied, the alarm reactions are not delayed. If no ESR is active in the cycle, the above-mentioned alarm reactions are delayed one IPO cycle (it takes one IPO cycle to check whether ESR is active).

For all other axes MD 37500: ESR_REACTION = 0 not mentioned, rapid deceleration with subsequent follow-up is initiated at the start of extended stop ($AC_ESR_TRIGGER = 1).

During processing of all commands, especially ones that result in an axis stop (e.g. RESET, Stop, Stopall, StopByAlarm), as well as the standard alarm reactions STOPBYALARM and NOREADY, the total for times ESR_DELAY_TIME1 and ESR_DELAY_TIME2 are delayed.

An NC-prompted stop interacts with the electronic gear (cf. 2.4). It contains the (selective) switchover of the electronic gear to actual value coupling if there is a fault in the leading axes, and also upholds interpolation and enable ("Continue travel") during a period which can be specified per MD 21380: ESR_DELAY_TIME1.

**Times T1 and T2**

The times T1 and T2 are parameterized via the machine data MD 21380: ESR_DELAY_TIME1 and MD 21381: ESR_DELAY_TIME2.

The timing for NC/prompted extended shutdown can be taken from the figure below.

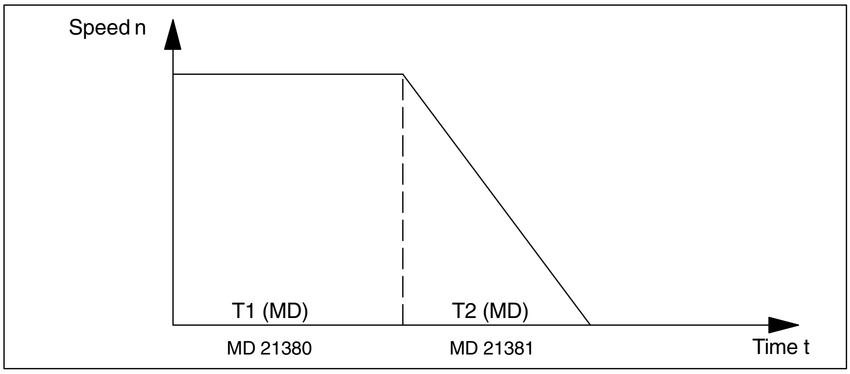



Fig. 2-8 Parameterizable/programmable control-driven shutdown

**Note**
The summer time T1 and T2 should not exceed a maximum value, e.g. 1 second, for security reasons.

**Effects of ESR_REACTION = 22**

Effects for a path axis
If MD 37500: ESR_REACTION = 22 is for a continuous-path axis, the response “Extended stop” is transferred to **all continuous-path axes** of the channel.

Effects for a leading axis
If MD 37500: ESR_REACTION = 22 is for a leading axis, the response “Extended stop” is transferred to **all following axes** of the channel.

---

**Note**

A following axis of the electronic gear follows the leading axis during the two phases of the extended stop according to the rule of motion, i.e. no separate braking is possible on transition from phase MD 21380: DELAY_TIME1 to phase MD 21381: ESR_DELAY_TIME2.

The precondition for a properly functioning ESR is that the activate signals are set and remain set.

---

### 2.6.4 NC-controlled retraction (SW 6 and higher)

**Initial conditions**

The following are significant for NC-prompted retraction:

- the axes selected with POLFMASK
- the axis-specific positions defined with POLF
- the time slots ESR_DELAY_TIME1 and ESR_DELAY_TIME2
- when triggered by system variable $AN_ESR_TRIGGER
- the agreed ESR reaction MD 37500: ESR_REACTION
- G code LFPOS of modal 46. G code group addressed.

**Response for ESR_REACTION = 21**

If the system variable $AC_ESR_TRIGGER = 1 is set and if a retraction axis is configured in this channel with MD 37500: ESR_REACTION = 21 and $AA_ESR_ENABLE=1 is set for this axis, then **LIFTFAST** becomes active in this channel.

**Precondition:**
The retraction position must be programmed in the parts program. The activate signals must be set for the retraction movement and remain set.

Fast retraction to the position defined with **POLF** is triggered via G code **LFPOS** of the modal 46th G code group.

The lift configured with **LFPOS**, **POLF** for the axes selected with **POLFMASK** <u>replaces</u> the <u>continuous-path motion</u> defined for these axes in the parts program. Lifting to a position is not interpolated as an overlaid movement. This can produce undesired effects for axes that are coupled to a leading axis.

The extended retraction (i.e. LIFTFAST/LFPOS initiated through $AC_ESR_TRIGGER) is **cannot be interrupted** and can only be terminated prematurely per EMERGENCY STOP.

Speed and acceleration limits for the axes involved in the retraction a monitored during extended retraction.

The maximum time available for retraction is the sum of the times MD 21380: ESR_DELAY_TIME1 and MD 21381: ESR_DELAY_TIME2. When this time has expired, rapid deceleration with follow-up is also initiated for the retraction axis.

**Programming**

The destination for the retraction axis is programmed with the language command**:**

**POLF**

**POLF[ geoaxis name I machine axis name ]**(**PO**sition **L**ift**F**ast).

**POLF** is modal.

**POLF** can also be programmed as incremental.
If this programming is carried out with a **geometry axis**, the position is interpreted as a position in the workpiece coordinate system WCS.

The frame valid at the time when lift fast was activated is considered. Important: frames with rotation also affect the direction of lift via **POLF**.

If **POLF** is programmed with a **channel/machine axis**, the position of the machine coordinate system MCS must be specified. Frames with rotation do not affect the position for retraction.

If the identifiers for the geoaxis and channel/machine axis are identical, retraction is carried out in the workpiece coordinate system.

**POLFMASK**

The language command **POLFMASK([ axisname1], [axisname2], ....)** enables selection of the axes that are to travel to a their position defined by **POLF** when fast list is activated. A variable parameter list can be used to select any number of axes for lift fast; however, all axes must be located in the **same coordinate system** (i.e. only geoaxes).

The parameters valid <u>at the triggering time</u> are decisive for the lift movement. If one of these parameters (G code, **POLF**, **POLFMASK**, Frame, etc.) changes during lifting (block change), this change does not affect the lift movement that has already started.

Before fast list to a fixed position can be activated via **POLFMASK**, a position must be programmed with **POLF** for the selected axis. No machine data is provided for presetting the values of **POLF**.

During interpretation of **POLFMASK**, Alarm 16016 is issued if **POLF** has not been programmed. The effect of the language command **POLFMASK** is also referred to below as “retraction is activated”.

If retraction is activated, the position for retraction can still be changed. However, it is no longer possible to change the coordinate system and an attempt is rejected with an alarm 16015.

If **POLF** is programmed again after activating retraction, the position at which this axis was first programmed must be specified in the coordinate system.

**Change coordinate system**

If the coordinate system is to be changed, fast retraction must first be deactivated with **POLFMASK()** and then **POLF** used to carry out programming in the new coordinate system.

**Deactivate fast retraction**

**POLFMASK()** without specifying an axis deactivates fast retraction for all axes. In the above terms, retraction is no longer activated.

**Start of parts program**

The positions programmed with **POLF** and the activation by **POLFMASK** are deleted when the parts program is started. This means that the user *must* program the values for **POLF** and the selected axes (**POLFMASK**) in each parts program.

**Example**

An example application for parameterizing with several axes and incremental programming is giving in Section 6.3.

### 2.6.5 Possible trigger sources

The trigger sources must be distinguished by evaluating the specified system variables. Any system variables which can be read in **synchronized actions** are available as error sources, e.g.

– Digital I/Os ($A_IN, $A_OUT)
– Synchronization differences ($VA_EG_SYNCDIFF)
– Channel status ($AC_STAT) ...

The drive states can be read in $AA_ESR_STAT:
Bit 0: Generator operation is triggered
Bit 1: Retraction is triggered
Bit 2: Extended stop is triggered

**References:** /PGA/, Programming Guide Advanced

The following error sources are possible for starting “Extended stop and retract”:

**General sources**

General sources (NC-external/global or mode group/channel-specific):

- Digital inputs (e.g. on NCU module or terminal box) or the readback digital output image within the control ($A_IN, $A_OUT)
- Channel status ($AC_STAT)
- VDI signals
  Access via $A_DBB. This approach is **not recommended for time-critical signals**, since the PLC cycle time is included in the overall time. However, it is an appropriate way for the PLC to influence the sequence or **activation** of the extended stop and retract function. It still makes sense to link PLC states, provided that these are powered/controlled **exclusively by the PLC** (e.g. EMERGENCY STOP, RESET key, Stop key).
- Group messages of a number of alarms ($AC_ALARM_STAT)

**Axial sources**

- Emergency retraction threshold of following axis (synchronization difference of electronic coupling, $VA_EG_SYNCDIFF[following axis])
- Drive: The system variable $AA_ESR_STAT[axis] “Status for extended stop and retract” displays: Bit 3: DC link undervoltage/generator operation)
- Drive: The system variable $AA_ESR_STAT[axis] “Status for extended stop and retract” displays: Bit 4: Generator minimum speed)

---

**Note**

If **NC-prompted ESR** is configured, it takes one IPO cycle to process the alarm reactions NOREADY and STOPBYALARM. This cycle checks whether the alarm source is for ESR. The reaction “Trigger ESR” or standard reaction (without ESR) occurs in the next IPO cycle. Self-resetting alarm 21600 displays this status; the checking time is included in the alarm response time. **You can use $MN_SUPPRESS_ALARM_MASK bit16 to suppress display of alarm 21600.**

---

## 2.6.6 Logic operation: Source/reaction logic operation

The flexible logic operation possibilities of the **static synchronous actions** can be used to trigger specific reactions based on sources. Logic operations of all relevant sources by means of static synchronous actions are the responsibility of the user/machine manufacturer. They can selectively evaluate the source system variables as a whole or by means of bit masks, and then make a logic operation with their desired reactions. The static synchronous actions are effective in all operating modes. For a detailed description on how to use synchronous actions, please refer to:

**References**: /FBSY/ Description of Functions Synchronized Actions
/PGA/ Programming Guide Advanced
(Synchronous actions, system variables)

You can use $AA_TYP (axis type) as required, for example, to configure axial sources or channel-specific sources.

## 2.6.7 Activation

**Function enable**

**$AA_ESR_ENABLE**
The functions generator operation, stop and retract are **enabled** by setting the associated control signal ($AA_ESR_ENABLE[axis]). This control signal can be modified by synchronized actions, by the parts program and (indirectly) by the PLC.

Writing in $A_DBB allows the PLC to extensively influence the execution of the ESR reactions, if appropriate access is also integrated into the synchronous actions. Thus the PLC can directly influence the ESR response.

**Function trigger**

**$AN_ESR_TRIGGER (drive-independent)**

- Generator operation "automatically" becomes active in the drive when the risk of DC link undervoltage is detected.
- Drive-independent stop and/or retract are activated when communication failure is detected (between NC and drive) as well as when DC link undervoltage is detected in the drive (providing they are configured and enabled).
- Independent drive stop and/or retract can also be triggered by parts programs/synchronous actions by setting the system variable $AN_ESR_TRIGGER (command to **all** drives). Precondition: activated.

**$AC_ESR_TRIGGER (NC-prompted)**

- NC-prompted shutdown is activated if configured
MD 37500: ESR_REACTION = 22 activated by setting the control signal "$AC_ESR_TRIGGER". Precondition: activated.
- NC-prompted retraction is activated if configured
MD 37500: ESR_REACTION = 21 and **POLF** and **POLFMASK** in the parts programby setting the control signal "$AC_ESR_TRIGGER". Precondition: activated.

## 2.6.8 Power failure detection and bridging

**Detection**

A power failure can be detected when the mains supply monitoring of the connected actuator is used as an external source via terminal 73 of the SIMODRIVE 611D I/RF module (e.g. external sources: NCU input or terminal box).

**Delay**

The time delay until the mains supply monitoring relay picks up corresponds to approx. 10–15 ms.

After the relay picks up, a minimum of 1/2 IPO cycle and a maximum of 3 IPO cycles pass.
The time for power failure detection is derived from this:

| | |
|---|---|
| Worst case | approx. 120ms |
| Best case | approx. 15ms |

**Limits of DC link overvoltage**

The DC link is monitored for the following voltage limits:

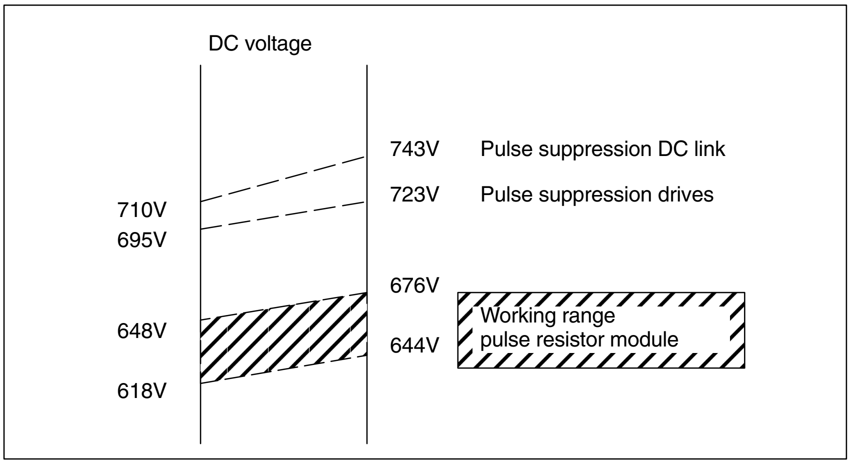



Fig. 2-9 Voltage level of SIMODRIVE 611D DC link

The drive and DC link pulses are deleted at specific voltage levels. This automatically causes the drives to coast down.

If this behavior is not desired, the user can use a resistor module to divert the surplus energy. This resistor module operates in the gray hatched area in the diagram, thus lying below the critical voltage level.

---

**Note**

The pulse power of the resistor module is greater than the I/RF power.

---

**Monitoring the DC link undervoltage**

The DC link voltage can be monitored for a threshold parameterized by the user (MD 1634: LINK_VOLTAGE_RETRACT).

Voltage below the threshold set in MD 1634: LINK_VOLTAGE_RETRACT can be utilized as internal error source for retracting. This is to avoid disconnection of the drive hardware without separation of workpiece and tool when the DC link voltage is less than the minimum of 280V.

In addition, you can program for one/several axis/axes (useful for one axis per I/RF area), whether a retraction is to be triggered when the voltage falls below the DC link threshold (MD 1634). Precondition is that the logic operation for the synchronous action is dependent on the system variable $AA_ESR_STAT.
This means that any parameterized and programmed ESR is carried out if it is enabled via system variable $AA_ESR_ENABLE.

The power required for ESR can be supplied to the DC link by parallel, regenerative braking: See DC link backup.

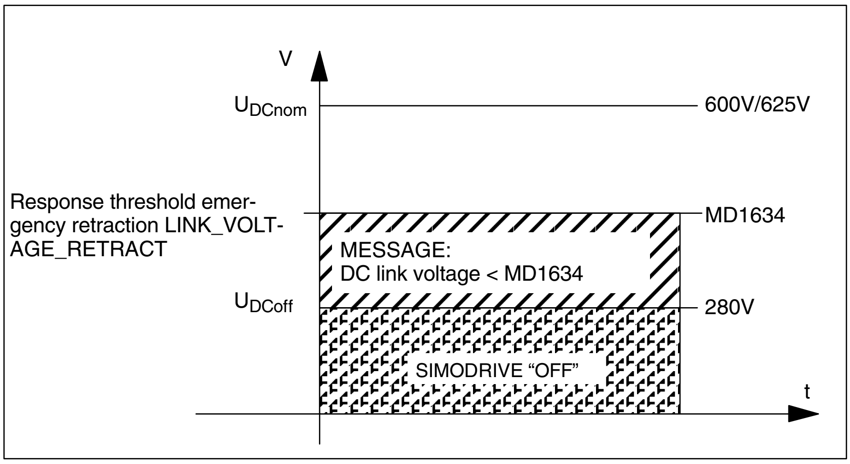



Fig. 2-10 DC link voltage monitoring SIMODRIVE 611D

**Communication/ control failure**

When the NC sign-of-life monitoring responds, a communication/control failure is detected on the drive bus and an independent drive ESR is performed if appropriately configured.

---

**Note**

As of SW 4.2, setting the preset value from 600 V to 0 V activates measurement of the DC link voltage per default.
To enable fault-free start-up of older hardware that does not have a DC link measurement MD 1161 “Calculate controller data” (FIELDVAL_FIXED_LINK_VOLTAGE) is set to 600 V.

---

## 2.6.9 Generator operation/DC link backup

**DC link backup**

You can compensate for temporary DC link voltage dips by configuring the drive MD and appropriately programming the system variable $AA_ESR_ENABLE via static synchronous actions. The bridged time depends on the energy stored by the generator that is used for DC link backup, as well as on the energy requirements for maintaining the current motions (DC link backup and monitoring for generator speed limit).

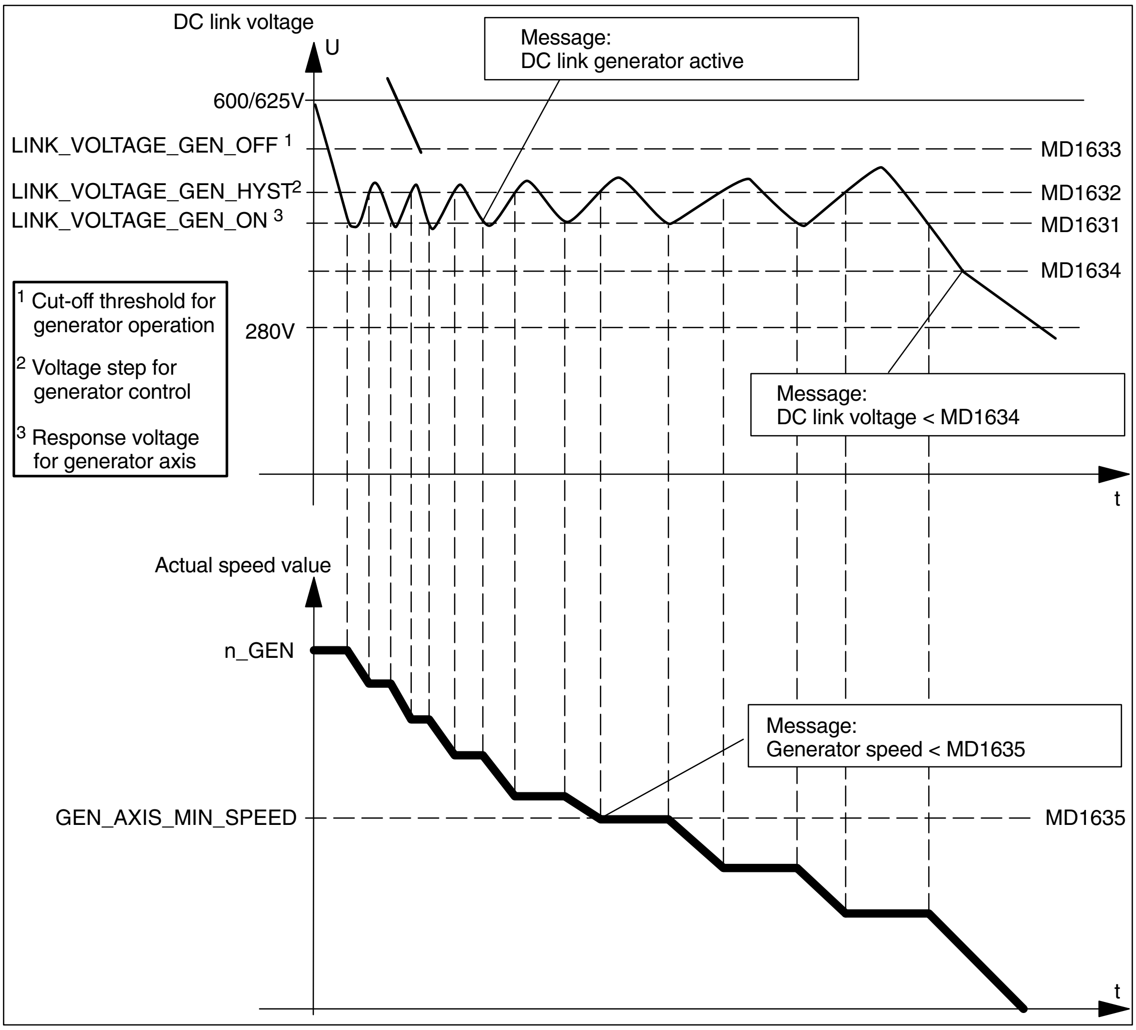



Fig. 2-11 Generator operation

When the DC-link voltage is below the minimum threshold (MD 1631: LINK_VOLTAGE_GEN_ON) the axis/spindle concerned switches from position-controlled or speed-controlled mode to DC-link voltage-controlled mode. By braking the drive (default speed setpoint = 0), regenerative feeback to the DC link takes place. The drive measures the DC link voltage cyclically (in the position control cycle). If the voltage exceeds the value set in MD1631: LINK_VOLTAGE_GEN_ON and MD1632: LINK_VOLTAGE_GEN_HYST, the two-step control is disabled, that is, the current actual speed value is preset as speed setpoint.

During active generator operation, bit 3 “DC link generator active” is output in system variable $AA_ESR_STAT.

The two-step behavior of the generator is machine and user-specific.

If the voltage exceeds the value set in MD1633: LINK_VOLTAGE_GEN_OFF, generator operation is exited and operation is switched back to speed-controlled operation.
This is not the case if the axis/spindle was previously in position-controlled mode. In this case, it is necessary to reset the drive (power ON).

**Monitoring the generator speed minimum limit**

In addition to generator operation to back up the DC link, the actual speed value of the axis/spindle in generator operation is monitored for any speeds lower than the minimum speed set in MD1635: GEN_AXIS_MIN_SPEED.
When values below this speed limit are detected, bit 4 “Generator speed < MD1635” is output in system variable $AA_ESR_STAT.

In addition, analogous to the detection for voltages below the permissible DC link voltage (MD1634: LINK_VOLTAGE_RETRACT), this signal can be defined as an internal source of error for ESR.

## 2.6.10 Independent drive stop

The drives of a previously coupled grouping can be stopped by means of time-controlled cutout delay with minimum deviations from each other, if this cannot be performed by the control.

Independent drive strop is configured via MD 37500: ESR_REACTION=12 activated with the system variables $AA_ESR_ENABLE and started after the delay time T1 (see below) with the system variable $AN_ESR_TRIGGER.

T1 is defined in MD 1637: GEN_STOP_DELAY.

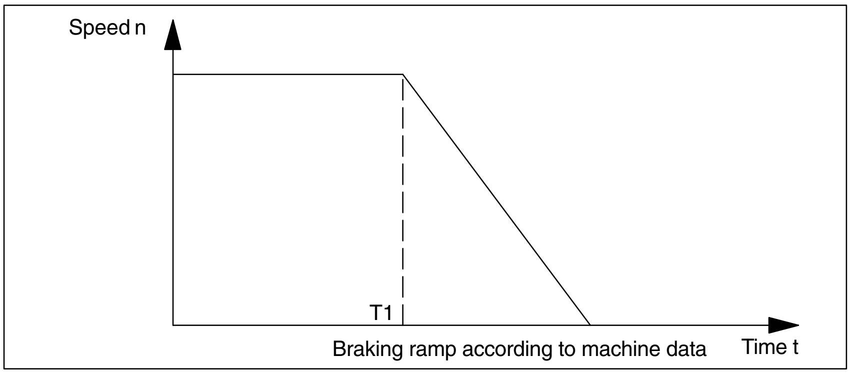



Fig. 2-12 Independent drive stop SIMODRIVE 611D

**Reactions**

The speed setpoint currently active as the error occurred will continue to be output for time period T1. This is an attempt to maintain the motion that was active before the failure, until the physical contact is annulled or the retraction movement initiated in other drives is completed. This can be useful for all leading/following drives or for the drives that are coupled or in a group.

After time T1 all axes are stopped at the current limit with zero speed setpoint and the pulses are deleted when zero speed is reached.

## 2.6.11 Independent drive retract

Axes with digital 611D drives can (if configured and enabled) independently conduct a retract motion

- when the control fails (sign-of-life failure detection).
- when the DC link voltage drops below a warning threshold.
- when triggered by system variable $AN_ESR_TRIGGER.

The retract movement is conducted independently through 611D.

After the beginning of the retraction phase the drive independently maintains its enables at the previously valid values. The emergency retraction is only conducted if pulse and servo enable (and system variable $AA_ESR_ENABLE) were set at the time the retraction was triggered and **the drive in question was therefore enabled**.

In the event of control failure the pulse enable set is sufficient. In this case the 611D drive independently generates its servo enable if it is still able to do so (subfunctionality for "Retract with clamped axes"). Any clamped axes have to be connected by the user.

**External safety logic**

The external safety logic for a control drive pair with drive emergency retraction must be implemented in such a way that the drive unit is still operable in the event of a control failure (for example, PLC stop and NC READY failure; the appropriate machine safety must be configured).

**Measuring system**

For the drive there is no reference to the NC geometry system. On the NC side, the unit system of the **motor measuring system** is only known if it is used **as a position measuring system**.

**Retraction path**

The retraction path is therefore specified to the drive using the following geometry-neutral data:

- Speed setpoint, direction (leading sign):
  MD 1639: RETRACT_SPEED
- Travel time: MD 1638: RETRACT_TIME

The drive traverses the programmed "retraction path" using a time-controlled speed that it specified in the drive.

It must be activated by the system variable $AA_ESR_ENABLE and triggered with $AN_ESR_TRIGGER.

The “retraction path” really traversed in the event of an error depends on the current actual speed at the time that the emergency retraction was started and can deviate slightly from the programmed path as the drive does not monitor a path (no interpolation).

After this process speed setpoint zero is preset for the retraction axes too and a standstill occurs at the current limit (comp. independent drive stop).

---

**Note**

- The independent drive emergency retraction is only effective if the bit “pulse suppression” is set to OFF in MD1612: ALARM_REACTION_POWER_ON and MD1613: ALARM_REACTION_RESET.
- With active emergency retraction it is not possible to parameterize the emergency retraction. Although data are transferred to the drive, they are not accepted. There is no message to the user.

---

### 2.6.12 Configuring aids for ESR

**Voltage failure**

The following hardware and software components are required:

- Hardware components
  - SINUMERIK 840D with e.g. NCU 573 and MMC 103
  - SIMODRIVE 611D with servo drive controls 6SN1 118–0DG... or 6SN1 118–0DH...
  - Closed-loop controlled I/RF module (16kW and greater) with suitable pulse resistor module and additional capacitors for the DC link if required.
  - Capacitor module (6FX2 006–1AA00) for backing up the 115–230VAC power supply for the central controller and the operator panel front or alternatively the 24VDC power supply.
- Software components
  - System software: V5
  - ESR option

The following points must be taken into account for configuring:

1. The electronics supply of the servo drive control must be provided by the DC link. For this, the user must connect the I/RF modules to the DC link (see Installation & Start-up Guide 611D).
2. A suitable backup system must be available for the NC and operator panel front; e.g. a capacitor module for 230V power supply or an accumulator for 24V power supply.
3. There must be a suitable backup system for supplying power to the PLC I/Os or the NCK terminal block I/Os, e.g. an accumulator.

**DC link backup**

The energy available in the drive DC links on power failure is calculated as follows:

$E = 1/2 * C * (U_{DC}^2 – U_{min}^2)$

With

E= Energy in watt seconds [Ws]
C= Overall capacitance of the DC link in Farad [F]
$U_{DC}$= Contents of MD1634: LINK_VOLTAGE_RETRACT
$U_{min}$= Minimum limit for safe operation
(taking the motor-specific electromotive force into account, but still above the deactivation threshold of 280 V in all cases)

**Example:**

For
C= 6000µF (see Table 2-4, 1st line) – 20% = 4800 µF
$U_{DC}$= 550V (MD 1634)
$U_{min}$= 350V (assum.)

E is calculated as:

$E = 1/2 * 4800\mu F *((550V)^2 – (350V)^2) = 432Ws$

Under load conditions, this energy is available for a time period of

$t_{min} = E / P_{max} * \eta$

to initiate the emergency retraction.

With

$t_{min}$ = Backup time in milliseconds [ms]
$P_{max}$ = Power in kilowatts [kW]
$\eta$ = Efficiency of the drive unit

The calculated values for the above example where:
E = 432Ws
$P_{max}$ = 16kW (see Table 2-4, 1st line)
$\eta$ = 0.90
are as follows:

$t_{min}$ = 432Ws / 16kW * 0.9 = 24.3ms

as the minimum backup time for emergency retraction.

The following table shows the values for different I/RF units. Nominal and minimum capacity are taken into account. The maximum possible capacitance (load limit) consists of the sum of the capacity of the I/RF module and the axis/spindle modules, as well as the external auxiliary capacitors (to be provided by the user). The minimum capacitance used in the table takes a component tolerance of –20% into account (worst case).

Table 2-4 Nominal and minimum backup time for different I/RF units

| Power $P_{max}$ of I/RF unit [kW] | Max. possible capacitance $C_{max}$ [μF] | Energy contents ($C_{max}$) [Ws] | Energy contents ($C_{min}$) [Ws] | Backup time $t_n$ with $P_{max}$ [ms] | Backup time $t_{min}$ with $P_{max}$ [ms] |
|---|---|---|---|---|---|
| 16 | 6000 | 540 | 432 | 30.38 | 24.30 |
| 36 | 20000 | 1800 | 1440 | 45.00 | 36.00 |
| 55 | 20000 | 1800 | 1440 | 29.46 | 23.56 |
| 80 | 20000 | 1800 | 1440 | 20.25 | 16.20 |
| 120 | 20000 | 1800 | 1440 | 13.50 | 10.80 |

### Energy balance

When configuring the emergency retraction, it is always necessary to establish an energy balance to find out whether you can do without an additional capacitor module or a generator axis/spindle (with correspondingly dimensioned centrifugal mass).

### Stopping as energy supply

From approx. the third interpolator cycle, changes are made to the speed setpoints for the configured stop/retraction axes/spindles.
After this time lapse, the braking phase starts (if no independent drive stop is configured in this axis).

As soon as the braking process is initiated, the energy released in this manner is available for retraction motion. Use an energy balance to ensure that the kinetic energy of the braking axes is sufficient for the retraction.

The energy balance shows the maximum setting for the interpolator cycle time which will allow a safe emergency retraction to be executed.

**Example:**
With a 16kW unit under maximum load and minimum DC link capacitance it should be possible to execute an emergency retraction without generator operation. For this, the interpolator cycle time should theoretically be a max. of 4.86ms, i.e., in this case you can set a max. of 4ms.
If required a more powerful NC-CPU can be used in order to achieve optimum conditions.

### Independent drive stop/retract

Independent drive stop and retract triggered by the NC is used when a very fast reaction is required. In this case, the drive reacts within one interpolator cycle and outputs a setpoint value for the configured axes/spindles.

**Important:**
After independent drive stop and retract it is necessary to perform a power ON.

---

**Note**

When the drive bus is interrupted between NC and drive (sign of life failure) a stop and retract can only be initiated by the drive.
Usually this situation does not occur at the same time as a power failure.

---

**Generator operation**

Generator operation is intended for situations where the DC link power is insufficient for safe retraction (for at least three IPO cycles). The mechanical power of a spindle/axis is used and the energy is optimally fed back to the DC link. The DC link voltage is kept within the limits set in the machine data by means of a two-step control.

In this case, the axis/spindle parameterized as a generator measures the DC link voltage if it falls below the value set in the ms cycle. Thus the DC link is backed up within 2 ms. (In normal conditions measurement every 4 ms.)

The energy stored in the drive

$$E = 1/2 * \Theta * \omega^2$$

where

$\Theta$ = Total mass moment of inertia
$\omega$ = Angular velocity at the time of switching to generator operation

is recovered with approx. 90% efficiency.

For generator operation, it is advisable, especially when using large machines with powerful I/RF units (55, 80, 120kW), to use a separate drive with centrifugal mass which, after acceleration to maximum speed, only has to generate the friction loss.

Of course, it is also possible to use any other drive as long as it is not directly participating in the controlled stop/retract.

Axes that are participating in gearbox links that must be specifically maintained are not suitable for this purpose.

---

**Note**

A minimum speed limit for the generator ($AA_ESR_STAT, bit 4) can also be the source for the retraction process. This is advisable, for example, when generator operation is to be used to bridge short voltage interruptions.

---

In order to prevent the DC link voltage from becoming too high when braking starts, and the drive from reacting with pulse suppression (which would cause uncontrolled coasting down), it is necessary to use suitable pulse resistor modules.

## 2.6.13 Control behavior

**Power OFF/ power ON**

If the retraction logic is stored in motion-synchronous actions, they are not yet active on power ON.
If logic operations are to be active after power ON, they must be activated in an asynchronous subroutine started by the PLC.
If an independent drive stop/retract is triggered, the drive software subsequently requires a power OFF/power ON.

As of Software Release 6, the function “Event-controlled program calls” may be used instead of ASUB.

**Mode change, NC Stop**

**Static** synchronous actions can be used for the logic operations (vocabulary word IDS). They are not affected by a mode change or NC Stop/reset.

Positioning a command axis/spindle is aborted with channel stop.

ESR is available in the modes AUTO, JOG, MDA.

**RESET**

Static synchronous actions are maintained after a reset.

On RESET, the values programmed for **POLF** and the activation by **POLFMASK** are not deleted.

**Start of parts program**

On start of a parts program, the values programmed for **POLF** and activation by **POLFMASK** are deleted.

The reason for resetting the programmed values **POLF/POLFMASK** is to force the ESR user (just as other users of the lift fast function) to explicitly program the matching retraction position for each workpiece programmed in each parts program, rather than trust that a suitable retract position has been stored in a previous machining process.

**Alarm behavior**

- Errors in an axis outside the EG axis grouping:
  This axis switches off “normally”. Stop and retract continue “undisturbed” or are triggered by this type of error.
- Error in a leading axis (LA):
  Selective switchover to actual-value linkage already during stop, otherwise as previously.
- Error in a following axis (FA):
  - Carry out retract: Retraction axis may not be a following axis, that is, no conflict.
  - Carry out stop: The following axis may react with uncontrollable behavior. Saving the workpiece/tool must be left to the retraction; however, the stop should not disrupt the process any further.
- Error in the retraction axis: There is no retraction.

- EMERGENCY STOP
  An EMERGENCY STOP is not a fault from the control's point of view, rather the response is the same as for any other control signal. For safety reasons, EMERGENCY STOP interrupts the interpolation and all traversing movements, and also dissolves the electronic coupling by canceling the servo enables.

  In applications where the coupling and traversing movements must remain valid after EMERGENCY STOP, this EMERGENCY STOP must be **delayed** long enough by the **PLC** for the required NC or drive-end reactions to terminate.
  The IS "ESR reaction is triggered" DB31, ... DBX98.7 is available as return signal to the PLC.

If an alarm with channel-independent NOREADY reaction is issued during the active phase of the ESR
(i.e. NOREADY |
NCKREACTIONVIEW |
BAGREACTIONVIEW),
then ESR is triggered in **all** channels.

**Block search, repositioning**

Extended stop and retract does not affect block search or repositioning motions.

## 2.6.14 Supplementary conditions

**Operational performance of the components**

The axis/spindle components participating in "Extended stop and retract", "drives, motors, encoders" must be operational. If one of these components fails, the full scope of the described reaction can no longer be guaranteed. Axis-specific servo or drive alarms describing the failure of one of these components are also implicitly signaling that the configured stop or retract reaction of the axis (axes)/spindle(s) is no longer (fully) available.

**Motion-synchronous actions**

Motion-synchronous actions are executed in the interpolator cycle. If there are many motion-synchronous actions, the runtime of the control for processing the cyclical interpretation of conditions in the synchronous actions is increased. The selected sources and the reactions to be assigned can "only" be evaluated/triggered in the interpolator cycle.

**Priority**

Each **drive-independent reaction** has a higher priority that the corresponding NC-prompted reaction (reason: when broadcast mode is activated for the drive, each drive-independent reaction becomes directly active)

**Power ON**

If independent drive stop/retract has been triggered, the drive software requires a subsequent power OFF/power ON (drive behavior as with serious errors, see also communications failure).

■

# Notes

# Supplementary Conditions 3

**Availability of “Coupled motion” function**

This function is available for

- SINUMERIK FM–NC with NCU 570, SW 2 and higher.
- SINUMERIK 840D with NCU 571/572/573, SW 2 and higher.
- SINUMERIK 810D, SW 3.2 and higher.
- SINUMERIK 840Di, 840DiE

**Availability of “Curve table” function**

This function is available for

- SINUMERIK 840D with NCU 571/572/573, SW 4 and higher
- From SW 6 also 5th degree polynomials.
- Option for SINUMERIK 840Di, 840DiE with restricted functionality

**Availability of “Leading value coupling” function**

This function is available for

- SINUMERIK 840D with NCU 571/572/573, SW 4 and higher
- SINUMERIK 810D with CCU 2, SW 2 and higher.
- Option for SINUMERIK 840Di, 840DiE with restricted functionality

**Availability of “Electronic gear” function**

The function is available on the

- SINUMERIK 840D with NCU 573, SW 5 and higher.
- As of SW 6 also non-linear coupling via curve tables and
- cascaded electronic gears.
- Option for SINUMERIK 840Di, 840DiE

**Availability of “Extended stop/re-tract” function**

The function is available on the

- SINUMERIK 840D with NCU 573, SW 5 and higher.
- Besides drive-independent (SW 5), also NC-prompted stop and retraction (SW 6 and higher).
- Not available for SINUMERIK 840Di, 840DiE

**Availability of the function "Independent drive retract"**

The function is not yet available for

- SINUMERIK 840Di with drive system SIMODRIVE 611 universal.

## Notes

# Data Descriptions (MD, SD) 4

## 4.1 General machine data

| | | | | |
|---|---|---|---|---|
| **11660**<br>MD number | **NUM_EG**<br>Number of possible EG axis groupings | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 31 |
| Changes effective after power ON | | | Protection level: 1 / 1 | Unit: – |
| Data type: Byte | | Applies from SW 5 | | |
| Meaning: | To allow implementation of the “Electronic gear” function, memory space corresponding to the size specified here is reserved in the S-RAM and D-RAM. The setting in this MD determines the maximum number of EG axis groupings which can be defined simultaneously with EGDEF. | | | |

| | | | | |
|---|---|---|---|---|
| **18400**<br>MD number | **MM_NUM_CURVE_TABS**<br>Number of curve tables (SRAM) | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: plus |
| Changes effective after power ON | | | Protection level: 1/1 | Unit: - |
| Data type: DWORD | | Applies from SW 4.1 | | |
| Meaning: | Defines the maximum number of curve tables that can be implemented in the entire system. A curve table comprises several curve segments. | | | |
| Related to .... | MD 18402: MM_NUM_CURVE_SEGMENTS | | | |

| | | | | |
|---|---|---|---|---|
| **18402**<br>MD number | **MM_NUM_CURVE_SEGMENTS**<br>Number of curve segments (SRAM) | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: plus |
| Changes effective after power ON | | | Protection level: 1/1 | Unit: - |
| Data type: DWORD | | Applies from SW 4.1 | | |
| Meaning: | Defines the maximum number of curve segments that can be implemented in the entire system. The curve segments are components of a curve table. | | | |
| Related to .... | MD 18400: MM_NUM_CURVE_TABS | | | |

| | | | | |
|---|---|---|---|---|
| **18404**<br>MD number | **MM_NUM_CURVE_POLYNOMS**<br>Number of curve table polynomials (SRAM) | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: plus |
| Changes effective after power ON | | | Protection level: 1/1 | Unit: - |
| Data type: DWORD | | Applies from SW 4.1 | | |
| Meaning: | Defines the maximum number of polynomials for curve tables that can be implemented in the entire system. The polynomials are components of a curve segment. A maximum of 3 polynomials are required for one curve segment. As a rule, only 2 polynomials are used per curve segment. | | | |
| Related to .... | MD 18400: MM_NUM_CURVE_TABS<br>MD 18402: MM_NUM_CURVE_SEGMENTS | | | |

| | | | | | |
|---|---|---|---|---|---|
| **18406**<br>MD number | **MM_NUM_CURVE_TABS_DRAM**<br>Number of curve tables (DRAM) | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: plus | |
| Changes effective after power ON | | | Protection level: 1/1 | | Unit: – |
| Data type: DWORD | | | | Applies from SW 6.3 | |
| Meaning: | Number of curve tables in DRAM available NCK-wide | | | | |
| Related to .... | MD 18408, MD 18410 | | | | |
| Further references | PGA | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **18408**<br>MD number | **MM_NUM_CURVE_SEGMENTS_DRAM**<br>Number of curve segment (DRAM) | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: plus | |
| Changes effective after power ON | | | Protection level: 1/1 | | Unit: – |
| Data type: DWORD | | | | Applies from SW 6.3 | |
| Meaning: | Number of segments for curve tables in DRAM available NCK-wide | | | | |
| Related to .... | MD 18406, MD 18410 | | | | |
| Further references | PGA | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **18410**<br>MD number | **MM_NUM_CURVE_POLYNOMS_DRAM**<br>Number of curve table polynomials (DRAM) | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: plus | |
| Changes effective after power ON | | | Protection level: 1/1 | | Unit: – |
| Data type: DWORD | | | | Applies from SW 6.3 | |
| Meaning: | Number or polynomials for curve tables in DRAM available for NCK-wide | | | | |
| Related to .... | MD 18408, MD 18406 | | | | |
| Further references | PGA | | | | |

## 4.2 Channel-specific machine data

| | | | | | |
|---|---|---|---|---|---|
| **20900**<br>MD number | **CTAB_ENABLE_NO_LEADMOTION**<br>Curve tables with jump of following axis | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 2 | |
| Changes effective after RESET | | | Protection level: 2/7 | | Unit: – |
| Data type: BYTE | | | | Applies from SW 6.3 | |
| Meaning: | This MD configures how jumps of the following axis are processed in the curve tables. A jump of the following axis arises when a movement of the following axis is present in a segment of the curve table but there is on movement of the leading axis. Such jumps of the following axis can either be programmed directly, or are produced internally in the control.<br>Such segments can be produced when a curve table is generated with active tool radius offset.<br><br>The following configuration options are available:<br><br>0: No curve tables are produced that contain a jump of the following axis. If a jump of the following axis occurs, Alarm 10949 (CTAB_NO_LEADMOTION) is output and the program processing is aborted. This setting is compatible with older software versions.<br><br>1: Curve tables that contain a jump of the following axis can be created. If a jump of the following axis occurs, Alarm 10955 (CTAB_NO_LEADMOTIONWARNING) is issued without aborting the program processing.<br><br>2: Curve tables with jumps of the following axis are created, without an alarm or message being issued. | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **20905**<br>MD number | **CTAB_DEFAULT_MEMORY_TYPE**<br>Default memory type for curve tables | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 1 | |
| Changes effective after RESET | | | Protection level: 2/7 | | Unit: – |
| Data type: BYTE | | | | Applies from SW 6.3 | |
| Meaning: | This MD sets the default memory type for curve tables:<br>0: Curve tables are created in buffered memory (SRAM).<br>1: Curve tables are created in dynamic memory (DRAM). | | | | |
| Related to .... | MD 18400, 18402, 18404, 18406, 18408, 18410 | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **21380**<br>MD number | **ESR_DELAY_TIME1**<br>Delay time (STOPBYALARM, NOREAD) for ESR axes | | | | |
| Default setting: 0 | | Min. input limit: – | | Max. input limit: plus | |
| Changes effective after NEW CONF | | | Protection level: 2 / 7 | | Unit: c |
| Data type: DOUBLE | | | | Applies from SW 6 | |
| Meaning: | If an alarm occurs, for example, this MD can be used to delay the braking time to enable retraction from the tooth gap in the case of gear hobbing, for example. | | | | |
| Application example(s) | See Section 6.3.2 | | | | |
| Related to .... | ESR_DELAY_TIME2 | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **21381**<br>MD number | **ESR_DELAY_TIME2**<br>Time for interpolatory braking for ESR axes | | | | |
| Default setting: 0 | | Min. input limit: – | | Max. input limit: plus | |
| Changes effective after NEW CONF | | | Protection level: 2 / 7 | | Unit: c |
| Data type: DOUBLE | | | Applies from SW 6 | | |
| Meaning: | When the time $MC_ESR_DELAY_TIME1 expires, the time specified here for interpolatory braking ($MC_ESR_DELAY_TIME2) still applies.<br>Diagnose.awl.<br>When the time $MC_ESR_DELAY_TIME2 expires, rapid deceleration with subsequent follow-up is initiated. | | | | |
| Application example(s) | See Section 6.3.2 | | | | |
| Related to .... | ESR_DELAY_TIME1 | | | | |

# 4.3 Axis-specific machine data

| | | | | |
|---|---|---|---|---|
| **30132**<br>MD number | **IS_VIRTUAL_AX**<br>Axis is virtual axis | | | |
| Default setting: 0 | | Min. input limit: *** | | Max. input limit: *** |
| Changes effective after NEW CONF | | | Protection level: 2/7 | Unit: - |
| Data type: BOOLEAN | | | Applies from SW 4.1 | |
| Meaning: | Virtual axis. An axis that is interpolated in follow-up mode, too. (Technology electronic transfer, virtual and real master value.)<br>This MD is equivalent to MD 30130: CTRLOUT_TYPE=4. Instead of<br>MD 30130: CTRLOUT_TYPE=4, MD 30130: CTRLOUT_TYPE=0 and IS_VIRTUAL_AX=1 should be set. | | | |
| Related to .... | MD 30130: CTRLOUT_TYPE | | | |

| | | | | |
|---|---|---|---|---|
| **37500**<br>MD number | **ESR_REACTION**<br>Reaction definition with extended stop and retract | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 22 |
| Changes effective after NEW CONF | | | Protection level: 2/7 | Unit: - |
| Data type: BYTE | | | Applies from SW 5.1 or 6 | |
| Meaning: | Selection of the reaction to be triggered via system variable “$AN_ESR_TRIGGER/ $AC_ESR_TRIGGER”:<br>0 = no reaction (or exclusively external reaction through synchronous action programming of fast digital outputs).<br>Drive MD MD 1636: RETRACT_AND_GENERATOR_MODE is set to **0**.<br>10 = independent drive generator mode<br>Drive MD 1636: RETRACT_AND_GENERATOR_MODE is set to **6**.<br>11 = independent drive retraction axis<br>Drive MD 1636: RETRACT_AND_GENERATOR_MODE is set to **4**.<br>12 = independent drive stop axis<br>(11 and 12 are activated together in the drive – as is the case with communication failure – by broadcast to all drives)<br>1636 RETRACT_AND_GENERATOR_MODE is set to **2**.<br>Selection of reaction to be triggered via system variable “$AN_ESR_TRIGGER”:<br>21 = NC-prompted retraction axis (SW 6 and higher)<br>Drive MD 1636: RETRACT_AND_GENERATOR_MODE is set to **5**.<br>22 = NC-prompted standstill axis (SW 6 and higher)<br>All axes involved in IPO or EG are brought to smooth stop even without this parameter setting. This parameter is used to configure the relevant independent drive reaction for communication failure or DC link undervoltage:<br>(21 and 22 contain independent drive standstill and retraction exclusively for communication failure or DC link undervoltage).<br>Drive MD 1636: RETRACT_AND_GENERATOR_MODE is set to **3**.<br>If the option “Extended stop and retract” (ESR) is not enabled, the values are reset to 0. | | | |
| Related to .... | | | | |

| | | | | |
|---|---|---|---|---|
| **37550**<br>MD number | **EG_VEL_WARNING**<br>Warning threshold for interface signals | | | |
| Default setting: 90 | | Min. input limit: 0 | | Max. input limit: 100 |
| Changes effective after NEW CONF | | | Protection level: 2 / 4 | Unit: % |
| Data type: REAL | | | Applies from SW 5 | |
| Meaning: | Threshold value for VDI signals<br>A velocity warning (signal) is output if an EG axis coupling is active and the current **velocity** of the axis has reached the percentage set here of the maximum velocities set in MD 32000: $MA_MAX_AX_VELO.<br>An acceleration warning (signal) is output if an EG axis coupling is active and the current **acceleration** of the axis has reached the percentage set here of the maximum acceleration rates set in MD 32300: $MA_MAX_AX_ACCEL. | | | |
| Related to .... | MD 32000: $MA_MAX_AX_VELO,<br>MD 32300: $MA_MAX_AX_ACCEL | | | |

| | | | | |
|---|---|---|---|---|
| **37560**<br>MD number | **EG_ACC_TOL**<br>Threshold value for VDI signal | | | |
| Default setting: 25 | | Min. input limit: 0 | | Max. input limit: – |
| Changes effective after NEW CONF | | | Protection level: 2/4 | Unit: % |
| Data type: REAL | | | Applies from SW 5 | |
| Meaning: | Threshold value for VDI signal “**Axis accelerating**”<br>This signal is set when the acceleration rate reaches the specified percentage of maximum acceleration. | | | |
| Related to .... | MD 32300: $MA_MAX_AX_ACCEL | | | |

# 4.4 Axis-specific setting data

| | | |
|---|---|---|
| **43100**<br>SD number | **LEAD_TYPE**<br>Master value type | |
| Default setting: 1 | Min. input limit: 0 | Max. input limit: 2 |
| Modification valid IMMEDIATELY | Protection level: 7/7 | Unit: – |
| Data type: DWORD | Applies from SW 4.1 | |
| Meaning: | Defines which value is to be used as master value:<br>0: Actual value<br>1: Setpoint<br>2: Simulated master value | |

| | | |
|---|---|---|
| **43102**<br>SD number | **LEAD_OFFSET_IN_POS**<br>Master value offset | |
| Default setting: 0 | Min. input limit: | Max. input limit: |
| Modification valid IMMEDIATELY | Protection level: 7/7 | Unit: – |
| Data type: DOUBLE | Applies from SW 4.1 | |
| Meaning: | Offset of master value before use for coupling. | |
| Related to .... | SD 43104: LEAD_SCALE_IN_POS<br>SD 43106: LEAD_OFFSET_OUT_POS<br>SD 43108: LEAD_SCALE_OUT_POS | |

| | | |
|---|---|---|
| **43104**<br>SD number | **LEAD_SCALE_IN_POS**<br>Master value scaling | |
| Default setting: 1 | Min. input limit: | Max. input limit: |
| Modification valid IMMEDIATELY | Protection level: 7/7 | Unit: – |
| Data type: DOUBLE | Applies from SW 4.1 | |
| Meaning: | Scaling of master value before use for coupling. | |
| Related to .... | SD 43102: LEAD_OFFSET_IN_POS<br>SD 43106: LEAD_OFFSET_OUT_POS<br>SD 43108: LEAD_SCALE_OUT_POS | |

| | | |
|---|---|---|
| **43106**<br>SD number | **LEAD_OFFSET_OUT_POS**<br>Curve table offset | |
| Default setting: 0 | Min. input limit: | Max. input limit: |
| Modification valid IMMEDIATELY | Protection level: 7/7 | Unit: POSN |
| Data type: DOUBLE | Applies from SW 4.1 | |
| Meaning: | Offset of curve table before use for coupling. | |
| Related to .... | SD 43102: LEAD_OFFSET_IN_POS<br>SD 43104: LEAD_SCALE_IN_POS<br>SD 43108: LEAD_SCALE_OUT_POS | |

| | | |
|---|---|---|
| **43108**<br>SD number | **LEAD_SCALE_OUT_POS**<br>Curve table scaling | |
| Default setting: 1 | Min. input limit: | Max. input limit: |
| Modification valid IMMEDIATELY | Protection level: 7/7 | Unit: |
| Data type: DOUBLE | Applies from SW 4.1 | |
| Meaning: | Scaling of function value of curve table. | |
| Related to .... | SD 43102: LEAD_OFFSET_IN_POS<br>SD 43104: LEAD_SCALE_IN_POS<br>SD 43106: LEAD_OFFSET_OUT_POS | |

## 4.5 System variables

| **Name** | **$AC_ALARM_STAT** | | | |
|---|---|---|---|---|
| Meaning | Contents not equal to 0: Alarms are present, the coded associated alarm reactions can be used as a source for “Extended stop and retract”.<br>The data is bit-coded, thus if needed, it is also possible to mask or evaluate individual states separately (bits that are not listed are reserved internally and always return “0” externally):<br>Bit 2=1: NOREADY (active rapid deceleration + cancellation of servo enable)<br>Bit6 = 1: STOPBYALARM (ramp stop for all channel axes)<br>Bit9 = 1: SETVDI (VDI interface alarm is set)<br>Bit13=1: FOLLOWUPBYALARM (follow-up) | | | |
| Data type | INT | | | |
| Access | Read in parts program | Write in parts program | Read in synchronous action | Write in synchronous action |
| | X | | X | |
| Implicit preprocessing stop | X | | | |

| **Name** | **$AA_ESR_STAT[axis]** | | | |
|---|---|---|---|---|
| Meaning | (Axial) status feedback of “Extended stop and retract” which can serve as input signals for the ESR logic operations (synchronous actions). The data is bit-coded, thus if needed, it is also possible to mask or evaluate individual states separately: Drive machine data:<br>Bit 0=1: Generator operation is triggered (MD 1631: LINK_VOLTAGE_GEN_ON)<br>Bit 1=1: Retraction is triggered (MD 1634: LINK_VOLTAGE_RETRACT)<br>Bit 2=1: Stop is triggered (MD 1634: LINK_VOLTAGE_RETRACT)<br>Bit 3=1: Potential undervoltage (DC link voltage monitoring, value less than warning threshold) (MD 1634: LINK_VOLTAGE_RETRACT)<br>Bit 4=1: Value below minimum generator speed threshold (that is, no more recovery rotation energy available) (MD 1635: GEN_AXIS_MIN_SPEED). | | | |
| Data type | INT | | | |
| Access | Read in parts program | Write in parts program | Read in synchronous action | Write in synchronous action |
| | X | | X | |
| Implicit preprocessing stop | X | | | |

| **Name** | **$AA_ESR_ENABLE[axis]** | | | |
|---|---|---|---|---|
| Meaning | 1 = (axial) enable of reaction(s) of “Extended stop and retract”. The required axial ESR reaction must have been parameterized in MD37500 ESR_REACTION. Corresponding stop/retract reactions can be triggered by means of the system variable $AN_ESR_TRIGGER (or independent drive with communication failure/DC link undervoltage), generator operation is automatically activated if voltage underflow occurs. | | | |
| Data type | BOOL | | | |
| Access | Read in parts program | Write in parts program | Read in synchronous action | Write in synchronous action |
| | X | X | X | X |
| Implicit preprocessing stop | X | X | | |

| **Name** | **$AN_ESR_TRIGGER** | | | |
|---|---|---|---|---|
| Meaning | (Global) control signal “Start stop/retract”. On signal transition from 0 to 1, the independent drive reactions parameterized in the axial MD 37500: ESR_REACTION and enabled via axial system variable $AA_ESR_ENABLE are started. **Independent drive** reactions require a subsequent power OFF/power ON or NCK RESET. | | | |
| Data type | BOOL | | | |
| Access | Read in parts program | Write in parts program | Read in synchronous action | Write in synchronous action |
| | | | X | X |
| Implicit preprocessing stop | | | | |

| **Name** | **$AC_ESR_TRIGGER** | | | |
|---|---|---|---|---|
| Meaning | Channel-specific control signal “Start stop/retract” On signal transition from 0 to 1 the reactions parameterized in the axial MD37500 ESR_REACTION and enabled via axial system variable $AA_ESR_ENABLE are started for **NC-prompted** stops and retract. | | | |
| Data type | BOOL | | | |
| Access | Read in parts program | Write in parts program | Read in synchronous action | Write in synchronous action |
| | | | X | X |
| Implicit preprocessing stop | | | | |

**For following axis dynamic response**

| Name | $PA_ACCLIMA[n] | | | |
|---|---|---|---|---|
| Meaning | Percentage (%) acceleration offset in run-in set with ACCLIMA[n] | | | |
| Data type | REAL | | | |
| Range of values | 1 – 200 | | | |
| Indexes | Meaning: Geo axis, channel axis, machine axis, spindle | | | Range of values |
| | | | | |
| Access | Read in parts program | Write in parts program | Read in synchronous action | Write in synchronous action |
| | X | | X | |
| Implicit preprocessing stop | | | | |

| Name | $AA_ACCLIMA[n] | | | |
|---|---|---|---|---|
| Meaning | Percentage (%) acceleration offset set in main run with ACCLIMA[n] | | | |
| Data type | REAL | | | |
| Range of values | 1 – 200 | | | |
| Indexes | Meaning: Geo axis, channel axis, machine axis, spindle | | | Range of values |
| | | | | |
| Access | Read in parts program | Write in parts program | Read in synchronous action | Write in synchronous action |
| | X | | X | |
| Implicit preprocessing stop | X | | | |

| Name | $PA_VELOLIMA[n] | | | |
|---|---|---|---|---|
| Meaning | Percentage (%) velocity offset set in run-in with VELOLIMA[n] | | | |
| Data type | REAL | | | |
| Range of values | 1 – 200 | | | |
| Indexes | Meaning: Geo axis, channel axis, machine axis, spindle | | | Range of values |
| | | | | |
| Access | Read in parts program | Write in parts program | Read in synchronous action | Write in synchronous action |
| | X | | X | |
| Implicit preprocessing stop | X | | | |

| Name | $AA_VELOLIMA[n,m] | | | |
|---|---|---|---|---|
| Meaning | Percentage (%) velocity offset set in main run with VELOLIMA[n] | | | |
| Data type | REAL | | | |
| Range of values | 1 – 200 | | | |
| Indexes | Meaning: Geo axis, channel axis, machine axis, spindle | | | Range of values |
| | | | | |
| Access | Read in parts program | Write in parts program | Read in synchronous action | Write in synchronous action |
| | X | | X | |
| Implicit preprocessing stop | X | | | |

| Name | $PA_JERKLIMA[n] | | | |
|---|---|---|---|---|
| Meaning | Percentage (%) jerk offset set in run-in with JERKLIMA[n] | | | |
| Data type | REAL | | | |
| Range of values | 0 – 200 | | | |
| Indexes | Meaning: Geo axis, channel axis, machine axis, spindle | | | Range of values |
| | | | | |
| Access | Read in parts program | Write in parts program | Read in synchronous action | Write in synchronous action |
| | X | | X | |
| Implicit preprocessing stop | | | | |

| Name | $AA_JERKLIMA[n] | | | |
|---|---|---|---|---|
| Meaning | Percentage (%) jerk offset set in main run with JERKLIMA[n] | | | |
| Data type | REAL | | | |
| Range of values | 0 – 200 | | | |
| Indexes | Meaning: Geo axis, channel axis, machine axis, spindle | | | Range of values |
| | | | | |
| Access | Read in parts program | Write in parts program | Read in synchronous action | Write in synchronous action |
| | X | | X | |
| Implicit preprocessing stop | X | | | |

■

## Notes

# Signal Descriptions 5

## 5.1 Signals from axis/spindle

| DB 31 – DBB 26.4 Data block | Active following axis overlay<br><br>Signal(s) from NC (PLC → NCK) | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW 5.1 |
| Signal state 1 or signal transition 0 ––> 1 | The following axis can overlay an additional traversing motion.<br><br>This signal is required for flying from leading to following axes.<br>As long as the signal “Enable following axis overlay” is set to 1, the following axis selected with EGONSYN in the EG coupling group travels to synchronization.<br>Modulo axes included in the EG coupling reduce their position values in the modulo, thereby ensuring that they approach the next possible synchronization. | |
| Signal state 0 or signal transition 1 ––> 0 | The following axis cannot be overlaid and traversed.<br><br>If “Enable following axis overlay” interface signal DB(30 +axis number), DBX 26, Bit 4 is not specified for the following axis, synchronization is not approached. Instead the program is stopped at the EGONSYN block and the self-inititating alarm 16771 is issued until the signal “Enable following axis overlay” is set to 1. | |

| DB 31 – DBB 99.3 Data block | Axis accelerated<br><br>Signal(s) from NC (NC → PLC) | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW 5.1 |
| Signal state 1 or signal transition 0 ––> 1 | If the following axis acceleration in the axis grouping of the electronic gear reaches or exceeds the % of acceleration contained in MD 37560: EG_ACC_TOL which is set in MD 32300: MAX_AX_ACCEL, then the signal is set to 1. | |
| Signal state 0 or signal transition 1 ––> 0 | The following axis acceleration in the axis grouping of the electronic gear is less than the operating value described above. | |
| Signal irrelevant | without electronic gear | |
| Related to .... | MD 37560, 32300 | |

| DB 31 – DBB 98.5 Data block | Velocity warning threshold<br><br>Signal(s) from NC (NC → PLC) | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW 5.1 |
| Signal state 1 or signal transition 0 ––> 1 | If the following axis velocity in the axis grouping of the electronic gear reaches or exceeds the % of velocity contained in MD 37550: EG_VEL_WARNING which is set in MD 32000: MAX_AX_VELO, then the signal is set to 1. | |
| Signal state 0 or signal transition 1 ––> 0 | The following axis velocity in the axis grouping of the electronic gear is less than the threshold value described above. | |
| Signal irrelevant | without electronic gear | |
| Related to .... | MD 37550, 32000 | |

| | | | |
|---|---|---|---|
| **DB 31 –<br>DBB 98.6**<br>Data block | **Acceleration warning threshold**<br><br>Signal(s) from NC (NC → PLC) | | |
| Edge evaluation: no | | Signal(s) updated: Cyclically | Signal(s) valid from SW 5.1 |
| Signal state 1 or signal transition 0 ——> 1 | If the following axis acceleration in the axis grouping of the electronic gear reaches or exceeds the % of acceleration contained in MD 37550: EG_VEL_WARNING which is set in MD 32300: MAX_AX_ACCEL, then the signal is set to 1. | | |
| Signal state 0 or signal transition 1 ——> 0 | The following axis acceleration in the axis grouping of the electronic gear is less than the threshold value described above. | | |
| Signal irrelevant | without electronic gear | | |
| Related to .... | MD 37550, 32300 | | |

| | | | |
|---|---|---|---|
| **DB 31 –<br>DBX 7**<br>Data block | **ESR reaction is triggered**<br><br>Signal(s) from NC (NC → PLC) | | |
| Edge evaluation: no | | Signal(s) updated: Cyclically | Signal(s) valid from SW 5.1 |
| Signal state 1 or signal transition 0 ——> 1 | Status signal<br>The VDI signal “ESR reaction is triggered” is available as return signal to the PLC.<br>The signal is set if $AA_ESR_STAT > 0, i.e. when<br>generator mode, standstill or retraction are active,<br>DC link undervoltage detected or<br>generator minimum speed not reached. | | |
| Signal state 0 or signal transition 1 ——> 0 | ESR is not active. | | |
| Application example(s) | For safety reasons, EMERGENCY STOP interrupts the interpolation and all traversing movements, and also dissolves the electronic coupling by canceling the servo enables. In applications where the coupling and traversing movements must remain valid after EMERGENCY STOP, this EMERGENCY STOP must be delayed long enough by the PLC for the required NC or drive-end reactions to terminate.<br>Writing in $A_DBB allows the PLC to extensively influence the execution of the ESR reactions, if appropriate access is also integrated into the synchronous actions. With 840, the PLC has a locking influence on the ESR response. With 840D, it is possible to link the relevant synchronized actions to produce the desired logic. | | |

# Examples 6

## 6.1 Curve tables

**Example**

Definition of a periodic curve table with table number 2
Master value range 0–360, the following axis moves from 0 to 45 and back again to 0 between N70 and N90.

```
N10 DEF REAL DEPPOS;
N20 DEF REAL GRADIENT;
N30 CTABDEF( Y, X, 2, 1)
N40 G1 X=0 Y=0
N50 POLY
N60 PO[X]=(45.0)
N70 PO[X]=(90.0) PO[Y]=(45.0, 135.0, –90)
N80 PO[X]=(270.0)
N90 PO[X]=(315.0) PO[Y]=(0.0, –135.0, 90)
N100 PO[X]=(360.0)
N110 CTABEND

N130 G1 F1000 X0            ; Test the curve by coupling Y to X
N140 LEADON(Y,X,2)
N150 X360
N160 X0
N170 LEADOF(Y,X)

N180 DEPPOS = CTAB(75.0, 2, GRADIENT)  ; Read the table position for
master value 75.0 from the curve table numbered 2

N190 G0 X75 Y=DEPPOS   ; Position the master and following axis

N200 LEADON(Y,X,2)          ; After activation of the coupling, it is not
                            ; necessary to synchronize the following axis
N210 G1 X110 F1000
N220 LEADOF(Y,X)
N190 M30
```

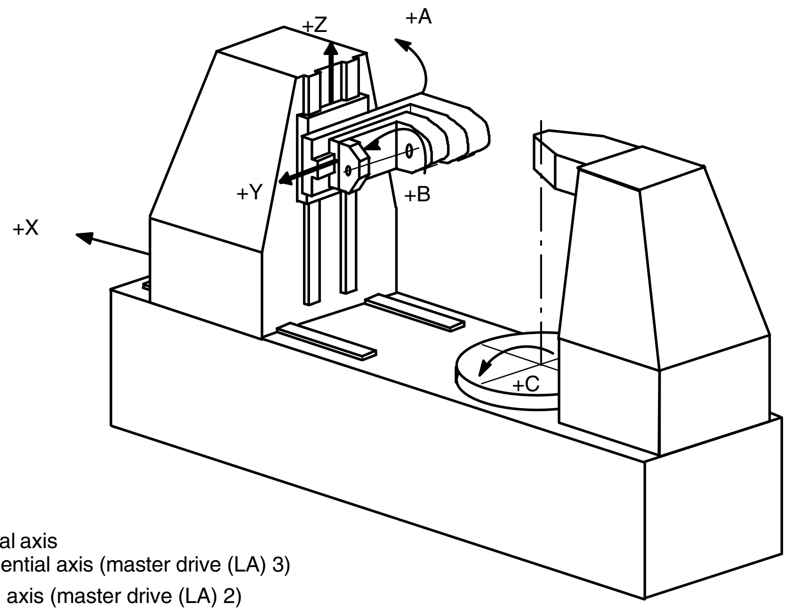
## 6.2 Electronic gear for gear hobbing

### 6.2.1 Example: (linear coupling SW 5)

**Use of axes**

The following diagram shows the configuration of a typical gear hobbing machine. The machine comprises five numerically closed-loop-controlled axes and an open-loop-controlled main spindle. The individual axes are as follows:

- The rotary motion of the workpiece table (C) and hobbing cutter (B).
- The axial axis (Z) for producing the feed motion over the entire workpiece width.
- The tangential axis (Y) for moving the hobbing cutter along its axis.
- The radial axis (X) for infeeding the cutter to depth of tooth.
- The cutter swivel axis (A) for setting the hobbing cutter in relation to the workpiece as a function of cutter lead angle and angle of inclination of tooth.




X = Radial axis
Y = Tangential axis (master drive (LA) 3)
Z = Axial axis (master drive (LA) 2)
A = Cutter swivel axis
C = Workpiece rotary axis (slave drive (FA))
B = Cutter rotary axis, main spindle (master drive (LA) 1)

Fig. 6-1 Definition of axes on a gear hobbing machine (example)

The functional interrelationships on the gear hobbing machine are as follows:

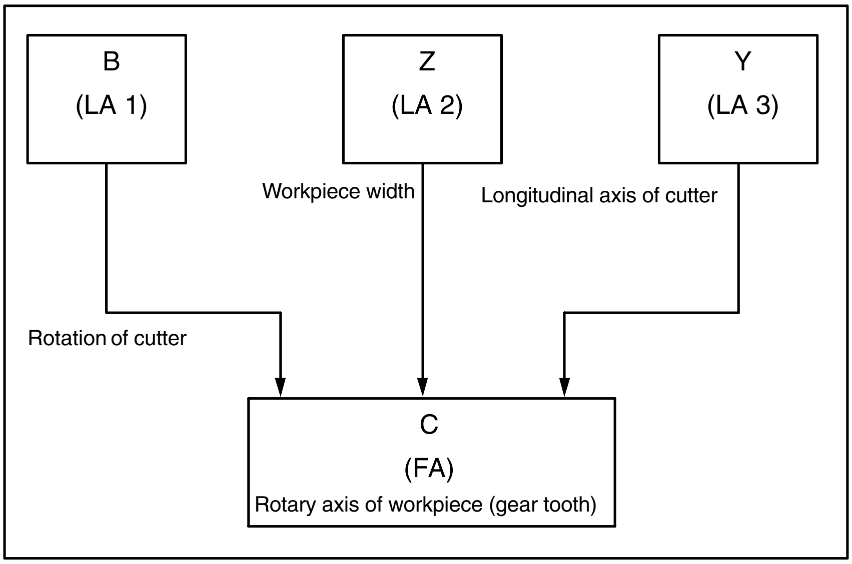



Fig. 6-2

In this case, the workpiece table axis (C) is the following axis which, in this example, is influenced by three master drives.

The setpoint of the following axis is calculated cyclically with the following logic equation:

$$n_c = n_b * \frac{z_0}{z_2} + v_z * \frac{u_{dz}}{z_2} + v_y * \frac{u_{dy}}{z_2}$$

| | |
|---|---|
| $n_c$ | Speed of workpiece axis (C) |
| $n_b$ | Speed of cutter spindle (B) |
| $z_0$ | Number of starts of hobbing cutter |
| $z_2$ | Number of teeth of workpiece |
| $v_z$ | Feed velocity of axial axis (Z) |
| $v_y$ | Feed velocity of tangential axis (Y) |
| $u_{dz}$ | Axial differential constant |
| $u_{dy}$ | Tangential differential constant |

**Quantities which influence the setpoint of workpiece axis C**

The first addend of the above equation determines the speed ratio between workpiece table and cutter, and thus the number of teeth of the workpiece.

The second addend effects the necessary additional rotation of the C axis as a function of the axial feed motion of the cutter to produce the tooth inclination on helical teeth.

The third component also makes allowance for additional rotation of the C axis to compensate for the tangential movement of the cutter in relation to the workpiece, thus ensuring that the tool is equally stressed over its entire length.

**Workpiece/ tool parameters**

The values $z_0$, $z_2$, $u_{dz}$ and $u_{dy}$ are determined by the workpiece or tool and are thus specified by the NC operator or parts program.

**Differential constants**

Differential constants $u_{dz}$ and $u_{dy}$ make allowance for the angle of the workpiece teeth and for the cutter geometry. These differential constants can be determined in user-specific cycles.

$$u_{dz} = \frac{\sin\beta^\circ}{m_n \cdot \pi} \cdot 360 \quad \left[\frac{\text{degrees}}{\text{mm}}\right]$$

$$u_{dy} = \frac{\cos\gamma^\circ}{m_n \cdot \pi} \cdot 360 \quad \left[\frac{\text{degrees}}{\text{mm}}\right]$$

where:

| | | |
|---|---|---|
| $m_n$ | = | Normal module (in mm) |
| $\beta^\circ$ | = | Angle of inclination of gear wheel |
| $\gamma^\circ$ | = | Lead angle of hobbing cutter. |

**Extract from part program**

; Definition of EG axis grouping with
; Setpoint coupling (1) of B, Z, Y to C (following axis)

EGDEF(C, B, 1, Z, 1, Y, 1)

; Activate coupling

EGON(C, “FINE”, B, $z_0$, $z_2$, Z, $u_{dz}$, $z_2$, Y, $u_{dy}$, $z_2$ )

## 6.2.2 Extended example with non-linear components (SW 6 and higher)

**Introduction**

The following example expands the example in Fig.6-1 to include

– Machine error compensations which are not linearly dependent on the Z axis, and

– a component of the gear geometry that depends on the Z axis.This can be used to produce a slightly ball-shaped a tooth surface in the center of the gear so that the load on the center of the tooth is greater than at the edges during operation.

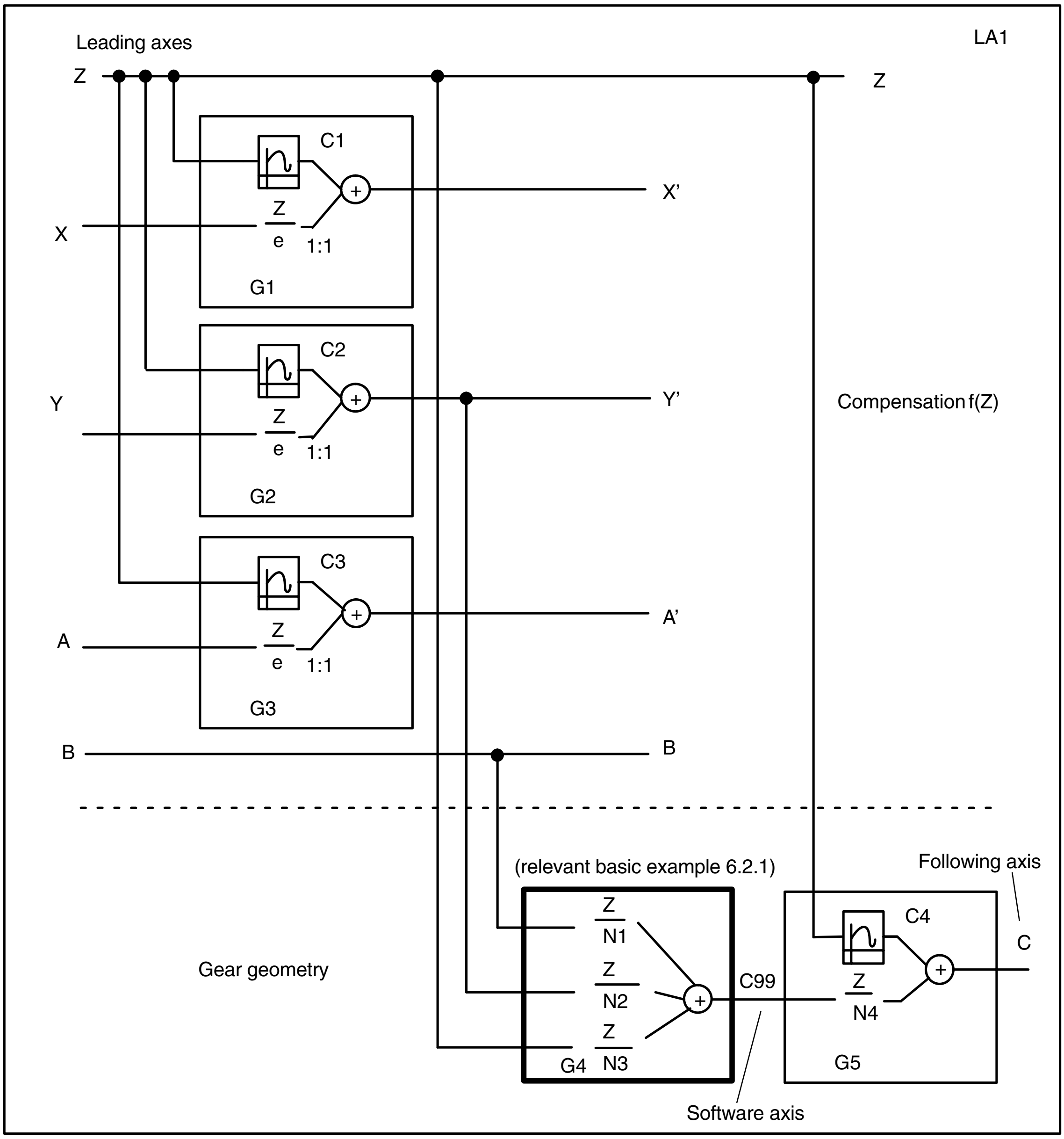



Fig. 6-3 Extended example with non-linear machine fault compensation and non-linear components on the tooth geometry

The following section of a parts program is intended to illustrate the general concept; supplementary curve tables and gear wheel/machine parameters are still to be added. Components to be added are marked < ... >. Stated parameters may also have to be modified, e.g. coupling factors.

```
N100    CTABDEF(X, Z, 1, 0)    ; Declaration and specification of non-periodic curve table C1
N110 < ... >                   ; Default from curve table: curve points or polynomial blocks
N190    CTABEND

N200    CTABDEF(Y, Z, 2, 0)    ; Declaration and specification of non-periodic curve table C2
N210 < ... >                   ; Default from curve table: curve points or polynomial blocks
N290    CTABEND

N300    CTABDEF(A, Z, 3, 0)    ; Declaration and specification of non-periodic curve table C3
N310 < ... >                   ; Default from curve table: curve points or polynomial blocks
N390    CTABEND

N400    CTABDEF(C, Z, 4, 0)    ; Declaration and specification of non-periodic curve table C4
N410 < ... >                   ; Default from curve table: curve points or polynomial blocks
N490    CTABEND

N500    EGDEF(X, Z, 1)         ; Path declaration via C1, setpoint coupling
N510    G1 F1000 X10           ; Declaration of command component of X
N520    EGONSYN(X, "NOC", <SynPosX>, Z, <SynPosX_Z>, 1, 0)          ; Activation of path via C1

N600    EGDEF(Y, Z, 1)         ; Path declaration via C2, setpoint coupling
N610    G1 F1000 Y10           ; Declaration of command component of Y
N620    EGONSYN(Y, "COARSE", <SynPosY>, Z, <SynPosY_Z>, 2, 0)       ; Activation of path via C2

N700    EGDEF(A, Z, 1)         ; Path declaration via C3, setpoint coupling
N710    G1 F1000 A10           ; Declaration of command component of A
N720    EGONSYN(A, "FINE", <SynPosA>, Z, <SynPosA_Z>, 3, 0)         ; Activation of path via C3

; 1st Gear stage, C99 is the software axis between the two electronic gears
N800    EGDEF(C99, Y, 1, Z, 1, B, 1 )
N810    EGONSYN(C99, "NOC", <SynPosC99>, B, <SynPosC99_B>, 18, 2, &  ; Activation of leading axis B
         Y, <SynPosC99_Y>, R1 * π, 1, &                              ; Activation of leading axis Y
         Z, <SynPosC99_Z>, 10, 1)                                    ; Activation of leading axis Z
        ; "&" character means: command continued in next line, no LF nor comment permissible in program

; 2nd gear stage
N900    EGDEF(C, C99, 1, Z, 1); Declaration of following axis C99 of Level as leading axis of Level 2,
                               ; Setpoint coupling
N910                           ; Path declaration via C4, setpoint coupling
N920    EGONSYN(C, "NOC", <SynPosC>, C99, <SynPosC_C99>, 1, 1, &     ; Activation of software axis C99
         Z, <SynPosC_Z>, 4, 0)                                       ; and of leading axis Z via C4
N999    M30
```

## Machine data

Only one section is specified, which extends beyond the necessary geometry/channel configuration and machine axis parameters.

```
$MN_NUM_EG = 5                          ; Maximum number of gears
$MN_MM_NUM_CURVE_TABS = 5 ; Maximum number of curve tables
$MN_MM_NUM_CURVE_SEGMENTS = 50 ;Max. number of
                                        ; curve segments
$MN_MM_NUM_CURVE_POLYNOMS = 100 ;Max. number of
                                        ; curve polynomials
```

**Setting data**

If the scaling described in Section 2.4. is used, the following applies for offset:

$SD_LEAD_SCALE_OUT_POS[**4**] = 1.2 ; Scaling for table C**4**

**System variables**

In accordance with the above definitions, the following values are entered in the associated system variables by the control. Access options to these system variables are described in

**References:** /PGA/, Programming Guide "Advanced"

The system variables listed below are only used **for explanatory purposes**!

| | |
|---|---|
| ; *************** Gear X (G1) | |
| $AA_EG_TYPE[X, Z] = 1 | ; Setpoint coupling |
| $AA_EG_NUMERA[X, Z] = 1 | ; Curve table No. = 1 |
| $AA_EG_DENOM[X, Z] = 0 | ; Denominator = 0 →curve table applies |
| $P_EG_BC[X] = "NOC" | ; Block change criterion |
| $AA_EG_NUM_LA[X] = 1 | ; Number of leading axes |
| $AA_EG_AX[0, X] = Z | ; Identifier of leading axis |
| $AA_EG_SYN[X,Z] = <SynPosX_Z> | ; Synchronized position of leading axis Z |
| $AA_EG_SYNFA[X] = <SynPosX> | ; Synchronized position of following axis |
| ; *************** Gear Y (G2) | |
| $AA_EG_TYPE[Y, Z] = 1 | ; Setpoint coupling |
| $AA_EG_NUMERA[Y, Z] = 2 | ; Curve table No. = 2 |
| $AA_EG_DENOM[Y, Z] = 0 | ; Denominator = 0 →curve table applies |
| $P_EG_BC[Y10] = "COARSE" | ; Block change criterion |
| $AA_EG_NUM_LA[Y] = 1 | ; Number of leading axes |
| $AA_EG_AX[0, Y] = Z | ; Identifier of leading axis |
| $AA_EG_SYN[Y, Z] = <SynPosY_Z> | ; Synchronized position of leading axis Z |
| $AA_EG_SYNFA[Y] = <SynPosY> | ; Synchronized position of following axis |
| ; *************** Gear A (G3) | |
| $AA_EG_TYPE[A, Z] = 1 | ; Setpoint coupling |
| $AA_EG_NUMERA[A, Z] = 3 | ; Curve table No. = 3 |
| $AA_EG_DENOM[A, Z] = 0 | ; Denominator = 0 →curve table applies |
| $P_EG_BC[A10] = "FINE" | ; Block change criterion |
| $AA_EG_NUM_LA[A] = 1 | ; Number of leading axes |
| $AA_EG_AX[0, A] = Z | ; Identifier of leading axis |
| $AA_EG_SYN[A, Z] = <SynPosA_Z> | ; Synchronized position of leading axis Z |
| $AA_EG_SYNFA[A] = <SynPosA> | ; Synchronized position of following axis |
| ; *************** Gear C99 (G4) | |
| $AA_EG_TYPE[C99, Y] = 1 | ; Setpoint coupling |
| $AA_EG_NUMERA[C99, Y] = 18 | ; Counter for coupling factor$_y$ |

```
$AA_EG_DENOM[C99, Y] = 2               ; Denominator for coupling factor_y

$AA_EG_TYPE[C99, Z] = 1                ; Setpoint coupling
$AA_EG_NUMERA[C99, Z] = R1 * π         ; Counter for coupling factor_z
$AA_EG_DENOM[C99, Z] = 1               ; Denominator for coupling factor_z

$AA_EG_TYPE[C99, B] = 1                ; Setpoint coupling
$AA_EG_NUMERA[C99, B] = 10             ; Counter for coupling factor_b
$AA_EG_DENOM[C99, B] = 1               ; Denominator for coupling factor_b
$P_EG_BC[C99] = "NOC"                  ; Block change criterion
$AA_EG_NUM_LA[C99] = 3                 ; Number of leading axes
$AA_EG_AX[0, C99] = Y                  ; Identifier of leading axis Y
$AA_EG_AX[1, C99] = Z                  ; Identifier of leading axis Z
$AA_EG_AX[2, C99] = B                  ; Identifier of leading axis B
$AA_EG_SYN[C99, Y] = <SynPosC99_Y>  ; Synchronized position of leading
                                      axis Y
$AA_EG_SYN[C99, Z] = <SynPosC99_Z> ; Synchronized position of leading
                                     axis Z
$AA_EG_SYN[C99, B] = <SynPosC99_B>  ; Synchronized position of leading
                                      axis B
$AA_EG_SYNFA[C99] = <SynPosC99>  ; Synchronized position of slave axis
; *************** Gear C (G5)
$AA_EG_TYPE[C, Z] = 1                  ; Setpoint coupling
$AA_EG_NUMERA[C, Z] = 4                ; Curve table No. = 4
$AA_EG_DENOM[C, Z] = 0                 ; Denominator = 0 →curve table
applies

$AA_EG_TYPE[C, C99] = 1                ; Setpoint coupling
$AA_EG_NUMERA[C, C99] = 1              ; Counter for coupling factor_C99
$AA_EG_DENOM[C, C99] = 1               ; Denominator for coupling factor_C99
$P_EG_BC[C] = "NOC"                    ; Block change criterion
$AA_EG_NUM_LA[C] = 2                   ; Number of leading axes
$AA_EG_AX[0, C] = Z                    ; Identifier of leading axis Z
$AA_EG_AX[1, C] = C99                  ; Identifier of leading axis C99
$AA_EG_SYN[C, Z] = <SynPosC_Z>  ; Synchronized position of leading axis Z
$AA_EG_SYN[C, C99] = <SynPosC_C99> ; Synchronized position of leading
                                   ; axis C99
$AA_EG_SYNFA[C] = <SynPosC>  ; Synchronized position of leading axis C
```

**Machine data**

Except from MD:

```
; *************** Channel 1
CHANDATA (1)
; *************** Axis 1, "X"
$MC_AXCONF_GEOAX_NAME_TAB[0] = "X"
$MC_AXCONF_CHANAX_NAME_TAB[0] = "X"
$MC_AXCONF_MACHAX_USED[0] = 1
$MN_AXCONF_MACHAX_NAME_TAB[0] = "X1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX1] = 0
$MA_IS_ROT_AX[AX1] = FALSE
; *************** Axis 2, "Y"
$MC_AXCONF_GEOAX_NAME_TAB[1] = "Y"
$MC_AXCONF_CHANAX_NAME_TAB[1] = "Y"
$MC_AXCONF_MACHAX_USED[1] = 2
$MN_AXCONF_MACHAX_NAME_TAB[1] = "Y1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX2] = 0
$MA_IS_ROT_AX[AX2] = FALSE
; *************** Axis 3, "Z"
$MC_AXCONF_GEOAX_NAME_TAB[2] = "Z"
$MC_AXCONF_CHANAX_NAME_TAB[2] = "Z"
$MC_AXCONF_MACHAX_USED[2] = 3
$MN_AXCONF_MACHAX_NAME_TAB[2] = "Z1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX3] = 0
$MA_IS_ROT_AX[AX3] = FALSE
; *************** Axis 4, "A"
$MC_AXCONF_CHANAX_NAME_TAB[3] = "A"
$MC_AXCONF_MACHAX_USED[3] = 4
$MN_AXCONF_MACHAX_NAME_TAB[3] = "A1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX4] = 0
$MA_IS_ROT_AX[AX4] = TRUE
$MA_ROT_IS_MODULO[AX4] = TRUE
; *************** Axis 5, "B"
$MC_AXCONF_CHANAX_NAME_TAB[4] = "B"
$MC_AXCONF_MACHAX_USED[4] = 5
$MC_SPIND_DEF_MASTER_SPIND = 1
$MN_AXCONF_MACHAX_NAME_TAB[4] = "B1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX5] = 1
$MA_IS_ROT_AX[AX5] = TRUE
```

```
$MA_ROT_IS_MODULO[AX5] = TRUE
; *************** Axis 6, "C"
$MC_AXCONF_CHANAX_NAME_TAB[5] = "C"
$MC_AXCONF_MACHAX_USED[5] = 6
$MN_AXCONF_MACHAX_NAME_TAB[5] = "C1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX6] = 0
$MA_IS_ROT_AX[AX6] = TRUE
$MA_ROT_IS_MODULO[AX6] = TRUE
; *************** Axis 10, "C99"
$MC_AXCONF_CHANAX_NAME_TAB[9] = "C99"
$MC_AXCONF_MACHAX_USED[9] = 10
$MA_SPIND_ASSIGN_TO_MACHAX[AX10] = 0
$MA_IS_ROT_AX[AX10] = TRUE
$MA_ROT_IS_MODULO[AX10] = TRUE
```

# 6.3 ESR

## 6.3.1 Use of independent drive reaction

**Example configuration**

- Axis A (spindle) must operate as generator drive;
- in the event of an error, axis X must retract by 10mm at maximum speed, and
- axes Y and Z must stop after a 100ms delay to give the retraction axis time to cancel the mechanical coupling.

**Parameterization**

1. Enable options “Ext. stop and retract” and “Static synchronous actions”.
2. Function assignment:
   $MA_ESR_REACTION[X]=11 $MA_ESR_REACTION[Y]=12
   $MA_ESR_REACTION[Z]=12 $MA_ESR_REACTION[A]=10
3. Drive configuration:
   MD1639: RETRACT_SPEED[X]=400000 ; max. speed in HEX format.
   MD1638: RETRACT_TIME[X]=10 ; mm/max. speed in ms
   MD1637: GEN_STOP_DELAY[Y]=100 ; in ms
   MD1637: GEN_STOP_DELAY[Z]=100 ; in ms
   MD1635: GEN_AXIS_MIN_SPEED[A]= 1 ; generator min. speed in
   ; rev/min
4. Function enable (from parts program or synchronous actions) by setting the system variables:
   $AA_ESR_ENABLE[X]=1 $AA_ESR_ENABLE[Y]=1
   $AA_ESR_ENABLE[Z]=1 $AA_ESR_ENABLE[A]=1
5. Accelerate generator drive to “momentum” speed
   (e.g. in spindle operation:
   M03 S1000 ; rotate CW, 1000 rev/min)
6. Formulate trigger condition as static synchronous action(s), e.g.:
   - dependent on intervention of generator axis:
     IDS=01 WHENEVER $AA_ESR_STAT[A]>0 DO $AN_ESR_TRIGGER=1
   - and/or dependent on alarms that trigger follow-up mode (bit13=2000H):
     IDS=02 WHENEVER ($AC_ALARM_STAT B_AND 'H2000')>0
     DO $AN_ESR_TRIGGER=1
   - and dependent on EG synchronization monitoring (if, for example, Y is defined as EG following axis and the maximum permissible synchronization deviation must be 100μm):
     IDS=03 WHENEVER ABS($VA_EG_SYNCDIFF[Y])>0.1
     DO $AN_ESR_TRIGGER=1
   - or (combination) dependent on all three described trigger conditions + PLC + input:
     IDS=01 WHENEVER ($AA_ESR_STAT[A] > 0) AND
     (($AC_ALARM_STAT B_AND 'H2000')> 0) AND
     (ABS($VA_EG_SYNCDIFF[Y]) > 0.1) OR
     ($A_DBB[0] > 0) OR
     ($A_PBB[0] > 0) DO $AN_ESR_TRIGGER=1

### 6.3.2 NC-prompted reactions

Example using NC-prompted reactions. The important details are specified.

**Task**

The A axis is to operate as the generator drive, while the X axis should retract 10 mm at maximum speed in the event of a fault, and axes Y and Z should stop after a delay of 100 ms so that the retraction axis has time to cancel the mechanical coupling.

**Preconditions**

The options ""Extended stop and retract", "Static synchronized actions" and "ASUB" must be available.

**Parameterization**

Parameterization or programming required for the example:

```
$MC_ASUB_START_MASK = 7     ; MD 11602

;Function group assignments
$MA_ESR_REACTION[X]=21      ; MD 37500
$MA_ESR_REACTION[Y]=22
$MA_ESR_REACTION[Z]=22
$MA_ESR_REACTION[A]=10

;Drive configuration for independent drive reactions
$MD_RETRACT_SPEED[X]=400000H              ;MD 1639,
                                          ;max. speed
$MD_RETRACT_TIME[X]=10       ; MD 1638, mm/max. retraction time.
$MD_GEN_STOP_DELAY[Y]=100 ; MD 1637, ms delayed shutdown
$MD_GEN_STOP_DELAY[Z]=100 ; MD 1637, ms delayed shutdown
$MD_GEN_AXIS_MIN_SPEED[A]= ...            ; MD 1635, Generator
                                          ; Min. speed (rev/min)

;Configuration of the NC-prompted retraction
LFPOS                        ; Axial retraction to a position
                             ; 46. G group
POLF[X]=IC(10)               ; Target position for retraction
POLFMASK(X)                  ; Enable retraction

;Configuration of the NC-prompted stop
$MC_ESR_DELAY_TIME1=0.1      ; MD 21380, duration of
                             ; Path interpolation in seconds
$MC_ESR_DELAY_TIME2=0.04     ; MD 21381, Braking duration in seconds

; Function enable (from parts program or synchronous actions) by setting the
system variables:
$AA_ESR_ENABLE[X]=1          ; Set system variables
$AA_ESR_ENABLE[Y]=1
$AA_ESR_ENABLE[Z]=1
$AA_ESR_ENABLE[A]=1
```

- Accelerate generator drive to "momentum" speed (e.g. in spindle operation M03 S1000)

**Synchronized actions**

Formulate trigger condition as **static synchronous action**(s), e.g.:

```
; dependent on intervention of generator axis:
IDS=01 WHENEVER $AA_ESR_STAT[A]>0 DO $AC_ESR_TRIGGER=1
```

```
; and/or dependent on alarms that trigger follow-up mode
; (bit13=2000H):
IDS=02 WHENEVER ($AC_ALARM_STAT B_AND 'H2000')>0 DO
$AC_ESR_TRIGGER=1
```

; and dependent on EG synchronization monitoring (if, for example, Y is defined as EG following axis and the maximum permissible synchronization deviation must be 100 μ):

```
IDS=03 WHENEVER $VA_EG_SYNCDIFF[Y]>0.1 DO $AC_ESR_TRIGGER=1
```

## 6.3.3 Fast retraction of an axis on stop thread cutting

**Suppressing path interpolation for an axis**

During thread cutting, the path interpolation of X is suppressed for a stop and a movement at maximum speed to position POLF[X] interpolated instead. The movement of the other axes is still determined by the programmed contour or the thread pitch and the spindle speed.

```
N10 G0 G90 X200 Z0 S200 M3      ;
N20 G0 G90 X170                 ;
N22 POLF[X]=210  LFPOS          ; Lift modes
POLFMASK(X)                     ; Activate lift fast from X axis
N25 G33 X100 I10  LFON          ; Interrupt thread cutting ON
N30 X130 Z–45 K10               ;
N40 X155 Z–128 K10              ;
N50 X145 Z–168 K10              ;
N55 X120 I10                    ;
N60 G0 Z0 LFOF                  ; Interrupt thread cutting OFF
N70 POLFMASK( )                 ; Block retraction for all axes
M30
```

## 6.3.4 Lift fast via a fast input with ASUB

**Activation**

Activation via a fast input with ASUB

```
N10 SETINT (1) PRIO=1 ABHEB_Y LIFTFAST  ; ASUB activation through lift fast
                                ; with fast input 1
N30 LFPOS                       ; Select lift mode
N40 POLF[X]=19.5 POLF[Y]=33.3   ; Program list positions for X and Y
N50 POLF[Z]=100                 ; Program lift positions for Z
N60 X0 Y0 G0
N70 POLFMASK(X, Y)              ; Select retraction for X axis and Y axis
N80 Z100 G1 F1000               ; lift off would position the X axis at 19.5mm
                                ; and the Y axis at 33.3mm
N90 POLFMASK(Z)                 ; Deselect retraction of the X axis and Y axis,
                                ; Select retraction of the Z axis
N100 Y10                        ; Lift off would position the Y axis at 100mm
N110 POLFMASK()                 ; Deselect retraction of Y axis, no axis
                                ; makes a retraction of lift movement
```

### 6.3.5 Lift fast with several axes

Parameterization with several axes and incremental programming

```
N10 SETINT (1) PRIO=1 ABHEB_Y LIFTFAST          ; ASUB activation through lift fast
                                                ; with interrupt input 1
N20 ENABLE(1)                         ; Reactivate interrupt routine 1 required after
                                      ; DISABL(1).
N30 LFPOS                             ; Select lift mode for lift fast
N40 POLF[X1]=IC(3.0) POLF[A1]=–4.0 ; Program lift positions for machine axis X1 and
                                      ; A1
N50 POLF[Z]=100                       ; Program lift positions for Z
N60 X0 Y0 A0 G0
N70 POLFMASK(X1, A1)                  ; Select retraction of X axis and A1 axis
N80 Z100 G1 F1000                     ; Lifting would position the machine axis X1
                                      ; to incremental 3.0mm a the A1 axis
                                      ; to –4.0mm absolute
N82 POLF[X1]=10                       ; Change target position of X1 to 10.0mm
                                      ; absolute
N80 Y0 G1 F1000                       ; Lifting would position the machine axis X1
                                      ; to 10.0mm absolute and the A1 axis to
                                      ; –4.0mm absolute
N90 POLFMASK(Z)                       ; Deselect retraction of X1 and A1 axes,
                                      ; Select retraction of the Z axis
N100 Y10                              ; Lift off would position the Y axis at 100mm
N110 POLFMASK()                       ; Deselect retraction of Y axis, no axis
                                      ; performs a retraction or lift motion.
```

■

# Data Fields, Lists 7

## 7.1 Interface signals

| DB number | Bit, byte | Name | Ref. |
|---|---|---|---|
| **Channel-specific** | | | |
| 21, ... | 0.3 | Activate DRF | H1 |
| **Axis-specific** | | | |
| 31, ... | 0.0–0.7 | Feed override | V1 |
| 31, ... | 1.3 | Axis disable | A2 |
| 31, ... | 2.1 | Controller enable | A2 |
| 31, ... | 4.0–4.2 | Activate handwheel | H1 |
| 31, ... | 4.3 | Feed stop | V1 |
| 31, ... | 26.4 | Enable following axis overlay | |
| 31, ... | 98.0 | Synchronism fine | S3 |
| 31, ... | 98.1 | Synchronism coarse | S3 |
| 31, ... | 98.5 | EG velocity warning threshold | |
| 31, ... | 98.6 | EG acceleration warning threshold | |
| 31, ... | 98.7 | ESR reaction is triggered | |
| 31, ... | 99.3 | EG following axis accelerated | |

## 7.2 Machine data

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| **General ($MN_ ...)** | | | |
| 11660 | NUM_EG | Number of possible electronic gears | |
| 18400 | MM_NUM_CURVE_TABS | Number of curve tables (SRAM) | |
| 18402 | MM_NUM_CURVE_SEGMENTS | Number of curve segments (SRAM) | |
| 18404 | MM_NUM_CURVE_POLYNOMS | Number of curve table polynomials (SRAM) | |
| 18406 | MM_NUM_CURVE_TABS_DRAM | Number of curve tables in DRAM | |
| 18408 | MM_NUM_CURVE_SEGMENTS_DRAM | Number of curve segments in DRAM | |

| | | | |
|---|---|---|---|
| **General ($MN_ ...)** | | | |
| 18410 | MM_NUM_CURVE_POLY-NOMS_DRAM | Number of curve polynomials in DRAM | |
| **Channel-specific ($MC_ ...)** | | | |
| 20110 | RESET_MODE_MASK | Definition of control basic setting after power-up and RE-SET/parts program end | K2 |
| 20112 | START_MODE_MASK | Definition of control basic setting after power-up and RE-SET | K2 |
| 20900 | CTAB_ENABLE_NO_LEADMO-TION | Curve table with jump of following axis | |
| 20905 | CTAB_DE-FAULT_MEMORY_TYPE | Default memory type for curve tables | |
| 21380 | ESR_DELAY_TIME1 | Delay time (STOPBYALARM, NOREAD) for ESR axes | |
| 21381 | ESR_DELAY_TIME2 | Time for interpolatory braking for ESR axis | |
| **Axis-specific ($MA_...)** | | | |
| 30130 | CTRLOUT_TYPE | Output type of setpoint | G2 |
| 30132 | IS_VIRTUAL_AX | Axis is virtual axis | |
| 35040 | SPIND_ACTIVE_AFTER_RESET | Own spindle RESET | S1 |
| 37200 | COUPLE_POS_TOL_COARSE | Threshold value for “synchronism coarse” | S3 |
| 37210 | COUPLE_POS_TOL_FINE | Threshold value for “synchronism fine” | S3 |
| 37500 | ESR_REACTION | Reaction definition with extended stop and retract | |
| 37550 | EG_VEL_WARNING | Warning threshold for interface signals | |
| 37560 | EG_ACC_TOL | Threshold value for VDI signal | |

## 7.3 Setting data

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| **Axis-specific ($SA_ ...)** | | | |
| 43100 | LEAD_TYPE | Definition of master value type | |
| 43102 | LEAD_OFFSET_IN_POS | Master value offset | |
| 43104 | LEAD_SCALE_IN_POS | Master value scaling | |
| 43106 | LEAD_OFFSET_OUT_POS | Curve table offset | |
| 43108 | LEAD_SCALE_OUT_POS | Curve table scaling | |

## 7.4 System variables

| | Identifier | Name | Ref. |
|---|---|---|---|
| | $AC_STAT | Channel state: invalid, in reset, interrupted and active | |
| | $A_IN | Digital input NC | |
| | $A_OUT | Digital output NC | |
| | $A_DBB | Read/write data byte (8Bit) fom/to PLC | |
| | $AC_ALARM_STAT | !=0: Alarms are present, the coded associated alarm reactions can be used as a source for “Extended stop and retract”. | |
| | $AN_ESR_TRIGGER | (Global) control signal “Start stop/retract”<br>Independent drive (from SW 5) | |
| | $AC_ESR_TRIGGER | Channel-specific control signal “Start stop/retract” NC-prompted (from SW 6) | |
| | $AA_ESR_STAT[axis] | (Axial) status feedback signals from “Extended stop and retract” | |
| | $AA_ESR_ENABLE[axis] | 1 = (axial) enable of reaction(s) of “Extended stop and retract”. | |
| | $AA_TYP[axis] | Axis type | |
| **Electronic gear (EG) and leading value coupling** | | | |
| | **Identifier** | **Name** | **Ref.** |
| | $AA_EG_SYNFA | Synchronous position of following axis a (SW 5 and higher) | |
| | $P_EG_BC | Block change criterion for EG activation calls: EGON, EGONSYN. WAITC => immediate synchronous run fine or coarse and setpoint synchronous run. (from SW 6.1) | |
| | $AA_EG_NUMLA | Number of leading axes defined with EGDEF (SW 5 and higher) | |
| | $VA_EG_SYNCDIFF | Difference in synchronous run (SW 5 and higher) | |
| | $AA_EG_AX | Identifier for nth leading axis (SW 6.1 and higher) | |
| | $AA_EG_TYPE | Type of coupling for leading axis b (SW 6.1 and higher) | |
| | $AA_EG_NUMERA | Counter of coupling factor for leading axis b (SW 6.1 and higher) | |
| | $AA_EG_DENOM | Denominator of coupling factor for leading axis b (SW 6.1 and higher) | |
| | $AA_EG_SYN | Synchronous position of leading axis b (SW 6.1 and higher) | |
| | $AA_EG_ACTIVE | Coupling for leading axis b is active, i.e. switched on (SW 6.1 and higher) | |
| | $AA_LEAD_SP | Simulated leading value – position in MCS (SW 4 and higher) | |
| | $AA_LEAD_SV | Simulated leading value – velocity (SW 4 and higher) | |
| | $AA_LEAD_P_TURN | Current leading value – position share that is lost through modulo reduction. (SW 4 and higher) | |
| | $AA_LEAD_P | Current leading value – position (modulo/reduced). (SW 4 and higher) | |
| | $AA_LEAD_V | Current leading value – velocity (SW 4 and higher) | |
| | $AA_SYNC | Coupling status of following axis for leading value coupling. (SW 4 and higher) | |

| | Identifier | Name | Ref. |
|---|---|---|---|
| **Dynamics of following axis** | | | |
| | $PA_ACCLIMA | Acceleration factor set with ACCLIMA in run-in | |
| | $AA_ACCLIMA | Acceleration factor set with ACCLIMA in main run | |
| | $PA_VELOLIMA | Velocity offset set with VELOLIMA in run-in | |
| | $AA_VELOLIMA | Velocity offset set with VELOLIMA in main run | |
| | $PA_JERKLIMA | Jerk offset set with JERKLIMA in run-in | |
| | $AA_JERKLIMA | Jerk offset set with JERKLIMA in main run | |

## 7.5 Alarms

A detailed description of the alarms which may occur is given in
**References:** /DA/, Diagnostics Guide
or in the online help in systems with MMC 101/102/103.

■

# SINUMERIK 840D/840Di/810D Description of Functions Special Functions (Part 3)

# Constant Workpiece Speed for Centerless Grinding (S8)



■

# Notes

# Brief Description 1

**Basic kinematics**

The basic kinematics of machines for “centerless grinding” are as follows:

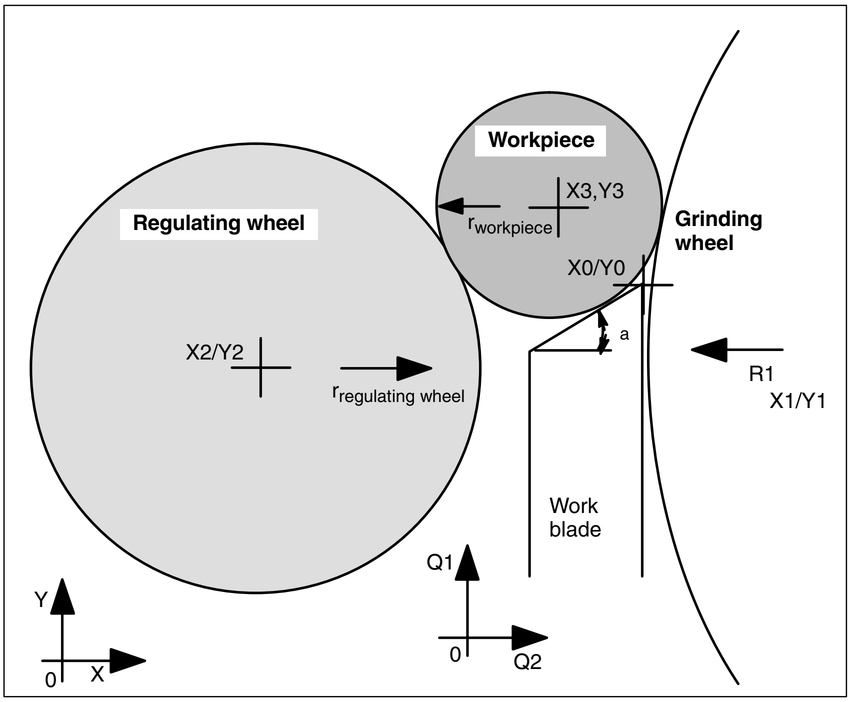



Fig. 1-1 Basic kinematics of machines for “Centerless grinding”

**Definition of workpiece diameter**

The position and size of the workpiece or workpiece to be ground (center point X3, Y3) is defined by a work blade (axes Q1, Q2), a regulating wheel (center point X2, Y2) and a grinding wheel (center point X1, Y1), i.e. the center point and radius of the workpiece are determined by a contact circle delineated by the regulating wheel, grinding wheel and the straight line which in turn is defined by the inclined support surface of the work blade.

**Motion of work blade, regulating wheel and grinding wheel**

The regulating wheel can be moved in the X direction. The grinding wheel is either stationary or it can be moved in the X direction. The work blade can be moved in the X direction and/or the Y direction with axes Q1 and Q2.

### Preconditions

- The axes of the regulating and grinding wheels and of the work blade must be positioned such that the workpiece can be ground from its initial dimension to its final dimension.
- The part must remain in a stable position during the infeed motion, i.e. the support point of the work blade must be within the top third.
- The workpiece speed (in rev/min) must be known.

### What does the function do?

The programmed workpiece speed is the setpoint speed. When the “Constant workpiece speed for centerless grinding” is active, the speed of the workpiece is kept constant during grinding.

| If | then |
|---|---|
| the diameter of the workpiece becomes smaller, | the speed of the regulating wheel is reduced when “Constant workpiece speed for centerless grinding” is active. |

■

# 2 Detailed Description

**Activation/deactivation of function**

The commands for activating/deactivating the “Constant workpiece speed for centerless grinding” function are as follows:

| Command | Meaning |
|---|---|
| CLGON (setpoint speed of workpiece)<br>Centerless Grinding On | Activation of “Constant workpiece speed for centerless grinding” function |
| CLGOF ()<br>Centerless Grinding Off | Deactivation of function |

**Note**

The “Constant workpiece speed for centerless grinding” function can be selected only for grinding tools (type 400–499).

The setpoint speed of the workpiece is programmed in rev/min.

CLGON is effective only if the spindle of the regulating wheel is operating in speed mode. No position actual value encoder is required.

G functions of the 15th group (G94, G95, ...) can be active simultaneously. However, they have no effect on the spindle of the regulating wheel. If the regulating wheel spindle is operating as a main spindle, G96 and GLGON are mutually exclusive.

**Calculating the speed of the regulating wheel**

The speed of the regulating wheel is calculated from the setpoint speed of the workpiece as follows:

$$S_{\text{regulating wheel}} = \frac{r_{\text{workpiece}}}{r_{\text{regulating wheel}}} \cdot S_{\text{programmed}} \qquad \text{S in rev/min}$$

$r_{\text{workpiece}}$ is calculated as the radius of the contact circle defined by the grinding wheel, regulating wheel and work blade.

The following data are required to calculate the speed of the regulating wheel:

- The programmed position of the axes
- The tool data of the grinding and regulating wheels
- The geometry data (stored in the machine data)

**Note**

The radii of the grinding and regulating wheels are derived from the current compensation data for T1, D1 (grinding wheel) and T2, D1 (regulating wheel).

Changes to the online tool offset are taken into account.

The spindle number of the regulating and grinding wheels as well as the geometry-defining parameters (axis numbers, direction vector of work blade, ...) are stored in machine data.

**Response on transition from motional blocks with and without G0**

When activated, the “constant workpiece speed for centerless grinding” function is active for motional blocks without G0 only when MD: TRACLG_G0_IS_SPECIAL = “1”.

Table 2-1 Response on transition from motional blocks with and without G0

| If ... | then ... |
|---|---|
| a transition from a motional block with G0 to a block without G0 takes place, | the speed of the regulating wheel is set during the G0 block to the desired initial speed in the block without G0. |
| a block with active function is followed by a motional block with G0, | the speed of the regulating wheel is frozen at the block end before G0<br>unless<br>this is followed by a motional block without G0 so that the spindle is accelerated up to the new setpoint speed during the G0 block. |

The valid values at the instant of preparation are used in the calculation of corrections which are effective in online operation.

**Gear stages**

The user must select gear stages appropriately to ensure that the regulating wheel can sweep the required speed range. The gear stage limits are not exceeded. When they are reached, the appropriate spindle signals are output at the interface.

**Monitoring functions**

The speed monitoring functions defined by G25 and G26 or the appropriate setting data are active. They monitor

- the work blade range within which the calculated contact point with the part to be ground must remain (as defined via MD).

**What happens in the case of range violation?**

When the range is violated,

- an appropriate alarm (self-resetting) and
- IS “Support range limits violated” (DB31–48, DBX83.4) are output.

DB31–48, DBX83.4 = 1 ⇒ Range violation monitoring has responded
DB31–48, DBX83.4 = 0 ⇒ Range violation monitoring has not responded

**Note**

There is no further reaction to the response of this monitoring function. If reactions are required, they must be programmed in the PLC program of the machine manufacturer.

**Interface signals**

The following axis/spindle-specific signals are provided for this function:

- IS “CLGON active” (DB31–48, DBX84.2)
- IS “Support range limits violated” (DB31–48, DBX83.4)

The signals must be interpreted as follows:

| Interface signal | | Meaning |
|---|---|---|
| CLGON active | Support range limits violated | |
| 0 | 0 | Centerless grinding not active |
| 1 | 0 | Centerless grinding active, no error |
| 0 | 1 | Centerless grinding active, but not functional owing to error |
| 1 | 1 | Centerless grinding active, range limits violated |

**Note**

The spindle (of the regulating wheel) can be braked with IS “Spindle reset” (DB31–48, DBX2.2).

The spindle override is operative.

**Response to RESET or program end**

The response to RESET or part program end is determined by channel-specific MD:SPIND_ACTIVE_AFTER_RESET.

If no spindle reset takes place on RESET, then the current spindle speed is frozen. The “Constant workpiece speed for centerless grinding” function is automatically deselected.

■

## Notes

# Supplementary Conditions 3

**Availability**

The "constant workpiece speed for centerless grinding" function is available for

- SINUMERIK 840D with NCU 572/573, with SW 2 and higher.

# Data Descriptions (MD, SD) 4

## 4.1 Channel-specific machine data

| | | | | | |
|---|---|---|---|---|---|
| **24100**<br>**24200**<br>**24300**<br>**24400**<br>MD number | **TRAFO_TYPE_1**<br>**TRAFO_TYPE_2**<br>**TRAFO_TYPE_3**<br>**TRAFO_TYPE_4**<br>Definition of transformation 1, 2, 3, 4 in channel | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: *** | |
| Changes effective after power ON | | | Protection level: 2 | | Unit: – |
| Data type: DWORD | | | | Applies from SW 2.1 | |
| Meaning: | The identifier of the desired transformation must be entered in this MD.<br>The identifier for transformation for the "Constant workpiece speed for centerless grinding" function is 2048. | | | | |
| Related to .... | MD: TRAFO_AXES_IN_1[n]<br>MD: TRAFO_AXES_IN_2[n]<br>MD: TRAFO_AXES_IN_3[n]<br>MD: TRAFO_AXES_IN_4[n] | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **24110**<br>**24210**<br>**24310**<br>**24410**<br>MD number | **TRAFO_AXES_IN_1[n]**<br>**TRAFO_AXES_IN_2[n]**<br>**TRAFO_AXES_IN_3[n]**<br>**TRAFO_AXES_IN_4[n]**<br>Axis assignment for transformation [Index]: | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: max. no. of axes | |
| Changes effective after power ON | | | Protection level: 2 | | Unit: – |
| Data type: BYTE | | | | Applies from SW 2.1 | |
| Meaning: | [0]: No. of axis in channel which moves the grinding wheel (e.g. axis 5 = X1)<br>[1]: No. of axis in channel which moves the regulating wheel (e.g. axis 4 = X2)<br>[2]: No. of axis in channel which moves the work blade (e.g. axis 3 = Q1)<br>[3]: No. of axis in channel which moves the work blade (e.g. axis 1 = Q2)<br>If TRAFO_TYPE_3 = 2048 and, for example, there is a regulating wheel (axis 4) and an axis for the work blade (axis 3), then the following applies:<br>TRAFO_AXES_IN_3[1] = 4 and TRAFO_AXES_IN_3[2] = 3 | | | | |
| Special cases, errors, ...... | An axis must always be specified for the regulating wheel. | | | | |
| Related to .... | MD: TRAFO_TYPE_1<br>MD: TRAFO_TYPE_2<br>MD: TRAFO_TYPE_3<br>MD: TRAFO_TYPE_4 | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **21522**<br>MD number | **TRACLG_GRINDSPI_NR**<br>No. of grinding spindle | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: max. no. of spindles | |
| Changes effective after power ON | | | Protection level: 2 | | Unit: – |
| Data type: BYTE | | | | Applies from SW 2.1 | |
| Meaning: | The number of the grinding spindle is specified in this MD. | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **21524**<br>MD number | **TRACLG_CTRLSPI_NR**<br>No. of regulating spindle | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: max. no. of spindles | |
| Changes effective after power ON | | | Protection level: 2 | | Unit: – |
| Data type: BYTE | | | | Applies from SW 2.1 | |
| Meaning: | The number of the regulating spindle must be specified in this MD. | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **21500**<br>MD number | **TRACLG_GRINDSPI_VERT_OFFSET**<br>Vertical offset of grinding spindle | | | | |
| Default setting: 0 | | Min. input limit: – | | Max. input limit: – | |
| Changes effective after power ON | | | Protection level: 2 | | Unit: mm, inches |
| Data type: DOUBLE | | | | Applies from SW 2.1 | |
| Meaning: | The vertical offset of the grinding axis is specified in this MD. | | | | |

| 21502<br>MD number | TRACLG_CTRLSPI_VERT_OFFSET<br>Vertical offset of regulating spindle | |
|---|---|---|
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 | Unit: mm, inches |
| Data type: DOUBLE | Applies from SW 2.1 | |
| Meaning: | The vertical offset for the regulating axis is specified in this MD. | |

---

**Note**

Three quantities in each case are required to define the vertical and horizontal offsets of the work blade (position X0/Y0).

- The $X_{offset}$ or the $Y_{offset}$
  is the position of the work blade referred to Q1 = 0 and Q2 = 0
- The direction vector Q1
  is the change in position if Q1 is traversed to 1 and Q2 remains in position 0
- The direction vector Q2
  is the change in position if Q2 is traversed to 1 and Q1 remains in position 0.

---

| 21506<br>MD number | TRACLG_SUPPORT_HOR_OFFSET<br>Horizontal offset for work blade | |
|---|---|---|
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 | Unit: mm, inches |
| Data type: DOUBLE | Applies from SW 2.1 | |
| Meaning: | X offset for work blade<br>Rule: $X_0 = X_{offset} + Q1 * X_{direction\ vector\ Q1} + Q2 * X_{direction\ vector\ Q2}$ | |

| 21504<br>MD number | TRACLG_SUPPORT_VERT_OFFSET<br>Vertical offset for work blade | |
|---|---|---|
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 | Unit: mm, inches |
| Data type: DOUBLE | Applies from SW 2.1 | |
| Meaning: | Y offset for work blade<br>Rule: The following equation applies: $Y_0 = Y_{offset} + Q1 * Y_{direction\ vector\ Q1} + Q2 * Y_{direction\ vector\ Q2}$ | |

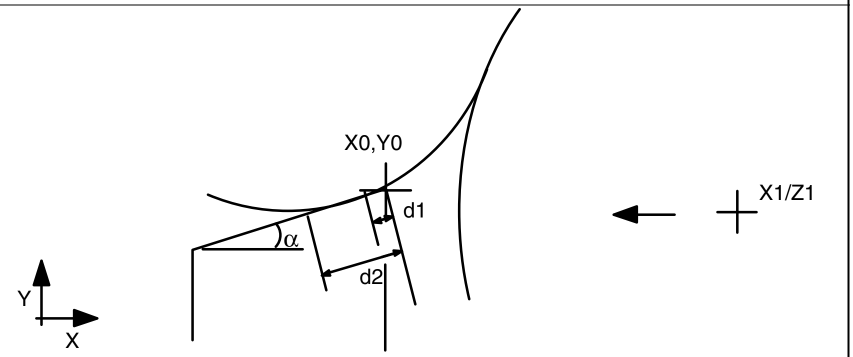
| 21510<br>MD number | TRACLG_HOR_DIR_SUPPORTAX_1<br>Horizontal direction vector for the 1st axis of work blade | |
|---|---|---|
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 | Unit: mm, inches |
| Data type: DOUBLE | Applies from SW 2.1 | |
| Meaning: | X component of blade direction vector for Q1<br>Rule: $X_0 = X_{offset} + Q1 * X_{direction\ vector\ Q1} + Q2 * X_{direction\ vector\ Q2}$ | |

| **21508**<br>MD number | **TRACLG_VERT_DIR_SUPPORTAX_1**<br>Vertical direction vector for the 1st axis of work blade | |
|---|---|---|
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 | Unit: mm, inches |
| Data type: DOUBLE | Applies from SW 2.1 | |
| Meaning: | Y component of blade direction vector for Q1<br>Rule: $Y_0 = Y_{offset} + Q1 * Y_{direction\ vector\ Q1} + Q2 * Y_{direction\ vector\ Q2}$ | |

| **21514**<br>MD number | **TRACLG_HOR_DIR_SUPPORTAX_2**<br>Horizontal direction vector for the 2nd axis of work blade | |
|---|---|---|
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 | Unit: mm, inches |
| Data type: DOUBLE | Applies from SW 2.1 | |
| Meaning: | X component of blade direction vector for Q2<br>Rule: $X_0 = X_{offset} + Q1 * X_{direction\ vector\ Q1} + Q2 * X_{direction\ vector\ Q2}$ | |

| **21512**<br>MD number | **TRACLG_VERT_DIR_SUPPORTAX_2**<br>axis of work blade axis of work blade | |
|---|---|---|
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 | Unit: mm, inches |
| Data type: DOUBLE | Applies from SW 2.1 | |
| Meaning: | Y component of blade direction vector for Q2<br>Rule: $Y_0 = Y_{offset} + Q1 * Y_{direction\ vector\ Q1} + Q2 * Y_{direction\ vector\ Q2}$ | |

| **21516**<br>MD number | **TRACLG_SUPPORT_LEAD_ANGLE**<br>Angle of lead of work blade | |
|---|---|---|
| Default setting: 0 | Min. input limit: –90 | Max. input limit: 90 |
| Changes effective after power ON | Protection level: 2 | Unit: Degrees |
| Data type: DOUBLE | Applies from SW 2.1 | |
| Meaning: | You can enter the lead angle of the work blade ($\alpha$) here. | |

| | | |
|---|---|---|
| **21518**<br>MD number | **TRACLG_CONTACT_UPPER_LIMIT**<br>Upper support range limit | |
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 | Unit: mm, inches |
| Data type: DOUBLE | Applies from SW 2.1 | |
| Meaning: | It is necessary to specify the upper contact limit of the blade with the part to be ground (d1) for the purpose of monitoring the support range limits. | |
| Diagram |  | |
| Related to .... | MD: TRACLG_CONTACT_LOWER_LIMIT | |

| | | |
|---|---|---|
| **21520**<br>MD number | **TRACLG_CONTACT_LOWER_LIMIT**<br>Lower support range limit | |
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 | Unit: mm, inches |
| Data type: DOUBLE | Applies from SW 2.1 | |
| Meaning: | It is necessary to specify the lower contact limit of the blade with the part to be ground (d2) for the purpose of monitoring the support range limits. | |
| Related to .... | MD: TRACLG_CONTACT_UPPER_LIMIT | |

| | | |
|---|---|---|
| **21526**<br>MD number | **TRACLG_G0_IS_SPECIAL**<br>Effect for G0 blocks | |
| Default setting: 1 | Min. input limit: 0 | Max. input limit: 1 |
| Changes effective after power ON | Protection level: 2 | Unit: – |
| Data type: BOOLEAN | Applies from SW 2.1 | |
| Meaning: | It is possible to define here how the speed of the regulating wheel must respond in the case of transitions between motion blocks with and without G0 (see Table 2-1).<br><br>**TRACLG_G0_IS_SPECIAL = 1:**<br>On transition from a motional block with G0 to one without, the speed of the regulating wheel is increased during the G0 block to the desired initial speed in the block without G0.<br><br>**TRACLG_G0_IS_SPECIAL = 0:**<br>The speed of the regulating wheel is controlled only for motional blocks without G0 (the transitions from a motional block with G0 to one without are not taken into account). | |

| | | |
|---|---|---|
| **21501**<br>MD number | **TRACLG_GRINDSPI_HOR_OFFSET**<br>Horizontal offset of grinding wheel | |
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 | Unit: mm, inches |
| Data type: DOUBLE | Applies from SW 2.1 | |
| Meaning: | The setting in this MD is significant only when MD: TRAFO_AXES_IN_n[0] = 0, i.e. no axis is programmed for the grinding wheel. | |

■

## Notes

# Signal Descriptions 5

## 5.1 Axis/spindle-specific signals

| | | | |
|---|---|---|---|
| **DB31–48 DBX83.4** | **Support range limits violated** | | |
| **DB31–48 DBX84.2** Data block | **CLGON active** Signal(s) from NCK → PLC | | |
| Edge evaluation: | | Signal(s) updated: | Signal(s) valid from SW 2.1 |
| Description | The signals must be interpreted as follows: | | |
| Further references | See Chapter 2 | | |

| CLGON active | Support range limits violated | Meaning |
|---|---|---|
| 0 | 0 | Centerless grinding not active |
| 1 | 0 | Centerless grinding active, no error |
| 0 | 1 | Centerless grinding active, but not functional owing to error |
| 1 | 1 | Centerless grinding active, range limits violated |

■

# Notes

# Example 6

## 6.1 Example of machining sequences

**Machining sequences**

The following machining sequences could be implemented:

```
...
CLGON(100)
S=1000 M3 M1=3
LABEL:
G0 X1=P1X1 X2=P1X2 Q1=P1Q1 Hyy
        ;Approach motion (+insert new part with auxiliary function):
        ;The spindle of regulating wheel accelerates up
        ;to initial speed of following block)
G1 X1=P2X1 X2=P2X2 Q1=P2Q1
        ;Infeed motion:
        ;The regulating wheel speed is determined acc. to
        ;"Constant workpiece speed for centerless grinding")
G0 X1=P3X1 X2=P3X2 Q1=P3Q1
        ;Eject motion:
        ;The speed of the regulating wheel is frozen at the
        ;end of the G01 block,
        ;"Constant workpiece speed for centerless
        ;grinding"/CLGON is suppressed.
        ;If "Constant workpiece speed for centerless
        ;grinding" were effective here, the speed of the
        ;regulating spindle would be changed unnecessarily
        ;and monitoring functions might signal an error).
GOTOB LABEL
        ;Start of a new machining process
...
```

## 6.2 Example of a machine configuration

**Requirements**

A machine for “Centerless grinding” is configured as follows:

- The work blade is defined.
- The axis for the grinding wheel is the 5th machine axis and designated as X1.
- The axis for the regulating wheel is the 4th machine axis and designated as X2.
- The grinding spindle is the 1st spindle.
- The regulating spindle is the 2nd spindle.

**Machine data**

TRAFO_TYPE_1 = 2048
TRAFO_AXES_IN_1[0] = 5
TRAFO_AXES_IN_1[1] = 4
TRAFO_AXES_IN_1[2] = 0
TRAFO_AXES_IN_1[3] = 0

TRACLG_GRINDSPI_NR = 1
TRACLG_CTRLSPI_NR = 2

TRACLG_GRINDSPI_VERT_OFFSET = 0
TRACLG_CTRLSPI_VERT_OFFSET = 0

TRACLG_SUPPORT_HOR_OFFSET = 370
TRACLG_SUPPORT_VERT_OFFSET = 50

TRACLG_HOR_DIR_SUPPORTAX_1 = 0
TRACLG_VERT_DIR_SUPPORTAX_1 = 0
TRACLG_HOR_DIR_SUPPORTAX_2 = 0
TRACLG_VERT_DIR_SUPPORTAX_2 = 0

TRACLG_SUPPORT_LEAD_ANGLE = 20
TRACLG_CONTACT_UPPER_LIMIT = 0
TRACLG_CONTACT_LOWER_LIMIT = 8

TRACLG_GO_IS_SPECIAL = 1
TRACLG_GRINDSPI_HOR_OFFSET = 0

■

# Data Fields, Lists 7

## 7.1 Interface signals

| DB number | Bit, byte | Name | Ref. |
|---|---|---|---|
| **Axis-specific** | | | |
| 31, ... | 83.4 | Support range limits violated | |
| 31, ... | 84.2 | CLGON active | |

**Note**

The axis/spindle-specific interface signals are valid.

| If ... | then ... |
|---|---|
| e.g. CLGON (500) is programmed | 500 is also output at the interface. However, that is the speed of the workpiece and not of the spindle. |

## 7.2 Machine data

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| **Channel-specific ($MC_ ...)** | | | |
| 21500 | TRACLG_GRINDSPI_VERT_OFFSET | Vertical offset of grinding spindle | |
| 21501 | TRACLG_GRINDSPI_HOR_OFFSET | Horizontal offset of grinding wheel | |
| 21502 | TRACLG_CTRLSPI_VERT_OFFSET | Vertical offset of regulating wheel | |
| 21504 | TRACLG_SUPPORT_VERT_OFFSET | Vertical offset for work blade | |
| 21506 | TRACLG_SUPPORT_HOR_OFFSET | Horizontal offset for work blade | |
| 21508 | TRACLG_VERT_DIR_SUPPORTAX_1 | Vertical direction vector for 1st axis of work blade | |
| 21510 | TRACLG_HOR_DIR_SUPPORTAX_1 | Horizontal direction vector for 1st axis of work blade | |

| Channel-specific ($MC_ ...) | | | |
|---|---|---|---|
| 21512 | TRACLG_VERT_DIR_SUPPORTAX_2 | Vertical direction vector for 2nd axis of work blade | |
| 21514 | TRACLG_HOR_DIR_SUPPORTAX_2 | Horizontal direction vector for 2nd axis of work blade | |
| 21516 | TRACLG_SUPPORT_LEAD_ANGLE | Angle of lead of work blade | |
| 21518 | TRACLG_CONTACT_UPPER_LIMIT | Upper support range limit | |
| 21520 | TRACLG_CONTACT_LOWER_LIMIT | Lower support range limit | |
| 21522 | TRACLG_GRINDSPI_NR | No. of grinding spindle | |
| 21524 | TRACLG_CTRLSPI_NR | No. of regulating spindle | |
| 21526 | TRACLG_G0_IS_SPECIAL | Effect on G0 blocks | |
| 24100 | TRAFO_TYPE_1 | Definition of transformation 1 in channel | |
| 24200 | TRAFO_TYPE_2 | Definition of transformation 2 in channel | |
| 24300 | TRAFO_TYPE_3 | Definition of transformation 3 in channel | |
| 24400 | TRAFO_TYPE_4 | Definition of transformation 4 in channel | |
| 24110 | TRAFO_AXES_IN_1 | Axis assignment for transformation | |
| 24210 | TRAFO_AXES_IN_2 | Axis assignment for transformation | |
| 24310 | TRAFO_AXES_IN_3 | Axis assignment for transformation | |
| 24410 | TRAFO_AXES_IN_4 | Axis assignment for transformation | |
| **Axis/spindle-specific ($MA_ ...)** | | | |
| 35040 | SPIND_ACTIVE_AFTER_RESET | Own spindle RESET | S1 |

## 7.3 Tool data

The tool data for the grinding spindle must be stored in T1, D1 and the data for the regulating spindle in T2, D1.

The base dimension is calculated depending on the tool type.

**References:** /FB/, W1, "Tool Offset"
/FB/, W4, "Grinding"

Table 7-1 Important tool data

| Parameters | Note |
|---|---|
| **Cutting-edge-specific offset data** | |
| $TC_DP1 | Only grinding tools are permitted. |
| **Tool-specific grinding data** | |
| $TC_TPG1 | The number of the regulating or grinding spindle must be entered here. The number specified must correspond to the appropriate machine data. |
| $TC_TPG8 | The angle of inclination stored here has no effect. |
| $TC_TPG9 | This parameter determines the radius data which are used in the center-less calculation. |

## 7.4 Alarms

A detailed description of the alarms which may occur is given in
**References:** /DA/, Diagnostics Guide
or in the online help in systems with MMC 101/102/103.

■

# Notes

# SINUMERIK 840D/840Di/810D Description of Functions Special Functions (Part 3)

# Tangential Control (T3)



■

## Notes

# Brief Description

# 1

**Tangential control**

The “Tangential control” function belongs to the category of NC functions with coupled axes. It is characterized by the following features:

- There are two *leading* axes which are moved *independently* by means of normal traversing instructions (leading axes). In addition there is a *following* axis whose position is determined *as a function* of the status of these leading axes (position, tangent).
- The leading axes and following axis are only coupled *at certain times*, i.e. the coupling can be switched on and off by program instructions.
- Tangential control is defined for the basic coordinate system/
- workpiece coordinate system.
- The leading axes are defined as geometry axes and the following axis as a rotary axis.
- The coupled axes are assigned to the same channel.
- The position of the following axis can be the input value for a transformation.
- Tangential control is only active in AUTOMATIC and MDA modes.

**Corners in the path contour**

If the contour defined by the leading axes contains a corner, the following points are to be noted with respect to the following rotary axis:

- **Up to and including SW 3.1,** the following rotary axis follows the leading axes in steps.
- **In SW 3.2 and higher** you can select one of two different types of response:
  – The path velocity is reduced to such an extent that the following axis reaches its target position synchronously with the other axes.
  – If TLIFT has been programmed, an intermediate block is inserted at any corner whose angle is greater than the “tangential angle for corner recognition” (EPS_TLIFT_TANG_STEP). In this inserted intermediate block, the rotary axis is moved as fast as possible to the position corresponding to the tangent after the corner. The limit values set for this axis are not violated.

**Canceling the follow-up grouping**

- **With SW 6.3 and higher**, the definition of a follow-up grouping can be canceled in order to track new leading axes with the following axis.

**Applications**

The tangential control function can be used to advantage for the following applications:

– Tangential positioning of a rotatable tool for nibbling operations.

– Follow-up control of tool alignment for a bandsaw.

– Positioning a dressing tool on a grinding wheel.

– Positioning of a gear shaping cutter in glass or paper processing applications.

– Tangential feed of a wire for 5-axis welding.

■

# Detailed Description 2

## 2.1 Characteristics of the tangential follow-up control

**Task definition**

The follow-up control for the rotary axis must be implemented such that the axis is always positioned at a specified angle on the programmed path of the two leading axes.

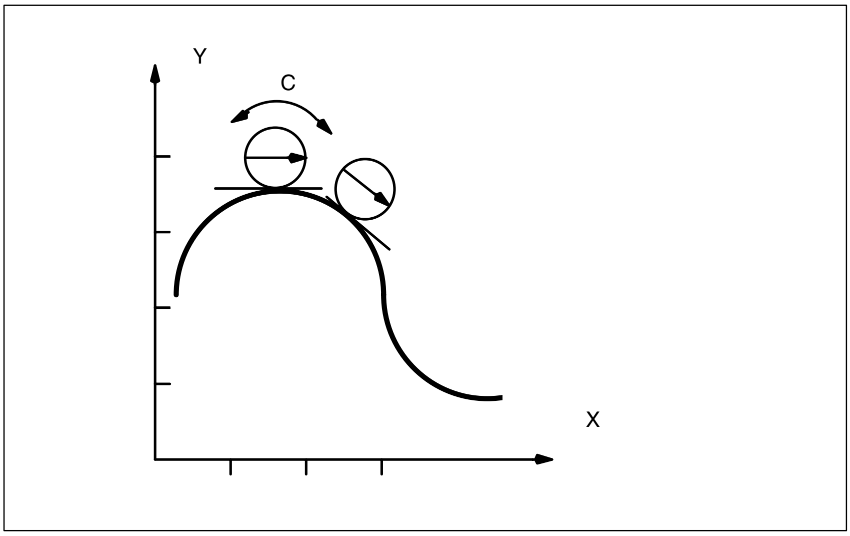



Fig. 2-1 Tangential control, offset angle of zero degrees to path tangent

In the diagram, X and Y are the leading axes in which the path is programmed; C is the following axis whose position is determined by the control as a function of the leading axis values and of the desired offset angle between tangent and alignment in C.

The tangential control will function only if the leading axes are used as path axes. A leading axis which is programmed as a positioning axis (POS or POSA) does not specify values required for the follow-up control function.

**Behavior of follow-up control SW 3.2 and higher**

A difference is to be made between the following cases:

– Without intermediate block (TLIFT)
The path velocity of the leading axes is reduced to such an extent that the following axis reaches its target position synchronously with the other axes.

– With intermediate block (TLIFT), without G641 rounding
The intermediate block causes the tangentially following axis to rotate as required. It is interpolated in such a way that the following axis travels at its limit velocity. The intermediate block is not rounded. At the beginning of the intermediate block, the path velocity of the leading axes is zero.

**Special cases**

– G641 rounding is possible between two blocks, both of which move at least one of the two leading axes of the tangentially following axis.

– G641 rounding is possible between two blocks, both of which do not move either of the leading axes of the tangentially following axis.

In both cases, an intermediate block for the tangentially following axis is not created. An intermediate block is not required because in the preprocessing run the rounded contour is detected and the limit values for the following axis are calculated.

– Hidden corner in space
A corner relevant for the tangential follow-up control can be hidden in space. (The projection of the contour on the plane defined by the two leading axes is relevant.)
If there is a hidden corner in space, an intermediate block is inserted before the block (here N6) causing the tangential jump. This intermediate block moves the following axis to the new position. The block transition is not rounded.

```
N1  TANG  (C, X, Y, 1)
N2  TLIFT  (C)
N3  G1  G641  X0  Y0  F1000
N4  TANGON (C)
N5  X10
N6  Y10                              ; the rotary axis is repositioned
                                       before the block is executed.
N7  M30
```

## 2.2 Using tangential follow-up control

**Activation**

The following axis can only be aligned if:

- the assignment between the leading and following axes is declared to the system (TANG)
- the follow-up control is activated explicitly (TANGON)
- the response at corners is specified, if required (TLIFT).

**Further functions**

Further functions are provided in order to

- terminate follow-up control of the following axis (TANGOF)
- deactivate the special response at corners (TANG() without a subsequent TLIFT)
- cancel the definition of a follow-up grouping (TANGDEL).

**Effect on transformation**

The position of the rotary axis to which follow-up control is applied can act as the input value for a transformation.

**References:** /FB/, M1, “Transmit/Peripheral Surface Transformation”

---

**Note**

The user is recommended to program TLIFT if tangential control is used together with a transformation. TLIFT prevents the follow-up axis from overtraveling and protects against excessive compensating movements.

---

**Explicit programming of follow-up axis**

If a following axis, which is being made to follow its leading axes, is positioned explicitly, then the position specification is added to the offset angle programmed in the activation instruction TANGON. See 2.2.2. The motional commands (AC, IC, DC, POS) are permitted.

**Reference point approach**

Follow-up control is deactivated while the following axis executes a reference point approach.

**Multi-channel block search in SW 6.1 and higher**

The cross-channel block search in Program Test mode (SERUPRO “**Se**rch **R**un by **Pro**gram Test”) can be used to simulate tangential follow-up of axes in SW 6.2 and higher.

For further information about the SERUPRO multi-channel block search please refer to:

**References:** /FB/, K1, “Mode Group; Channel, Program Operation” 2.4 Program testing

## 2.2.1 Assignment between leading axes and following axis

**Programming**

The programming instructions are provided in the predefined subprogam **TANG**. The following parameters are transferred to the control:

| | |
|---|---|
| Following axis (additional rotary axis) | here C |
| Leading axis 1 (geometry axis) | here X |
| Leading axis 2 (geometry axis) | here Y |
| Coupling factor | default 1 |
| Identifier of coordinate system<br>"B" → basic coordinate system, optional<br>["W" → workpiece coordinate system] | default "B" |

The appropriate axis identifiers are used to specify the axes. The coupling factor is generally "1".

The coupling factor can be omitted.

**TANG(C, X, Y)**

## 2.2.2 Activation of follow-up control

**Programming**

The activation is programmed via a predefined subprogam **TANGON**. When the tangential control is activated, the name of the following axis which must be made to follow is transferred to the control. This specification refers to the assignment between master and following axes made beforehand with TANG. See 2.2.1. An angle between the tangent and the position of the following axis can be specified optionally when follow-up is activated. This angle is maintained by the control for as long as the following axis is made to follow. The angle is added to the angle stored in machine data

$MA_TANG_OFFSET.

If the angle is zero both in TANGON and in the MD, the following axis takes the direction of the tangent.

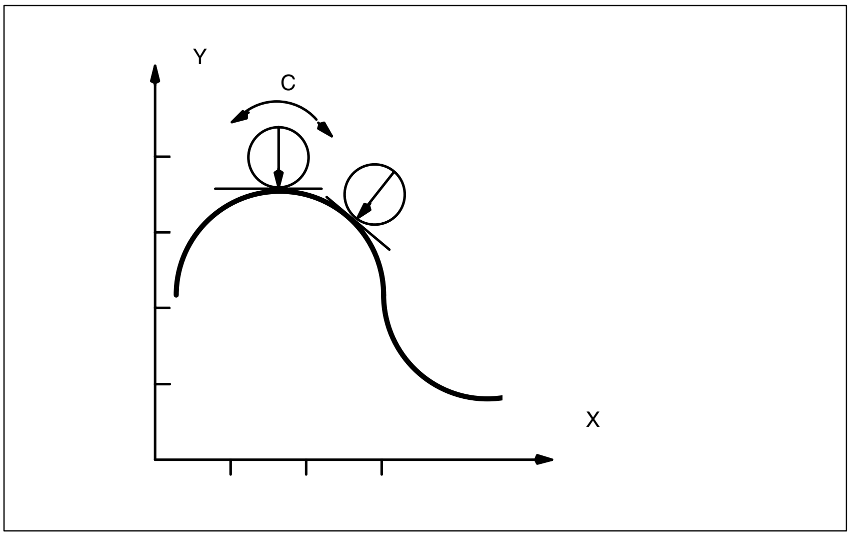



Fig. 2-2 Tangential control, offset angle of 90 degrees to path tangent

Activation is programmed as follows for the above example and an offset angle of 90 degrees:

**TANGON(C, 90)**

In response to every motion in path axes X and Y, following axis C is rotated to an angle of 90 degrees in the relation to the path tangent.

## 2.2.3 Switching on corner response

After axis assignment with TANG(), the TLIFT() instruction must be inserted if the corner response is to be contained in an intermediate block.

**TLIFT (C)**

The control reads machine data $MA_EPS_TLIFT_TANG_STEP
for the tangential follow-up axis C. If the tangential angle jump exceeds the angle (absolute value) of the angle set in the MD, the control recognizes a "corner" and approaches the new position of the follow-up axis via an intermediate block.

**SW 6.4 and higher**

System variable $MC_TLIFT_BLOCK indicates whether the current block is an intermediate block generated by TLIFT. If the value of the system variable is 1, TLIFT inserted the current block as an intermediate block.

### 2.2.4 Termination of follow-up control

**Programming**

The activation is programmed via a predefined subprogam **TANGOF**. The name of the following axis to be decoupled from its leading axes for the remainder of the machining operation must be transferred to the control in conjunction with the subroutine name TANGOF.

The termination command with respect to the example in 2.2.1 is:

**TANGOF(C)**

The follow-up control process initiated with TANGON is terminated.

Termination of follow-up control initiates a preprocessing stop internally in the control.

**RESET/part program end**

An activated tangential control can remain active for further machining operations. For further details, please refer to

**References:** /FB/, K2, "Coordinate Systems, Axis Types, Axis Configurations, Workpiece-Related Actual-Value System, External Zero Offset".

---

**Note**

The assignment between 2 leading axes and one following axis specified by TANG( ... ) is not canceled by TANGOF. See 2.2.6.

---

### 2.2.5 Switching off intermediate block generation

In order to stop generating the intermediate block at corners during program execution with active tangential follow-up control, the block TANG() must be repeated without following TLIFT().

### 2.2.6 Canceling the definition of a follow-up axis assignment.

A follow-up axis assignment specified by TANG() remains active after TANGOF. This inhibits a plane change or geometry axis switchover.

The predefined subprogam **TANGDEL** is used to cancel the definition of a follow-up axis assignment so that the follow-up axis can be operated dependent on new leading axes when a new follow-up axis assignment is defined.

**TANGDEL(C)**

The existing definition in the example of TANG(A, X, Y) is canceled.

**Example for plane change**

```
N10 TANG(A, X, Y, 1)
N20 TANGON(A)
N30 X10 Y20
......
N80 TANGOF(A)
N90 TANGDEL(A)            ; Cancel defined link between A and X and Y
                          ; as leading axes
......
N120 TANG(A, X, Z)        ; A can be linked to new leading axes
                          ;
N130 TANGON(A)
......
N200 M30
```

**Example for geometry axis switchover**

If the definition of the follow-up axis assignment is not canceled, an attempt to execute a geometry axis switchover is suppressed and an alarm is output.

```
N10 GEOAX(2, Y1)
N20 TANG(A, X, Y)
N30 TANGON(A, 90)
N40 G2 F8000 X0 Y0 I0 J50
N50 GEOAX(2, Y2)          ; Alarm 14415 geometry axis to be canceled
                          ; is still leading axis of follow-up
                          ; axis assignment
```

**Geometry axis switchover with TANGDEL**

The following example shows how TANGDEL is used correctly in association with an axis switchover.

```
N10 GEOAX(2, Y1)              ; Geometry axis group is defined
N20 TANG(A,X, Y)              ; Channel axis Y1 is assigned
N30 TANGON(A, 90)             ; Follow-up grouping with Y1 is activated
N40 G2 F8000 X0 Y0 I0 J50     ; Traversing block for the leading axes
N50 TANGOF(A)                 ; Deactivate follow-up
N60 TANGDEL(A)                ; Cancel definition
                              ; of follow-up axis assignment
N70 GEOAX(2, Y2)              ; Geometry axis switchover possible
N80 TANG(A, X, Y)             ; New def. of follow-up axis grouping
N90 TANGON(A, 90)             ; Follow-up grouping with Y2 is activated
.......
```

## 2.3 Limit angle

**Definition of problem**

When the axis moves backwards and forwards along the path, the tangent turns abruptly through 180 degrees at the path reversal point. This response is not generally desirable for this type of machining operation (e.g. grinding of a contour). It is far better for the reverse motion to be executed at the same offset angle (negative) as the forward motion.

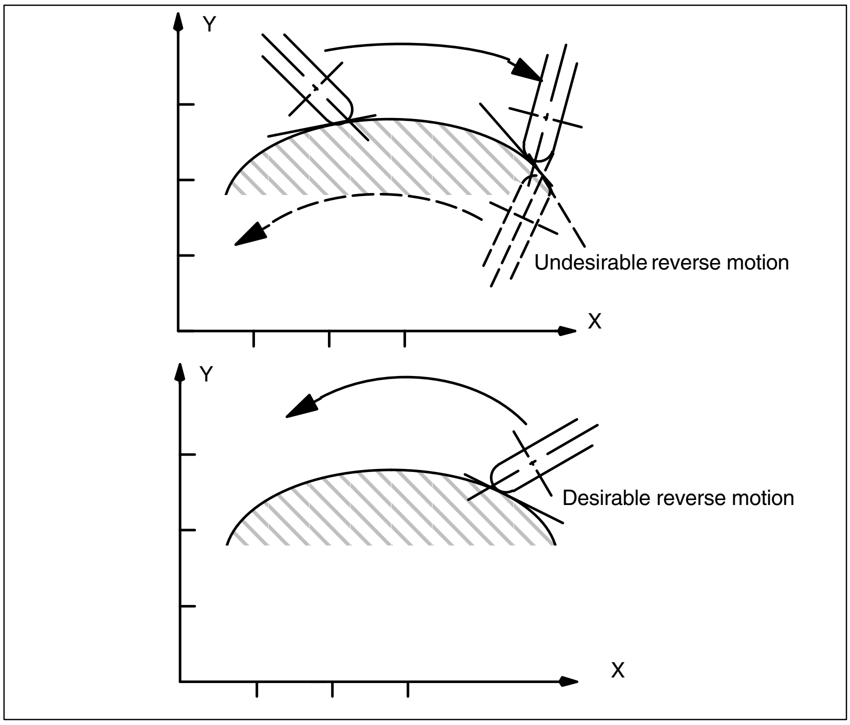



Fig. 2-3 Backward and forward motion on the path

**Programming**

A minimum and a maximum value for the position of the axis made to follow (“C” in example) referred to the base coordinate system are transferred to the control with G25 and G26. These working area limitations are activated with WALIMON and deactivated again with WALIMOF. The working area limitation must be active at the instant of path reversal.

**References:** /PA/, Programming Guide Fundamentals

**Effect**

If the current offset angle is outside the active working area limitation for the following axis, an attempt is made to return to within the permissible working area by means of the negative offset angle. This response corresponds to that shown in the lower diagram of Fig. 2-3.

■

## Notes

# Supplementary Conditions 3

**Availability**

The “Tangential control” function is an option and available for

- SINUMERIK 840D with NCU 572/573, with SW2 and higher.

As of SW 3.2, the special response at path corners, controlled by TLIFT () is available.

# Data Descriptions (MD, SD) 4

## 4.1 Machine data

| | | | | |
|---|---|---|---|---|
| **37400**<br>MD number | **$MA_EPS_TLIFT_TANG_STEP**<br>Tangential angle for corner recognition | | | |
| Default setting: 5 | Min. input limit: 0 | | Max. input limit: 180 | |
| Changes effective after RESET | | Protection level: 2 / 7 | | Unit: Degrees |
| Data type: DOUBLE | | | Applies from SW 3.2 | |
| Meaning: | If TLIFT has been programmed and the axis is under tangential follow-up control, a step change in the position setpoint larger than EPS_TLIFT_TANG_STEP causes an intermediate block to be inserted. The intermediate block moves the axis to the position corresponding to the initial tangent in the next block. | | | |
| MD irrelevant for ...... | TLIFT not activated | | | |
| Related to .... | TLIFT instruction | | | |

| 37402<br>MD number | $MA_TANG_OFFSET<br>Default angle for tangential follow-up control | |
|---|---|---|
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: Degrees |
| Data type: DOUBLE | Applies from SW: 3.2 | |
| Meaning: | Default offset (angle) which the following axis forms with the tangent. The angle acts additively to the angle programmed in the TANGON block. | |
| MD irrelevant for ...... | If no tangential follow-up control. | |
| Related to .... | TANGON instruction | |

## 4.2 System variable

| **Name** | **$AC_TLIFT_BLOCK** | | | |
|---|---|---|---|---|
| Meaning | The system variable indicates whether the current block was generated by TLift as an intermediate block.<br>0 Current block is **not** an intermediate block generated by TLIFT<br>1 Current block **is** an intermediate block generated by TLIFT | | | |
| Data type | INT | | | |
| Access | Read in part program | Write in part program | Read in synchronous action | Write in synchronous action |
| | X | | X | |
| Implicit preprocessing stop | X | | | |

■

# Signal Descriptions 5

**Special response to signals**

The movement of the axis under tangential follow-up control to compensate for a tangent jump at a corner of the path (defined by the movements of the leading axis) can be stopped by the following signals (e.g. for test purposes):

- NC Stop and override = 0
- Removal of the axis-specific feed enable

The signals are described in

**References:** /LIS/, Lists

■

# Notes

# Examples 6

### Positioning of workpiece

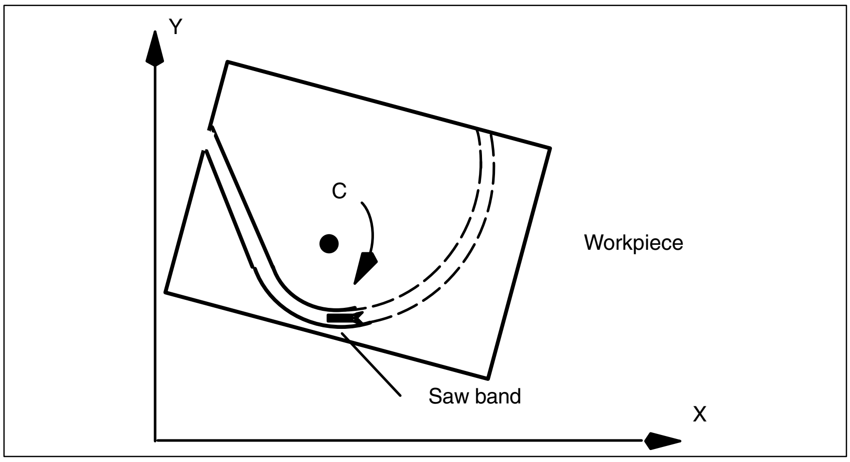



Fig. 6-1 Tangential positioning of a workpiece on a bandsaw

### Positioning of tool

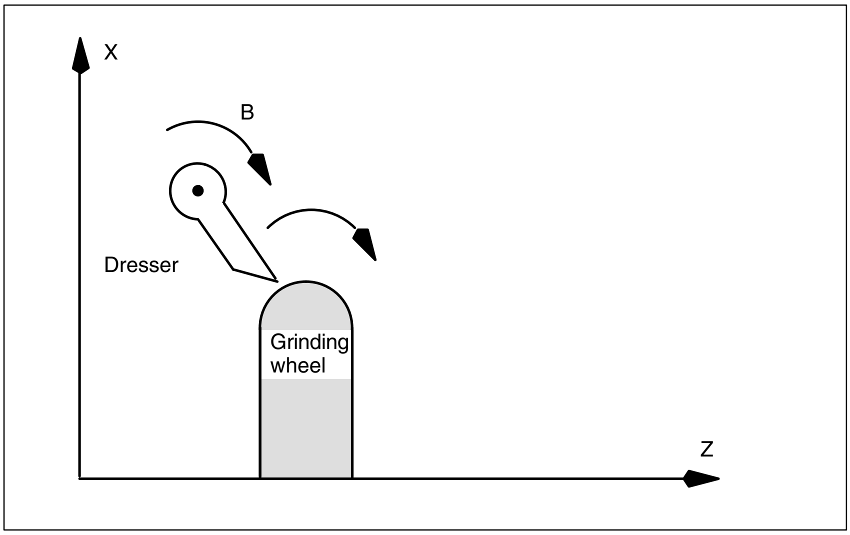



Fig. 6-2 Positioning of a dressing tool on a grinding wheel

**Example**
**Corner in area**

```
TANG(A,X,Y,1.0,“B”)
TLIFT(A)
G1 G641 X0 Y0 Z0 A0
TANGON(A,0)
N4 X10
N5 Z10
N6 Y10
M30
```

Here, a corner is hidden in the area between N4 and N6. N6 causes a tangent jump. That is why there is no rounding between N5 and N6 and an intermediate block is inserted.

In the case of a hidden corner in area, an intermediate block is inserted before the block that has caused the tangent jump. The intermediate block moves the following axis to the new tangent position.

■

# 7 Data Fields, Lists

## 7.1 Alarms

When the “Tangential control” function is used, special situations may arise which are indicated by alarm messages on the operator panel front. In systems with the MMC 101/102/103 operator interface, the online help functions can be called to display more detailed information about the active alarm than that shown in the message line.
Users of an operator panel front with MMC 100 can find detailed information about the alarms in the Diagnostics Guide.

**References:** /DA/, Diagnostics Guide

## 7.2 Machine data

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| **Axis-specific ($MA_...)** | | | |
| 37400 | EPS_TLIFT_TANG_STEP | Tangent angle for corner recognition | |
| 37402 | TANG_OFFSET | Default angle for tangential follow-up control | |

## 7.3 System variables

| | Identifier | Name | Ref. |
|---|---|---|---|
| | $AC_TLIFT_BLOCK | Current block is an intermediate block generated by TLIFT | |

■

# SINUMERIK 840D/840Di
# Description of Functions Special Functions (Part 3)

# Installation and Activation of Loadable Compile Cycles (TE0)



■

# Notes

# Brief Description 1

## 1.1 Brief description (840D/810D)

**Aim of the description**

This section describes how technology functions in the form of loadable compile cycles are installed and activated. The description applies to all of the following technology functions available from Siemens:

- 1D/3D clearance control in position control cycle
  Order no.: 6FC5 251-0AC05-0BA0
  Compile cycle: CCCLC.ELF
  **References**: Clearance Control chapter (TE1)
- Handling Transformation Package
  Order no.: 6FC5 251-0AD07-0BA0
  Compile cycle: CCRCTRA.ELF
  **References**: Handling Transformation Package chapter (TE4)
- Setpoint exchange
  Order no.: 6FC5 251-0AC05-0BA0
  Compile cycle: CCSETP.ELF
  **References**: Setpoint Exchange chapter (TE5)
- Axial coupling in machine coordinate system (MCS coupling)
  Order no.: 6FC5 251-0AD11-0BA0
  Compile cycle: CCMCSC.ELF
  **References**: MCS Coupling chapter (TE6)
- Retrace/Continue Machining Support
  Order no.: 6FC5 251-0AE72-0BA0
  Compile cycle: CCRESU
  **References**: Retrace/Continue Machining Support chapter (TE7)
- High-speed laser switching signal
  Order no.: 6FC5 251-0AE74-0BA0
  Compile cycle: CCHSLC.ELF
  **References**: Cycle-Clock-Independent Path-Synchronous Signal Output chapter TE8

as well as to user-specific technology functions.

The following technology functions are not available in the form of compile cycles:

- Analog axis
  The compile cycle is now available as a hardware solution.
- Speed/torque coupling
  The compile cycle is available as a general function SW 6.4 and higher.

**Supply format**

Up to and including SW 6.3, the technology functions were supplied in the form of compile cycles on the technology PC card. Several functions were normally stored on each PC card but only one function was used. With SW 6.4, the technology functions are supplied individually as loadable compile cycles.

**Tips for use**

The following chapters describe how to load and activate the compile cycles and set the necessary NCK machine data.

Please follow the instructions in Chapter 2 if you have not already used compile cycles.

Follow the instructions in Chapter 3 if you have made an archive from an operational control using compile cycles from a technology PC card and want to replace these compile cycles with more recent versions in the form of loadable compile cycles.

**Prerequisites**

One of the following programs is required for the installation:

- SinuCom NC
- SinuCopy FFS
- HMI Advanced, SW 6.3 and higher.

Furthermore, a PG/PC with MPI connection to the NCU must also be available.

---

**Note**

The following must be observed for system start-/up:

– Installation and Start-Up Guide /IAD/

– Installation and Start-Up Guide HMI/MMC and if necessary.

– the current Standard Upgrade Guide.

---

## 1.2 Brief Description (840Di)

The description of how to load and activate compile cycles in conjunction with the SINUMERIK 840Di can be found in:

**References**: /HBI/ SINUMERIK 840Di Manual
Section: NC Installation with HMI Advanced,
Loadable compile cycles

■

# Detailed Description 2

## 2.1 Design (840D/810D)

**Supply format**

Technology functions which are available as loadable compile cycles must be purchased as extensions:

**References**: Ordering information in Catalog NC 60.2002

To obtain the compile cycle in the form of a loadable file (.ELF extension for executable and linking format), please contact your regional Siemens sales partner.

**Installation and activation**

The following steps are necessary in order to install and activate a loadable compile cycle.

1. Copy the compile cycle into the flash file system (FFS) of the NCK.
2. Load the compile cycle into the NCK system software.
3. Activate the compile cycle in the NCK.

---

**Note**

Chapter 3 describes how to use existing archives and compile cycles.

---

### 2.1.1 Copying compile cycles into the flash file system (FFS)

**Target directory**

Copy the ELF file of the compile cycle to the directory "_N_CCOEM_DIR" of the FFS on the PC card of the NCU. You can do this in several ways:

**Outside the control (a)**

Create a PC card outside the control:

Use the **SINUCOPY FFS** program to create a "_N_CCOEM_DIR" directory in the FFS of the PC card and copy the required ELF files to this directory.

**Via PC/PG with MPI connection to NC (b)**

The installation software **SinuCom Nc** offers the menu item <File> <Load compile cycle> as of Version 6.2.12.
You can use this menu item to copy ELF files (<load> button) from any source directory (bottom window) under Windows to the FFS of the PC card (top window) while the PC card is inserted in the NCU.

SinuCom NC then offers to trigger an NCU reset to load the ELF file (see 2.1.2) and also permits deletion of ELF files from the FFS.

**The program ddetest.exe (c)**

The program is part of the an HMI Advanced delivery.

It copies files to the FFS in 3 steps:

1. Create a variables (here copyelf, for example):
   Execute <Doit> new(copyelf, –1)
2. Start hotlink on this variable:
   Hotlink <Start> copyelf
3. PI service for loading an ELF file stored on a diskette a:\, for example, start ccmcsc.elf:
   Execute <Doit> copy_to_nc (a:\ccmcsc.elf,
   /NC/_N_NC_CARD_DIR/_N_CCOEM_DIR/_N_CCMCSC_ELF, copyelf)
   The variable "copyelf" in the example has the initialization value –1 after starting the hot link.
   The value runs from 0 through 99 while loading the ELF file.
   The value 100 indicates that loading was successful.

   HMI openness allows this PI service to be integrated into an OEM operating menu.

**Storage capacity**

The storage capacity specified in the table below must be free in the FFS in order to load the functions. Please note that some functions load further functions implicitly.

Table 2-1 Storage requirements in the flash file system

| Function | Description | Size [ KB ] |
|---|---|---|
| CLC | TE2 | 81 |
| RCTR | TE | 130+100 |
| SETP | TE5 | 52 |
| MCSC | TE6 | 80 |
| TPM | – | 59 |
| HSLC | TE8 | 47 |
| RESU | TE7 | 89 |
| DST | – | 407 |
| MATH1 | – | 58 |
| MATH2 | – | 98 |

**With HMI Advanced SW 6.3**

As of SW 6.3, the standard scope of HMI Advanced offers an operating menu for loading ELF files. See /BAD/.

### 2.1.2 Loading the compile cycles into the NCK system software

Each time the NCK is booted, **all** of the compile cycles in directory

– _N_CCOEM_DIR

(ELF files) are automatically loaded into the NCK system software.

**Version display**

Once the cycles have been loaded, the current compile cycle versions are displayed in the HMI Advanced user interface at: **Operating area switchover > Diagnosis > Service > Version > NCU Version**

---

**Note**

The start addresses of the code and data areas are displayed in addition to the current versions for diagnostics purposes.

---

**Load abort**

The following conditions can cause the load operation to be aborted:

a)
Alarm 7200 "Version conflict with CCNCK–Interface-Version"
A version of an ELF file does not match the current NCK system:

Reason 1:
An attempt was made to load an ELF file that is too old for the current system.
Sign:
The ELF file could be loaded in an older NCK system.
Remedy:
Obtain a newer version of the ELF file.

Reason 2:
The ELF file uses interfaces provided only by more recent NCK systems.
Sign:
The ELF file has been obtained.
Remedy:
Update of the NCK system required (or older ELF version)

b)
Alarm 7200 “CCXXXX_ELF Loader problem from dFixup”
A further file is needed in order to load the CC application successfully:
For example CCMATH1_ELF (mathematics library for particular OEM transformations)
or CCSEC_ELF (SpaceErrorCompensation for particular OEM transformations)

The file name output in this alarm does not always refer precisely to the ELF file that caused the problem.

c)
Alarm 7200 “CCXXXX_ELF NO EMBARGO”
Alarm 7200 “CCXXXX_ELF NO 840Di”
Alarm 7200 “CCXXXX_ELF NO 810D”
Use of the ELF file is not enabled in the specified system.

---

**Caution**

If **any** Alarm 7200 is active after start-up, **NONE** of the compile cycles were loaded!

---

## 2.1.3 Activating the technology functions in the NCK

**Option**

The corresponding option must be enabled before activating a technology function as described below (see Section 4.2).

If the corresponding option is not enabled, the following alarm is displayed each time the NCK is booted, and the technology function is not activated:

- Alarm “7201 XXX_ELF_option_bit_missing: <*bit number*>”

---

**Note**

The previous compile cycle option data

– MD 19600: $ON_CC_EVENT_MASK[#]

which were used to activate the compiled technology functions are not relevant for the loadable compile cycles.

---

**Activation**

Each technology function loaded by compile cycle creates a function-specific global NCK machine data:

- $MN_CC_ACTIVE_IN_CHAN_<*identifier*>[n], where n = 0, 1

in the number range from 60900 to 60999.

Example: Technology function MCS coupling (CC*MCSC*.ELF)

– $MN_CC_ACTIVE_IN_CHAN_*MCSC*[0]
– $MN_CC_ACTIVE_IN_CHAN_*MCSC*[1]

**Activation for 1st NC channel**

The technology functions are activated in the first NC channel via

– $MN_CC_ACTIVE_IN_CHAN_<*identifier*>[0], bit 0 = 1

The meanings of all further machine data bits are described in the function descriptions (TE1 – TE8).

---

**Note**

Please refer also to the following documents for system installation and start-up:

**References**:

– /IAD/ Installation & Start-up Guide 840D/611D
– /IAM/ Installation & Start-up Guide HMI/MMC
– or the current standard upgrade instructions

---

**Caution**

The first time a bit is set in one of the function-specific NCK machine data:

- $MN_CC_ACTIVE_IN_CHAN_XXXX[0],

the control outputs the following alarm:

- Alarm “4400 MD modification causes reorganization of the buffered memory (data will be lost!)”

and you are warned that all user data (parts programs, tool data, etc.) will be deleted on the next power-up. It may be necessary to create an archive AFTER setting this data and **PRIOR** to triggering an NCK RESET.

---

The technology functions activated by function-specific NCK machine data are effective after the next NCK power-up.

**Function-spec. installation**

The further function-specific installation routines are described in the corresponding function descriptions (TE1–TE8).

## 2.2 Design (840Di)

The description of how to load and activate compile cycles in conjunction with the SINUMERIK 840Di can be found in:

**References**: /HBI/ SINUMERIK 840Di Manual
Section: NC Installation with HMI Advanced,
Loadable compile cycles

■

## Notes

# Updating Technology Functions From Existing Archives For Newer NCK Versions

# 3

## 3.1 Migrating to NCK 06.03.23 or higher (840D/810D)

**Prerequisities**

The following are required:

- A PC card with standard system version 06.03.23 or higher
- The ELF files for the technology functions to be activated
- The conversion program arc4elf.exe (archive conversion).

**Required procedure**

Please proceed as follows to use technology functions from an existing archive in conjunction with an NCK version as of NCK 06.03.23:

1. Create back-up archive
   a) standard procedure
   b) with optimized S-RAM utilization
2. Insert new PC card
3. Incorporate the ELF files
4. Reset the NCU
5. Activate technology function
6. NCU RESET
7. Convert archive using arc4elf.exe
8. Load converted archive.

### 3.1.1 Create back-up archive

**Standard procedure**

a)
Create a data back-up as described in /BAD/.

**Optimized**

b)
Create a data back-up with optimized SRAM utilization:

This step is necessary only if an archive from SW 6.3.xx is used and when the SRAM (buffered memory) is to be optimized:

New CC functions loaded per ELF reserve the SRAM required for the CC machine data autonomously.

In order to optimize the use of the S-RAM, you can reset the global machine data MD 18238: MM_CC_MD_MEM_SIZE to the default value 1. This first frees the SRAM space explicitly reserved for the compile cycles and then reserves the space actually needed for the ELF files.

If MD 18238: CC_MD_MEM_SIZE is not reset, the reserved space in S-RAM (in kB ) remains unused.
Resetting $MN_MM_CC_MD_MEM_SIZE to 1 produces alarm 4400
"MD modification causes reorganization of the buffered memory (data will be lost !)"

**In this state, you must create a new archive PRIOR to a further NCU RESET.**

**Follow steps 3.1.2 to 3.1.8 below.**

### 3.1.2 Insert new PC card

**PC card**

Replace the previous PC card with the one that contains the new system and clear the SRAM with:

NCU RESET with NCU switch S3 to position 1.

After this start–up, alarm 4060 "Standard machine data loaded is present".

### 3.1.3 Loading the compile cycles into the FFS

Use the option described in that is most suitable 2.1.1.

### 3.1.4 NCU RESET

When the NCK is rebooted after an NCU reset, the compile cycles are loaded from the FFS to the NCK system software.

You can check the versions of the loaded compile cycles (see Subsection 2.1.2).

### 3.1.5 Activate technology function

**Option**

The option bits for the loaded ELF files can be set (see Subsection 2.1.3).

**Channel activation**

Subsection 2.1.3 describes the channel settings for the individual technology functions in
MD $MN_CC_ACTIVE_IN_CHAN_XXX[0] and
MD $MN_CC_ACTIVE_IN_CHAN_XXX[1]
The associated MD number is derived from the loading sequence in 3.1.4.

### 3.1.6 NCU RESET

The NCK is rebooted; no alarms should appear.

### 3.1.7 Convert archive

The archive created in step 3.1.1 a) and/or b) must be converted. The **arc4elf.exe** program is required for this purpose (available from E-Support).

You can call up help for using the program by entering
**arc4elf –h**.
The general call format is:

**arc2elf ORIGINAL.ARC CONVERTED.ARC**

Replace ORIGINAL:ARC and CONVERTED.ARC with the actual archive names.

The converted archive is created in the same directory as the original archive.

### 3.1.8 Load converted archive

Load the converted archives as described in /BAD/.

Activate the imported data by NC-RESET.

■

## Notes

# Machine Data 4

## 4.1 Activating the technology functions

| | | | | | |
|---|---|---|---|---|---|
| **60900 + i**<br>where i = 0. 1. 2. 3 ...<br>MD number | **CC_ACTIV_IN_CHAN_XXXX[n]**<br>where: XXXX = function identifier, n = 0 or 1<br>n = 0: Activate technology function in NC channels<br>n = 1: Additional functions within the technology function | | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: | |
| Changes effective after RESET | | | Protection level: 2 / 7 | | Unit: – |
| Data type: UINT16 | | | | Applies from SW: 6.4 | |
| Meaning: | Activate technology function in NC channels:<br>The technology function is activated in the NC channels by means of index n = 0.<br>Bit 0 = 1: Technology function activated in NC channel 1<br>Bit n = 1: Technology function activated in NC channel n+1<br>See function descriptions TE1 – TE8 for details of the NC channels for which a technology function can be activated.<br>Additional functions within the technology function:<br>The MD with index n = 1 activates additional functions within the relevant technology function. See function descriptions TE1 – TE8. | | | | |

## 4.2 Activating the technology functions (option)

Technology functions which are available as loadable compile cycles must be purchased as extensions.

**References**: Ordering information in Catalog NC 60.2002

To obtain the compile cycle in the form of a loadable file (*.elf), please contact your regional Siemens sales partner.

■

# Notes

# SINUMERIK 840D/840Di/810D Description of Functions Special Functions (Part 3)

# Clearance Control (TE1)

■

# Brief Description 1

**Function description**

The “clearance control” technology function is used to maintain a technological (1D) or three-dimensional (3D) clearance during a machining process. The clearance to be maintained is the distance between the tool and the workpiece surface.

**Compile cycle**

The “clearance control” technological function is a compile cycle. The system-specific availability and handling of compile cycles is described in the section entitled “Installation and activation of loadable compile cycles” (TE0), Page 3/TE0/1-3 ff.

■

## Notes

# Detailed Description 2

## 2.1 Requirements

The “clearance control” technology function is used to maintain a technological (1D) or three-dimensional (3D) clearance during a machining process. The clearance to be maintained is the distance between the tool and the workpiece surface. The laser cutting technology is used as an example for the further description of the “clearance control” functionality.

**Laser cutting**

During laser cutting, a divergent parallel laser beam is directed across a fiber-optic cable or via a mirror to a light-collecting lens mounted on the laser machining head. The collecting lens focuses the laser beam at its focal point. Typical focal lengths are from 5 to 20cm.

The position of the focal point in relation to the workpiece is an extremely critical process parameter in laser cutting operations and must be kept constant with a tolerance of approx $\leq 100\mu m$.

The distance between the focal point and the workpiece, which is also a key process variable, is usually measured by means of a high-speed capacitive sensor.

The analog signal of the sensor is approximately proportional to the deviation from the setpoint distance. It is transferred via an analog I/O module to the NC, where it generates an additional speed setpoint for the motion axes of the machining head.

**System overview (840D)**

An overview of the system components required for clearance control in conjunction with SINUMERIK 840D is provided in Fig. 2-1.

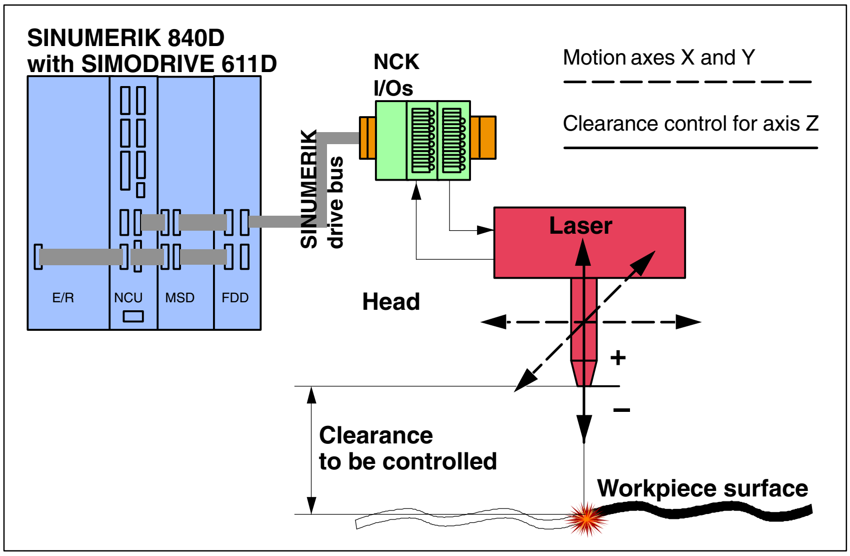



Fig. 2-1 System components for clearance control with SINUMERIK 840D

**System overview (840Di)**

An overview of the system components required for clearance control in conjunction with SINUMERIK 840Di is provided in Fig. 2-2.

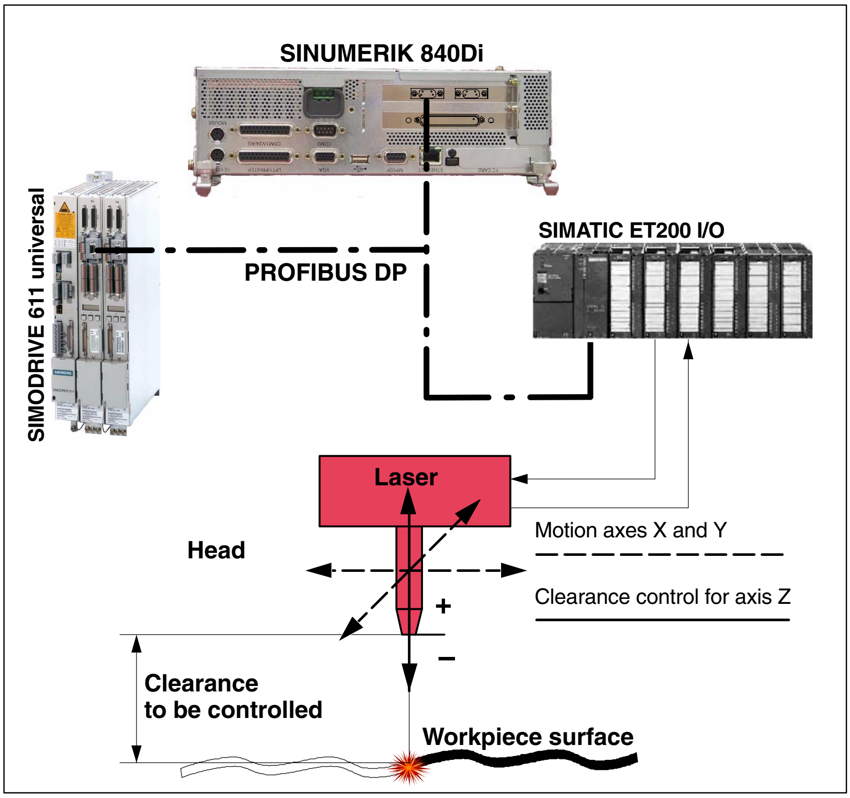



Fig. 2-2 System components for clearance control with SINUMERIK 840Di

**1D/3D machining**

The clearance control can be used for 1D and 3D machining with up to 5 interpolating geometry axes.

- 1D machining

  In the case of 1D machining, only one machine axis is affected by the clearance control, e.g. axis Z, as shown in the example machine configurations in the system overview, Fig. 2-1 and Fig. 2-2. The clearance control acts in the direction of the Z axis.

- 3D machining

  In the case of 3D machining, up to 5 machine axes are affected by the clearance control, e.g. 3 linear axes and 2 rotary axes. The clearance control acts in the direction of the current tool offset vector.

## 2.1.1 Application categories

The applications of the clearance control can be divided into three categories according to the technological requirements of the process:

1. Category 1 (low dynamic requirements)

   Low dynamic requirements arise, e.g. when cutting flat metal sheets or 3D metal parts (mostly deep-drawing workpieces) which have onlyslight ripples and little deviation from the ideal geometry.

Key data:

– Clearance error: approx. 10 mm per meter of the machining path
– Cutting rate: approx. 20 m/min
– Superimposed closed-loop control speed: approx. 200 mm/min
– Tolerable control loop deadtime: up to approx. 40ms

2. Category 2 (medium dynamic requirements)

   Medium dynamic requirements arise in conjunction with the following conditions in addition to the requirements of Category 1:

   - Abrupt steps in metal thickness have to be controlled without explicit programming in the parts program.
   - The approach to the workpiece surface takes place rapidly and is exclusively sensor-driven.

3. Category 3 (high dynamic requirements)

   High dynamic requirements arise, e.g., during radial cutting of rod material with unrounded (typically rectangular) cross-sections. The rod material rotates around its longitudinal axis (axis X) through the programmed movement of rotary axis A.

   The compensating movements of the head (axis Z) are not programmed, but are influenced exclusively by the sensor-driven clearance control. The maximum available dynamic response on the controlled axes is utilized in this type of application.

   Key data:

   – Servo gain Kv of position controller: approx. 4 1000/min
   – Maximum axis velocity: approx. 10 m/min
   – Maximum axis acceleration: approx. 10 m/s$^2$

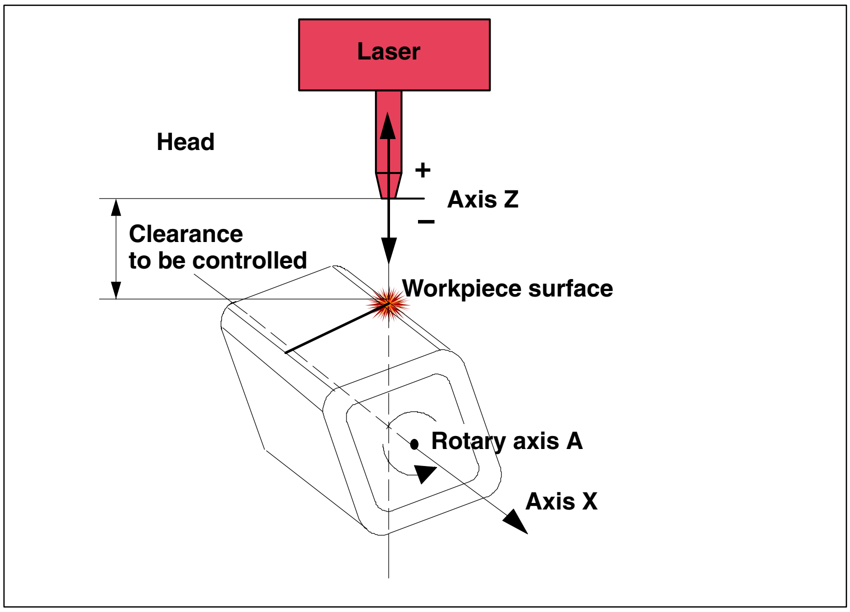



Fig. 2-3 Laser cutting of rotating square

## 2.1.2 Correlation between dynamic control response and deadtimes

**Closed-loop control gain Kv**

The dynamic response of the closed control loop (sensor – open-loop control - axis) is determined by the maximum closed-loop control gain Kv.

The closed-loop control gain Kv is defined as:

$$Kv = \frac{\text{velocity [m/min]}}{\text{following error [mm]}} \quad ; \quad \text{in } \left[ \frac{\text{[m/min]}}{\text{[mm]}} \right]$$

**Sensor characteristic**

The sensor measures the actual distance from the workpiece surface and returns as its output variable a voltage in [volt] which is directly proportional to the distance.

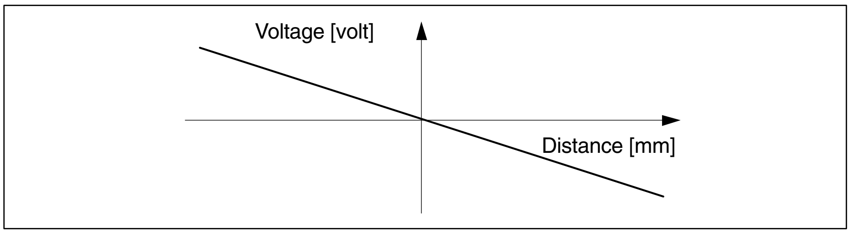



Fig. 2-4 Principle characteristic of the clearance sensor

From the perspective of the CNC, the closed-loop control gain unit is [(mm/min)/volt]. The gain can be standardized in [(mm/min)/mm] and the setpoint distance in [mm] only if corresponding settings are made in the sensor electronics.

**Max. closed-loop control gain**

The maximum achievable closed-loop control gain is determined by the following delay and reaction times of the overall system:

1. Reaction time of sensor
2. Delay time of the A/D converter
3. Delay time for signal processing in control
4. Reaction time of position controller
5. Reaction times of speed and current controllers
6. Time constants of motor and mechanical components.

In practice, only items 3 and 4 are relevant.

The influencing variables together produce an effective time constant. If the closed-loop control gain is set too high in relation to this time constant, natural oscillations on the axis/axes to be controlled result. The frequency of the natural oscillation depends on the effective time constant and is typically several Hertz in magnitude.

The objective when starting up the clearance control is to minimize important time constants in such a way that the closed-loop control gain required by the process can be set without inducing natural oscillation of this type.

## 2.2 Velocity feedforward control

**Delay time**

The closed-loop control gain set for the position controller corresponds to a delay time $\Delta t$. The delay time $\Delta t$ is the time which elapses until the actual position of the axis to be controlled correlates with the setpoint in response to a velocity specification v.

Where $\Delta t = \frac{1}{KV}$

and a closed-loop control gain Kv in seconds:

$$Kv \text{ in } \left[\frac{m/min}{mm}\right] = \left[\frac{1000\,mm/60\,s}{mm}\right] = 16.667\left[\frac{1}{s}\right]$$

for an assumed closed-loop control gain Kv = 4, the corresponding delay time $\Delta t$ for:

$$\Delta t = \frac{1}{4 * 16.667\left[\frac{1}{s}\right]} = 14.999ms$$

By activating the velocity feedforward control for the clearance-controlled axis, it is possible to almost completely eliminate this delay.

**Optimizing the control response**

If the control response of the axis is too rigid due to the velocity feedforward control, the control response can be optimized with the following axis-specific NC machine data:

- MD32410: AX_JERK_TIME (time constant for the axial jerk filter)
- MD32610: VELO_FFW_WEIGHT (feedforward control factor for the speed feedforward control)

**(840D)**

The velocity filters of the SIMODRIVE 611D drive provide an additional means of damping:

- MD1502: SPEED_FILTER_1_TIME (time constant for speed setpoint filter 1)
- MD1503: SPEED_FILTER_2_TIME (time constant for speed setpoint filter 2)

**(840Di)**

The velocity filters of the SIMODRIVE 611 universal / E and POSMO SI, CD, CA drives provide an additional means of damping:

- Parameter 1502: (time constant for speed setpoint filter 1)
- Parameter 1503: (time constant for speed setpoint filter 2)

**Caution**

Every damping measure implemented contributes to increasing the overall time constant of the control loop!

**References**

You will find a complete description of the velocity feedforward control in:

**References**: /FB2/ Description of Functions Extended Functions
Chapter: Compensation K3,
Following error compensation (feedforward control)

## 2.3 1D/3D distance control in position control cycle

To obtain a highly dynamic control response, the control loop needs to be closed in the position control cycle rather than in the interpolator cycle. The average deadtime caused by the CNC is then reduced to 1.5 position control cycles (see Fig. 2-4).

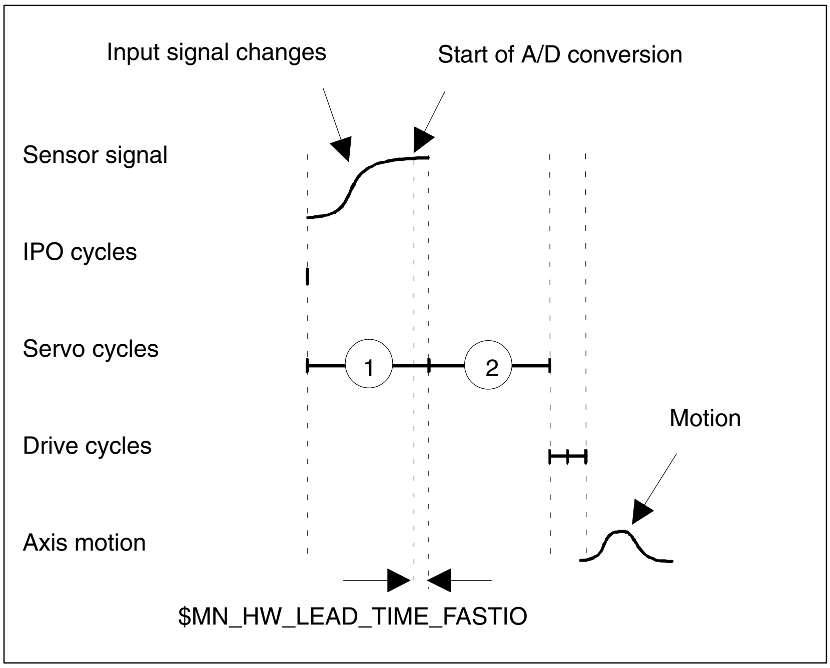



Fig. 2-4 Deadtime during processing of sensor signal in position control cycle

In addition to the highly dynamic control response, the 3D clearance control in position control cycle offers the following additional functionality:

- **Dynamic response**

  The overlaid sensor motion uses the current residual dynamic response that is still in reserve after the programmed axis motion (velocity and acceleration). The proportion of residual acceleration that must be used can be set as a percentage in a machine data.

- **Sensor characteristic (840D up to SW 5.3)**

  The gain characteristics of a sensor can be defined with up to 6 interpolation points.

- **Sensor characteristic**

  The gain characteristic of a sensor can be defined with up to 10 interpolation points.

- **Sensors**

  Two sensors with different gain characteristics (e.g. a mechanical and a capacitive sensor) can be connected simultaneously. The active sensor characteristic can be switched over block-synchronously by means of an NC command in the parts program.

- **Closed-loop control gain of clearance control**

  The closed-loop control gain configured in the NC machine data for clearance control can be changed block-synchronously by means of an NC command in the parts program.

- **Motion limitation**

  The lower and upper limits configured in the NC machine data for the axis movements induced by the clearance control can be changed block-synchronously by means of an NC command in the parts program.

  An alarm is output when a limit is reached. The alarm response (stop all traversing movements or display only) can be configured.

  The current position offset can be frozen by means of a PLC signal.

- **Response on deactivation**

  The deactivation response of the control can be programmed either for synchronization with the current axis positions (no compensating movement) or for compensating axis movements to the programmed axis positions (axis positions without clearance control).

- **Programmable clearance setpoint**

  An additional voltage value can be programmed in order to alter the setpoint distance set in the sensor electronics on a block-related basis.

- **Control options via the PLC interface**

  The following signals are available at the PLC interface:

  Status signals:

  – Closed-loop control active
  – Overlaying movement at standstill
  – Lower limit reached
  – Upper limit reached.

  Control signals:

  – Path override for sensor movement active

- **Status data of clearance control**

  Both the current values and the min/max values of the sensor signal and of the position offset are available as OPI and/or GUD variables.

- **Signal: Sensor collision**

  An additional “Sensor collision” signal is applied via a high-speed digital input and causes an instantaneous “escape” motion in the positive control direction as well as braking of the path motion. This additional function is activated and deactivated through programming measures in the parts program.

- **Sensor signal**

  The sensor signal can be smoothed via a PT1 filter with adjustable time constant.

**Block diagram**

The following diagram shows the operating principle of a clearance control

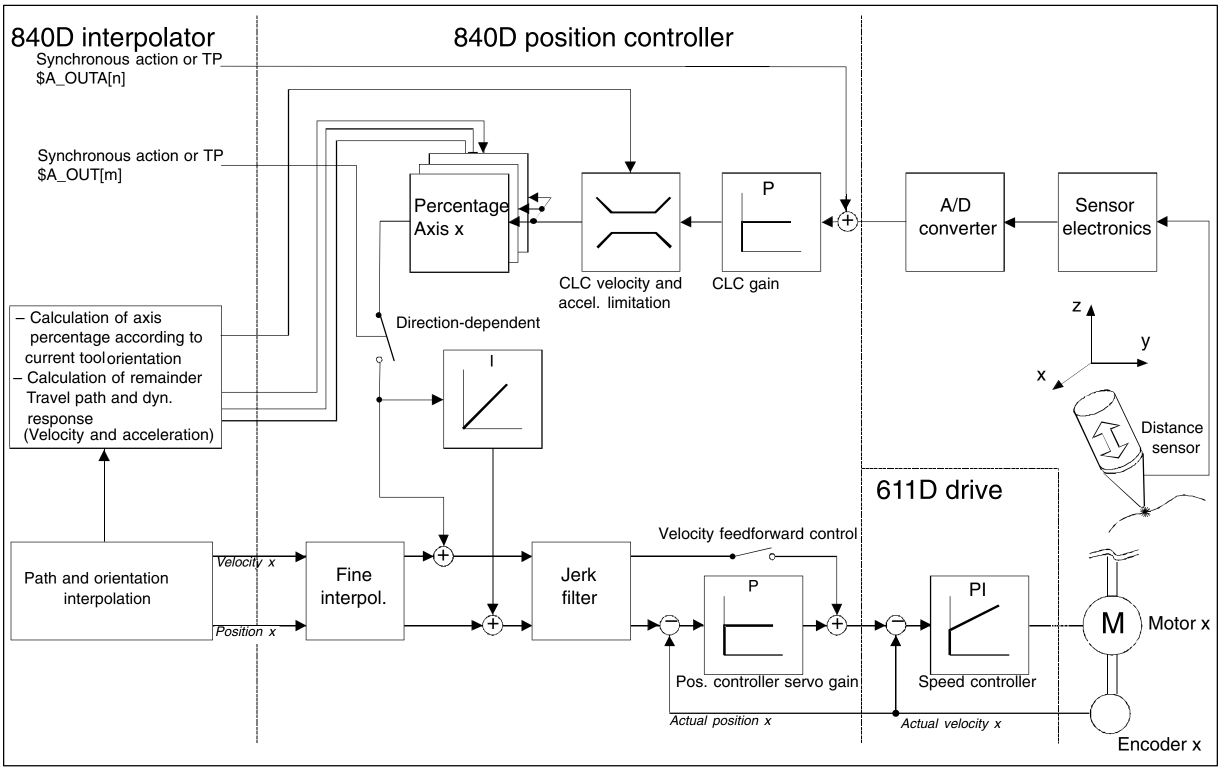



Fig. 2-5 Block diagram of clearance control in position control loop

# 2.4 Programming

The identifiers of all program commands and machine data contain the abbreviation “CLC” which is derived from the English term:
Clearance control.

## 2.4.1 Activating and deactivating the clearance control

**CLC(2)**

Activates the clearance control function.
Evaluation of the additional “Sensor collision” signal is active.

**CLC(1)**

Activates the clearance control.
The collision signal is not evaluated with this setting.

**CLC(0)**

Deactivates the clearance control function.

If the axes are still in motion as a result of the sensor signal, this motion is stopped first. The position reached at zero speed is transferred to the WCS so that the next programmed motion starts at the actual machine position. This function is linked to an automatic preprocessing stop.

**CLC(–1)**

Deactivates the clearance control after retraction to zero offset position.
The axes are moved from the offset position generated by the sensor to the zero offset position so that the next block is started at the originally programmed position.

**RESET**

CLC (0) is applied internally on RESET and at end of program.

**RESET with active clearance control**

SW 5.3 MD 62524: CLC_ACTIVE_RESET and higher can determine whether the 1-dimensional clearance control remains active or is deactivated with RESET (end of program or operator RESET).

**Effectiveness of MD 62524**

MD 62524: CLC_ACTIVE_RESET (only effective for 1-dimensional clearance control).

- MD 62524 = 0: RESET deactivates the clearance control as if CLC(0) has been programmed.
- MD 62524 = 1: With RESET the current clearance control remains unchanged.

If the clearance control is deactivated, the display and the current position in the parts program is synchronized to the position reached by the sensor movement in the position controller.

The machine data MD 62524 is not effective for the 3-dimensional clearance control. As the clearance control is, in this case, realized by the path axes, CLC must always be internally deactivated in case of RESET.

**Syntax**

The CLC (< mode>) command is implemented as a procedure call, i.e. it must be programmed in a separate NC block.

**Error messages**

Call arguments other than those described are rejected by CLC alarm 75005.

The same alarm locks the CLC(2) call if no high-speed input is configured for the collision signal.

If the clearance control is not available on the PC card or has not been activated via machine data, the activation command is rejected in the same way as other unknown commands with standard alarm 12550.

**Path response**

Programming of CLC (< mode>) interrupts G64/G641 path motions with constant feedrate. To activate or deactivate the sensor motion without a drop in path velocity, the control gain can be switched to zero by means of command CLC_GAIN.

As long as the sensor is generating an axis motion, the condition “Exact stop coarse” or “Exact stop fine” cannot be fulfilled and, accordingly, the program does not advance to the next block with G601/G602.

**Sensor collision monitoring**

Via the machine data MD 62504: CLC_SENSOR_TOUCHED_INPUT a digital input for an additional collision signal from the sensor can be configured. This collision monitor can be activated and deactivated block-synchronously through alternate programming of CLC(1)/CLC(2).

As a reaction to the sensor collision signal, the clearance control moves, irrespective of the feedrate override setting, at maximum preset velocity in the plus direction until it reaches the currently valid upper limitation. The path motion is stopped simultaneously.
The machining operation can be continued again after NC start.

**3D control direction with no transformation active**

If the 2D or 3D clearance control is enabled before a 5-axis transformation has been activated with TRAORI(1) or TRAORI(2), the control works in the current tool direction defined by G17/G18/G19 in parallel to one of the Cartesian coordinates.

When the transformation is subsequently activated, the tool orientation determined by the rotary axis positions must correspond to this control direction or else activation of the transformation will be rejected by CLC alarm 75016. When the transformation is temporarily disabled while the clearance control is active, the last tool orientation before the transformation was deactivated determines the control direction.

**Tool radius compensation**

Supplementary condition for tool radius compensation:

In 3D applications, deactivation of the function with CLC(0) may be programmed only if tool radius compensation is not active (G40). If G41/G42 is still active when CLC is deactivated, CLC alarm 75015 is output, causing a block interpretation stop. The synchronization of the WCS position with the offset position of the sensor motion that occurs when CLC(0) is programmed requires an empty internal block buffer and produces a “discontinuous” contour, both of which contradict the geometrically “continuous” operating principle of tool radius compensation.

### 2.4.2 Altering the control gain

**CLC_GAIN=*Gain***

The real number <*gain*> specifies the factor by which the active (see CLC_SEL) gain characteristic (set in machine data) is multiplied (see Fig. 2-6).

**RESET**

CLC_GAIN=1.0 becomes effective automatically after power ON, RESET or end of program.

**Syntax**

Command CLC_GAIN is an NC address, i.e. it can be programmed together with other instructions in one block.

**Application of CLC_GAIN=0.0**

Programming of CLC_GAIN=0.0 “freezes” the position offset value currently reached. This feature can be used, for example, to make the sensor “fly over” blanks that have already fallen out without descending or to prevent it from lifting slightly off concave edges of a 3D contour. If the orientation is changed under CLC_GAIN=0.0 in 3D applications, then the offset vector is rotated simultaneously in space.

**Effectiveness of CLC_GAIN**

The programmed factor for the control gain remains active when the gain characteristic is changed over with CLC_SEL, i.e. it is immediately applied to the newly selected characteristic.

CLCGAIN=1.0 reactivates the gain characteristic set in the machine data.

An alteration in gain becomes effective in the block in which it has been programmed or, if this block does not contain an executable instruction, then in the next executable block.

When a negative factor is programmed, its absolute value is used without an alarm output.

**Caution**

Increasing the gain (CLC_GAIN > 1.0) can lead to oscillation in the controlled axes!

### 2.4.3 Changing the limitations of the control range

**CLC_LIM(*llim*, *ulim*)**

The real numbers <*llim*> and <*ulim*> overwrite the lower and upper limitations of the sensor motion that are set in machine data.
The same unit ([mm] or [inch]) is used as that applied for the programming of positions.

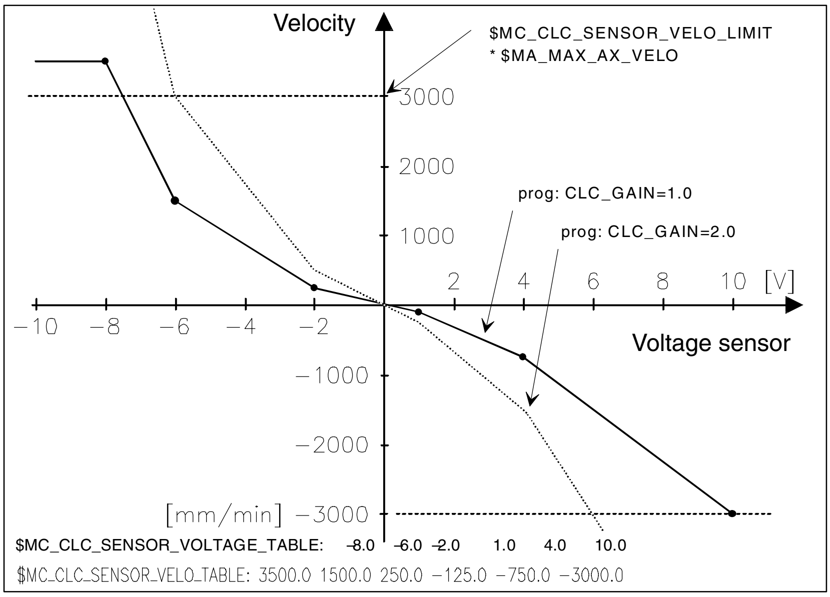



Fig. 2-6 Parameterization of gain characteristic

The limitations act in relation to the currently programmed position. If the limitations are altered such that the instantaneous position is outside the limitation, the sensor is moved back to within the limitation position.

**RESET**

The default setting from the machine data becomes effective automatically after power ON, RESET and end of program.

**Syntax**

The CLC_LIM(...) command is implemented as a procedure call, i.e. it must be programmed in a separate NC block.

**Error messages**

CLC alarm 75005 is output if more than two arguments are programmed.

If CLC_LIM() is programmed without arguments, the default setting from the machine data is restored.

MD 62505: CLC_SENSOR_LOWER_LIMIT[0] and
MD 62506: CLC_SENSOR_UPPER_LIMIT[0].

The second field element in each of these two machine data defines the lowest programmable lower limit or the highest programmable upper limit. If values outside this range are programmed, alarm 75010 is output.

### Block-independent: Direction-dependent blocking of the distance-encoder (SW 5.3 and higher)

**Effectiveness of MD 62523**

Via the machine data MD 62523: CLC_LOCK_DIR_ASSIGN_DIGOUT you can determine that a specified direction of the sensor movement can be blocked. The sensor movement is block direction-dependently by defining the corresponding digital output as follows:

- MD 62523: CLC_LOCK_DIR_ASSIGN_DIGOUT[ ] = 0: Function is inactive

By means of MD 62523 = 0 a blocking of the assigned traversing direction is no longer possible.

- MD 62523: CLC_LOCK_DIR_ASSIGN_DIGOUT[0] = 1: Negative direction
- MD 62523: CLC_LOCK_DIR_ASSIGN_DIGOUT[1] = 2: Positive direction

With the following setting the digital output blocks:

- $A_OUT[1] = 1 the overlaid movement in the negative direction
  (i.e. in the direction of MD 62505: CLC_SENSOR_LOWER_LIMIT)
- $A_OUT[2] = 1 the overlaid movement in the positive direction
  (i.e. in the direction of MD 62506: CLC_SENSOR_UPPER_LIMIT)

Blocking via the $A_OUT[n] parameter can be realized both block-synchronously by the machining program and by synchronous actions.

The function can also be used to block the overlaid sensor movement without disabling the clearance control.

**Inversion of the evaluation**

The input of a negative number in the machine data MD 62523 inverts the internal evaluation of the digital output.

**Example**

In case of the inverted evaluation of the level the setting

MD 62523: CLC_LOCK_DIR_ASSIGN_DIGOUT[0] = –1
causes

the blocking of the movement in the negative direction by means of
$A_OUT[1] = 0 and

the enabling of the movement in the negative direction by means of
$A_OUT[1] = 1.

## 2.4.4 Modification of the setpoint distance

**CLC_VOFF=*offvolt***

The real number <*offvolt*> specifies a signed voltage value in volts that is deducted from the sensor voltage.
This voltage offset alters the control distance vis-à-vis the value set on the operating unit of the sensor.

The quantitative effect of a change in voltage on the distance depends on the relevant setting in the sensor electronics and cannot therefore be standardized generally.

**RESET**

CLC_VOFF=0.0 becomes effective automatically after power ON, RESET and end of program.

**Syntax**

Command CLC_VOFF is an NC address, i.e. it can be programmed together without other instructions in one block.

**Effectiveness of CLC_VOFF**

An alteration in the setpoint distance becomes effective in the block in which it has been programmed or, if this block does not contain an executable instruction, then in the next executable block.

### Block-independent modification of the setpoint distance (SW 5.3 and higher)

**Effectiveness of MD 62522**

Up to now, the modification of the setpoint distance has only been possible block-synchronously with CLC_VOFF. In SW 5.3 and higher, the setpoint distance modification can be programmed block-independently via the machine data MD 62522: CLC_OFFSET_ASSIGN_ANAOUT. This enables an online modification of the setpoint distance.

This machine data defines the number of an analog output, the voltage value of which is subtracted from the input signal of the encoder.

- MD 62522: CLC_OFFSET_ASSIGN_ANAOUT = number of analog output
- MD 62522: CLC_OFFSET_ASSIGN_ANAOUT = 0 the function is inactive

The entry of MD 62522 = 0 deactivates the function.

The analog output can be overlaid by means of the $A_OUTA[n] parameter both block-synchrously by the machining program and by synchronous actions.

**Example**

By means of the block-independent modification of the setpoint distance with the setting MD 62522: CLC_OFFSET_ASSIGN_ANOUT = 2 the synchronous action

ID=1 DO SYNFCT(1, $A_OUTA[2], $A_INA[3])

overlays the CLC encoder input with a voltage that can be specified by the operator during machining via $A_INA[3].
The evaluation function SYNFCT can be parameterized in such a way that the voltage specification for a certain encoder results in a proportional modification of the machining distance.

## 2.4.5 Selecting a sensor characteristic

**CLC_SEL(*KLNo*)**

The integer number <*KLNo*> selects the gain characteristic to be used. The two possible characteristics are entered via the following machine data:

MD 62510: CLC_SENSOR_VOLTAGE_TABLE_1
MD 62511: CLC_SENSOR_VELO_TABLE_1

and

MD 62512: CLC_SENSOR_VOLTAGE_TABLE_2
MD 62513: CLC_SENSOR_VELO_TABLE_2

Characteristic 2 is selected when CLC_SEL(2) is programmed. Any other number selects characteristic 1.

**SW 5.3 and higher**

In SW 5.3 and higher the gain characteristics can be specified with up to 10 intermediate points instead of the usual six.

**RESET**

Characteristic 1 is automatically used after RESET and end of program.

**Syntax**

The CLC (<mode>) command is implemented as a procedure call, i.e. it must be programmed in a separate NC block.

■

## Notes

# Supplementary Conditions 3

## 3.1 Reading in sensor signal in synchronism with NC clock cycle

The sensor is connected via an NCU terminal block and an analog input module to the CNC.

---

**Caution**

No module other than the high-speed DMP compact module with order number: **6FC5 211–0AA10–0AA0** may be used. This is the only module that is capable of the high-speed A/D conversion time of 75μs which clearance control functions require.

The module of identical design with order number: 6FC5 111–0CA04–0AA0 is suitable only for taking temperature measurements. It has a conversion time of 80 ms and is therefore unsuitable for the clearance control functions.

---

**External RC filters**

If RC filters for smoothing the sensor signal are fitted between the sensor output and the input of the A/D converter, then it must be ensured that the resultant time constant is small in comparison to the processing clock cycle of the CNC (1 ms should generally be sufficient).

In principle, a higher signal-to-noise ratio of the analog signal can be obtained by means of efficient screening and not by strong filters.

**Machine data for configuring the analog module**

The following system machine data are relevant for ensuring correct read-in of the sensor signal:

- MD 10362: HW_ASSIGN_ANA_FASTIN (for each analog module)
  Specification of its physical address activates the analog module.
- MD 10384: HW_CLOCKED_MODULE_MASK (for each terminal block)
  The slot of the analog input module on the terminal block must be set to clock-synchronous operation.
  This is done by setting the bit corresponding to the module slot on the terminal block (e.g.: 5th slot: 10 Hex)
- MD 10380: HW_UPDATE_RATE_FASTIO (for each terminal block)
  Selection of the cycle to which the A/D converter is synchronized
  2 = position control cycle for the clearance control in the position controller.
  3 = IPO clock cycle for clearance control via synchronous actions.
- MD 10382: HW_LEAD_TIME_FASTIO (for each terminal block)
  Setting of the period by which the A/D converter is activated before the NC cycle. A setting of 100 starts the A/D conversion 100 ms before the corresponding clock pulse so that the supply of read sensor information is synchronized as closely as possible with the actual positions.

## 3.2 General secondary conditions

**NCU 572.2**

The Clearance Control function can be utilized on NCU 572.2 hardware only on condition that is has been specifically enabled for the customer.

**Response to NC stop from PLC**

If the sensor motion must be stopped at the same time as the path motion, the signal “NC stop axes and spindles” DB21.DBB7.4 in the channel-specific interface can be used.

In this case, the sensor motion is enabled again when NC start is activated to continue program processing.

**Response to “Follow-up” mode**

The overlaid motion is automatically stopped when one of the controlling axes is switched to “Follow-up” mode by an alarm or a PLC input.

**Travel without software limit switches**

If axes involved in implementation of the clearance control function are to be operated in an unreferenced state, then software limit switches with axial machine data

- MD 36100: POS_LIMIT_MINUS,
- MD 36110: POS_LIMIT_PLUS,
- MD 36120: POS_LIMIT_MINUS2,
- MD 36130: POS_LIMIT_PLUS2,

must be set to values outside the actual traversing range since these machine data are always included in calculations for the clearance control.

**No PLC control of sensor input**

High-speed digital or analog input signals that are evaluated in the position control cycle cannot be controlled via the standard PLC interface in DB10.DBB0 or B10.DBB146. This is valid for the analog input of the distance encoder as well as for the digital signal used by the CLC special function “Rapid lift in position controller cycle” (refer to the description of MD 62508: CLC_SPECIAL_FEATURE_MASK).

**Clearance control with gantry axes**

Only one of the axes involved in the clearance control function may be configured as the leading axis in a gantry grouping;

MD 37100: GANTRY_AXIS_TYPE

It is not generally permissible to use gantry following axes for the clearance control.

**Display of axis positions**

The actual axis positions derived from the programmed position and the positional offset resulting from the sensor motion are not displayed in the “Machine” basic display by the standard MMC.

To check these values, they can be monitored separately for each axis under "Actual position". For this purpose, screen "Service display" must be called from the "Diagnosis" area.

**Availability**

The clearance control function is only available in the first NC channel, even on controls with more than one NC channel.

---

**Note**

The clearance control function is available only in the first channel!

---

**Computing time requirements**

The additional computing time required through activation of the clearance control function must be taken into account in control systems in which the cycle setting of the interpolator and position control cycle have been substantially optimized as compared to the default setting:

- NCU2 and 3D clearance control:
  The computing time required for the 3D clearance control function is approximately 300µs per servo cycle and approximately 400µs per interpolation cycle on an NCU2.
- NCU2 and 1D clearance control:
  The computing time requirements are reduced to approximately 260µs per servo cycle and approximately 270µs per interpolation cycle for a 1D clearance control.
- NCU3:
  The computation times stated are cut by half is a NCU573.2 is used.

## 3.3 Displaying status variables

The clearance control function allows various internal variables to be displayed via the external communications system and via global user data (GUDs).

The variables of the external communications system can be displayed by HMI Advanced via NCDDE access. All variables are stored in a data block. The following tables shows a list of all available variables.

| Description of CLC variables | Unit | NCDDE LinkItem | Channel-specific GUD | Access |
|---|---|---|---|---|
| Current overlaid CLC position offset | mm | CLC[0] | CLC_DISTANCE[0] | Read only |
| Minimum CLC position offset | mm | CLC[1] | CLC_DISTANCE[1] | Read/write |
| Maximum CLC position offset | mm | CLC[2] | CLC_DISTANCE[2] | Read/write |
| Current input voltage of sensor | V | CLC[3] | CLC_VOLTAGE[0] | Read only |
| Minimum input voltage | V | CLC[4] | CLC_VOLTAGE[1] | Read/write |
| Maximum input voltage | V | CLC[5] | CLC_VOLTAGE[2] | Read/write |
| 1st component of standardized tool orientation vector | | CLC[6] | — not available — | Read only |
| 2nd component of standardized tool orientation vector | | CLC[7] | — not available — | Read only |
| 3. component of standardized tool orientation vector | | CLC[8] | — not available — | Read only |

**Application of info variables**

Minimum and maximum variables can also be written, making it possible to measure the range of compensated form deviations (position offset) and control errors of brief duration (sensor voltage as dimension before the deviation from setpoint distance).
At the start of a measurement of this type, the relevant minimum must be set to a high value and the maximum to a low value.

With a constant distance and CLC_GAIN=0.0, the noise on the sensor signal actually measured by the converter can be checked on the basis of the voltage values. The min/max calculation is calculated in every position control cycle.

### 3.3.1 Variables available via the external communications system

The following steps must be taken in order to display these variables:

**NCK file**

1. Create a CLC.NCK file which contains the following line:
   LINK("CLC",200,"2 1 1 1 1F# /NC 5 0 1",100)
2. Add the call
   CALL(clc.nsk)
   at the end of the NCDDE311.NSK file stored in directory mmc2.

**LinkItem**

1. The LinkItem property of a DDE control must be set according to the following example:
   label1.LinkItem = "CLC[u1,1,9](""!d%15.4lf"")"
2. You can adjust the format string if necessary.
3. You can use the following code lines to spread the result of the NCDDE access operation among an array of labels:
   ```
   For i = 0 To 8
      label2(i).Caption = Trim$(Mid$(label1.Caption, 1 + 15 * i, 15))
   Next
   ```

**References:** For further information, please refer to "OEM MMC 102/103 Documentation".

### 3.3.2 Variables available via channel-specific GUD

The most important status variables of the clearance control are available on the operator panel front in the display area "Parameters" – "User data".

For this purpose, the appropriate GUDs must be set up. For a detailed description of the procedure to be followed, please refer to Section "File and program management" in the document "SINUMERIK 840D/810D/FM-NC Programming Guide Advanced".

Proceed as follows:

1. Create an INITIAL.INI back-up file.
2. Write a text file containing the following lines on an external PC:
   ```
   %_N_SGUD_DEF
   ;$PATH=/_N_DEF_DIR
   DEF CHAN REAL CLC_DISTANCE[3]  ; CLC variable
   DEF CHAN REAL CLC_VOLTAGE[3]   ; CLC variable
   M30
   ```
3. Load this file to the NC.
4. Load the INITIAL.INI backup file to the NC.

### 3.3.3 Creating alarm texts

Please follow the steps below to create alarm texts:

1. Add an entry for the alarm text files of the technology board in the [TextFiles] section of the C:\OEM\MBDDE.INI file:
   CZYK=C:\OEM\TF_
2. Create language-specific text files TF_xx.COM in directory C:\OEM. xx stands for the language code, e.g. GR for German and UK for English.
3. Enter the following alarm texts here:
   - In TF_GR.COM:
     075000 0 0 “Channel %1 CLC: Incorrect MD configuration, error no: %2”
     075005 0 0 “Channel %1 block %2 CLC: general programming error”
     075010 0 0 “Channel %1 block %2 CLC_LIM exceeds limit set in MD”
     075015 0 0 “Channel %1 block %2 CLC(0) with active TRC”
     075016 0 0 “Channel %1 block %2 CLC: Orientation changed with TRAFOOF”
     075020 0 0 “Channel %1 CLC position offset at lower limit: %2”
     075021 0 0 “Channel %1 CLC position offset at upper limit: %2”
     075025 0 0 “Channel %1 CLC stopped since sensor tip touched”
   - In TF_UK.COM:
     075000 0 0 “Channel %1 CLC: MD configuration error no: %2”
     075005 0 0 “Channel %1 block %2 CLC general programming error”
     075010 0 0 “Channel %1 block %2 CLC_LIM exceeds limit set in MD”
     075015 0 0 “Channel %1 block %2 CLC(0) while CRC is active”
     075016 0 0 “Channel %1 block %2 CLC: orientation changed with TRAFOOF”
     075020 0 0 “Channel %1 CLC position offset at lower limit %2”
     075021 0 0 “Channel %1 CLC position offset at upper limit %2”
     075025 0 0 “Channel %1 CLC stopped since sensor tip touched”
4. Set up the appropriate text files for any further languages you require.
5. Make your changes effective by restarting the MMC.

---

**Note**

For MMC 100, the alarm texts in file ALC.TXT stored on the “System and Application Diskette” supplied with the system can be extended.

---

# Data Descriptions (MD, SD) 4

## 4.1 Machine data of standard system

The clearance control function in the position controller is implemented as a compile cycle application. In addition to the function-specific machine data, the following standard machine data must therefore be set also:

- Option data.

---

**Warning**

This symbol appears whenever material damage **can** occur if the appropriate precautions are not taken.

The functions activated by the option data trigger the corresponding compile cycles. These cycles can have a significant effect on the behavior of the control system and may cause hazardous situations by accessing the NC.
Before a compile cycle is activated, the necessary safety precautions must be taken to protect operating personnel and machine (safety precautions against parameterization or programming errors in the compile cycles may need to be taken).

---

- MD 28090: MM_NUM_CC_BLOCK_ELEMENTS = 4
  Number of internal block elements available to the compile cycle application.
- MD 28100: MM_NUM_CC_BLOCK_USER_MEM = 20
  Memory reserved (in KB) for the internal block elements assigned to the compile cycle application.

---

**Caution**

Modifying this data erases the user memory.

Please save your data before you modify this data !!

---

- MD 10712: NC_USER_CODE_CONF_NAME_TAB[0] =“OMA1”
  MD 10712: NC_USER_CODE_CONF_NAME_TAB[1] =“CLC_GAIN”
  renames the reserved OEM-NC address “OMA1” to “CLC_GAIN”
- MD 10712: NC_USER_CODE_CONF_NAME_TAB[2] =“OMA2”
  MD 10712: NC_USER_CODE_CONF_NAME_TAB[3] =“CLC_VOFF”
  renames the reserved OEM-NC address “OMA2” to “CLC_VOFF”

**Exact stop coarse/ fine reached**

In order to meet the condition for “Exact stop coarse/fine reached”, the axis velocity generated by the sensor must be lower than the zero speed tolerance for the duration of the zero speed delay period. The following machine data can be adjusted in order to optimize block change times when G601 and G602 are active:

- MD 36000: STOP_LIMIT_COARSE[ <clcaxis> ]
  MD 36010: STOP_LIMIT_FINE[ <clcaxis> ]
- MD 36020: POSITIONING_TIME [ <clcaxis> ]
- MD 36040: STANDSTILL_DELAY_TIME[ <clcaxis> ]
  MD 36060: STANDSTILL_VELO_TOL[ <clcaxis> ]

The configuration of the analog input is explained in Section 3.1.

### 4.1.1 Special machine data relating to clearance control

| **62500**<br>MD number | **$MC_CLC_AXNO**<br>Axis assignment for clearance control | |
|---|---|---|
| Default setting: 0 | Min. input limit: –2 | Max. input limit:<br>Maximum number of axes in channel |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: INT | | |
| Meaning: | = 0:<br>Deactivates the clearance control function<br>> 0:<br>Activates 1D clearance control (single-axis clearance control) with the specified channel axis number. This axis must not be a modulo rotary axis.<br>< 0:<br>Activates 3D clearance control.<br>The precondition for this control variant is that at least one of the two possible 5-axis transformations in the channel is configured.<br>–1 selects the first 5-axis transformation configured with $MC_TRAFO_TYPE_x (16 <=transtype <=149) for clearance control function<br>–2 selects the second 5-axis transformation configured in the channel.<br>The overlaid motion acts on the axes that are configured as linear axes in the first three elements of $MC_TRAFO_AXES_IN_x of the selected transformation.<br>It is permissible to configure 3-axis and 4-axis transformations (2D clearance control).<br>Restriction:<br>Only one of the linear axes involved in the clearance control function may be configured as master axis of a gantry grouping in 37100 $MA_GANTRY_AXIS_TYPE.<br>No axis involved in the clearance control function may be configured as the slave axis of a gantry grouping.<br>Incorrectly parameterized configurations are rejected with CLC alarm 75000 during power ON. | |

| | | |
|---|---|---|
| **62502**<br>MD number | **$MC_CLC_ANALOG_IN**<br>Analog input for clearance control function | |
| Default setting: 1 | Min. input limit: 1 | Max. input limit: 8 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: INT | | |
| Meaning: | This machine data defines the number of the analog input that is used for the distance sensor.<br>The hardware module for this input must be correctly configured with the following machine data (see Section 3.1).<br>10362 $MN_HW_ASSIGN_ANA_FASTIN<br>10380 $MN_HW_UPDATE_RATE_FASTIO<br>10382 $MN_HW_LEAD_TIME_FASTIO<br>10384 $MN_HW_CLOCKED_MODULE_MASK<br>Machine data 10300 $MN_FASTIO_ANA_NUM_INPUTS has no relevance with respect to the clearance control function.<br>In contrast to the functions (synchronous actions) implemented in the interpolator, the analog input cannot be controlled via PLC interface DB10.DBW148 ff. | |

| | | |
|---|---|---|
| **62504**<br>MD number | **$MC_CLC_SENSOR_TOUCHED_INPUT**<br>Input bit assignment for the sensor collision signal | |
| Default setting: 0 | Min. input limit: –40 | Max. input limit: 40 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: INT | | |
| Meaning: | If the distance sensor has a special switching signal for indication of sensor tip collision, then it can be applied to the high-speed digital input of the control system programmed in this MD.<br>The inputs are numbered according to the same method used for programming with $A_IN[ <inNr> ]:<br>E.g.: 3rd input on 2nd input byte:<br>$MC_CLC_SENSOR_TOUCHED_INPUT = 19 ; 3 + 2 * 8<br>When negative values are set, the corresponding input signal is inverted internally for processing (fail-safe method).<br>For NC reaction to the collision signal, see Subsection 2.5.1. | |

| 62505<br><br>MD number | $MC_CLC_SENSOR_LOWER_LIMIT<br><br>Lower motion limit of clearance control | |
|---|---|---|
| Default setting: –5.0, –10.0 | Min. input limit: – | Max. input limit: 0.0 |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: mm / inch |
| Data type: REAL | | |
| Meaning: | The first field element of this machine data sets the lower limit for the deviation between the sensor-controlled machine position and the programmed position.<br><br>The limit specified here becomes effective after power ON, RESET and end of program.<br><br>If this limit is reached, PLC signal DB21.DBB37.4 is set and CLC alarm 75020 output.<br><br>The currently effective limit can be modified by the parts program (see Subsection 2.5.3).<br><br>The second field element ($MC_CLC_SENSOR_LOWER_LIMIT[1]) restricts the programmable limit. | |

| 62506<br><br>MD number | $MC_CLC_SENSOR_UPPER_LIMIT<br><br>Upper motion limit of clearance control | |
|---|---|---|
| Default setting: 10.0, 40.0 | Min. input limit: 0.0 | Max. input limit: – |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: mm / inch |
| Data type: REAL | | |
| Meaning: | The first field element of this machine data sets the upper limit for the deviation that may occur as a result of the sensor input between the sensor-controlled machine position and the programmed position.<br><br>The limit specified here becomes effective again after power ON, RESET and end of program.<br><br>If this limit is reached, PLC signal DB21.DBB37.5 is set and CLC alarm 75021 output.<br><br>The currently effective limit can be modified by the parts program (see Subsection 2.5.3).<br><br>The second field element ($MC_CLC_SENSOR_UPPER_LIMIT[1]) restricts the programmable limit. | |

| 62508<br><br>MD number | $MC_CLC_SPECIAL_FEATURE_MASK<br><br>Special functions and operating modes of the clearance control | |
|---|---|---|
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: INT HEX format | | |

| | |
|---|---|
| Meaning: | **Bits 0 and 1:**<br><br>Alarm reaction when CLC motion limits are reached:<br><br>This machine data configures the alarm reaction when motion limits set in MD 62505 and MD 62506 or programmed with CLC_LIM are reached.<br><br>Bit 0 = 0: Alarm 75020 does not stop program processing. This alarm can be acknowledged with the Cancel key.<br><br>Bit 0 = 1: Alarm 75020 stops program processing at the lower limit. The alarm must be acknowledged with RESET.<br><br>Bit 1 = 0: Alarm 75021 does not stop program processing. This alarm can be acknowledged with the Cancel key.<br><br>Bit 1 = 1: Alarm 75021 stops program processing at the upper limit. The alarm must be acknowledged with RESET.<br><br>**Bit 4:**<br><br>Operation as online tool length compensation in direction of orientation<br><br>Bit 4 = 0: The clearance control function works normally.<br><br>Bit 4 = 1: The analog input does not specify a velocity – as under clearance control – but a direct offset position. In this case, the ordinate of the selected sensor characteristic $MC_CLC_SENSOR_VELO_TABLE_x is interpreted in the unit mm or inch instead of in mm/min (inch/min).<br><br>This operating mode can be activated for test purposes and implementation of 3D tool length compensation.<br><br>In this mode, the analog value is read in the position controller clock cycle rather than in the interpolator cycle.<br><br>In this mode, the PLC is able to control the input normally or input analog values via DB10.DBW148 ff.<br><br>The input used must be activated via machine data<br><br>MD 10300: FASTIO_ANA_NUM_INPUTS must be activated!<br>**Bit 5:**<br><br>Mode for rapid retraction in position control cycle<br><br>Bit 5 = 0: The clearance control function works normally.<br><br>Bit 5 = 1: The analog input is not operative. If the digital input configured with MD 62504: CLC_SENSOR_TOUCHED_INPUT is activated (possibly inverted), a retract movement is started in the same position control cycle corresponding to an analog signal setting of +10V during operation as “online tool length compensation” (see bit 4).<br><br>The digital input signal that initiates the retraction motion cannot be controlled via the PLC. In addition to the reaction in the position controller, the “Sensor collision” input with subsequent stop of path motion is treated in the interpolator. This signal branch can be controlled by the PLC via standard signals DB10. DBB0 ff. |

| 62508<br><br>MD number | $MC_CLC_SPECIAL_FEATURE_MASK<br><br>Special functions and operating modes of the clearance control | |
|---|---|---|
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: INT HEX format | Applies from SW: 5.3 | |
| Meaning: | **Bit 8:**<br><br>Mode for alarm output if the lower movement limit has been reached<br><br>Bit 8 = 0: Alarm 75020 is output.<br><br>Bit 8 = 1: Alarm 75020 is not output under the following prerequisite: When the lower movement limit has been reached, the alarm output must not be configured with the alarm reaction Stop and the cancel criterion RESET.<br>Thus, bit 0 = 0 must be set.<br><br>**Bit 9:**<br><br>Mode for alarm output if the upper movement limit has been reached<br><br>Bit 9 = 0: Alarm 75021 is output.<br><br>Bit 9 = 1: Alarm 75021 is not output under the following prerequisite: When the upper movement limit has been reached, the alarm output must not be configured with the alarm reaction Stop and the cancel criterion RESET.<br>Thus, bit 1 = 0 must be set. | |

<table>
<tr><td>62510<br><br>MD number</td><td colspan="2">$MC_CLC_SENSOR_VOLTAGE_TABLE_1<br><br>Coordinate voltage of interpolation points sensor characteristic 1</td></tr>
<tr><td>Default setting:<br><br>–10.0, 10.0, 0.0, 0.0, 0.0, 0.0</td><td>Min. input limit: –10.0</td><td>Max. input limit: +10.0</td></tr>
<tr><td>Changes effective after RESET</td><td>Protection level: 2 / 7</td><td>Unit: Volt</td></tr>
<tr><td>Data type: REAL</td><td colspan="2">In SW 5.3 and higher with up to 10 intermediate point instead of the usual six</td></tr>
<tr><td>Meaning:</td><td colspan="2">This machine data defines the voltage values of the maximum of 10 interpolation points of sensor characteristic 1.<br><br>The associated velocity value of an interpolation point must be entered under the same field index i in machine data $MC_CLC_SENSOR_VELO_TABLE_1[ i ].<br><br>The characteristic is normally entered in the form of two interpolation points as a symmetrical straight line through the zero point:<br><br>E.g.:<br><br>$MC_CLC_SENSOR_VOLTAGE_TABLE_1[ 0 ] = –10.0 ; Volt<br><br>$MC_CLC_SENSOR_VOLTAGE_TABLE_1[ 1 ] = 10.0 ; Volt<br><br>$MC_CLC_SENSOR_VELO_TABLE_1[ 0 ] = 500.0 ; mm/min<br><br>$MC_CLC_SENSOR_VELO_TABLE_1[ 1 ] = –500.0 ; mm/min<br><br>(not all field elements listed are 0.0)<br><br>If this characteristic produces the wrong control direction, i.e. the sensor “escapes” from the workpiece after power ON, then the control direction must be corrected either through polarity reversal of the sensor signal or by alteration of the following data:<br><br>$MC_CLC_SENSOR_VELO_TABLE_1[ 0 ] = –500.0 ; mm/min<br>$MC_CLC_SENSOR_VELO_TABLE_1[ 1 ] = 500.0 ; mm/min<br><br>The following must be noted when more complex characteristics with knee-bends are input:<br><br>• An interpolation point with velocity value 0 must not be positioned at the end of the table. The order in which points are entered in the table is otherwise meaningless.<br><br>• The characteristic must be monotone, i.e. the velocity over voltage values must either be exclusively ascending or exclusively descending.<br><br>• It must not exhibit any velocity step changes, i.e. no differences in velocity at the same voltage value.<br><br>• It must have at least two interpolation points.<br><br>• No more than three interpolation points with positive or negative velocity may be entered.<br><br>• Characteristics in which the line does not pass exactly through the zero point may affect the distance standardization set in the sensor hardware.</td></tr>
</table>

| | | |
|---|---|---|
| **62511**<br><br>MD number | **$MC_CLC_SENSOR_VELO_TABLE_1**<br><br>Coordinate velocity of interpolation points sensor characteristic 1 | |
| Default setting:<br><br>2000.0, –2000.0, 0.0, 0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: mm/min inch/min |
| Data type: REAL | In SW 5.3 and higher with up to 10 intermediate point instead of the usual six | |
| Meaning: | This machine data defines the velocity values of the maximum of 10 interpolation points of sensor characteristic 1.<br><br>The associated voltage value of an interpolation point must be entered under the same field index i in machine data $MC_CLC_SENSOR_VOLTAGE_TABLE_1[ i ].<br><br>Please refer to machine data $MC_CLC_SENSOR_VOLTAGE_TABLE_1 for an exact description of characteristic inputs. | |

| | | |
|---|---|---|
| **62512**<br><br>MD number | **$MC_CLC_SENSOR_VOLTAGE_TABLE_2**<br><br>Coordinate voltage of interpolation points sensor characteristic 2 | |
| Default setting:<br><br>–10.0, 10.0, 0.0, 0.0, 0.0, 0.0 | Min. input limit: –10.0 | Max. input limit: +10.0 |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: Volt |
| Data type: REAL | In SW 5.3 and higher with up to 10 intermediate point instead of the usual six | |
| Meaning: | This machine data defines the voltage values of the maximum of 10 interpolation points of sensor characteristic 2.<br><br>The associated velocity value of an interpolation point must be entered under the same field index i in machine data $MC_CLC_SENSOR_VELO_TABLE_2[ i ].<br><br>Please refer to machine data $MC_CLC_SENSOR_VOLTAGE_TABLE_1 for an exact description of characteristic inputs. | |

| | | |
|---|---|---|
| **62513**<br><br>MD number | **$MC_CLC_SENSOR_VELO_TABLE_2**<br><br>Coordinate velocity of interpolation points sensor characteristic 2 | |
| Default setting:<br><br>2000.0, –2000.0, 0.0, 0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: mm/min inch/min |
| Data type: REAL | In SW 5.3 and higher with up to 10 intermediate point instead of the usual six | |
| Meaning: | This machine data defines the velocity values of the maximum of 10 interpolation points of sensor characteristic 2.<br><br>The associated voltage value of an interpolation point must be entered under the same field index i in machine data $MC_CLC_SENSOR_VOLTAGE_TABLE_2[ i ].<br><br>Please refer to machine data $MC_CLC_SENSOR_VOLTAGE_TABLE_1 for an exact description of characteristic inputs. | |

| | | |
|---|---|---|
| **62516**<br>MD number | **$MC_CLC_SENSOR_VELO_LIMIT**<br>Velocity of the distance control movement | |
| Default setting: 100.0 | Min. input limit: 0.0 | Max. input limit: 100.0 |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: Per cent |
| Data type: REAL | | |
| Meaning: | This machine data defines the maximum velocity of the overlaid control movement as percentage of the maximum axis velocity MD 32000: MAX_AX_VELO.<br>With the 2D and 3D clearance control variants, the reference value used is the maximum velocity of the slowest axis multiplied by $\sqrt{2}$ or $\sqrt{3}$. | |

| | | |
|---|---|---|
| **62517**<br>MD number | **$MC_CLC_SENSOR_ACCEL_LIMIT**<br>Acceleration of clearance control motion | |
| Default setting: 100.0 | Input limit: 0.0 | Max. input limit: 100.0 |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: Per cent |
| Data type: REAL | | |
| Meaning: | This machine data specifies the percentage of the residual acceleration that must be used by the clearance control function.<br>“Residual acceleration” is the acceleration capacity left in reserve by the programmed machining motion of the overlaid control motion in the current control direction. | |

| | | |
|---|---|---|
| **62520**<br>MD number | **$MC_CLC_SENSOR_STOP_POS_TOL**<br>Positional tolerance for status message “Clearance control zero speed” | |
| Default setting: 0.05 | Input limit: 0.0 | Max. input limit: – |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: mm / inch |
| Data type: REAL | | |
| Meaning: | In order to fulfill the axial “Exact stop coarse/fine” conditions, the clearance control must be in the stop state. Since the axes will never be completely still in the control loop closed by means of the sensor, it must be determined using positioning tolerance and dwell time settings at what point axis zero speed must be enabled:<br>If the programmed motion has been completed and the motion component overlaid by the sensor has remained with the position window specified in $MC_CLC_SENSOR_STOP_POS_TOL for the duration of the dwell time set in $MC_CLC_SENSOR_STOP_DWELL_TIME, the exact stop coarse/fine signals are enabled in the PLC interface. The next block cannot be executed until these conditions are fulfilled when G601/G602 is active. | |

| | | |
|---|---|---|
| **62521**<br><br>MD number | **$MC_CLC_SENSOR_STOP_DWELL_TIME**<br><br>Wait time for status message "Clearance control zero speed" | |
| Default setting: 0.1 | Min. input limit: 0.0 | Max. input limit: – |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: s |
| Data type: REAL | In SW 5.3 and higher measure to prevent the system alarm 1011 | |
| Meaning: | This machine data sets the dwell time for detection of zero speed in axes involved in the clearance control function.<br><br>See MD 62520: CLC_SENSOR_STOP_POS_TOL<br><br>If the encoder cannot maintain the position window $MC_CLC_SENSOR_STOP_POS_TOL during distance control operation within the period defined in $MC_CLC_SENSOR_STOP_DWELL_TIME, the system alarm 1011 is consequently output with the ID 140002.<br><br>This alarm message can be prevented by observing the following measures:<br><br>• Switch on the CLC with the distance encoder with the typical operating distance towards a thin sheet metal.<br><br>• Knock against the sheet metal until the laser head moves visibly in order to maintain the defined distance.<br><br>• If this distance has been reached, the sheet metal should no longer be touched and the IS signal "Position reached with exact stop fine" (DB3x, ... DBB60.7) on the MMC should be observed.<br><br>• If this IS signal flashes now or after discharge of the process gas, the following machine data have to be set:<br>MD 36010: STOP_LIMIT_FINE and increase<br>MD 62520: CLC_SENSOR_STOP_POS_TOL.<br>MD 62521: CLC_SENSOR_STOP_DWELL_TIME must possibly be decreased until the IS signal "Position reached with exact stop fine" is stable unless someone knocks against the sheet metal.<br><br>The machine data becomes effective with RESET, i.e. the CLC must possibly be switched on several times. | |
| Related to .... | The time set must not be longer than the maximum waiting time for the output of the exact stop signal set in the axial machine data MD 36020: POSITIONING_TIME. | |

| | | |
|---|---|---|
| **62522**<br><br>MD number | **$MC_CLC_OFFSET_ASSIGN_ANAOUT**<br><br>Modification of the setpoint distance by means of sensor signal override | |
| Default setting: 0 | Min. input limit: –8 | Max. input limit: 8 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: INT | Applies from SW: 5.3 | |
| Meaning: | This machine data defines the number of an analog output, the voltage of which is subtracted from the voltage value of the sensor input signal. The analog output can be overlaid via the $A_OUTA[n] parameter both block-synchronously from the machining program and by means of synchronous actions.<br>Entering zero deactivates this function. | |

| 62523 MD number | $MC_CLC_LOCK_DIR_ASSIGN_DIGOUT Assignment of the digital outputs for disabling the CLC movement | |
|---|---|---|
| Default setting: 0.0 | Min. input limit: –40 | Max. input limit: 40 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: INT | Applies from SW: 5.3 | |
| Meaning: | This machine data defines the number of two digital outputs, each of those can block one direction of the sensor movement.<br>$MC_CLC_LOCK_DIR_ASSIGN_DIGOUT[0] = 1 negative direction<br>$MC_CLC_LOCK_DIR_ASSIGN_DIGOUT[1] = 2 positive direction<br><br>This setting blocks the digital output:<br>$A_OUT[1] = 1 blocks the overlaid movement in the negative direction.<br>$A_OUT[2] = 1 blocks the overlaid movement in the positive direction.<br><br>The blocking can be activated and/or deactivated via the $A_OUT[n] parameter both block-synchronously by the machining program and by synchronous actions.<br><br>By entering zero the assigned traversing direction cannot be blocked.<br>$MC_CLC_LOCK_DIR_ASSIGN_DIGOUT = 0 this function is inactive.<br><br>Entering a negative number in the machine data inverts the level evaluation.<br>$MC_CLC_LOCK_DIR_ASSIGN_DIGOUT = –1 evaluation inversion:<br>$A_OUT[1] = 0 blocks the movement in the negative direction and<br>$A_OUT[1] = 1 enables the movement in the negative direction. | |

| 62524 MD number | $MC_CLC_ACTIVE_AFTER_RESET Clearance control remains active after RESET | |
|---|---|---|
| Default setting: 0 | Min. input limit: – | Max. input limit: – |
| Modification effective after NEW_CONF | Protection level: 2 / 7 | Unit: – |
| Data type: BOOL | Applies from SW: 5.3 | |
| Meaning: | This machine data defines whether the 1-dimensional clearance control remains active or is deactivated during RESET (end of program or operator RESET). $MC_CLC_ACTIVE_AFTER_RESET is effective for 1-dimensional clearance control $MC_CLC_ACTIVE_AFTER_RESET = 0 deactivates the clearance control $MC_CLC_ACTIVE_AFTER_RESET = 1 maintains the current clearance control state during RESET.<br><br>MD 62524 = 0 RESET deactivates the analog clearance control CLC(0).<br>MD 62524 = 1 RESET maintains the current clearance control state during RESET.<br><br>This machine data is not effective for the 3-dimensional clearance control. As the clearance control is, in this case, realized by the path axes, CLC must always be deactivated in case of RESET. | |

| 62525<br><br>MD number | $MC_CLC_SENSOR_FILTER_TIME<br><br>PT1 filtering time constant of sensor signal | |
|---|---|---|
| Default setting: 0.0 | Min. input limit: 0.0 | Max. input limit: – |
| Changes effective immediately | Protection level: 2 / 7 | Unit: s |
| Data type: REAL | Applies from SW: 4.0 | |
| Meaning: | Sets the time constant of the PT1 filter (corresponds to RC element) with high-frequency noise components in the sensor signal can be attenuated.<br><br>Important:<br><br>Every time constant in the control loop reduces the achievable dynamic response of the control system.<br><br>The effect of the filter can be monitored via GUD variables (see Subsection 3.3.2) or the external communications system variables (see Subsection 3.3.1) through resetting of the min/max value for the input voltage (with CLC_GAIN=0.0).<br><br>A setting of zero deactivates the filter completely. | |

■

## Notes

# Signal Descriptions 5

## 5.1 Signals to channel

| DB21, ...<br>**DBB1.4**<br>Data block | **Stop CLC motion**<br>Signal(s) to channel (PLC -> NCK) | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW: 3.6<br>for technology card only |
| Signal state 1 or signal transition 0 ——> 1 | The input signal from the sensor is set internally to zero, i.e. the overlaid sensor motion is stopped and the offset position reached after the stop is “frozen”. The effect is identical to that obtained by programming CLC_GAIN=0.0. In the case of a 2D or 3D clearance control, the orientation of the offset vector continues to change in line with the change in tool orientation, but its length remains unchanged. | |
| Signal state 0 or signal transition 1 ——> 0 | The offset is specified by the current analog signal. | |

| DB21, ...<br>**DBB1.5**<br>Data block | **Feedrate override acts on CLC**<br>Signal(s) to channel (PLC -> NCK) | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW: 3.6<br>for technology card only |
| Signal state 1 or signal transition 0 ——> 1 | The channel-specific override DB21.DBB4 also acts on the clearance control. Override settings of < 100% reduce in the gain characteristic (see Fig. 2-6) the velocity limitation for the overlaid motion set in machine data $MC_CLC_SENSOR_VELO_LIMIT. With settings of >= 100%, the limitations from the machine data are applied. | |
| Signal state 0 or signal transition 1 ——> 0 | The maximum velocity of the control motion is not dependent on the override setting | |
| Application | The difference for the operator is particularly dependent on whether the sensor motion is stopped or not with a 0 override. | |
| Related to .... | Channel-specific override settings DB21.DBB4 and DB21. DBB6.7 | |

## 5.2 Signals from channel

| DB21, ...<br>**DBB37.3**<br>Data block | **CLC is active**<br>Signal(s) from channel (NCK -> PLC) | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW: 3.6<br>for technology card only |
| Signal state 1 or signal transition 0 ––> 1 | The clearance control has been activated from the parts program with command CLC(1) or CLC(2). | |
| Signal state 0 or signal transition 1 ––> 0 | The clearance control is deactivated CLC(0) or CLC(–1) or status after power ON, Reset of end of parts program | |

| DB21, ...<br>**DBB37.4–5**<br>Data block | **CLC motion has stopped**<br>Signal(s) from channel (NCK -> PLC) | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW: 3.6<br>for technology card only |
| Signal state 1 or signal transition 0 ––> 1 | The clearance control has just reached zero speed<br>• either in accordance with the condition specified in machine data $MC_CLC_SENSOR_STOP_POS_TOL and $MA_CLC_SENSOR_STOP_DWELL_TIME<br>• or as a result of a programmed CLC_GAIN=0.0<br>• or in response to PLC signal “Stop CLC motion” DB21.DBB1.4. | |
| Signal state 0 or signal transition 1 ––> 0 | The clearance control is currently causing the axes to move.<br>While the axes are moving as a result of the sensor signal, axial signals “Position reached exact stop coarse/fine” cannot be set. | |
| Related to .... | DB3x.DBB30.6/7 “Position reached, exact stop coarse/fine” | |

| DB21, ...<br>**DBB37.4**<br>Data block | **CLC motion at lower motion limit**<br>Signal(s) from channel (NCK -> PLC) | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW: 3.6<br>for technology card only |
| Signal state 1 or signal transition 0 ––> 1 | The overlaid motion has been stopped at the limit set in $MC_CLC_SENSOR_LOWER_LIMIT or programmed with CLC_LIM(...). Alarm 75020 has been set.<br>If signal DB21.DBB37.5 is set at the same time, the CLC motion has reached a standstill for other reasons. | |
| Signal state 0 or signal transition 1 ––> 0 | The CLC motion has left the lower limitation. | |

| DB21, ...<br>**DBB37.5**<br>Data block | **CLC motion at upper motion limit**<br>Signal(s) from channel (NCK -> PLC) | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW: 3.6<br>for technology card only |
| Signal state 1 or signal transition 0 ——> 1 | The overlaid motion has been stopped at the limit set in $MC_CLC_SENSOR_UPPER_LIMIT or programmed with CLC_LIM(...). Alarm 75021 has been set.<br>If signal DB21.DBB37.4 is set at the same time, the CLC motion has reached a standstill for other reasons. | |
| Signal state 0 or signal transition 1 ——> 0 | The CLC motion has left the upper limitation. | |

■

# Notes

# 6 Examples

## 6.1 General start-up of a compile cycle function

---

**Note**

With SW 6.4, the compile cycles are supplied as loadable modules. The general procedure for installing such compile cycles can be found in TE0. The specific installation measures for this compile cycle can be found from Section 6.2 onwards.

---

**Requirements**

The following requirements must be met with reference to the HMI and NCK:

- HMI Advanced SW 3.5 or higher
- An NCK technology card with the “1D/3D clearance control in position controller” function must be installed.

**Saving SRAM contents**

As the first step in installing a compile cycle function, the original card inserted in the NCU must be replaced by the technology card.
This is identical to the procedure followed for upgrading the NCU to a later software version and likewise requires the static (battery-backed) control system memory to be erased. When this is done, all data stored in the SRAM are lost and they must therefore be saved beforehand.

**References:** For a detailed description, please refer to the Manufacturer/Service Documentation “SINUMERIK 840D/ SIMODRIVE 611D Installation and Start-Up Guide”:

1. Enter the machine manufacturer password.
2. Change to the “Services” operating area.
3. Press softkey “Series start-up”.
4. Select “NC” and “PLC” as the areas to be saved and enter a name of your choice for the archive file to be created on the hard disk. Finish by pressing the RETURN key.
5. If the control system contains machine-specific compensation data, then these must be saved in a separate archive file:
   Press softkey “Data Out” and select under item “NC active data” the following data as required:
   “Measuring system compensations”,
   “Sag/Angularity comp.” and
   “Quadrant error compensation”.
   Save these data by selecting softkey “Archive ...” and specify another file name for a second archive file.

Keep the archive files you have created in a safe place. They will allow you to restore original settings in your system.

**Insert the PC card**

- Switch off control system
- Insert the PC card with the new firmware ( technology card ) in the PCMCIA slot of the NCU
- Then proceed as follows:

1. Turn switch S3 on the front panel of the NCU to 1
2. Switch the control system back on again.
3. When the system powers up, the firmware is copied from the PC card into the NCU memory.
4. Wait until number “6” is displayed on the NCU digital display (after approximately one minute).
5. Turn switch S3 back to zero.

---

**Note**

If number “6” does not appear, then an error has occurred.
- Incorrect PC card (e.g. card for NCU2 in NCU3 hardware)
- Card hardware defective

---

**Copy back SRAM contents**

To copy the saved data back into the control system, proceed as described in Section 12.2 (series start-up). Please read all information provided by the manufacturer about new software versions.

- Enter the machine manufacturer password.
- Select “Data In” and “Archive...”. Then load the archive with backup compensation data (if applicable).

## 6.2 Start-up of clearance control

**Memory configuration for clearance control**

To start up the clearance control on the newly installed technology card, the SRAM must be reformatted because two memory-configuring machine data need to be changed so that the internal block memory required can be configured.

To do so, proceed as follows:

- Modify the memory-configuring channel machine data
  MD 28090: MM_NUM_CC_BLOCK_ELEMENTS = 4 and
  MD 28100: MM_NUM_CC_BLOCK_USER_MEM = 20
  Subsequently, do **not** power up the control but create **another** archive file via the "Series start-up" softkey.
- Power up the control system **now** (softkey "NCK reset" or power ON) to re-format the SRAM.
- To restore the SRAM contents, load the last archive file you created plus the compensation data archive (if you created one for the card replacement) back into the control system.

**Option data for compile cycles**

Set the option for compile cycle function 1

**Machine data for CLC program commands**

Rename the standard identifiers "OMA1" and "OMA2" as CLC-specific identifiers.

MD 10712: NC_USER_CODE_CONF_NAME_TAB[0] = "OMA1"

MD 10712: NC_USER_CODE_CONF_NAME_TAB[1] = "CLC_GAIN"

MD 10712: NC_USER_CODE_CONF_NAME_TAB[2] = "OMA2"

MD 10712: NC_USER_CODE_CONF_NAME_TAB[3] = "CLC_VOFF"

**High-speed analog input**

Configure this analog input with the following machine data:

MD 10362: HW_ASSIGN_ANA_FASTIN

MD 10384: HW CLOCKED_MODULE_MASK

MD 10380: HW_UPDATE_RATE_FASTIO = 2

MD 10382: HW_LEAD_TIME_FASTIO = 100

Run up the NCK software again.

After the software has run up, the special data of the clearance control should be visible at the end of the channel machine data list (look for "CLC" or "62500").

**Special CLC machine data**

First of all, set the axis configuration and the number of the analog input (with reference to all analog inputs that may be activated in the control) that you wish to use.

MD 62500: CLC_AXNO

MD 62502: CLC_ANALOG_IN

**Sensor connection analog input**

Connect the bipolar sensor output to terminals 3 and 4 of the analog input module on the terminal block. Check the wiring of the ground/screen against the wiring diagrams in hardware documentation “SINUMERIK 840 Configuring of DMP Compact Module 1E NC Analog”.
Check the control direction of the clearance control by starting a simple parts program containing activation command CLC(1). Make sure that there is no risk of collision between the axes within the preset motion area of –5 mm to + 10 mm.
If you find that the control direction is wrong (sensor moves away from workpiece), reverse the polarity of the analog input or change the signs of all values in machine data $MC_CLC_SENSOR_VELO_TABLE_x.

**Check the sensor signal**

To check the noise level on the sensor signal and calculate the sensor characteristic if necessary, you can set up special GUD variables according to the method described in Section 3.3.

Variable CLC_VOLTAGE[0] indicates the response of the sensor at various distances from the workpiece as interpreted by the control system when the (CLC(1)) function is activated, but the control motion is blocked (CLC_GAIN=0.0).

**Optimization**

By optimizing the CLC-specific machine data and standard machine data (especially feedforward control) that are relevant in terms of dynamic response, you can adapt the clearance control optimally to the meet the requirements of your machining process.

**Alarm texts**

Extend the language-dependent alarm text files as described in Section “Creating alarm texts”.

Finish the start-up process by creating a complete backup of all your data.

■

# 7 Data Fields, Lists

## 7.1 Alarms

**12550** **Channel %1 block %2 name %3 not defined or option not installed**

Explanation
%1 = channel number
%2 = block number, label
%3 = source symbol; in this case: CLC
If the clearance control is not available on the PC card or has not yet been activated via machine data, the corresponding parts program commands (prefix "CLC") are rejected as invalid syntax.

Reaction
Abortion of parts program interpretation.
Alarm signal in PLC interface.

Remedy
Use technology PC card and install function.

Reset criterion
RESET

**75000** **Channel %1 CLC: Incorrect MD configuration, error no: %2**

Explanation
The following error has been detected in the clearance control machine data during power-up:

Error no. = –1:
The interpolation points of one of the two sensor characteristics are not strictly monotone ascending or descending.

Error no. = –2:
One of the two sensor characteristics has less than 2 valid interpolation points.

Error no. = –3:
One of the two sensor characteristics has more than three interpolation points with negative velocity or more than three interpolation points with positive velocity.

Error no. = –4:
The digital input set in MD $MC_CLC_SENSOR_TOUCHED_INPUT for monitoring sensor collision is not activated in the control system (10350 $MN_FASTIO_DIG_NUM_INPUTS).

Error no. = –5:
No high-speed input has been assigned to special function "Rapid retraction in position controller" via MD $MC_CLC_SENSOR_TOUCHED_INPUT.

Error no. = –6:
The axis selected in MD $MC_CLC_AXNO for the clearance control function is not active in the channel.

Error no. = –7:
The 5-axis transformation (24100 $MC_TRAFO_TYPE_x) selected in MD $MC_CLC_AXNO for the clearance control function is not configured in the channel.

Error no. = –8:
More than one of the axes involved in the clearance control is a leading axis in a gantry grouping 37100 $MA_GANTRY_AXIS_TYPE.

Error no. = –9:
One of the axes involved in the clearance control is a following axis in a gantry grouping 37100 $MA_GANTRY_AXIS_TYPE.

| | |
|---|---|
| Reaction | Dropout of signal “Mode group READY”.<br>Alarm signal in PLC interface. |
| Remedy | Correct the relevant machine data. |
| Reset criterion | Power ON |

**75005** **Channel %1 block %2 CLC: General programming error**

| | |
|---|---|
| Explanation | The activation/deactivation command for the clearance control “CLC(..)” accepts only the values 2, 1, 0 and –1 as call parameters.<br>This alarm signals that parameters are incorrect or missing.<br><br>The activation command CLC(2) with monitoring of sensor collision signal is accepted only if a valid digital input is configured for the monitoring signal in MD $MC_CLC_SENSOR_TOUCHED_INPUT. |
| Reaction | Abortion of parts program interpretation.<br>Alarm signal in PLC interface. |
| Remedy | Correct parts program.<br>Configure digital input for collision evaluation in MD if necessary. |
| Reset criterion | RESET |

**75010** **Channel %1 block %2 CLC_LIM value higher than MD limit**

| | |
|---|---|
| Explanation | One of the limitations programmed with CLC_LIM(...,...) for the position offset of the clearance control is greater than the permissible limitation set in the associated MD $MC_CLC_SENSOR_LOWER_LIMIT[1] or $MC_CLC_SENSOR_UPPER_LIMIT[1]. |
| Reaction | Abortion of parts program interpretation.<br>Alarm signal in PLC interface. |
| Remedy | Correct parts program.<br>Raise limitation in appropriate machine data if necessary. |
| Reset criterion | RESET |

**75015** **Channel %1 block %2 CLC(0) with active TRC**

| | |
|---|---|
| Explanation | The 3D clearance control has been switched off with CLC(0) while tool radius compensation is still active (G41/G42).<br>Since CLC(0) empties the internal block buffer and transfers the current position offset of the clearance control as a “contour jump” to the interpreter, TRC must be deactivated when this command is issued. |

| | |
|---|---|
| Reaction | Abortion of parts program interpretation.<br>Alarm signal in PLC interface. |
| Remedy | Correct parts program.<br>Switch off active G41/G42 before CLC(0) or<br>do not switch of clearance control, but just “freeze” temporarily (CLC_GAIN=0.0) or<br>cancel the position offset mechanically with CLC(–1). |
| Reset criterion | RESET |

**75016** **Channel %1 block %2 CLC: Orientation changed with TRAFOOF active**

| | |
|---|---|
| Explanation | 1.<br>The 2D/3D clearance control has been switched off before the transformation. The tool direction according to G17/G18/G19 has been applied as the control direction. Switching on the transformation with rotary axis settings that define a different tool orientation requires an orientation step change and is therefore rejected.<br>2.<br>The transformation has been switched off temporarily (TRAFOOF) while clearance control is still active. When the transformation is switched on again, the tool orientation must be the same as when it was switched off, i.e. the rotary axes must not be moved while the transformation is deactivated. |
| Reaction | Abortion of parts program interpretation.<br>Alarm signal in PLC interface. |
| Remedy | Correct parts program.<br>Do not switch on the clearance control until the transformation is already active or make sure that the required conditions relating to orientation are observed. |
| Reset criterion | RESET |

**75020** **Channel %1 CLC position offset at lower limit %2**

| | |
|---|---|
| Explanation | The position offset generated by the overlaid motion has reached the lower limit set in MD $MC_CLC_SENSOR_LOWER_LIMIT or programmed with CLC_LIM(...,...). |
| Reaction | Depending on setting in bit 0 of MD $MC_CLC_SPECIAL_FEATURE_MASK:<br>Bit 0 = 0: Alarm display only, no internal reaction<br>Bit 0 = 1: Stop programmed motion, NC start interlock |
| Remedy | Check position and shape of workpiece.<br>Program extended limitation range if necessary. |
| Reset criterion | Depending on setting in bit 0 of MD $MC_CLC_SPECIAL_FEATURE_MASK:<br>Bit 0 = 0: Cancel key<br>Bit 0 = 1: RESET request |

**75021** **Channel %1 CLC position offset at upper limit %2**

| | |
|---|---|
| Explanation | The position offset generated by the overlaid motion has reached the limit set in MD $MC_CLC_SENSOR_UPPER_LIMIT or programmed with CLC_LIM(...,...). |

Reaction
Depending on setting in bit 1 of MD
$MC_CLC_SPECIAL_FEATURE_MASK:
Bit 1 = 0: Alarm display only, no internal reaction
Bit 1 = 1: Stop programmed motion, NC start interlock

Remedy
Check position and shape of workpiece.
Program extended limitation range if necessary.

Reset criterion
Depending on setting in bit 1 of MD
$MC_CLC_SPECIAL_FEATURE_MASK:
Bit 1 = 0: Cancel key
Bit 1 = 1: RESET

**Reference**
Program command CLC_LIM(...,...);
MD $MC_CLC_SENSOR_UPPER_LIMIT
MD $MC_CLC_SPECIAL_FEATURE_MASK

**75025** **Channel %1 CLC stopped since sensor tip touched**

Explanation
The collision monitor of the sensor tip has signaled “Sensor touched”.

Reaction
A retraction motion to the upper limitation of the position offset ($MC_CLC_SENSOR_UPPER_LIMIT) is started. The maximum velocity and acceleration reserves available in the control direction are used for this purpose.
The feedrate override setting has no effect on this retraction motion.
The path motion is stopped simultaneously.

Remedy
The parts program can be continued with NC start.
The overlaid motion then returns to the control distance.

Reset criterion
NC start or RESET

## 7.2 Machine data

| Number | Identifier name | Ref. |
|---|---|---|
| **Drive machine data** | | |
| 1502 | **SPEED_FILTER_1_TIME**[**n**]<br>Time constant, speed setpoint filter 1 | /DD2/ |
| 1503 | **SPEED_FILTER_2_TIME**[**n**]<br>Time constant, speed setpoint filter 2 | /DD2/ |
| **General ($MN_ ...)** | | |
| 10300 | FASTIO_ANA_NUM_INPUTS<br>Number of active analog NCK inputs | A2 |
| 10350 | FASTIO_DIG_NUM_INPUTS<br>Number of active digital NCK input bytes | A2 |
| 10362 | HW_ASSIGN_ANA_FASTIN<br>Hardware assignment of external analog NCK inputs: 0...7 | A2 |
| 10380 | **HW_UPDATE_RATE_FASTIO**<br>Update cycle of synchronously clocked external NCK input/output modules | A2 |
| 10382 | **HW_LEAD_TIME_FASTIO**<br>Pretrigger time of synchronously clocked external NCK inputs/outputs. Terminal block: 0...3 | A2 |
| 10384 | **HW_CLOCKED_MODULE_MASK**<br>Synchronous processing of individual external input/output modules. Terminal block: 0...3 | A2 |
| 10712 | **NC_USER_CODE_CONF_NAME_TAB** List of renamed NC identifiers | /PA/ |
| 19600 | **$ON_CC_EVENT_MASK[0]** Enabling of compile cycle events | |
| **Channel-specific ($MC_ ...)** | | |
| | | |
| 28090 | MM_NUM_CC_BLOCK_ELEMENTS Number of compile cycle block elements (DRAM) | S7 |
| 28100 | MM_NUM_CC_BLOCK_USER_MEM Memory space for compile cycle block elements (DRAM) in KB | S7 |
| **Axis-specific ($MA_...)** | | |
| 32070 | CORR_VELO Axis velocity for handwheel, external zero offsets, SA clearance control | **H1, K2, W4** |
| 32410 | AX_JERK_TIME Time constant for axial jerk filter | B2 |
| 32610 | VELO_FFW_WEIGHT Feedforward control factor for velocity feedforward control | K3 |
| 36000 | STOP_LIMIT_COARSE Exact stop coarse | B1 |
| 36010 | STOP_LIMIT_FINE Exact stop fine | B1 |
| 36040 | STANDSTILL_DELAY_TIME Delay time zero speed monitoring | A3 |
| 36060 | STANDSTILL_VELO_TOL Max. velocity for axis/spindle stopped | A2 |
| 36750 | AA_OFF_MODE Mode of value calculation with axial position override | S5 |
| **Channel-specific machine data of clearance control ($MC_ ...)** | | |
| 62500 | CLC_AXNO Axis assignment for clearance control | |
| 62502 | CLC_ANALOG_IN<br>Analog input for clearance control | |
| 62504 | CLC_SENSOR_TOUCHED_INPUT<br>Assignment of an input bit for the “Sensor collision” signal | |
| 62505 | CLC_SENSOR_LOWER_LIMIT<br>Lower motion limit of clearance control | CLC_LIM() |
| 62506 | CLC_SENSOR_UPPER_LIMIT<br>Upper motion limit of clearance control | CLC_LIM() |
| 62508 | CLC_SPECIAL_FEATURE_MASK<br>Special functions and operating modes of clearance control | |

| | | |
|---|---|---|
| 62510 | CLC_SENSOR_VOLTABE_TABLE_1<br>Coordinate voltage of interpolation points of sensor characteristic 1 | CLCGAIN |
| 62511 | CLC_SENSOR_VELO_TABLE_1<br>Coordinate velocity of interpolation points of sensor characteristic 1 | |
| 62512 | CLC_SENSOR_VOLTABE_TABLE_2<br>Coordinate voltage of interpolation points of sensor characteristic 2 | |
| 62513 | CLC_SENSOR_VELO_TABLE_2<br>Coordinate velocity of interpolation points of sensor characteristic 2 | |
| 62516 | CLC_SENSOR_VELO_LIMIT<br>Velocity of clearance control motion | MAX_AX_<br>VELO |
| 62516 | CLC_SENSOR_ACCEL_LIMIT<br>Acceleration of clearance control motion | MAX_AX_<br>ACCEL |
| 62520 | CLC_SENSOR_STOP_POS_TOL<br>Positional tolerance for status message “Clearance control zero speed” | |
| 62521 | CLC_SENSOR_STOP_DWELL_TIME<br>Wait time for status message “Clearance control zero speed” | |
| 62522 | CLC_OFFSET_ASSIGN_ANAOUT<br>Modification of the setpoint distance via sensor signal override (SW 5.3 and higher) | |
| 62523 | CLC_LOCK_DIR_ASSIGN_DIGOUT<br>Assignment of the digital outputs for CLC movement deactivation (SW 5.3 and higher) | |
| 62524 | CLC_ACTIVE_AFTER_RESET<br>Clearance control remains active after RESET (SW 5.3 and higher) | |
| 62525 | CLC_SENSOR_FILTER_TIME Time constant of PT1 filter of sensor signal | |

## 7.3 Interface signals

| DB no. | Bit, byte name | | Ref. |
|---|---|---|---|
| **Channel-specific** | | | |
| 21 | 1.4 | Stop CLC motion | |
| 21 | 1.5 | Feedrate override acts on CLC | |
| 21 | 37.3 | CLC is active | |
| 21 | 37.4–5 | CLC motion has stopped | |
| 21 | 37.4 | CLC motion at lower motion limit | |
| 21 | 37.5 | CLC motion at upper motion limit | |

■

# SINUMERIK 840D/840Di/810D Description of Functions Special Functions (Part 3)

# Analog Axis (TE2)



■

# Notes

# 1 Brief description

**840D**

The “analog axis” function was supplied as a compile cycle up to SW 6. This function can now be implemented with the aid of the hydraulics module. It is therefore no longer available as a compile cycle.

**840Di**

On the 840Di, analog axis is implemented via an ADI4 board.

■

# Notes

# Detailed Description 2

## 2.1 General

With SW 4.3, the “Analog axis” function allows up to 8 of the available NC axes to be controlled by an analog drive (e.g. SIMODRIVE 611A) via a +/– 10V speed interface. The function is designed for individual motors on a machine that cannot be controlled by digital drive systems such as, for example, large spindle motors or single motors for tool changers.

**Number of analog axes**

The maximum number of analog axes depends on the maximum number of NC axes available.

Axes controlled by analog drives are included in the maximum number of available NC axes in the system and, when used for this purpose, reduce the number of available digital axes.

| | NC axes | Analog axes |
|---|---|---|
| NCU2 | 12 | 3 |
| NCU3s | 12 | 3 |
| NCU3 | 31 | 8 |

**Function**

As regards scope of applications, analog axes are not subject to any particular restrictions that do not also apply to digital axes. There is no special dependency on particular channels, i.e. analog axes can be programmed in different channels. Axes can also be exchanged between channels. All functions of normal NC axes/spindles such as those listed below are also available for analog axes:

- Programming from parts programs
- Traversal from PLC
- Manual traversal, etc.

**Speed setpoint**

The speed setpoint of an analog axis is made available on a DMP module on the NCU terminal block from where it is taken to the analog drive.

**Actual position value**

The actual position value of the axis is acquired by a signal generator. An unassigned measured-value input for the direct measuring system of an active digital drive is used as the measurement input.

**Caution**

You must observe the different dynamic responses of the drives in an interpolation group that has analog and digital drives (following error, drift).

## 2.2 Hardware configuration

Fig. 1 shows the hardware configuration of an analog axis:

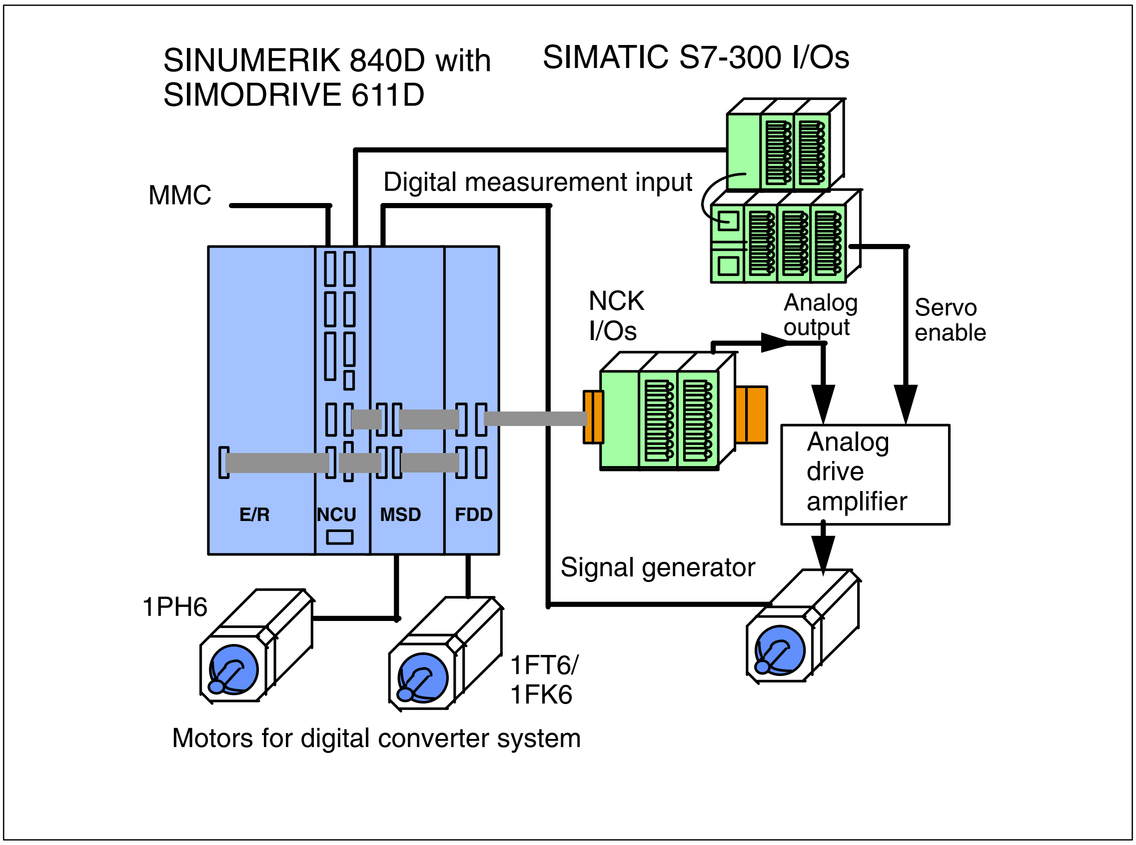



Fig. 2-1 Hardware configuration of an analog axis

Hardware requirements

All hardware components required are listed below:

- NCU terminal block (6FC5 211–0AA00–0AA0)
- DMP output module (6FC5 111–0CA05–0AA0) for each analog axis
- Cable for setpoint from DMP output module to analog drive.
- Analog drive amplifier, e.g.: SIMODRIVE 611A
- Signal generator on motor
- Actual-value cable for direct measuring system (15-pin connector)
- Active SIMODRIVE 611D drive system with unassigned actual-value input for direct measuring system (socket connector X422) and submodule for direct measuring system

## 2.3 Configuration

The following description explains how an NC axis can be configured via machine data such that it can be controlled by means of an analog drive amplifier.

**Declare axis as analog axis**

Every NC axis can be operated as an analog axis. The maximum number depends on the number of NC axes.

Since the function is implemented via compile cycles, machine data MD 19600: CC_EVENT_MASK[1] = 64 (HEX) must be set to activate compile cycle application 2.

---

**Warning**

Failure to take appropriate precautions **can** have undesirable consequences.

The functions activated via MD 19600: CC_EVENT_MASK[n] initiate corresponding compile cycles. These cycles can have a significant effect on the behavior of the control system and may cause hazardous situations by accessing the NC.
Before a compile cycle is activated, the necessary safety precautions must be taken to protect operating personnel and machine (safety precautions against parameterization or programming errors in the compile cycles may need to be taken).

---

An NC axis is then declared to be an analog axis in axial machine data MD 63530: ANALOG_AXIS. The input value in this case is the modulo number of the DMP module on the NCU terminal block (input setting can be 1 to 8). The analog axis is deactivated when 0 is input.

---

**Note**

MD 63530: ANALOG_AXIS is not displayed until the control has booted once after setting of MD 19600: CC_EVENT_MASK[0].

---

**Alarms in response to configuring errors**

If more than the permissible number of analog axes are activated, then alarm 75100 “Too many analog axes configured” is displayed after power-up.

If the same DMP module is assigned to more than one analog axis, then alarm 75101 “DMP module no.: %1 assigned more than once” is displayed.

The DMP modules must not be assigned to other control system functions (e.g. synchronous actions) since this would interfere with the servo output.
DMP modules are assigned to the interpolator via machine data MD 10310: FASTIO_ANA_NUM_OUTPUTS. For this reason, the numbers of DMP modules used for analog axes must be higher than the contents of machine data 10310 or else alarm 75102 “DMP module no.: %1 assigned to system functions” will be activated.

**Machine data for configuring the analog module**

The following system machine data are relevant in ensuring correct output of the speed setpoint via the analog module:

- MD 10364: HW_ASSIGN_ANA_FASTOUT (for each analog module)
  Specification of its physical address activates the analog module.
- MD 10384: HW_CLOCKED_MODULE_MASK (for each terminal block)
  The slot of the analog output module on the terminal block must be set to clock-synchronous operation.
  This is done by setting the bit corresponding to the module slot on the terminal block (e.g.: 5th slot: 10 Hex)
- MD 10380: HW_UPDATE_RATE_FASTIO (for each terminal block)
  Selection of clock cycle with which A/D converter is synchronized
  2 = position control cycle for analog axis.

## 2.4 Setpoint

**Setpoint in hardware**

The speed setpoint of an analog axis is written to a digital/analog converter in every position controller cycle and is available there as a signal in the +/– 10V range.

A digital/analog converter is required for each analog axis. It is inserted in the NCU terminal block as a DMP output module. A maximum of 8 DMP modules can be inserted in an NCU terminal block.

The setpoint is taken to the analog drive amplifier (e.g. SIMODRIVE 611A) which is driving the motor.

**Servo enable**

Other signals such as servo enable or NC Ready required by the analog drive amplifier are not applied at the NCU terminal block, but must be derived from the SIMATIC I/O devices.

**Configure setpoint**

Analog axes do not transfer an internal setpoint to a digital drive. For this reason, they must be configured as simulated axes with respect to setpoints. This configuration is programmed via machine data 30100–30130.

| | |
|---|---|
| MD 30100: CTRLOUT_SEGMENT_NR = 1 | Bus segment 840D |
| MD 30110: CTRLOUT_MODULE _NR= | The module number of an unassigned module must be entered here even through the setpoint output is simulated. This module need not actually exist in the hardware. |
| MD 30120: CTRLOUT_NR = 1 | Setting is always 1 for 840D |
| MD 30130: CTRLOUT_TYPE = 0 | Simulated setpoint |

**Normalize setpoint**

Axial machine data MD 32250: RATED_OUTVAL and MD 32260: RATED_VELO are set to normalize and limit the output voltage. The maximum motor speed in rev/min is entered in MD 32260: RATED_VELO. The percentage value in MD 32260: RATED_VELO specifies the voltage at maximum motor speed with respect to +/– 10 V. A setting of 80% means +/– 8V on the DMP module at maximum motor speed. The percentage value must be adjusted according to the analog drive amplifier used.

**Example:**

| | |
|---|---|
| Maximum motor speed 6000 rev/min | MD 32260: RATED_VELO = 6000 |
| 8V at motor speed of 6000 rev/min | MD 32250: RATED_OUTVAL = 80 |

4V are present on the DMP module at a motor speed of 3000 rev/min.

---

**Note**

Changes to machine data MD 32250: RATED_OUTVAL and MD 32260: RATED_VELO are not activated by NewConfig, but only after a RESET. This is applicable only in relation to the analog axis function.

---

**Drift compensation**

A drift that needs to be compensated by the position controller occurs in every analog drive. The “Analog axis” functions provides two different options for compensating drift. One of these options involves a constant drift value that is entered in machine data MD 36720: DRIFT_VALUE. This value is added to the position controller setpoint in every position controller cycle and output.

The second method involves automatic drift compensation. This is activated via machine data MD 36700: DRIFT_ENABLE = 1. A maximum drift value for automatic compensation is activated in machine data MD 36710: DRIFT_LIMIT. The drift is compensated as soon as the analog axis is operating under closed-loop control, no setpoints are applied from the IPO and the axis is stationary. As soon as the axis moves again, the last compensation value is “frozen” and added to the setpoint in every position control cycle. If the compensation value increases above the value set in machine data MD 36710: DRIFT_LIMIT, alarm 75110 “Axis X1 has reached drift limit” and the drift value is limited.

---

!

**Important**

Changes to machine data MD 36700: DRIFT_ENABLE, MD 36710: DRIFT_LIMIT, MD 36720: DRIFT_VALUE are not activated by NewConfig, but by a RESET. This is applicable only in relation to the analog axis function.

---

## 2.5 Actual value

**Actual value in hardware**

The actual position value of the analog axis is acquired by a signal generator. An unassigned measured-value input for the direct measuring system on an active digital drive (SIMODRIVE 611D) is used as the measured-value input. This is the lower 15-pin measured-value input with designation X422 on the digital drive.

As an example, a machine has 3 digital and one analog axis. The actual value of the analog axis can be taken to a direct measured-value input of one of the 3 digital drives. The selected digital drive must have a submodule for a direct measuring system. In other words, an active digital drive is required for every analog axis.

**Configure actual value**

The analog axis must be configured such that its actual value is applied to an unassigned measured-value input for the direct measuring system on an active digital drive. Since the analog axis has only one position measuring system, only one measuring system is activated in the machine data. Machine data 30200–30240 are used for this purpose.

| | |
|---|---|
| MD 30200: NUM_ENCS = 1 | Analog axis has one measuring system |
| MD 30210: ENC_SEGMENT_NR = 1 | Bus segment 840D |
| MD 30220: ENC_MODULE_NR = | Module number of active digital drive |
| MD 30230: ENC_INPUT_NR[0] = 2 | Direct measuring system |
| MD 30240: ENC_TYPE[0] = 1 | Encoder type: Signal generator |

The first measuring system is activated by the PLC-to-axis signal DB31–48, DBX1.5. It is not possible to switch over to measuring system 2 and any attempt to do so is ignored.

■

# Supplementary Conditions 3

**NCU 572.2**

The Analog Axis function can be utilized on NCU 572.2 hardware only on condition that is has been specifically enabled for the customer.

**SINUMERIK 840Di**

The operation of analog axes via the PROFIBUS DP of the SINUMERIK 840Di is available soon.

The compile cycle function of the SINUMERIK 840D is, for the time being, only available on request for the SINUMERIK 840Di.

## 3.1 Effectiveness of machine data

Changes to the following machine data do not take effect with NewConfig, but only on RESET:

**Caution**

MD 32250: RATED_OUTVAL

MD 32260: RATED_VELO

MD 36700: DRIFT_ENABLE

MD 36710: DRIFT_LIMIT

MD 36720: DRIFT_VALUE

This is applicable only in relation to the analog axis function.

## 3.2 Displaying setpoints in NCK GUD

To support the start-up process, it is possible to display the voltages of individual analog axes in the “Parameters – User Data” display area on the operator panel front.

For this purpose, the appropriate GUDs must be set up. For a detailed description of the procedure to be followed, please refer to Section “File and program management” in the document “SINUMERIK 840D/810D/FM-NC Programming Guide Advanced”.

Please proceed as follows:

1. Create an INITIAL.INI back-up file.
2. Write a text file containing the following lines on an external PC:
   %_N_SGUD_DEF
   ;$PATH=/_N_DEF_DIR
   DEF NCK REAL ANALOG_AXIS_VOLTAGE[n]
   M30

   n = number of analog axes
3. Load this file to the NC.
4. Load the INITIAL.INI backup file to the NC.

After the next power ON, the voltages of the analog axes (maximum of 3) are displayed in the GUD array standardized to a maximum of +/– 10V.

## 3.3 Creating alarm texts

1. Add an entry for the alarm text files of the technology card in the [TextFiles] section of the C:\MMC2\MBDDE.INI file:
   CZYK=C:\DH\MB.DIR\TK1_
2. Set up language-specific text files TK1_GR.COM and TK1_UK.COM in directory C:\DH\MB.DIR.
3. Enter the following alarm texts here:
   in TK1_GR.COM:
   075100 0 0 “Too many analog axes configured”
   075101 0 0 “DMP module no. %1 assigned twice”
   075102 0 0 “DMP module no. %1 assigned to other system functions”
   075110 0 0 “Axis %1 drift compensation limit reached”

# Data Descriptions (MD, SD)

4

## 4.1 Machine data of standard system

The “Analog axis” function is implemented as a compile cycle application. In addition to the function-specific machine data, the following option data must be set.

**Warning**

An undesirable event or status **can** arise if the appropriate precautions are not taken.

The functions activated by the option data trigger corresponding compile cycles. These cycles can have a significant effect on the behavior of the control system and may cause hazardous situations by accessing the NC.
Before a compile cycle is activated, the necessary safety precautions must be taken to protect operating personnel and machine (safety precautions against parameterization or programming errors in the compile cycles may need to be taken).

## 4.2 Machine data for the analog axis function

| 63530<br><br>MD number | $MA_ANALOG_AXIS<br><br>Configuration of an analog axis | |
|---|---|---|
| Default setting:<br><br>0 | Min. input limit: 0 | Max. input limit: 8 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: INT | | |
| Meaning: | An NC axis is declared as an analog axis in this machine data.<br><br>Input value = 0: Axis is digital axis<br><br>Input value > 0: Axis is one of the 3 possible analog axes.<br><br>The value to be entered here is the number of the DMP module on the NCU terminal block that must output an analog +/– 10V speed setpoint to the drive amplifier of the axis. | |

■

# Signal Descriptions 5

No separate signals to the PLC are provided for the “Analog axis” function.

# Examples 6

## 6.1 General start-up of a compile cycle function

**Precondition**

- The software version installed on the MMC must be 3.5 or higher.
- An NCK technology card with the “Analog axis” function must be available.

**Saving SRAM contents**

As the first step in installing a compile cycle function, the original card inserted in the NCU must be replaced by the technology card.
This is identical to the procedure followed for upgrading the NCU to a later software version and likewise requires the static (battery-backed) control system memory to be erased. When this is done, all data stored in the SRAM are lost and they must therefore be saved beforehand. For a detailed description, please refer to the Manufacturer/Service Documentation “SINUMERIK 840D/SIMODRIVE 611D Installation and Start-Up Guide”:

1. Enter the machine manufacturer password.
2. Change to the “Services” operating area.
3. Press softkey “Series start-up”.
4. Select “NC” and “PLC” as the areas to be saved and enter a name of your choice for the archive file to be created on the hard disk. Finish by pressing the RETURN key.

5. If the control system contains machine-specific compensation data, then these must be saved in a separate archive file:
   Press softkey “Data Out” and
   select under item “NC active data” the following data as required:
   “Measuring system compensations”,
   “Sag/Angularity comp.” and
   “Quadrant error compensation”.
   Save these data by selecting softkey “Archive ...” and specify another file name for a second archive file.

Keep the archive files you have created in a safe place. They will allow you to restore original settings in your system.

**Insert the PC card**

- Switch off control system.
- Insert the PC card with the new firmware ( technology card ) in the PCMCIA slot of the NCU.
- Then proceed as follows:

1. Turn switch S3 on the front panel of the NCU to 1.
2. Switch the control system back on again.
3. When the system powers up, the firmware is copied from the PC card into the NCU memory.
4. Wait until number “6” is displayed on the NCU digital display (after approximately one minute).
5. Turn switch S3 back to zero.

---

**Note**

If number “6” does not appear, then an error has occurred.
- Incorrect PC card (e.g. card for NCU2 in NCU3 hardware)
- Card hardware defective

---

**Copy back SRAM contents**

To copy the saved data back into the control system, proceed as described in Section 12.2 (series start-up). Please read all information provided by the manufacturer about new software versions.

- Enter the machine manufacturer password.
- Select “Data In” and “Archive...”. Then load the archive with backup compensation data (if applicable).

## 6.2 Start-up of analog axis

To start up the “Analog axis” function, you next need to active the compile cycle.

**Option data for compile cycles**

- Set the option for compile cycle application 2
- **Run up the software again.** Machine data 63530 should now be displayed at the end of the axial machine data list (look for “ANALOG_AXIS” or “63530”):

**Analog output**

Start up the DMP module for the analog setpoint with machine data 10362 $MN_HW_ASSIGN_NUM_INPUTS.

**Alarms**

Enter the alarm texts in language-specific text files TK1_GR.COM and TK1_UK.COM.

**GUD**

Set up GUD “ANALOG_AXIS_VOLTAGE” for monitoring the voltage output if required.

**Analog axis**

Declare axis as an analog axis with machine data 63530 $MA_ANALOG_AXIS and set axial machine data for the setpoint output and actual value input for the analog axis.

## 6.3 Example of how to configure an analog axis

One of the axes on the following machine is to be controlled by an analog drive.

- Channel 1: Machine axes AX1, AX2, AX3
- Channel 2: Machine axes AX4, AX5
- Axes 1–4 are digital axes, drive 5 is the NCU terminal block.
- Axis AX5 must be operated as an analog axis. Analog value must be available on DMP module 2. This module is inserted in slot 3 on the NCU terminal block.
- The direct measuring system of digital drive 3 is to be used as the actual value input.
- The maximum motor speed is 3000 rev/min.
- The maximum motor speed is reached at +/– 8V.

The machine data for the DMP module, setpoint output and actual value input need to be set as follows:

Axis-specific machine data for axis 5:

| | |
|---|---|
| 63530 $MA_ANALOG_AXIS = 2 | DMP module number |

Setpoints:

| | |
|---|---|
| 30100 $MA_CTRLOUT_SEGMENT_NR = 1 | Bus segment 840D |
| 30110 $MA_CTRLOUT_MODULE _NR= 6 | Free module (need not actually exist in hardware) |
| 30120 $MA_CTRLOUT_NR = 1 | Setting is always 1 for 840D |
| 30130 $MA_CTRLOUT_TYPE = 0 | Simulated setpoint |
| 32250 $MA_RATED_OUTVAL = 80 | 80% rated voltage at max. motor speed |
| 32260 $MA_RATED_VELO = 3000 | Max. motor speed |

Actual values:

| | |
|---|---|
| 30200 $MA_NUM_ENCS = 1 | Analog axis has one measuring system |
| 30210 $MA_ENC_SEGMENT_NR = 1 | Bus segment 840D |
| 30220 $MA_ENC_MODULE_NR = 3 | Module number 3 |
| 30230 $MA_ENC_INPUT_NR[0] = 2 | Direct measuring system |
| 30240 $MA_ENC_TYPE[0] = 1 | Encoder type: Signal generator |

Global machine data:

| | |
|---|---|
| 10310 $MN_FASTIO_ANA_NUM_INPUTS = 0 or 1 | DMP modules assigned to system functions |
| 10364 $MN_HW_ASSIGN_ANA_FASTOUT[1]= 1090301 | Assign 2nd DMP module slot 3 on NCU terminal block, see also A4 |
| 10383 $MN_HW_CLOCKED_MODULE_MASK=8H | Set DMP module for clock-synchronous operation (3rd slot) |
| 10380 $MN_HW_UPDATE_RATE_FASTIO = 2 | Position control cycle |

After power ON: Carry out drift compensation on the 5th axis by programming axial machine data 36700–36720.

■

# Notes

# 7 Data Fields, Lists

## 7.1 Alarms

| | |
|---|---|
| **75100** | **Too many analog axes configured** |
| Explanation | More than 3 NC axes are configured as analog axes in machine data 63530 $MA_ANALOG_AXIS. |
| Reaction | All axes switch to follow-up mode, alarm signal at PLC interface. |
| Remedy | Reduce the number of analog axes. |
| Reset criterion | Power ON |

| | |
|---|---|
| **75101** | **DMP module no. %2 is assigned twice** |
| Explanation | A DMP module has been assigned twice in axial machine data 63530 $MA_ANALOG_AXIS. |
| Reaction | All axes switch to follow-up mode, alarm signal at PLC interface. |
| Remedy | Use different DMP modules. |
| Reset criterion | Power ON |

| | |
|---|---|
| **75102** | **DMP module no. %2 assigned to other system functions** |
| Explanation | A module has been defined in machine data 63530 $MA_ANALOG_AXIS that has already been assigned to other system functions in machine data 10310 $MN_FASTIO_ANA_NUM_OUTPUTS. |
| Reaction | All axes switch to follow-up mode, alarm signal at PLC interface. |
| Remedy | The module number defined in MD 63530 must be greater than the setting in MD 10310. |
| Reset criterion | Power ON |

| | |
|---|---|
| **75110** | **Axis %1 has reached drift limit** |
| Explanation | The automatic drift compensation has reached the value set in MD 36710 $MA_DRIFT_VALUE. |
| Reaction | Drift value is limited. |
| Remedy | Increase either the value in MD 36710 or the fixed drift value in MD 36720 $MA_DRIFT_VALUE. |
| Reset criterion | RESET |

## 7.2 Machine data

| Number | Identifier name | Re-ference |
|---|---|---|
| **General ($MN_ ...)** | | |
| 10310 | FASTIO_ANA_NUM_OUTPUTS<br>Number of active NCK outputs | A4 |
| 10364 | HW_ASSIGN_ANA_FASTOUT Hardware assignment of external analog NCK outputs: 0...7 | A4 |
| 10380 | HW_UPDATE_RATE_FASTIO Update cycle of synchronously clocked external NCK input/output modules | A4 |
| 10384 | HW_CLOCKED_MODULE_MASK<br>Synchronous processing of individual external input/output modules. Terminal block: 0...3 | A4 |
| **Axis-specific ($MA_...)** | | |
| 30100 | CTRLOUT_SEGMENT_NR<br>Setpoint assignment drive type | G2 |
| 30110 | CTRLOUT_MODULE<br>Setpoint assignment drive number | G2 |
| 30120 | CTRLOUT_NR Setpoint output on drive module | G2 |
| 30130 | CTRLOUT_TYPE Type of setpoint output | G2 |
| 30200 | NUMS_ENC Number of encoders | G2 |
| 30210 | ENC_SEGMENT_NR Actual value input drive type | G2 |
| 30220 | ENC_MODULE_NR Actual value input drive number | G2 |
| 30230 | ENC_INPUT_NR Actual value input on drive module | G2 |
| 30240 | ENC_TYPE Type of actual value acquisition | G2 |
| 32250 | RATED_OUTVAL Rated output voltage | G2 |
| 32260 | RATED_VELO Maximum motor speed | G2 |
| 36700 | DRIFT_ENABLE Automatic drift compensation | K3 |
| 36710 | DRIFT_LIMIT Drift limit value for aut. drift compensation | K3 |
| 36720 | DRIFT_VALUE Drift basic value | K3 |
| 63530 | ANALOG_AXIS Configuration of an analog axis | |

■

# SINUMERIK 840D/840Di/810D Description of Functions Special Functions (Part 3)

# Speed/Torque Coupling, Master–Slave (TE3)

# 1 Brief Description

**SW 6 and higher**

The speed/torque coupling function (master–slave) is used for mechanically coupled axes that are driven by two separate motors. A further application is the compensation of gears and backlash in the gear tooth flank due to mutual tension in the drives.

Speed/torque coupling (master–slave) is a speed setpoint coupling between a master and a slave axis, involving a torque compensatory controller for even torque distribution.
Each slave axis has exactly one master axis.
Conversely, a master axis can also belong to several slaves; this is done by configuring several master–slave relationships using the same master axis.
A configured slave axis must not be the master axis in one of the other master–slave relationships.

**Difference compared to previous solution (SW 5.x and lower)**

- If traversing is programmed for a slave axis that has already been linked, an alarm is issued.
- The setpoint position of the coupled axis corresponds to the current actual position.
- On request, the coupling is made or released independent of the channel status the next time the axis stops. This allows the coupling status to be changed even during parts program processing.
- For brake control, the interface signal “Master–slave coupling status active” should be used.
- If a master axis is simultaneously configured as the slave, an alarm is issued. So cascading is not possible.
- If a coupling is requested and closed, the control activation signals are derived directly from the master axis.
- If the coupling is closed, the slave axis is speed-controlled; status signal DB3x.DBX61.5 “Position control active” is not set.

Please refer to Chapter 3 for more information about the differences.

**SW 6.4 and higher**

The function of the speed/torque coupling has been expanded to include the following options:

- Coupling/decoupling of **rotating, speed/controlled spindles**
- **Dynamic configuration** of couplings
- A separate machine data has been provided for reversing the direction of the slave axis in coupled state.

**SW 5 and lower**

The speed/torque coupling function (master–slave) is used for mechanically coupled axes that are driven by two separate motors.
This function was available in SW 5 and lower only as a technology card. It was not included in the standard scope of functions.

■

# Detailed Description 2

## 2.1 Speed/torque coupling, master–slave (SW 6 and higher)

### 2.1.1 General

Speed/torque coupling (master–slave) is a speed setpoint coupling between a master and a slave axis, involving a torque compensatory control which ensures even torque distribution.

This function is mainly used for boosting the power of mechanically coupled drives. Other application: Compensation of gears and backlash in the gear tooth flank due to mutual tension in the drives.

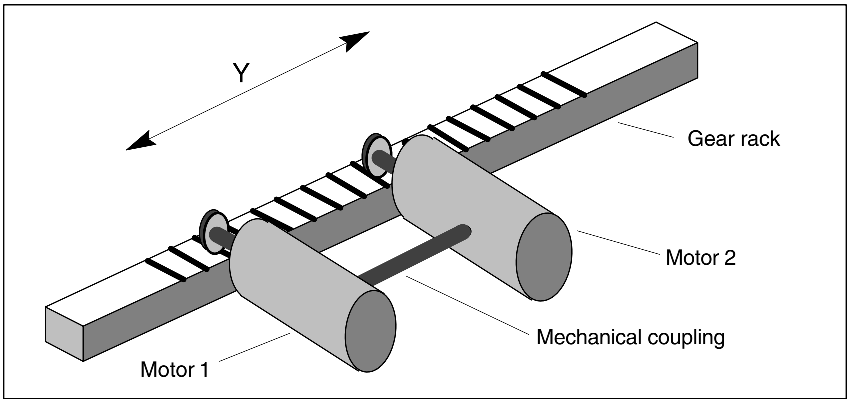



Fig. 2-1 Permanent mechanical coupling

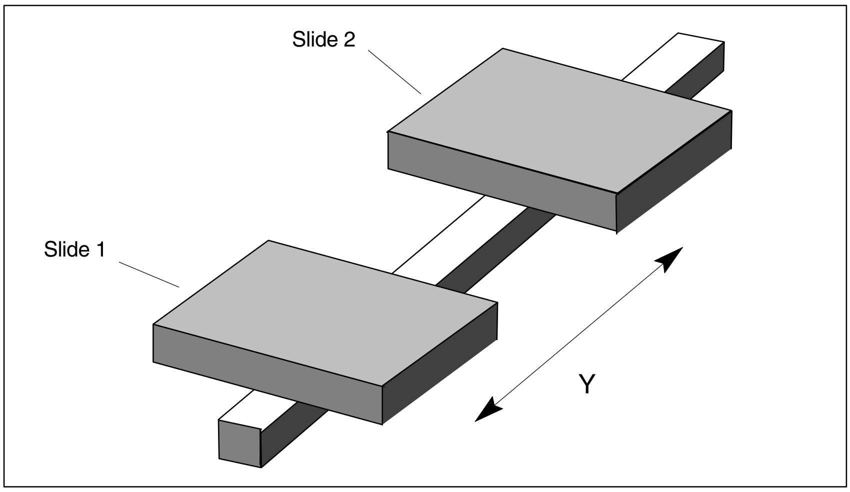



Fig. 2-2 Slides (linear motor) for temporary coupling

Each slave axis has exactly one master axis.
Conversely, a master axis can also belong to several slaves; this is done by configuring several master–slave relationships using the same master axis.
A configured slave axis must not be the master axis in one of the other master–slave relationships.

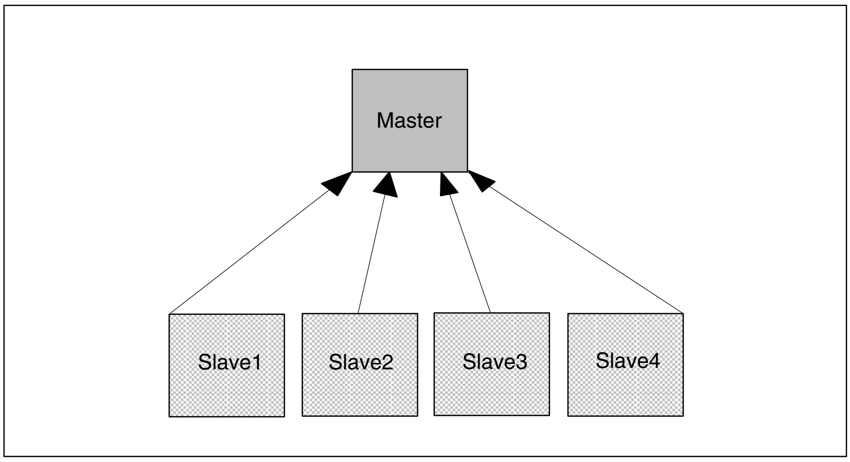



Fig. 2-3 Four coupling relationships with the same master axis

### 2.1.2 Coupling diagram

If the coupling is closed, the slave axis is traversed only with the load-side setpoint speed of the master axis. It is therefore only speed-controlled, not position-controlled.
No positional deviation control is implemented between master and slave axes.
A torque compensatory controller divides the torque evenly over the master and slave axes.
An additional torque can be used to achieve a tension between the master and slave axis.
If different motors are used, individual weighting factors can be used to adapt the torque distribution.

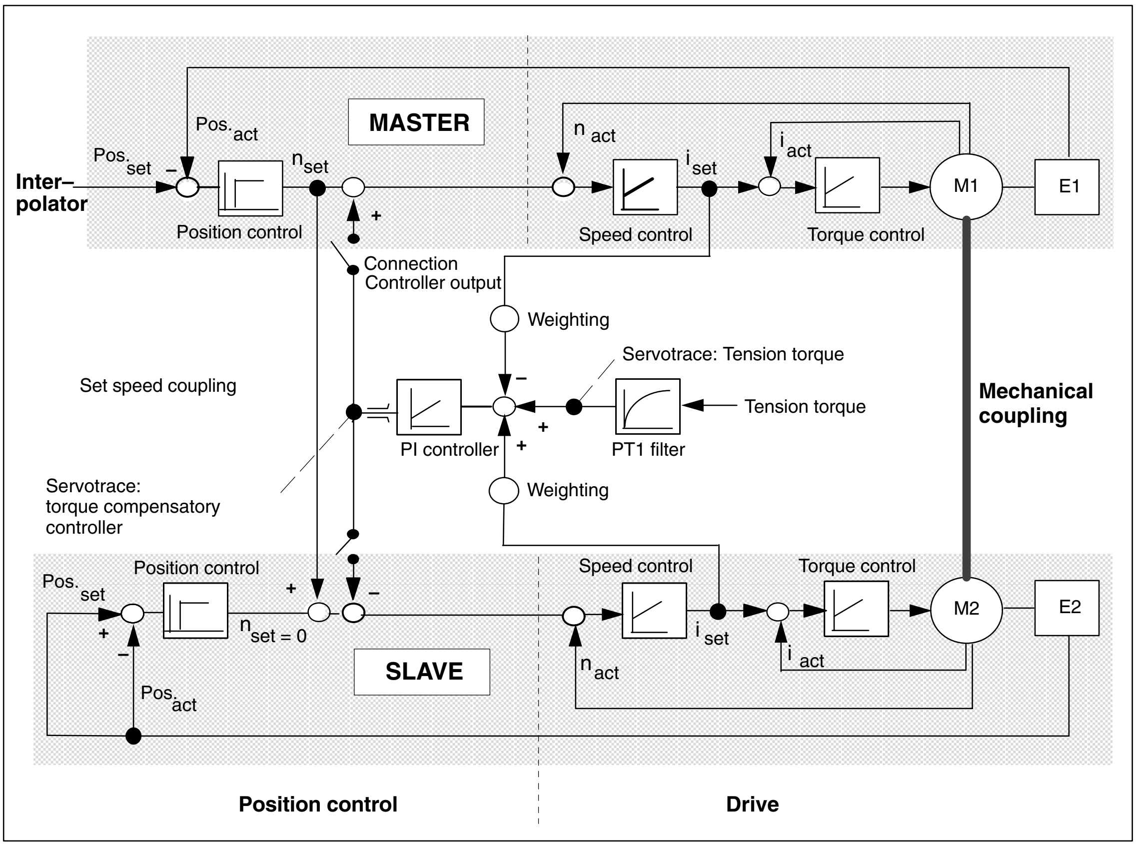



Fig. 2-4 Control structure

## 2.1.3 Configuring a coupling

**Static**

A master–slave coupling is configured only in the slave axis.
This must be assigned permanently to one of the channels. Each slave axis is assigned one master axis for speed setpoint coupling and one for torque compensatory control.

In the default setting, the same master axis is used for torque compensatory control as for speed setpoint coupling.

The assigned made in MD 37252: MS_ASSIGN_MASTER_TORQUE_CTR and MD 37250: MS_ASSIGN_MASTER_SPEED_CMD is automatically active in each control start-up.

**Dynamics**
**SW 6.4 and higher**

The program commands MASLDEF and MASLDEL can be used to change the assignment from the parts program dynamically. This type of configuration can change the static configuration but does not have any reverse effect on the associated machine data.

The statement
MASLDEF (slv1, slv2, ..., master axis)
assigns a master axis to one or more slave axes.

MASLDEL (slv1, slv2, ...)
cancels the assignment of the slave axes to the master axis and simultaneously disconnects the current coupling analogous to MASLOF.

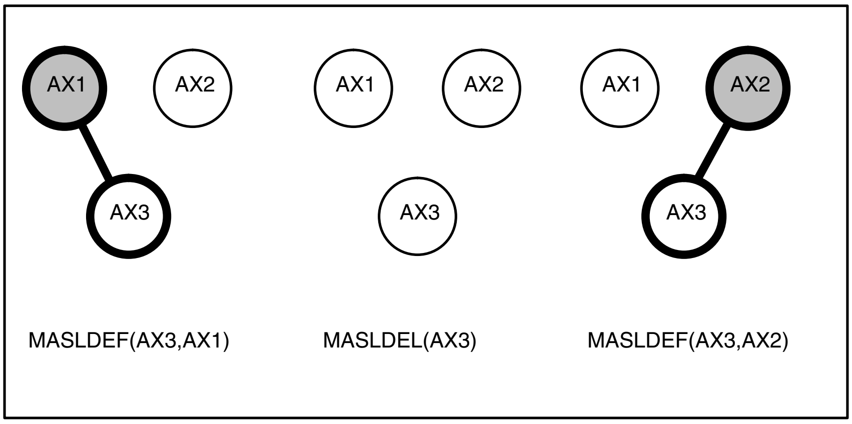



Fig. 2-5 Varying configuration of master axis

Changing the configuration has no effect in the coupled state. The change is not accepted until the axes are next uncoupled.

Unlike static assignment, the master axis for torque compensatory control always corresponds to the speed setpoint coupling.

A plausibility check is not carried out until the coupling is closed. In the event of multiple assignment, Alarm 26031 is issued.

An assignment made with MASLDEF is retained after a mode change, reset or parts program end.

---

**Note**

To implement a standard assignment in each reset, you can add the corresponding MASLDEF and MASLDEL statements to the PROG_EVENT.SPF user application. The event-driven call is configured via MD 20108: PROG_EVENT_MASK = 4.

---

### 2.1.4 Torque compensatory controller

A PI controller calculates a load-side additional speed setpoint from the torque difference between the master and slave axes. This is applied as standard to the command speed setpoint in the master and slave axes with different signs in each case.

If one master and several slaves axes are used, this distribution can cause to instabilities. The output of the torque compensatory controller should only be applied in the slave MD 37254. MS_TORQUE_CTRL_MODE = 1.

The torque setpoints used for torque compensation control are smoothed in the drive. The corner frequency of the PT1 filters is entered in MD1252: TORQUE_FILTER_FREQUENCY. The same value should be set in the master and slave axes.

The gain factor MD 37256: MS_TORQUE_CTRL_P_GAIN corresponds to the percentage ratio of the maximum axis velocity MD 32000: MAX_AX_VELO to the drive torque = MD1725 / 8 of the slave axis.

The I component is disabled in the default setting.

The integration time MD 37258: MS_TORQUE_CTRL_I_TIME is entered in seconds.

The output of the torque compensatory control is actively limited to MD 37260: MS_MAX_CTRL_VELO.

With the setting MD 37256: MS_TORQUE_CTRL_MODE = 3 or MS_TORQUE_CTRL_P_GAIN = 0, the torque compensatory control is inactive.

The torque distribution can be parameterized via the input variables of the torque compensatory controller. The drive torque of the slave axis is weighted with MD 37268: MS_TORQUE_WEIGHT_SLAVE, the drive torque of the master axis with (100 – MS_TORQUE_WEIGHT_SLAVE).

If motors with different nominal torque values are used, the 50% to 50% standard distribution must be adapted to suit.

A mechanical coupling is absolutely necessary when the torque compensatory controller is used. Otherwise, the drives involved could accelerate from standstill.

**Activation/ deactivation via the PLC SW 6.4 and higher**

The torque compensatory control can be activated and deactivated directly via the PLC interface signal DB31, ... DBX24.4.
You must set MD 37255: MS_TORQUE_CTRL_ACTIVATION=1. The activation status can be returned in DB31, ... DBX96.4. MD 37254: MS_TORQUE_CTRL_MODE is then only used for configuring the torque distribution.

### 2.1.5 Tension torque

Specifying an additional torque MD 37264: MS_TENSION_TORQUE, you can achieve a tension between the master and slave axis when the torque compensatory controller is active. The tension torque is entered as a percentage of the nominal torque and is active straight away.

The tension torque tension torque is applied via a PT1 filter. Specifying a filter time constant MD 37266: MS_TENSION_TORQ_FILTER_TIME > 0 activates the filter.

The tension torque chosen must be high enough to ensure that the resulting torque does not drop below the minimum required tension even during acceleration. To prevent unnecessary heating in the motor, you can reduce the tension torque when the motor is at standstill.

Specifying a tension torque without a mechanical coupling produces axis movement.

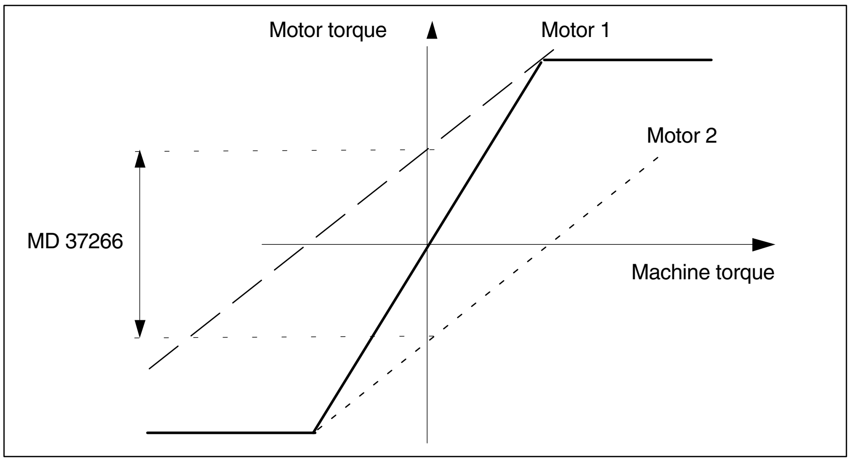



Fig. 2-6 Resulting tension torque

### 2.1.6 Activating a coupling

The type of activation for a master–slave coupling is defined in MD 37262: MS_COUPLING_ALWAYS_ACTIVE.

Depending on the machine configuration, a distinction is made between a permanent and a temporary master–slave coupling.

Temporary master–slave coupling (MD 37262: MS_COUPLING_ALWAYS_ACTIVE = 0) configured via machine data can be closed and disconnected via the axial PLC interface signal "Master/Slave On" DB31, ... DBX24.7 and inside a parts program also with the language commands MASLON(Slave axis 1, Slave axis 2, ...) and MASLOF(Slave axis 1, Slave axis 2, ...).

The setpoint status of the coupling always corresponds to the last specification made.

The current coupling status can be read back to the slave axes via the PLC interface signal DB31–DB3x.DBX96.7 "Master–slave coupling active" in the slave axis.

In the parts program and from the synchronous actions, the current coupling status can be output via the **system variable** of the slave axis $AA_MASL_STAT.

**SW 6.4 and higher**

The statement
MASLOFS(Slv1, Slv2, ...)
can be used too disconnect the coupling analogous to MASLOF and decelerate the slave spindle automatically.

---

**Note**

A permanent coupling (MD 37262 MS_COUPLING_ALWAYS_ACTIVE=1) does not require explicit activation.

---

**Example**

For an example of how to configure the master–slave coupling between AX1=master and AX2=slave, please refer to:

**Section 6.1.** Speed/Torque Coupling (SW6 and higher)

**Control system response**

The control system response on power ON, mode changes, RESET, block searches and Repos is as follows:

- A master–slave coupling activated via PLC or MASLON statement is retained after Mode change, RESET, Parts program end.
- MASLON / MASLOF/ MASLOFS becomes effective on block search. Changes in the positions of coupled slave axes and spindle speeds must be computed separately by the user (see “Block search”, Subsection 2.1.10).

## 2.1.7 Response on activation/deactivation

**Activating/ deactivating during axis standstill**

Activation/deactivation is not active until the axis next comes to a standstill. If the specification is changed, the sequence is the same as for axis replacement. The coupling is closed when the axis comes to a standstill. The coupled axes must be in feedback control mode.

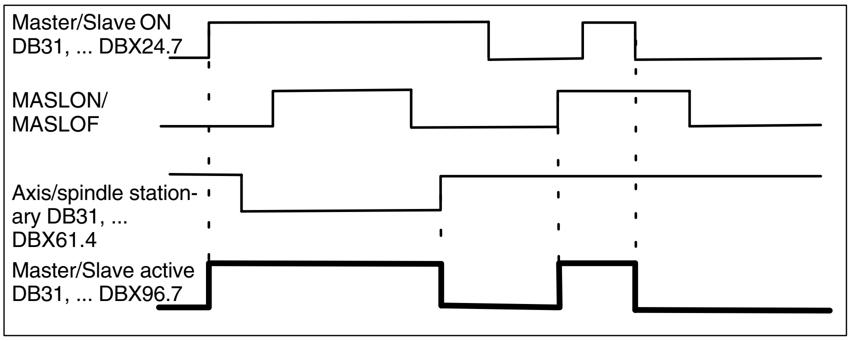



Fig. 2-7 Activation procedure

Block stepping is halted for MASLON until the coupling has actually been closed. At the MMC/HMI, the associated channel operating message “Master–slave switchover inactive” is output.

**Activating/ deactivating in motion SW 6.4 and higher**

Activation/deactivation of the coupling in motion has been implemented only for spindles in speed control mode. For axes and spindles in positioning mode, switchover is still carried out when the axis is at a standstill.

**Activation during a motion**

The coupling procedure at different speeds is divided into two phases.

**Phase 1**

The signal DB31, ... DBX24.7 request closing of the coupling. The slave spindle accelerates or decelerates according to the ramp of the setpoint speed of the master spindle with the dynamic response available.

When the setpoint speed is reached, the coupling is closed and the interface signal “Coupling active” DB31, ... DBX96.7 is set.

If the master spindle is accelerated during the coupling process, the first phase is extended according to the existing difference in dynamics between the master and slave spindles.

**Phase 2**

In the second phase, the actual difference speed between the master and slave spindles is used to generate the synchronous run signals

“Speed tolerance, coarse” DB31, ... DBX96.3 and
“Speed tolerance, fine” DB31, ... DBX96.2.

The associated limits are defined via the following machine data:
MD 37270: MS_VELO_TOL_COARSE (“Tolerance coarse”)
MD 37272: MS_VELO_TOL_FINE (“Tolerance fine”).

---

**Note**

The signal “Tolerance coarse” can be used to implement a PLC monitoring function that checks a coupled group for loss of speed synchronism. The signal “Tolerance fine” can be used to derive the time for mechanical closure of the coupling and to activate the torque compensatory control directly.

---

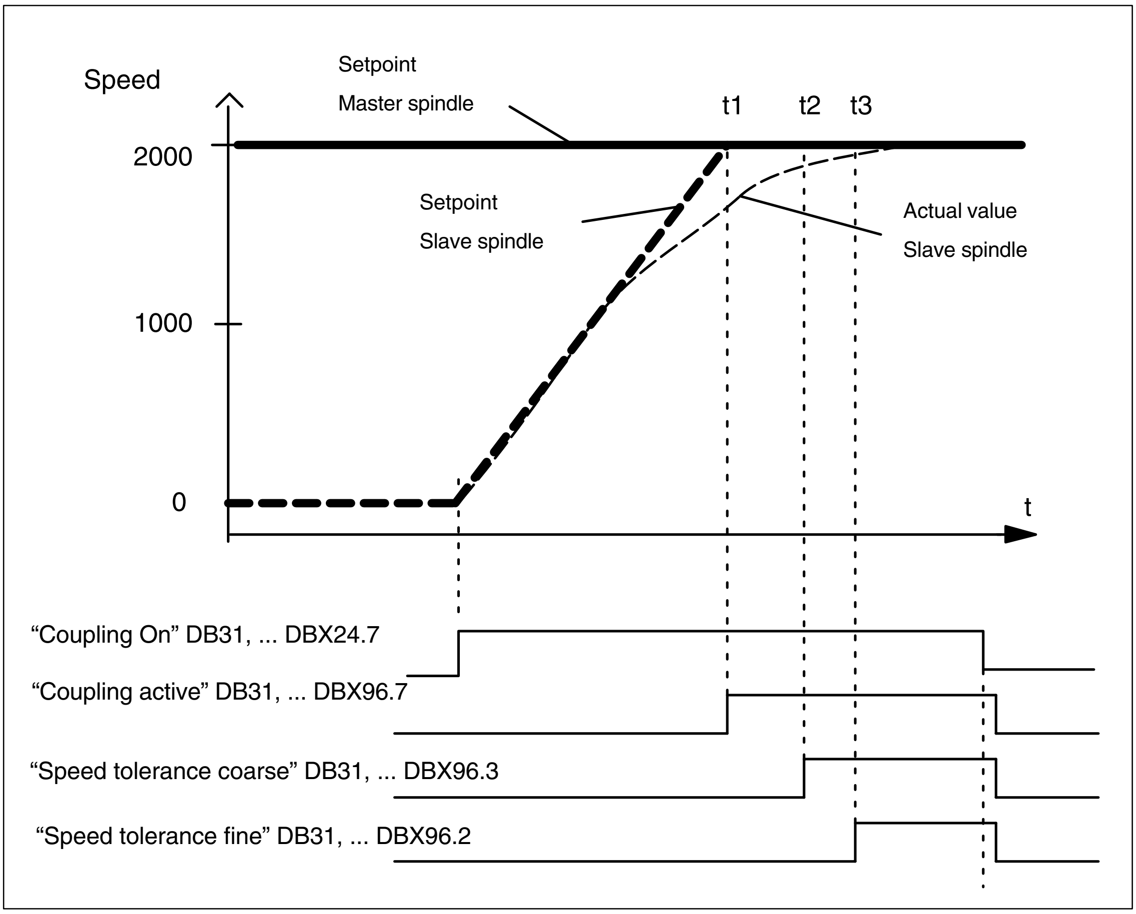



Fig. 2-8 Coupling procedure between two spindles with difference speeds

**Deactivating during motion**

An active coupling is disconnected using the MASLOF statement.

This statement is executed directly for spindles in speed control mode. The slave spindles that are rotating at this point in time retain their last speed until a new speed is programmed.

To decelerate slave spindles automatically when disconnecting the coupling, you can use the MASLOFS statement. For axes and spindles in positioning mode, the coupling is still only disconnected at standstill.

**Note**

The implicit stop preprocessor is omitted for MASLON and MASLOF. The missing stop preprocessor means that the $P system variables of the slave spindle do not supply updated values until re/programmed.

### 2.1.8 Axial interface signals

If a master–slave coupling is requested, the PLC axis enable signals “Servo enable” DB31, ... DBX2.1 and “Pulse enable” DB31, ... DBX21.7 of the slave axis are derived directly from the master axis. The separate PLC axis enable signals have no effect.

Cancellation of the servo enable in the master axis results in interpolative braking of the slave axis within the time configured in MD 36610: AX_EMERGENCY_STOP_TIME. The associated speed and current control enable signals are not canceled until MD 36620: SERVO_DISABLE_DELAY_TIME expires for the relevant axes.

To ensure identical braking behavior, the time set in machine data MD 36620: SERVO_DISABLE_DELAY_TIME should be identical for all coupled axes if possible. The same applies to drive machine data
MD1403: PULSE_SUPPRESSION_SPEED and
MD1404: PULSE_SUPPRESSION_DELAY.

If the drive status signals “Current controller active” DB31, ... DBX61.7 or “Speed controller active” DB31, ... DBX61.6 are missing on the master or slave axis, PLC interface signal “Master/slave active” DB31, ... DBX96.7 must be reset on the slave when it is stationary. When the master and slave axes are operating in closed-loop control mode again, IS “Master/slave active” DB31, ... DBX96.7 is set on the slave axis.

DB31, ... DBX24.4 is used to activate the torque compensatory control via the PLC. DB31, ... DBX96.4 can be used to read the status of torque compensatory control.

---

**Note**

If the coupling is closed, the slave axis operates in speed control mode; status signal “Position controller active” DB31, ... DBX61.5 is thus not set.

---

### 2.1.9 Axial monitoring functions

Except for speed setpoint and actual velocity monitoring, the axial monitoring function, such as contour and zero speed monitoring, are inactive in the slave axis because the position controller is missing.
The position-control circuit parameters, such as gain factor, feedforward control, symmetry, can thus be set differently in the master and slave axes without triggering the watchdogs.

To achieve the same braking response for all coupled axis in the event of a fault, the same alarm reaction is applied to entire coupling grouping when the coupling is active.

When correcting fault states, a repositioning of slave axes on the interrupt point is suppressed.

### 2.1.10 Response in conjunction with other functions

**Function generator**

To calibrate the speed control circuit for a master–slave coupling, MD 37268: MS_TORQUE_WEIGHT_SLAVE should be set to a small value in the slave axis.
Traversing of a mechanically controlled following slave axis is not prevented by the torque compensatory control in this case.

**Reference point approach**

If the coupling is closed, only the master axis can be referenced. Referencing of slaves axes is suppressed. The referencing requirement need not be explicitly canceled for the slave axis in order to do this. The referencing status of coupled slave axes remains unchanged. The slave axis position is generally not the same as the master axis position. This difference in position is not significant. In the case of separate coupling, each axis can be referenced separately as usual.

**Compensation**

Position offsets of the slave axis, such as spindle pitch errors, backlash, temperature and sag offsets are computed but not active because there is no position controller.

Correct calculation of the backlash compensation requires that the backlash of the slave axis is always overtraveled by the motion of the master axis in coupled mode. If you cancel the coupling during an axis reversal error, an incorrect actual value for slave axis results.

**Dynamic Stiffness Control**

The Kv factor of the master axis is copied to the slave axis for an existing coupling and is thus also active in the slave axis. This is an attempt to achieve the same control response in the drive of the master and slave axis as far as possible. MD 32640: STIFFNESS_CONTROL_ENABLE must be configured identically in all coupled axes.

**Speed/torque feedforward control**

The feedforward control in the slave axis need not be activated explicitly. The current settings of the master axis apply. The speed feedforward value of the master axis is already incorporated in the speed setpoint of the slave axis.
When the torque feedforward control is active, the torque feedforward control on the load side of the master axis is also active in the slave drive.
When coupled, the mechanical ratios change.
Axial settings must be adapted to suit. All coupled drives should have the same speed control dynamics.

**Gantry**

If one master–slave relationship is defined on each side of the gantry grouping to increase the gain, only the leading axis or following axis may be operated as master axis.

**Travel to fixed stop**

The Travel to Fixed Stop function can be programmed only in the master axis when a coupling is active and has a different effect on the master and slave axes. The programmed value is expressed

- as a percentage of the rated drive torque of the master axis. The master axis detects when the fixed stop has been reached.
- The programmed value is also active on the slave axis, but refers to the drive torque of the slave axis.

If the rated torque values of the master and slave axes are different, machine data MD 37014: FIXED_STOP_TORQUE_FACTOR on the slave axis can be set to compensate the difference. Specifying a factor < 1 reduces the programmed clamping torque in the slave axis.

Please note the following boundary conditions:

- Torque distribution between the master and slave axes is not possible during clamping as the torque compensatory controller is deactivated during clamping operations.
- Status changes to the master–slave coupling have no effect during travel to fixed stop. Specification of a new status is only accepted when the fixed stop function has been completed.

**Safety Integrated**

Since the slave axis is traversed via the speed setpoint of the master axis, the axial setpoint limitation MD 36933: SAFE_DES_VELO_LIMIT in the coupled slave axes is inoperative. All safety monitoring functions remain active in the slave axes however.

**Counterweight**

The additional torque for the electronic counterweight MD 32460: TORQUE_OFFSET is computed in the slave axis irrespective of its coupling status.

**Axis container**

If a coupled slave axis is configured in an axis container, alarm “4025 Switch axis container %3 not permitted: Master–slave active channel %1 Axis %2” is output when the axis container advances. Advance of the axis container is not permitted because the coupling is active.

**SW 6.4 and higher**

Dynamic configuration can be used for changing masters to make the spindle relevant after a rotation of the axis container the master spindle. Both master and slave spindles can be container spindles.

For a coupling to be closed after container rotation using a different spindle in each case, the old coupling must be disconnected before the rotation, the configuration deleted and the new coupling closed after the rotation.

Example of a cyclic coupling sequence (Position=3 / Container=CT1)

```
MASLDEF(AUX,SPI(3))      ; S3 Master for AUX
MASLON(AUX)              ; Coupling in for AUX
M3=3 S3=4000             ; Processing ...
MASLDEL(AUX)             ; Delete configuration and
                         ; disconnect coupling
AXCTSWE(CT1)             ; Container rotation
```

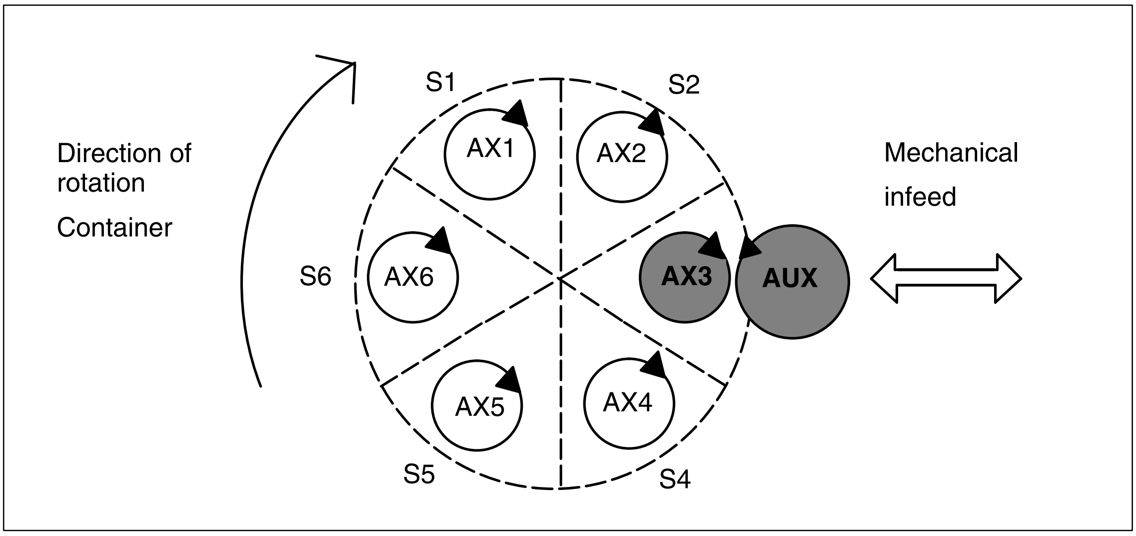



Fig. 2-9 Coupling between container spindle S3 and auxiliary motor AUX (prior to rotation)

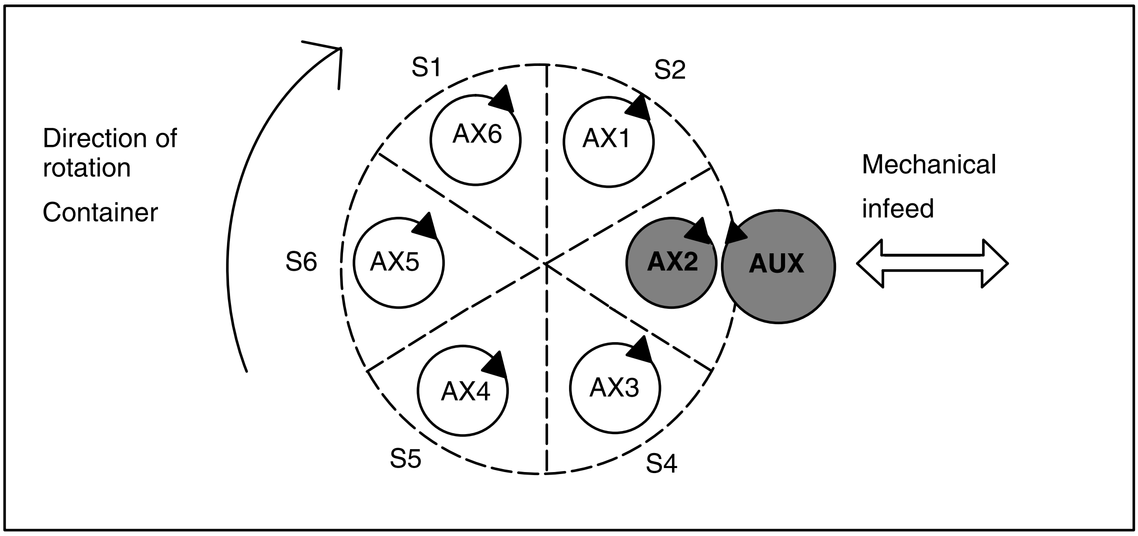



Fig. 2-10 Coupling between container spindle S3 and auxiliary motor AUX (after to rotation)

**Hardware and software limit switches**

Crossing of hardware and software limit switches is detected in coupled axes; in the coupled state, the software limit switch is generally crossed on slave axes. The alarm is output on the slave axis, while braking is initiated via the master axis.

The path traveled after detected of the slave software limit switch equals the distance required by the master axis to brake the coupling.

The master axis controls the movement away from the limit switch, since the coupling cannot be separated until the cause of the alarm has been eliminated.

**Block search**

The SERUPRO function “Block search with calculation” can be used without restriction in combination with a permanent master–slave coupling when MD 37262: MS_COUPLING_ALWAYS_ACTIVE=1.

The following restrictions apply when the coupling is programmed using MASLON and MASLOF commands:

- The coupled axes must be in the same channel when the block search is executed. If they are not in the same channel, the block search is aborted with alarm 15395.
- The coupled axes are operated on the same NCU.
- After block search, the associated axis positions and speeds must be influenced later by the user via a system ASUB (asynchronous subprogram) “PROGEVENT.SPF” of the coupling status. System variables are available for this purpose:

$P_PROG_EVENT
This variable provides information about the event which activated the subroutine. A value of 5 stands for block search.

$P_SEARCH_MASLC[slave axis identifier]
The variable stands for alteration of the coupling status during a block search.

$P_SEARCH_MASLD[slave axis identifier]
This variable indicates the positional offset calculated in the block search between the slave and master axes at the instant the coupling was closed.

$AA_MASL_STAT[slave axis identifier]
This variable indicates the current coupling status.

- The system ASUB “PROGEVENT.SPF” must be stored under /_N_CMA_DIR/_N_PROG_EVENT_SPF so that it can be accessed by the control system.

**Example 1** for PROGEVENT.SPF:

```
N10 IF $P_PROG_EVENT==5              ; Block search active
N20 IF (($P_SEARCH_MASLC[Y]<>0)      ; The coupling status has
    AND ($AA_MASL_STAT[Y]<>0))       ; changed during the block search
                                     ; and current status is coupled.
N30 MASLOF(Y)                        ; Disconnect coupling
N40 SUPA Y=$AA_IM[X]–$P_SEARCH_MASLD[Y]
                                     ; Position offset via the
                                     ; via the slave axis
N50 MASLON(Y)                        ; Close coupling
N60 ENDIF
N70 ENDIF
N80 REPOSA
```

**Example 2** for PROGEVENT.SPF:

```
N10 IF $P_PROG_EVENT==5                  ; Block search active
N20 IF (($P_SEARCH_MASLC[SPI(2)]<>0)     ; In block search,
    AND ($AA_MASL_STAT[SPI(2)]==0))      ; the coupling status of the 2nd
                                         ; spindle has changed and
                                         ; the current status is disconnected
N30 M2=$P_SEARCH_SDIR[2]                 ; Update direction of rotation
N40 S2= $P_SEARCH_S[2]                   ; Update speed
N50 ENDIF
N60 ENDIF
N70 REPOSA
```

- To allow subroutine PROGEVENT.SPF to start automatically, the following machine data must be parameterized accordingly:

– MD 11450 SEARCH_RUN_MODE = H02
– MD 11602 ASUB_START_MASK = H03
– MD 11604 ASUB_START_PRIO_LEVEL = 100

For further application examples, see **Chapter 6.**

---

**Note**

For further information about event-driven program calls and block searches in Program Test mode (SERUPRO), please refer to:

**References:** /FB/, K1, Mode Groups, Channel, Program Operation

---

### 2.1.11 Compatibility of SW 6.4 compared to lower versions

**Implicit stop preprocessor**

The implicit stop preprocessor is omitted for MASLON, MASLOF.

For spindles in speed control mode, the time at which the coupling is closed or disconnected changes. The coupling is closed or disconnected immediately, without waiting for standstill.

If activation/deactivation is to remain the same despite the new function, a WAITS must be programmed explicitly before MASLOF as in the example on the right. The coupling is not disconnected until all coupled spindles have come to a standstill.

| SW 6.4 and lower | SW 6.4 and higher |
|---|---|
| `MASLON(S3)` | `MASLON(S3)` |
| `M2=3 S2=1000` | `M2=3 S2=1000` |
| `G4 F4` | `G4 F4` |
| `M2=5` | `M2=5` |
| `MASLOF(S3)` | `WAITS(2); For compatibility reasons` |
| | `MASLOF(S3)` |

**Multiple assignment**

The time for output of the configuration alarms 26031 changes from the control start-up to the time at which an attempt is made to close the coupling. The alarm is acknowledged with a reset.

## 2.1.12 Boundary conditions with SW 6.4 and higher

See Chapter 3. In addition:

The coupling for axes and spindles in positioning mode is still closed and disconnected only at standstill.

Accelerating spindles up to the current limit in coupled mode may leave spare capacity for the torque compensatory control. The desired torque application between master and slave is then no longer guaranteed.

Prior to gear change or a star/delta switchover, the master/slave coupling must be deactivated.

The maximum chuck speed for the master spindle
MD 35100: SPIND_VELO_LIMIT must be configured less than or equal to that of the slave spindles.

The axis velocity monitoring MD 36200: AX_VELO_LIMIT should be adapted to the chuck speed.

For dynamic configuration, no distinction is made between the speed and torque master. The response corresponds to that of the standard setting
MD 37252: MS_ASSIGN_MASTER_TORQUE_CTR = 0.

# 2.2 Speed/torque coupling (SW 5.x and lower)

## 2.2.1 General

The speed/torque coupling (master–slave) function is required for configurations in which two drives are mechanically coupled to one axis. With this type of axis, a torque controller must ensure that each motor produces exactly the same torque or else the two motors would work in opposition.

Master–slave operation possible only with digital 611D drives.

One of the two drives is programmed (the master), while the other drive, referred to as the slave, is linked via the set speed coupling.

This function essentially consists of

- a set speed coupling and
- a torque control between the master and slave axes.

A master–slave operation without permanent mechanical coupling does not make sense because no torque distribution to the common mechanical connection can take place in this case.

When you activate a master–slave coupling, the position reference of the slave axis is lost for the NC. It is maintained on the real axis by way of a fixed mechanical coupling.

The function is not implemented as a difference position control but only as a coupling on the speed/torque plane. A difference position control would not make sense, since the controller between the master and slave would counteract one another.

This function allows each axis to be assigned to a master as a slave, which means that several master–slave couplings can co-exist.

To achieve a tensioning between the master and slave, it is possible to apply a settable tension torque to the torque control per machine data.

Master and slave axis need not be programmed in the same channel.
The set speed coupling is processed in the position control cycle.

## 2.2.2 Control structure

The control structure of a master–slave coupling is shown in Figure 2-11. For a better overview, only one master/one slave coupling is illustrated.

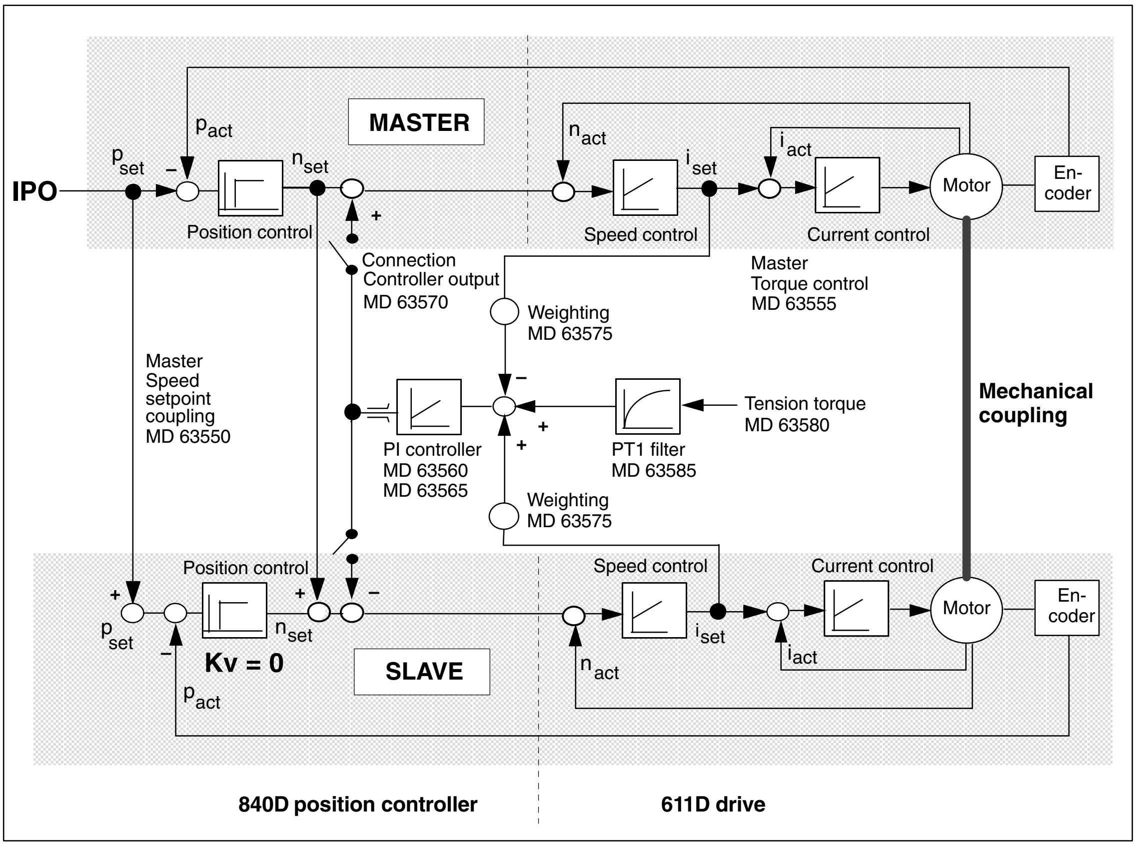



Fig. 2-11 Control structure

## 2.2.3 Configuring a coupling

**Defining a coupling**

Each axis involved in a master–slave coupling must be assigned to a channel as an NC axis. The axis/specific
MD 63550: MS_ASSIGN_MASTER_SPEED_CMD and
MD 63555: ASSIGN_MASTER_TORQUE_CTRL are used to assign a master axis for speed setpoint coupling and a master axis for torque control to each potential slave axis.

In most cases, the same master is used for set speed coupling and torque control. If MD ASSIGN_MASTER_TORQUE_CTRL is set to 0, the master axis for torque control is identical to that for set speed coupling.

**Several couplings**

A master can be assigned to each slave axis to produce several couplings. In a simple case, the couplings are mutually independent, i.e. each axis is involved in only one coupling. An example of this is a gantry axis with a master–slave coupling on each side.

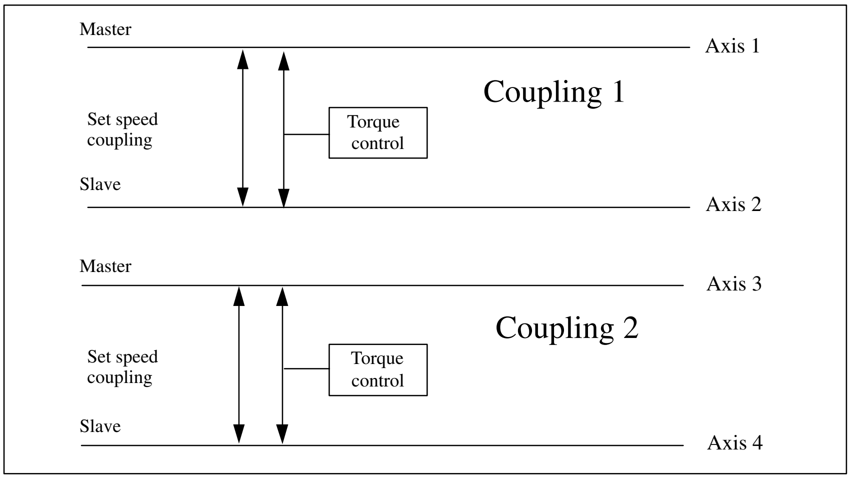



Fig. 2-12 Independent master–slave couplings

**One master several slaves**

It is also possible to configure master–slave couplings where one axis is the master axis for several couplings. In this example, axis 1 is the master axis for coupling 1 and coupling 2. Note the following:

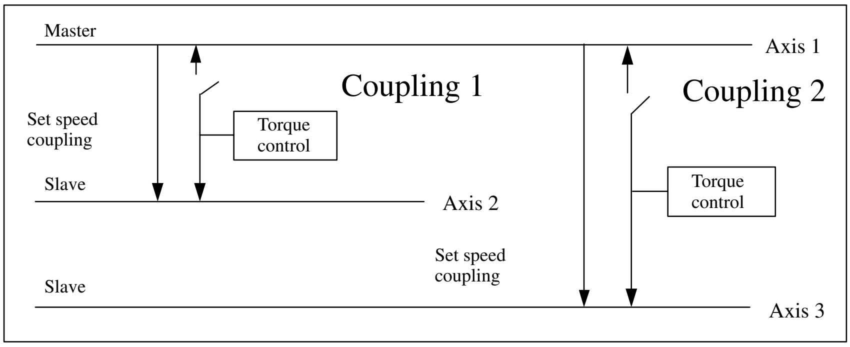



Fig. 2-13 One master, two slaves

The torque control of coupling 1 attempts to maintain the same torque between axis 1 and axis 2 by writing a set speed to axis 1 and axis 2. The torque control of coupling 2 also tries to maintain the same torque between axis 1 and axis 3. Both controllers would write set speeds to axis 1.

In order to ensure a stable system, both controllers must be parameterized such that the controller output is added only to the slave axes (axis 2 and axis 3) but not to the master axis (axis 1). This is achieved by setting MD 63570: MS_TORQUE_CTRL_MODE = 1 (controller output only on slave axis) for both couplings. Both torque controls now try to match the torque of the slave axis to the torque of the master axis, without adding set speeds to the master axis.

**Axis in the channel**

When the coupling is active, the motion of the slave axis is not displayed in the AUTOMATIC mode basic display, the actual value is "frozen". If a coupling is always active, i.e. the slave axis is never traversed individually, it is useful to display this axis as the last axis in the AUTOMATIC mode basic display. This is achieved by entering this axis as the last axis in the channel (MD 20070: AXCONF_MACHAX_USED).

**Several channels**

Master axis and slave axis need not be programmed in the same channel. Multi-channel couplings are possible for several active channels.

**Exchange**

Exchanging axes between channels can be prepared (MD 30550: AXCONF_ASSIGN_MASTER_CHAN) but can only be carried out with restrictions. These restrictions are described in Section 3.1.

**Spindles**

A master–slave coupling can also include spindles. The slave axis must then always operate in speed control mode, the position controller is deactivated. (DB3x.DBB61.5 = 0).
The master axis can be operated in all spindle modes, open-loop control mode with/without position controller, oscillation mode or positioning mode, even changeover between spindle modes is possible. Restrictions relating to the actual-value display are described in Section 3.1.

**Rotary axes**

Master and slave axes can also be rotary axis. Please note the restrictions outlined in Section 3.2.

**Motors rotating in opposite directions**

If the motors have been mounted to run in opposite directions, the traversing direction is reversed for one of the drives with MD 32100: MOTION_DIR. In this case, the set speed and the output of the torque control are calculated correctly; there is no need to set further machine data.

**Different motor speeds**

Master and slave axis can have different gear reduction ratios between the motors and the mechanical coupling. With these types of axes, the master and slave rotate at different speeds. When the coupling is active, the same load speed is standardized internally so that different motor speeds are possible for the master and slave without having to set further machine data.

**Speed feedforward control**

If the speed feedforward control is active in the master axis, the speed feedforward control must also be activated in the slave axis. Non-active speed feedforward control in the slave axis causes a "Contour monitoring" alarm in the slave axis.

**CPU load**

Each master slave coupling places a load on the position control level and the interpolation level. The table shows the CPU time required depending on the NCU hardware.

| NCU | Position control | | Interpolator level | |
|---|---|---|---|---|
| 572 | 1st coupling<br>each further coupling | 0.120ms<br>+ 0.050ms | 1st coupling<br>each further coupling | 0.100ms<br>+ 0.020ms |
| 573 | 1st coupling<br>each further coupling | 0.040ms<br>+ 0.020ms | 1st coupling<br>each further coupling | 0.030ms<br>+ 0.010ms |

**Configuration alarms**

During power ON of the control, the configuration machine data are checked and alarms set as necessary:

If the master and slave axes are identical for speed coupling, the alarm "75150 Slave axis AX1 and master axis are identical for set speed coupling" is present after power ON.

If the master axis and slave axis are identical for torque control, alarm "75151 Slave axis AX1 and master axis identical for torque control" is present.

All axes of the mode group follow on; the alarms can only be reset with power ON.

## 2.2.4 Torque control

The torque control between master and slave ensures an even distribution of the torque between the master and slave axis. The input variable of the controller is the torque difference Mdiff between the master and slave axis; the output is a set speed nset, which is applied to the master and the slave axis.

The controller consists of a P component and an I component. Both parts must be parameterized separately.

The machine data of the slave axis is always relevant for the configuration of the particular master–slave torque control.

**P controller**

The P controller calculates a set speed nset by multiplying the torque difference Mdiff by a gain factor Kp. The resulting set speed is added to the master and slave axis.

nset = Mdiff * Kp

The P gain Kp of the torque compensator has the dimension [( mm / min) / Nm].

In the axial MD 63560: MS_TORQUE_CTRL_P_GAIN, the P gain is entered as a percentage value of the ratio:
Maximum drive velocity [mm / min] to rated torque [Nm].

The maximum drive velocity is the contents of MD 32000: MAX_AX_VELO. The rated torque is obtained from the product
of drive MD 1113: TORQUE_CURRENT_RATIO and
drive MD 1118: MOTOR_STANDSTILL_CURRENT.

Only the data of the slave axis are relevant for the torque control.

| Example: | |
|---|---|
| Maximum drive velocity of the slave axis | 30000 mm/min |
| Motor rated torque of the slave axis | 10Nm |
| MS_TORQUE_CTRL_P_GAIN | 15% |
| Kp: (30000/10) * 15% | 450 (mm / min) / Nm |

## I controller

The I controller calculates a speed setpoint nset by multiplying torque difference Mdiff by a gain factor Ki:

nset = Mdiff * Ki

The gain factor Ki of the I controller is parameterized via the reset time of the torque compensator I_TIME. Ki can only be calculated if the gain factor of the P controller Kp $\neq$ 0. The I controller is not active unless the P component is also activated.

Ki = 1/ position controller cycle * I_TIME * Kp

In the axial MD 63565: MS_TORQUE_CTRL_I_TIME, the reset time is entered in seconds.
The default setting 0 deactivates the I component if the P component already ensures a matching torque distribution.

## Limiting the controller output

You can use MD 63600: MS_MAX_CTRL_VELO to limit the output of the controller to a maximum value. The value is entered as a percentage value relative to the maximum speed of the slave-axis. Default is 100%. The limit applies in the positive and negative directions.

## Interconnecting the torque compensatory control

You can use a further MD 63570: MS_TORQUE_CTRL_MODE to connect the output of the torque controls freely to the master and slave axis. In most cases, the output value is applied to the master and slave. The user is responsible for setting parameters meaningfully. The MD of the slave axis is the important setting.

| Meaning | 0: Switch the controller output to master and slave |
|---|---|
| | 1: Switch the controller output to the slave only |
| | 2: Switch the controller output to the master only |
| | 3: The controller is deactivated; if the coupling is active, only the set speed coupling is operative. |

Even if the controller output is not connected to an axis, the controller is calculated.

## Weighting

MD 63575: MS_TORQUE_WEIGHT_SLAVE is used to apply a weighting to the input variables of the torque compensator to enable a parameterizable torque distribution over the two drives. If the motors are identical and the same drive parameters are to be set for the motors to produce the same torque, the standard parameterization 50% is recommended. The MD refers to the torque of the slave axis, the torque of the master axis is weighted with the difference between the MD and 100%.

Example:
Slave axis is to produce 30% of the total torque.
70% is supplied by the master axis.
MS_TORQUE_WEIGHT_SLAVE = 30

**Tension**

An axial MD 63580: MS_TENSION_TORQUE can be used to apply a constant tension torque as input to the torque control. This tension torque is applied continuously and causes mutual tensioning of the coupled drives.
The MD of the slave axis is relevant for the tensioning of a coupling.

The tension torque can be positive or negative. The value to be input is a percentage value relative to the rated torque of the slave axis. The rated torque is obtained from the product of drive MD 1113: TORQUE_CURRENT_RATIO and drive MD 1118: MOTOR_STANDSTILL_CURRENT.
The tension torque is active immediately after a change. In this way, it is possible to implement various tension torques which suit individual machining situations.

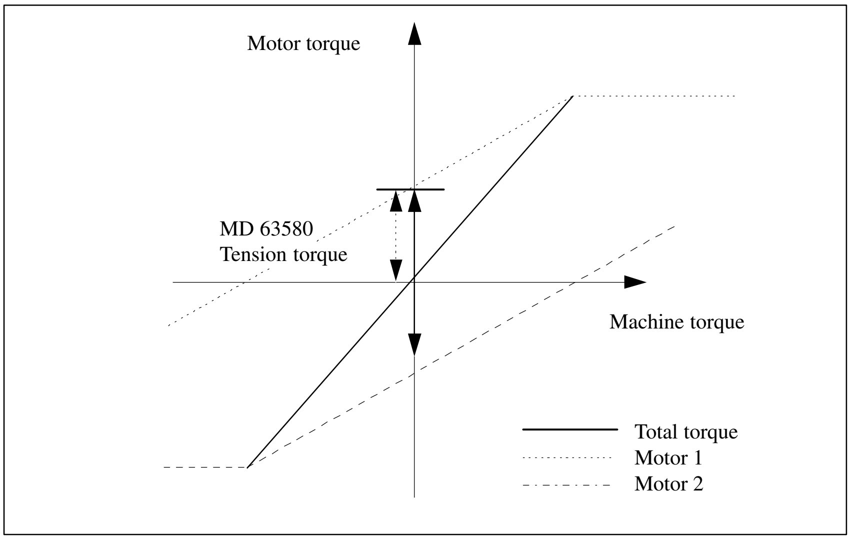



Fig. 2-14 Tension torque

**PT1 filter**

The tension torque is taken to the torque controller via a PT1 filter. The PT1 filter ensures a continuous increase or decrease of the tension torque when the tension torque value is changed. Without the PT1 filter, changing the tension torque causes a step change in the set speed at the torque control output when the controller is operated without I component. The PT1 filter is configured using MD 63585: MS_TENSION_TORQ_FILTER_TIME. The time is input in seconds. Input 0 to deactivate the PT1 filter.

**Note**

The functions ensures distribution of the torque-forming currents (Iq) and not distribution of the torques.
This means that torque distribution is also assured for FDD synchronous motors (no field weakening). In contrast, however, only current distribution is assured for MSD asynchronous motors in the field-weakening range. Distribution of torques is guaranteed only for motors of the same type which are operating simultaneously at the same speed. If MSD motors are not operated in the field-weakening range, then torque distribution can also be assured for different motor types operating at different speeds.

### 2.2.5 Presetting the drive machine data

**P component in the speed controller**

If the axes are put into operation separately in a master–slave coupling, whereby the single axis takes the full load, the P component in the speed controller must then be halved in the two axes. Only in this way is it possible to avoid overshoot when traversing the axis with active coupling.

### 2.2.6 Activating and deactivating a coupling

**Conditions for activation and deactivation**

A coupling is activated or deactivated only under the following conditions:

- Master and slave axes are operating in position control mode (DB3x.DBB 61.5) or, in the case of spindles, in speed control mode.
- Master and slave axis are at standstill (DB3x.DBB 61.4).
- The channels of the master and slave axis are in the “RESET” state (DB2x.DBB35.7). This condition can be activated/deactivated via a bit in MD 63595: TRACE_MODE.

If the master axis and slave axis are in different channels, both channels must be in the “RESET” state. With an axis exchange the state of the master channel is decisive.
(MD 30550: AXCONF_ASSIGN_MASTER_CHAN)
A channel is in the “RESET” state after the end of a program (M30) or after a “RESET” from the operator panel

**Master–slave coupling after power ON always active**

MD 63590: MS_COUPLING_ALWAYS_ACTIVE defines a coupling as always active. The coupling is activated as soon as the conditions for activation of a coupling are satisfied after a power ON. It can no longer be deactivated, i.e. it is no longer possible to operate the drives separately.

The machine data of the slave axis are always relevant for a coupling.

If it is not possible to activate a coupling after power-on because, for example, the axes are not in the position control state, Alarm “75160 slave axis AX1, master–slave coupling not active” is output. Further attempts are made to close the coupling. When all the conditions have been satisfied, the coupling is closed and the alarm deleted.

**Activating and deactivating a master–slave coupling via PLC signal**

A coupling is activated or deactivated via an axis-specific PLC signal “to axis”. Only the signal to the slave axis is relevant here. The signal resides in the technologies area.

| | |
|---|---|
| DB3x.DBB24.7 | “Activate master–slave coupling” |
| | 1 = Activate master–slave coupling |
| | 0 = Deactivate master–slave coupling |

If one of the conditions for activation or deactivation is not satisfied, the slave axis does not react to the PLC signal, i.e. the status of the coupling remains unchanged. No NC alarm is output.

**Example:**

- A parts program is processed in channel 1, channel state: “active”.
- A master–slave coupling is active, master axis and slave axis are in channel 1, PLC signal to slave axis DB3x.DBB24.7 = 1.
- The coupling is to be deactivated, PLC sets DB3x.DBB24.7 = 0.
- Since the channel is not in the “RESET” state, the coupling is not deactivated.
- The coupling is not deactivated until the parts program is terminated with M30 or RESET.

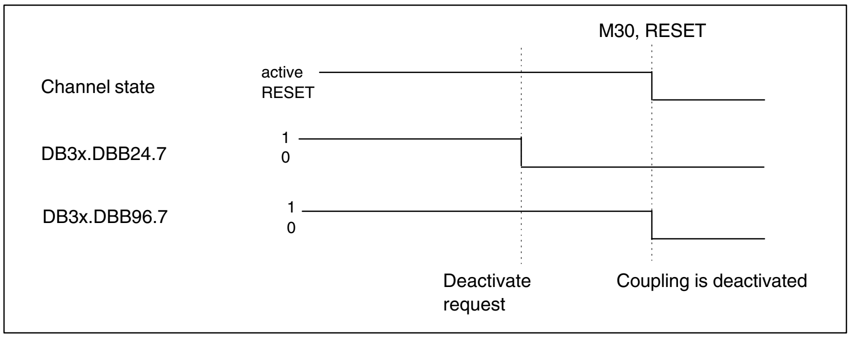



Fig. 2-15 Deactivating a coupling

**PLC signal: State of a master–slave coupling**

The status of a master–slave coupling is displayed in an axis-specific VDI signal “from axis”. The machine data of the slave axis are always relevant for a coupling. This signal is set irrespective of whether the coupling is always active (MD 63590) or has been activated per PLC (DB3x.DBB24.7).

| | |
|---|---|
| DB3x.DBB96.7 | “State master–slave coupling” |
| | 1 = Master–slave coupling active |
| | 0 = Master–slave coupling not active |

### 2.2.7 System response when a coupling is active

**PLC signal: Traversing the slave axis**

If a slave axis is traversed via the master axis when the coupling is active, the following PLC signals are output depending on the travel state:

| | |
|---|---|
| DB3x.DBB60.6 | “Exact stop fine” |
| DB3x.DBB60.7 | “Exact stop coarse” |
| DB3x.DBB61.4 | “Axis/spindle stop” |

Since the coupling is processed in the position control cycle, the travel command signals are not output:

DB3x.DBB64.7 “Travel command +/–”

All other signals show the current state of the axis.

**Actual-value display**

In the automatic basic display, the motion of the slave axis is not displayed for an active coupling, the actual value is “frozen”. If the coupling is deactivated, the actual-value display jumps to the current actual position. The next time the NC is started, the slave axis is synchronized with the NC so that the slave axis can be traversed from this position.
In the “Dialog” menu, “Service display” softkey, the motion of the slave axis is always displayed, even when the coupling is active.

**Traversing the slave axis**

A slave axis in an active coupling must not be traversed by the parts program, by the PLC or manually in JOG mode. If a coupled slave axis is traversed, this produces the reset alarm “75170 Axis AX1 overlaid motion not permissible”.

**Reference point approach**

The status of the coupling determines the method of reference point approach. This applies to referencing in JOG Ref mode, and referencing from within the parts program (G74).

If a master–slave coupling can be activated per PLC signal (DB3x,DBB24.7), the master and slave axes are referenced individually in the coupling state “not active”.

If a master–slave coupling is always active after power ON (MD 63590 = 1), only the master axis is referenced. In this case, the slave axis is never referenced. Since the coupling is active, the slave axis follows when the master axis is referenced.

MD 34110: REFP_CYCLE_NR of the slave axis must be set to –1 so that NC start is possible without having to reference the slave.

**Response in the event of an error**

In the event of error conditions with alarm reaction Follow-up in master and/or slave, each axis is braked down to 0 speed. The master–slave coupling is deactivated.

To prevent mechanical tension,
MD 36620: SERVO_DISABLE_DELAY_TIME and
MD 36610: AX_ENERGY_STOP_TIME
and the drive machine data
MD 1403: PULSE_SUPPRESSION_SPEED and
MD 1404: PULSE_SUPPRESSION_DELAY
must be set to the same values for the master and slave axes. This is the responsibility of the user.
The master–slave coupling does not become active again until the both axes have the state “Controller active” following a channel reset.

■

# Notes

# Supplementary Conditions 3

**NCU 572.2**

The Master/Slave for Drives function can be utilized on NCU 572.2 hardware only on condition that is has been specifically enabled for the customer.

**SINUMERIK 840Di**

The compile cycles function of the SINUMERIK 840D are currently available only on request for the SINUMERIK 840Di.

## 3.1 Speed/torque coupling (SW 6 and higher)

**Option**

The speed/torque coupling function is an option and not available in every control variant.

The master–slave function requires the master and slave axes to be operated on the same NCU.

**Further notes**

The master–slave function requires the master and slave axes to be operated on the same NCU.

- A coupled slave axis cannot be rotated around the **axis container**.
- **Closing and separating** the master–slave coupling is carried out when the axis has stopped.
- **Traversing a slave axis** with the coupling closed is possible only via the master axis.
- **Axis replacement** is not performed for coupled slave axes.
- When the coupling is closed via the slave axis, the master axis is **braked automatically** (if defined in the same channel). This produces an asymmetric response on closure and separation of the coupling. In contrast to closing, there is no automatic braking on separation.
- **Block search with calculation (SERUPRO)** takes into account the positional changes of coupled slave axes after a block search only if a system ASUB (asynchronous subroutine) “PROGEVENT.SPF” has been generated. This can be used to subsequently adjust the coupling state and associated axis positions so as to update changes to the coupling state.

**Differences compared to previous solution (SW 5.x and lower)**

- If a traversing movement is programmed for a slave axis that has already been coupled, the alarm "14092 Channel %1 Block %2 Axis %3 has the wrong type" appears.
- The setpoint position of the linked axis corresponds to the current actual position.
- On request, the coupling is closed or separated in the next axis standstill independent of the channel status. This enables the coupling status to be changed even during processing of a parts program.
- PLC interface signal DB3x.DBX61.5 "Position control" is no longer interpreted in the braking control logic of the slave axes. This is no longer set for an active coupling. Instead, the interface signal "Master–slave coupling status active" should be used.
- If a master axis is simultaneously configured as a slave axis, the alarm "26031 Axis %1 Configuration error master–slave" appears. Cascading is therefore not possible.
- If a coupling is requested and closed, the control activation signals are derived directly from the master axis.

## 3.2 Speed/torque coupling (SW 5.x and lower)

### 3.2.1 Changing of axes

Exchanging axes between channels is subject to the following restrictions:
To activate or deactivate a coupling, the channels of the slave and master axes must have the status RESET. The states of the default channels of the axes are scanned prior to activation/deactivation. At the time of activation and deactivation, the axes must be located in the default channel assigned by MD30550. A change of axis is possible in between times, even if the coupling is active.

### 3.2.2 Modulo rotary axis, spindles

**Modulo rotary axes**

Master and slave axes can also be rotary axes. The following must be observed:
On the slave axis, the actual value in the "Diagnosis" menu under softkey "Service" exceeds 360 degrees, even if MD 30310: ROT_IS_MODULO has been set to select modulo operation for the axis. The automatic basic display and the service display do not show the actual value modulo 360 until the coupling is deactivated.

**Spindles**

If a master–slave coupling is activated with spindles, the slave axis is in speed control mode. In this case too, the actual value of the slave axis exceeds 360 degrees in the service display. No modulo calculation is active. However, the value shown in the automatic basic display is modulo 360 degrees.

### 3.2.3 Simultaneous operation of master/slave coupling and clearance control function

The "Speed/torque coupling (master–slave)" and "Clearance control" functions can be operated simultaneously with the following restriction:
An axis that is traversed by the clearance control must be neither a master nor a slave axis in the master–slave function.

### 3.2.4 Displaying torque values and controller output in NCK GUD

To support installation, the current axial torque values in [Nm] and the set speed in [mm/min] or [rpm] of the P controller and the I controller of a torque control can be displayed on the operator panel front in the "Parameter – user data" area.

For this purpose, the appropriate GUDs must be set up. For a detailed description of the procedure to be followed, please refer to Section "File and program management" in the document "SINUMERIK 840D/810D/FM-NC Programming Guide Advanced".

Please proceed as follows:

Create SGUD

- "Services" menu
- If directory "Definitions" is not displayed, select definitions using the "Data selection" softkey
- Open directory "Definitions"
- "Manage data" softkey
- "New" softkey
- Create file

  Name: SGUD
  File type: Select global data/system
- OK
- File is opened

DEF NCK REAL MASTER_SLAVE_TORQUE[number of active axes]
DEF NCK REAL TORQUE_CTRL_P[number of active axes]
DEF NCK REAL TORQUE_CTRL_I[number of active axes]
Enter
M30

- Close
- Load

Create Initial.ini:

- Services menu
- In the root
- “Manage data” softkey
- “New” softkey
- Create new directory type “NC data backup”
- Create Initial.ini file in this directory:

  Name: initial
  Type: initialization program

- OK
- File is opened
  Enter
  M17
- Close
- Load

The following axis data are then displayed:

| | |
|---|---|
| MASTER_SLAVE_TORQUE[0] | Current torque in [Nm] |
| TORQUE_CTRL_P[0] | P component of an active torque control in [mm/min] or [rpm] |
| TORQUE_CTRL_I[0] | I component of an active torque control in [mm/min] or [rpm] |

### 3.2.5 Servo Trace

To support installation, the current torque values and the output of the torque control can be displayed on the MMC in the Servo Trace function.

---

**Caution**

The existing Servo Trace function has been expanded for master and slave. The operation of the “Servo Trace” is described in Chapter 10 of the Installation Guide.

---

In order to be able to select the data of a master–slave coupling in the menu in the servo trace, the following files must be created on the MMC. You can use the DOS shell and the editor “edit” to do this.

File: \ oem \ ibsvt.ini
Contents:
[OemSignalList]

| | | | |
|---|---|---|---|
| Item0 | = Type := Title, | Signal index := - 1, | Unit := No |
| Item5 | = Type := Signal, | Signal index := 200, | Unit := Torque I Force |
| Item10 | = Type := Signal, | Signal index := 201, | Unit := Torque I Force |
| Item15 | = Type := Signal, | Signal index := 202, | Unit := NcSpeed |

File: \ oem \ language \ Ibsvt_gr.ini
Contents:
[OemComboBoxItemNames]

| | | |
|---|---|---|
| Item5 | = | “MASTER–SLAVE” |
| Item10 | = | “Master torque” |
| Item15 | = | “Slave torque” |
| Item20 | = | “Controller output” |

This file is language-specific and must be created with the corresponding language code (uk for English) for all available languages.

Following the next MMC power ON, you can use the selection menu to select the following signals in the Servo Trace menu.

- Master torque
- Slave torque
- Controller output.

---

**Caution**

To achieve a higher signal resolution, the data are displayed in the following units:
Torque in [milli Nm]
Controller output in [internal increments / s]

---

No further machine data need be set to activate a measurement.

Up to 4 signals can be recorded in one measurement. The associated machine axis is selected in the axis selection for the torque values; for the controller output, the machine axis of the slave axis of this control is selected.

**Example:**
Master axis: X1
Slave axis: Y1

The following are to be recorded:

| | | |
|---|---|---|
| Master torque | axis selection | X1 |
| Slave torque | axis selection | Y1 |
| Controller output | axis selection | Y1 |

With 4 active couplings, it is possible to record all 4 torque values of the master axes or 4 controller outputs.

With automatic scaling, the measured curves of a display are always overlaid. In order to compare the values of the curves properly, the scaling must be set the same for both curves (see graphic 2 in Figure 3-1). The scaling can still be modified in the Scale menu after the measurement.

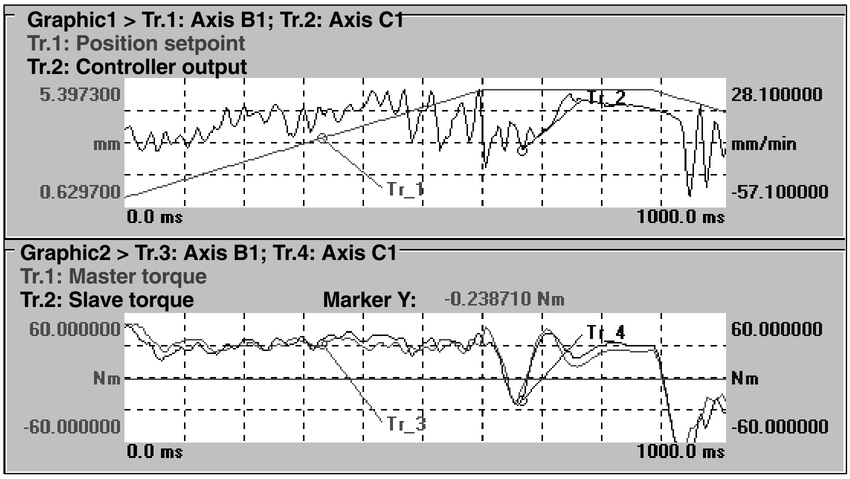



Fig. 3-1 Example of a measurement with 4 measured values

### 3.2.6 Controller data to analog output

Machine data MD 63595 TRACE_MODE bit 0 can be set to activate output of controller data to an analog output.
The following data are output to the analog output at the terminal block:

- Torque of the master axis at analog converter 1
- Torque of the slave axis at analog converter 2
- Torque control output at analog converter 3

The torque values are standardized to 8 V relative to the nominal torque; the torque controller output is standardized to 8 V as mm/min relative to the maximum speed of the slave axis.

The slots occupied by the analog converter on the terminal block are defined by MD 10364 HW_ASSIGN_ANA_FASTOUT.

### 3.2.7 Creating alarm texts

Add an entry for the alarm text files of the technology card in the [Text Files] section of the C:\OEM\MBDDE.INI file:
CZYK=C:\OEM\TF_

Create language-specific text files TF_xx.COM in directory C:\OEM\. xx stands for the language code, e.g. GR for German and UK for English.

Enter the following alarm texts here: in TF_GR.COM:

075150 0 0 “Slave axis %1 and master axis for setspeed coupling are identical”
075151 0 0 “Slave axis %1 and master axis for torque control are identical”
075160 0 0 “Slave axis %1, master–slave coupling is not active”
075170 0 0 “Axis %1 overlaid motion not permissible”

# Notes

# Data Descriptions (MD, SD) 4

## 4.1 Machine data of speed/torque coupling (SW 6 and lower)

| **37250**<br>MD number | **MS_ASSIGN_MASTER_SPEED_CMD**<br>Machine axis number of master axis for speed setpoint coupling | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 31 |
| Change effective after power ON | Protection level: 2 | Unit: – |
| Data type: DWORD | Applies from SW: 6.1 | |
| Meaning: | A master–slave speed setpoint coupling is configured by specifying the machine axis number of the master axis associated with this slave. | |
| Related to .... | MD 37252 MS_ASSIGN_MASTER_TORQUE_CTR | |

| **37252**<br>MD number | **MS_ASSIGN_MASTER_TORQUE_CTR**<br>Master axis number for torque control | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 31 |
| Change effective after power ON | Protection level: 2 | Unit: – |
| Data type: DWORD | Applies from SW: 6.1 | |
| Meaning: | A master–slave speed setpoint coupling is configured by specifying the machine axis number of the master axis associated with this slave. Even torque distribution is achieved by means of the torque compensatory controller.<br>In the default setting = 0, the same master axis is used for torque distribution as for set speed coupling MS_ASSIGN_MASTER_TORQUE_CTR. | |
| Related to .... | MD 37250 MS_ASSIGN_MASTER_SPEED_CMD<br>MD 37254 MS_TORQUE_CTRL_MODE<br>MD 37256 MS_TORQUE_CTRL_P_GAIN<br>MD 37258 MS_TORQUE_CTRL_I_TIME<br>MD 37260 MS_TORQUE_WEIGHT_SLAVE | |

| | | | | |
|---|---|---|---|---|
| **37254**<br>MD number | **MS_TORQUE_CTRL_MODE**<br>Connection of torque control output | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 3 |
| Change effective afterImmediately | | Protection level: 2 / 7 | | Unit: – |
| Data type: DWORD | | | Applies from SW: 6.1 | |
| Meaning: | The output of the torque compensatory control is applied when the torque control is active:<br>0: Master and slave axes<br>1: Slave axis<br>2: Master axis<br>3: None of the axes | | | |
| Special cases, errors, ...... | for | | | |
| Related to .... | MD 37250 MS_ASSIGN_MASTER_SPEED_CMD<br>MD 37252 MS_ASSIGN_MASTER_TORQUE_CTR<br>MD 37255 MS_TORQUE_CTRL_ACTIVATION | | | |
| Further references | Tab. 2.2 Combination options for machine data | | | |

| | | | | |
|---|---|---|---|---|
| **37255**<br>MD number | **MS_TORQUE_CTRL_ACTIVATION (SW 6.4 and higher)**<br>Activate torque compensatory control | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 1 |
| Changes effective after NEW CONF | | Protection level: 2 / 7 | | Unit: – |
| Data type: BYTE | | | Applies from SW: 6.4 | |
| Meaning: | The torque compensatory controller can be activated/deactivated via MD 37254 or via PLC (DB31, ... DBX24.4).<br>In the case of the PLC, MD 37254 is used only to onterconnec the torque compensatory controller.<br>Value 0: Activation/deactivation via MD 37254<br>Value 1: Activation/deactivation via DB31, ... DBX24.4 | | | |
| Related to .... | MD 37254: MS_TORQUE_CTRL_MODE | | | |

| | | | | |
|---|---|---|---|---|
| **37256**<br>MD number | **MS_TORQUE_CTRL_P_GAIN**<br>Gain factor of torque compensatory controller | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 100 |
| Change effective afterNEW_CONF | | Protection level: 2/7 | | Unit: % |
| Data type: DOUBLE | | | Applies from SW: 6.1 | |
| Meaning: | Gain factor of torque compensatory control<br>The gain factor is entered as a percentage ratio between the maximum load-side axis speed of the slave axis and the rated torque.<br>The maximum axis speed is derived from MD32000, the nominal torque is the product of drive machine data MD1725. | | | |
| Related to .... | MD 37254 MS_TORQUE_CTRL_MODE<br>MD 37258 MS_TORQUE_CTRL_I_TIME<br>MD 32000 MAX_AX_VELO | | | |

| | | | | |
|---|---|---|---|---|
| **37258**<br>MD number | **MS_TORQUE_CTRL_I_TIME**<br>Reset time for torque compensatory controller | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 100 |
| Change effective afterNEW_CONF | | Protection level: 2/7 | | Unit: s |
| Data type: DOUBLE | | | Applies from SW: 6.1 | |
| Meaning: | Reset time for torque compensatory control<br>The reset time does not become active until the P gain factor > 0. | | | |
| Related to .... | MD 37254 MS_TORQUE_CTRL_MODE<br>MD 37256 MS_TORQUE_CTRL_P_GAIN<br>MD 32000 MAX_AX_VELO | | | |

| | | | | |
|---|---|---|---|---|
| **37260**<br>MD number | **MS_MAX_CTRL_VELO**<br>Limitation of torque compensatory control | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 100 |
| Change effective afterNEW_CONF | | | Protection level: 2/7 | Unit: % |
| Data type: DOUBLE | | | Applies from SW: 6.1 | |
| Meaning: | Limitation of torque compensatory control<br>The speed setpoint computed by the torque compensatory control is limited.<br>The percentage limitation refers to MD 32000 MAX_AX_VELO of the slave axis. | | | |
| Related to .... | MD 37254 MS_TORQUE_CTRL_MODE<br>MD 37256 MS_TORQUE_CTRL_P_GAIN<br>MD 37258 MS_TORQUE_CTRL_I_TIME<br>MD 32000 MAX_AX_VELO | | | |

| | | | | |
|---|---|---|---|---|
| **37262**<br>MD number | **MS_COUPLING_ALWAYS_ACTIVE**<br>Master/slave coupling active after power ON | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 1 |
| Change effective afterNEW_CONF | | | Protection level: 2/7 | Unit: – |
| Data type: Byte | | | Applies from SW: 6.1 | |
| Meaning: | Power-on response of a master–slave coupling.<br>0: Temporary coupling<br>The coupling is activated/deactivated via the PLC interface signals and using the language commands.<br>1: Permanent coupling<br>The coupling is activated permanently via this machine data.<br>The PLC interface signals and language commands have no effect. | | | |
| Related to .... | MD 37252 MS_ASSIGN_MASTER_TORQUE_CTR<br>MD 37250 MS_ASSIGN_MASTER_SPEED_CMD | | | |

| | | | | |
|---|---|---|---|---|
| **37264**<br>MD number | **MS_TENSION_TORQUE**<br>Master–slave tension torque | | | |
| Default setting: 0 | | Min. input limit: –100 | | Max. input limit: 100 |
| Modification valid IMMEDIATELY | | | Protection level: 2/7 | Unit: – |
| Data type: PERCENT | | | Applies from SW: 6.1 | |
| Meaning: | You can enter a constant tension torque between the master and slave axis as a percentage of the nominal drive torque of the slave axis | | | |
| Related to .... | MD 37252 MS_ASSIGN_MASTER_TORQUE_CTR<br>MD 37266 MS_TENSION_TORQ_FILTER_TIME | | | |

| | | | | |
|---|---|---|---|---|
| **37266**<br>MD number | **MS_TENSION_TORQ_FILTER_TIME**<br>Filter time constant master–slave tension torque | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 100 |
| Changes effectiveImmediately | | | Protection level: 2/7 | Unit: s |
| Data type: DOUBLE | | | Applies from SW: 6.1 | |
| Meaning: | The tension torque between the master and slave axes can be applied via a PT1 filter.<br>Each change to MD 37264 is then traversed with the time constant of the filter.<br>The filter is inactive per default, each change in torque is left unfiltered. | | | |
| Related to .... | MD 37264 MS_TENSION_TORQUE | | | |

| | | | | |
|---|---|---|---|---|
| **37268**<br>MD number | **MS_TORQUE_WEIGHT_SLAVE**<br>Weighting of the torque value for the slave axis | | | |
| Default setting: 50 | | Min. input limit: 0 | | Max. input limit: 100 |
| Change effective afterNEW_CONF | | | Protection level: 2/7 | Unit: _ |
| Data type: PERCENT | | | Applies from SW: 6.1 | |
| Meaning: | You can use the weighting to configure the torque of the slave axis relative to the total torque. This enables a different torque control to be implemented for the master and slave axes. A 50 : 50% torque control makes sense for motors that have the same torque.<br>The torque for the master axis is obtained implicitly as 100% – MD37268. | | | |
| Related to .... | MD 37252 MS_ASSIGN_MASTER_TORQUE_CTR<br>MD 37266 MS_TENSION_TORQ_FILTER_TIME | | | |

| | | | | |
|---|---|---|---|---|
| **37270**<br>MD number | **MS_VELO_TOL_COARSE (from SW 6.4)**<br>Master–slave velocity tolerance “coarse” | | | |
| Default setting: 10.0 | | Min. input limit: | | Max. input limit: |
| Changes effective after NEW CONF | | | Protection level: 2 / 7 | Unit: % |
| Data type: DOUBLE | | | Applies from SW: 6.4 | |
| Meaning: | Tolerance window “coarse” for the difference velocity between master and slave. The PLC interface signal DB3x.DBX96.3 is set for a velocity difference within the tolerance window. The MD is specified as a percentage (%) of MD 32000: MAX_AX_VELO. | | | |
| Related to .... | MD 32000: MAX_AX_VELO | | | |

| | | | | |
|---|---|---|---|---|
| **37272**<br>MD number | **MS_VELO_TOL_FINE (from SW 6.4)**<br>Master–slave velocity tolerance “fine” | | | |
| Default setting: 10.0 | | Min. input limit: | | Max. input limit: |
| Changes effective after NEW CONF | | | Protection level: 2 / 7 | Unit: % |
| Data type: DOUBLE | | | Applies from SW: 6.4 | |
| Meaning: | Tolerance window “fine” for the difference velocity between master and slave. The PLC interface signal DB3x.DBX96.2 is set for a velocity difference within the tolerance window. The MD is specified as a percentage (%) of MD 32000: MAX_AX_VELO. | | | |
| Related to .... | MD 32000: MAX_AX_VELO | | | |

| | | | | |
|---|---|---|---|---|
| **37274**<br>MD number | **MS_MOTION_DIR_REVERSE (from SW 6.4)**<br>Invert master–slave direction of travel | | | |
| Default setting: 0 | | Min. input limit: 0 | | Max. input limit: 1 |
| Changes effective after NEW CONF | | | Protection level: 2 / 7 | Unit: – |
| Data type: BYTE | | | Applies from SW: 6.4 | |
| Meaning: | The direction of travel of the slave axis is to be inverted in coupled state.<br>1: Invert direction of travel<br>0: Direction of travel unchanged | | | |
| Related to .... | MD 32100: AX_MOTION_DIR | | | |

## 4.2 Machine data of speed/torque coupling (SW 5.x and lower)

The speed/torque coupling (master–slave) is implemented as a compile cycles application. In addition to the function-specific machine data, the following standard machine data must therefore be set also:

- Option data.

**Warning**

Failure to take appropriate precautions **can** have undesirable consequences.

The function activated by the option data initiate the corresponding compile cycles. These cycles can have a significant effect on the behavior of the control system and may cause hazardous situations by accessing the NC.
Before a compile cycle is activated, the necessary safety precautions must be taken to protect operating personnel and machine (safety precautions against parameterization or programming errors in the compile cycles may need to be taken).

| **63550**<br><br>MD number | **$MA_MS_ASSIGN_MASTER_SPEED_CMD**<br><br>Configuration of a master/slave coupling | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: number of axes available |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: machine axis number |
| Data type: INT | | |
| Meaning: | This machine data is used to assign a master axis to a potential slave axis for set speed coupling. This configures a master–slave coupling.<br><br>Master and slave axis do not have to be programmed in the same channel. | |

| **63555**<br><br>MD number | **$MA_MS_ASSIGN_MASTER_TORQUE_CTRL**<br><br>Configuration of a master/slave coupling | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: number of axes available |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: machine axis number |
| Data type: INT | | |
| Meaning: | This machine data assigns a master axis to the slave axis for torque control. If the value 0 is entered, the same master is used for the torque control as for the set speed coupling. This applies to most cases.<br><br>Master and slave axis do not have to be programmed in the same channel. | |

| **63560**<br><br>MD number | **$MA_MS_TORQUE_CTRL_P_GAIN**<br><br>P gain of the torque control | |
|---|---|---|
| Default setting: 0.0 | Min. input limit: 0 | Max. input limit: 100.0 |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: % |
| Data type: DOUBLE | | |
| Meaning: | The P controller calculates a set speed nset by multiplying the torque difference by the P gain Kp.<br><br>nset = Mdiff * Kp<br><br>The dimension of the P gain is [(mm/min)/Nm].<br><br>A percentage value of the ratio is input:<br>Maximum drive speed [mm/min] / rated torque [Nm]<br><br>The data of the slave axis are relevant for a torque control. | |

| **63565**<br><br>MD number | **$MA_MS_TORQUE_CTRL_I_TIME**<br><br>Reset time I controller of the torque control | |
|---|---|---|
| Default setting: 0.0 | Min. input limit: 0.0 | Max. input limit: 100.0 |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: s |
| Data type: DOUBLE | | |
| Meaning: | The I controller calculates a set speed by multiplying the sum of the torque difference Mdiff by the I gain.<br><br>nset = Mdiff * Ki<br>The reset time of the torque control I_TIME is used to parameterize the gain factor Ki of the I controller. Ki can only be calculated if the gain factor of the P controller Kp <> 0, i.e. the I controller can only be active if the P component is also calculated.<br>Ki = 1/ position controller cycle * I_TIME * Kp<br><br>The reset time is input in seconds. | |

| 63570 MD number | $MA_MS_TORQUE_CTRL_MODE Connection of the torque control output | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 2 |
| Modifications take immediate effect | Protection level: 2 / 7 | Unit: – |
| Data type: INT | | |
| Meaning: | This machine data enables the set speed calculated in the torque control to be freely connected to the master and slave axes. Even if the set speed is not applied to the axis, the torque control calculates the set speed.<br>Meaning:<br>0: Switch controller output to master and slave<br>1: Switch controller output only to slave<br>2: Switch controller output only to master<br>3: Controller is deactivated, only the set speed coupling is active | |

| 63575 MD number | $MA_MS_ASSIGN_MASTER_SPEED_CMD Weighting of the current torque values | |
|---|---|---|
| Default setting: 50.0 | Min. input limit: 0.0 | Max. input limit: 100.0 |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: % |
| Data type: DOUBLE | | |
| Meaning: | This machine data performs a weighting of the input variables of the torque compensator to enable a parameterizable torque distribution over both drives. If the motors are identical and the same drive parameters are to be set for the motors to produce the same torque, the standard parameterization 50% is recommended. The MD refers to the torque of the slave axis, the torque of the master axis is weighted with the difference between the MD and 100%.<br>Example:<br>The slave axis must produce 30% of the overall torque. 70% are supplied by the master axis.<br>$MA_MS_TORQUE_WEIGHT_SLAVE = 30 | |

| 63580 MD number | $MA_MS_TENSION_TORQUE Tension torque | |
|---|---|---|
| Default setting: 0.0 | Min. input limit: –100.0 | Max. input limit: 1000.0 |
| Modifications take immediate effect | Protection level: 2 / 7 | Unit: % |
| Data type: Double | | |
| Meaning: | This machine data can be used to apply a constant tension torque as input to the torque control. This tension torque is applied continuously and produces a mutual tensioning of the coupled drives. The MD of the slave axis is relevant for the tension of a coupling. The tension torque can be positive or negative.<br><br>The value to be input is a percentage of the rated torque of the slave axis.<br><br>The MD is active immediately after a change. This enables a different tension torque to be implemented as appropriate to the machining situation. A STOPRE must be programmed to achieve block-synchronous activation of a change in tension torque from the parts program. | |

| 63585<br><br>MD number | $MA_MS_TENSION_TORQ_FILTER_TIME<br><br>Time constant of the PT1 filter for tension torque | |
|---|---|---|
| Default setting: 0.0 | Min. input limit: 0.0 | Max. input limit: 100.0 |
| Modifications take immediate effect | Protection level: 2 / 7 | Unit: s |
| Data type: DOUBLE | | |
| Meaning: | The tension torque is applied to the torque control via a PT1 filter. This machine data is used to parameterize the PT1 filter. The time constant is input in seconds.<br>If the tension torque is changed, the torque is increased continuously.<br>Input zero to completely deactivate the filter. | |

| 63590<br><br>MD number | $MA_MS_COUPLING_ALWAYS_ACTIVE<br><br>Master–slave coupling active after power ON | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 1 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: INT | | |
| Meaning: | This machine data specifies the status of a coupling after power ON.<br><br>Value 1:<br>As soon as the conditions for activation of a coupling are satisfied after a power ON, the coupling is activated. It can no longer be deactivated, i.e. it is no longer possible to operate the drives separately. Modifying the PLC signal at axis DB3x.DBB24.7 has no effect.<br><br>Value 0:<br>The coupling can be activated via the PLC signal at axis DB3x.DBB24.7. | |

| 63595<br><br>MD number | $MA_TRACE_MODE<br><br>Activate/deactivate master–slave trace | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 2 |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: – |
| Data type: INT | | |
| Meaning: | This machine data activates a trace for start-up of a master–slave coupling<br><br>Bit 0: 0: No trace active<br>1: Analog trace is active: from this coupling, the torque of the master axis, slave axis and control output is output to analog outputs on the terminal box.<br><br>Bit 1: 0: Open and close coupling in channel status RESET only.<br>1: Open and close coupling without channel status RESET.<br><br>Bit 2: 0: Open coupling when master or slave axis are in follow-up mode and signal "Axis stopped" = 1.<br>1: Open coupling when master or slave axis are not in control and signal "Axis stopped" = 1. | |

| **63600** | **$MA_MS_MAX_CTRL_VELO** | |
|---|---|---|
| MD number | Limit value for controller output | |
| Default setting: 100 | Min. input limit: 0 | Max. input limit: 100 |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: % |
| Data type: DOUBLE | | |
| Meaning: | This machine data limits the controller output of a master–slave coupling to a maximum value. The value is entered as a percentage value relative to the maximum speed of the slave-axis. The controller output is limited by this value in the positive and negative direction.<br>The default is 100%. | |

## 4.3 System variables (SW 6 and higher)

After a block search, the coupling status and associated axis positions can be adjusted subsequently by means of a system ASUB (asynchronous subroutine) “PROGEVENT.SPF”. System variables $P_SEARCH_MASL_C, $P_SEARCH_MASL_D and $AA_MASL_STAT are available for this purpose; they can be used to alter the positional offset between the coupled axes as well as the coupling status:

| Identifier | Meaning | Ref. |
|---|---|---|
| **Axis/spindle-specific($MA_ ...)** | | |
| $P_SEARCH_MASL_C[slave axis identifier] | This variable registers a change in the coupling status during the block search. | |
| $P_SEARCH_MASL_D[slave axis identifier] | This variable indicates the positional offset between the slave and master axes at the instant the coupling was closed | |
| $AA_MASL_STAT[slave axis identifier] | This variable outputs the current coupling status.<br>A value ≠ 0 “Master–slave coupling active”. In this case, it contains the current machine number of the master axis and, if the NCU link is active (several operating panel fronts and NCUs), also the NCU No. at the hundreds position. Example: 201 for Axis 1 on NCU2. | |

■

## Notes

# Signal Descriptions 5

## 5.1 Axis-specific signals

| **DB31 – DB38**<br>**DBX24.4**<br><br>Data block | **Activate torque compensatory controller**<br><br>Signal(s) from axis spindle (PLC–> NCK) |
|---|---|
| Edge evaluation: yes | Signal(s) updated: Cyclically |
| Signal state 1<br>or signal transition 0 –> 1 | Torque compensatory control is to be activated<br><br>The following conditions must be satisfied for activation:<br>– Difference “Fine” reached (DB3x.DBX96.2) |
| Signal state 0 or<br>signal transition 1–> 0 | Torque compensatory controller is to be deactivated. |

| **DB31 – DB38**<br>**DBX24.7**<br><br>Data block | **Activate master–slave coupling**<br><br>Signal(s) from axis spindle (PLC–> NCK) |
|---|---|
| Edge evaluation: yes | Signal(s) updated: Cyclically |
| Signal state 1<br>or signal transition 0 –> 1 | Torque compensatory controller is to be activated |
| Signal state 0 or<br>signal transition 1–> 0 | Master–slave coupling must be deactivated.<br>The following conditions must be fulfilled to activate or deactivate the exchange function:<br>– Master and slave axis in position control (DB3x.DBB61.7)<br>– Master and slave axis at standstill (DB3x.DBB61.4)<br>– The channels of the master and slave axes are in the “RESET” state (DB2x.DBB35.7)<br>If a condition is not satisfied, the coupling is not activated or deactivated. No alarm appears and the status of the coupling remains the same.<br>If all the conditions are satisfied at a later point in time, the coupling is activated or deactivated depending on the status of the signal.<br>The signal at the slave axis of a coupling is relevant. |

| DB31 – DB38<br>**DBX96.2**<br><br>Data block | **Difference speed "Fine"**<br><br>Signal(s) from axis spindle (NCK –> PLC) |
|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically |
| Signal state 1<br>or signal transition 0 –> 1 | The difference speed lies in the range defined by MD 37272:<br>MS_VELO_TOL_FINE. |
| Signal state 0 or<br>signal transition 1–> 0 | The difference speed has not reached the range defined in MD 37272:<br>MS_VELO_TOL_FINE. |

| DB31 – DB38<br>**DBX96.3**<br><br>Data block | **Difference speed "Coarse"**<br><br>Signal(s) from axis spindle (NCK –> PLC) |
|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically |
| Signal state 1<br>or signal transition 0 –> 1 | The difference speed lies in the range defined by<br>MD 37270: MS_VELO_TOL_COARSE. |
| Signal state 0 or<br>signal transition 1–> 0 | The difference speed has not reached the range defined in MD 37270:<br>MS_VELO_TOL_COARSE. |

| DB31 – DB38<br>**DBX96.4**<br><br>Data block | **Status of the torque compensatory control**<br><br>Signal(s) from axis spindle (NCK –> PLC) |
|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically |
| Signal state 1<br>or signal transition 0 –> 1 | Torque compensatory control is active. |
| Signal state 0 or<br>signal transition 1–> 0 | Torque compensatory control is not active.<br>The signal at the slave axis of a coupling is relevant. |

| DB31 – DB38<br>**DBX96.7**<br><br>Data block | **Status of the master/slave coupling**<br><br>Signal(s) from axis spindle (NCK –> PLC) |
|---|---|
| Edge evaluation: no | Signal(s) updated: Cyclically |
| Signal state 1<br>or signal transition 0 –> 1 | Master–slave coupling is active. |
| Signal state 0 or<br>signal transition 1–> 0 | Master–slave coupling is not active.<br>The signal at the slave axis of a coupling is relevant. |

# Examples 6

## 6.1 Speed/torque coupling (SW 6 and higher)

### 6.1.1 Master–slave coupling between AX1=Master and AX2=Slave

**Configuration**

Master–slave coupling between AX1=Master and AX2=Slave.

1. Machine axis number of master axis with speed setpoint coupling
   MD 37250: MS_ASSIGN_MASTER_SPEED_CMD[AX2] = 1
2. Master axis with torque distribution identical to master axis with speed setpoint coupling
   MD 37252: MS_ASSIGN_MASTER_TORQUE_CTR[AX2] = 0
3. Permanent coupling
   MD 37262: MS_COUPLING_ALWAYS_ACTIVE[AX2] = 1
4. Torque is injected in both the master and slave axes
   MD 37254: MS_TORQUE_CTRL_MODE[AX2] = 0
5. Torque distribution between the master and slave axes is 50% to 50%
   MD 37268: MS_TORQUE_WEIGHT_SLAVE[AX2] = 50
6. Parameters of torque compensatory controller
   MD 37256: MS_TORQUE_CTRL_P_GAIN[AX2] = 0.5
   MD 37258: MS_TORQUE_CTRL_I_TIME[AX2] = 5.0

## 6.1.2 Close coupling via the PLC

This application allows you to close or separate a master–slave coupling between the machine axes AX1=Master axis and AX2=Slave axis during operation.

**Preconditions**

- A configured master axis
  MD 37250: MS_ASSIGN_MASTER_SPEED_CMD $\neq$ 0
- Activation of a master–slave coupling via
  MD 37262: MS_COUPLING_ALWAYS_ACTIVE=0
- The coupling is open.

**Typical sequence of operations**

| Action | Effect/comment |
|---|---|
| 1. Approach coupling position | Each axis moves to the coupling position. |
| 2. Close coupling mechanically | The two axes are connected mechanically. |
| 3. Request to close the coupling | PLC interface signal “Master/Slave ON” DB32, ... DBX24.7 is set. |
| 4. Read back coupling status | When the axis is at standstill, the coupled slave axis sets PLC interface signal “Master/slave active” DB32, ... DBX96.7 and cancels signal “Position controller active” DB32, ... DBX61.5.<br>Wait for checkback signal. |
| 5. Move master–slave grouping | The master axis is moved. |

### 6.1.3 Close/separate coupling via parts program

This application allows you to close or separate a master–slave coupling between the machine axes AX1=Master axis and AX2=Slave via the parts program.

**Preconditions**

- A configured master axis MD 37250 ≠ 0.
- Activation of a master–slave coupling via MD 37262 = 0.
- The coupling is open.

**Parts program**

```
N10 G0 AX1=0 AX2=0; Approach coupling position. Each of the axes
                    approaches the coupling position.
N20 MASLON (AX2);   Close the coupling mechanically. Both axes
                    are mechanically coupled to one another.
N30 AX1=100;        Move master-slave grouping
                    The master axis is moved.
                    The slave follows the master coupled
                    via the speed setpoint.
N40 MASLOF (AX2);   Open coupling. The axes are mechanically
                    separated from one another.
N50 AX1=200 AX2=200;Move the master and slave axes.
                    The master axis is moved, decoupled from the
                    slave axis.
N60 M30
```

### 6.1.4 Release the mechanical brake

This application allows implementation of a brake control for machine axes AX1=Master axis and AX2=Slave axis in a master–slave coupling.

**Preconditions**

- Master–slave coupling is configured.
- Axes are stationary.
- No servo enable signals.

**Typical sequence of operations**

| Action | Effect/comment |
|---|---|
| 1. Request to close the coupling | PLC interface signal “Master/Slave ON” DB32, ... DBX24.7 is set. |
| 2. Set servo enable signal | PLC interface signal “Servo enable” DB31, .... DBX2.1 is set for both axes. |
| 3. Evaluate checkback signals | AND the PLC interface signals of the master axis:<br>– DB31, ... DBX61.7 “Current controller active”<br>– DB31, ... DBX61.6 “Speed controller active”<br>– DB31, ... DBX61.5 “Position controller active” |
| | AND the PLC interface signals of the slave axis:<br>– DB32, ... DBX61.7 “Current controller active”<br>– DB32, ... DBX61.6 “Speed controller active”<br>– DB32, .... DBX96.7 “Master/slave active” |
| 4. Release brake | If the result of the AND operations on the master and slave axes is $\neq$ 0, the brake may be released. |

## 6.2 Speed/torque coupling (SW 5.x and lower)

### 6.2.1 General start-up of a compile cycle function

**Preconditions**

- MMC software version must be 3.5 or higher.
- An NCK technology card with the speed/torque coupling (master–slave) function must be available.

**Saving SRAM contents**

As the first step in installing a compile cycle function, the original card inserted in the NCU must be replaced by the technology card.
This measure is identical to the procedure followed for upgrading the NCU to a higher software version and likewise requires the static (battery-backed) control system memory to be erased. When this is done, all data stored in the SRAM are lost and they must therefore be saved beforehand. For a detailed description, please refer to the Manufacturer/Service Documentation "SINUMERIK 840D/SIMODRIVE 611D Installation and Start-Up Guide":

1. Enter the machine manufacturer password.
2. Switch to the "Services" operating area.
3. Press the "Series start-up" softkey.
4. Select "NC" and "PLC" as the areas to be saved and enter a name of your choice for the archive file to be created on the hard disk. Finish by pressing the RETURN key.
5. If the control system contains machine-specific compensation data, then these must be saved in a separate archive file:
   Press softkey "Data Out" and
   select under item "NC active data" the following data as required:
   "Measuring system compensations",
   "Sag/Angularity comp." and
   "Quadrant error compensation".
   Save these data by selecting softkey "Archive ..." and specify another file name for a second archive file.

Keep the archive files you have created in a safe place. They will allow you to restore original settings in your system.

**Insert the PC card**

1. Deactivate the control.
2. Insert the PC card with the new firmware (technology card) in the PCMCIA slot of the NCU.
3. Turn switch S3 on the front panel of the NCU to 1.
4. Switch the control system back on again.
   During power-up, the firmware is copied from the PC card to the NCU memory.
5. Wait until number "6" appears on the NCU digital display (after approximately 1 minute).
6. Turn switch S3 back to zero.

**Caution**

If number “6” does not appear, then an error has occurred.

- Incorrect PC card (e.g. card for NCU2 in NCU3 hardware).
- Card hardware defective.

**Copy back SRAM contents**

To copy the saved data back into the control system, proceed as described in Section “Series Start-Up”. Please read all information provided by the manufacturer about new software versions.

1. Enter the machine manufacturer password.
2. Select “Data In” and “Archive...”.
3. Then load the archive with backup compensation data (if applicable).

### 6.2.2 Start-up of a master–slave coupling

Start-up of the speed/torque coupling (master–slave) function requires activation of the compile cycle as the next step.

**Option data for compile cycles**

1. Set the option for compile cycle application 3
2. **Run up the software again.** Then MD 63550–63590 (search for “ANALOG_AXIS” or “63550”) are at the end of the axial machine data list:

**Warning**

Failure to take appropriate precautions **can** have undesirable consequences.

The functions activated via MD 19600: CC_EVENT_MASK[n] initiate corresponding compile cycles. These cycles can have a significant effect on the behavior of the control system and may cause hazardous situations by accessing the NC.
Before a compile cycle is activated, the necessary safety precautions must be taken to protect operating personnel and machine (safety precautions against parameterization or programming errors in the compile cycles may need to be taken).

**Alarms**

3. Enter the alarm texts in the language-specific text files TK1_GR.COM and TK1_UK.COM.

**GUD**

4. If necessary, create GUDs from Section  to check the torque control 3.

**Configure the master/slave coupling**

5. Assign a master axis to a slave axis via MDs 63550, 63555 and configure the torque control with MDs 63560–63590.

### 6.2.3 Sample configuration for two master/slave couplings

Two master/slave couplings must be configured for the following machine
Channel 1: Machine axes AX1, AX2, AX3
Channel 2: Machine axes AX4, AX5

**Coupling 1:**

- Master speed coupling AX1
- Master torque control AX1
- Slave AX3
- The coupling must be active after power ON
- Set speed of the torque control must be applied to master and slave
- Weighting of torque for master–slave 50%–50%
- No tension torque

Enter the following MDs for coupling 1:

MD 63550: MS_ASSIGN_MASTER_SPEED_CMD[AX3]= 1
MD 63555: ASSIGN_MASTER_TORQUE_CTRL[AX 3] = 1 or 0
MD 63590: MS_COUPLING_ALWAYS_ACTIVE[AX 3] = 1
MD 63575: MS_TORQUE_WEIGHT_SLAVE[AX 3] = 50
MD 63570: MS_TORQUE_CTRL_MODE[AX 3] = 0
MD 63580: MS_TENSION_TORQUE[AX 3]= 0

The P component and the I component of the torque control must be configured with the following MDs:

MD 63560: MS_TORQUE_CTRL_P_GAIN[AX 3]=
MD 63565: MS_TORQUE_CTRL_I_TIME[AX 3] =

**Coupling 2:**

- Master speed coupling AX4 (channel 2)
- Master torque control AX4
- Slave AX2 (channel 1)
- Coupling must be activated via PLC signal
- Set speed of the torque control must be applied to master and slave
- Torque weighting for slave-master 30%–70%
- Tension torque 10% of rated torque

The following MDs are entered for coupling 2:

MD 63550: MS_ASSIGN_MASTER_SPEED_CMD[AX2]= 4
MD 63555: ASSIGN_MASTER_TORQUE_CTRL[AX 2] = 4 or 0
MD 63590: MS_COUPLING_ALWAYS_ACTIVE[AX 2] = 0
MD 63575: MS_TORQUE_WEIGHT_SLAVE[AX 2] = 30
MD 63570: MS_TORQUE_CTRL_MODE[AX 2] = 0
MD 63580: MS_TENSION_TORQUE[AX 2]= 10

The P component and the I component of the torque control must be configured with the following MDs:

MD 63560: MS_TORQUE_CTRL_P_GAIN[AX 2]=
MD 63565: MS_TORQUE_CTRL_I_TIME[AX 2] =

■

# Data Fields, Lists 7

## 7.1 Interface signals

| DB number | Bit, byte | Name | Ref. |
|---|---|---|---|
| **Axis/spindle-specific** | | | |
| DB3x. | DBX24.4 | "Activating the torque compensatory control" (SW 6.4 and higher) | |
| DB3x. | DBX24.7 | "Activate master/slave coupling" | |
| DB3x. | DBX96.2 | "Difference speed Fine" (SW 6.4 and higher) | |
| DB3x. | DBX96.3 | "Difference speed Coarse" (SW 6.4 and higher) | |
| DB3x. | DBX96.4 | "Status of torque compensatory control" (SW 6.4 and higher) | |
| DB3x. | DBX96.7 | "Status master/slave coupling" | |

## 7.2 NC machine data

### 7.2.1 Speed/torque coupling, master–slave (SW 6 and higher)

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| **Axis/spindle-specific($MA_ ...)** | | | |
| 37250 | MS_ASSIGN_MASTER_SPEED_CMD | Master axis for speed setpoint coupling | |
| 37252 | MS_ASSIGN_MASTER_TORQUE_CTR | Master axis for torque control | |
| 37254 | MS_TORQUE_CTRL_MODE | Connection of torque control output | |
| 37255 | MS_TORQUE_CTRL_ACTIVATION | Activating the torque compensatory control (SW 6.4 and higher) | |
| 37256 | MS_TORQUE_CTRL_P_GAIN | Gain factor of torque compensatory controller | |
| 37258 | MS_TORQUE_CTRL_I_TIME | I component of the torque control | |
| 37260 | MS_MAX_CTRL_VELO | Limitation of torque compensatory control | |
| 37262 | MS_COUPLING_ALWAYS_ACTIVE | Master/slave coupling active after power ON | |
| 37264 | MS_TENSION_TORQUE | Master–slave tension torque | |
| 37268 | MS_TORQUE_WEIGHT_SLAVE | Weighting of the torque value for the slave axis | |
| 37270 | MS_VELO_TOL_COARSE | Master slave velocity tolerance "coarse" (SW 6.4 and higher) | |
| 37272 | MS_VELO_TOL_FINE | Master slave velocity tolerance "fine" (SW 6.4 and higher) | |
| 37274 | MS_MOTION_DIR_REVERSE | Invert master–slave traversing direction (SW 6.4 and higher) | |

## 7.2.2 Speed/torque coupling (SW 5.x and lower)

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| **Axis/spindle-specific ($MA_ ...)** | | | |
| 34110 | REFP_CYC_NR | NC start without referencing the axis | R1 |
| 36620 | SERVO_DISABLE_DELAY_TIME | Cutout delay for control enable | A2 |
| 36610 | AX_ENERGY_STOP_TIME | Duration of braking slope | A3 |
| | | | |
| 63550 | MS_ASSIGN_MASTER_SPEED_CMD | Master axis for speed setpoint coupling | |
| 63555 | MS_ASSIGN_MASTER_TORQUE_CTRL | Master axis for torque control | |
| 63560 | MS_TORQUE_CTRL_P_GAIN | P gain of the torque control | |
| 63565 | MS_TORQUE_CTRL_I_TIME | I component of the torque control | |
| 63570 | MS_TORQUE_CTRL_MODE | Connection of torque control output | |
| 63575 | MS_TORQUE_WEIGHT_SLAVE | Weighting of the torque values | |
| 63580 | MS_TENSION_TORQUE | Tension torque | |
| 63585 | MS_TENSION_TORQ_FILTER_TIME | Time constant for PT1 filter tension torque | |
| 63590 | MS_COUPLING_ALWAYS_ACTIVE | Master/slave coupling active after power ON | |
| 63595 | MS_TRACE_MODE | Trace setting | |
| 63600 | MS_MS_MAX_CTRL_VELO | Control output limit | |

# 7.3 Alarms

See **References:** /DA/, Diagnostics Guide

# 7.4 System variables

| Identifier | Meaning | Ref. |
|---|---|---|
| $P_SEARCH_MASLC[slave axis identifier] | This variable registers a change in the coupling status during the block search SERUPRO in SW 6.2 and higher. | PGA |
| $P_SEARCH_MASLD[slave axis identifier] | This variable indicates the positional offset between the slave and master axes at the instant the coupling was closed in SW 6.2 and higher. | PGA |
| $AA_MASL_STAT[slave axis identifier] | This variable outputs the current coupling status in SW 6 and higher. | PGA/ FBSY |

■

# SINUMERIK 840D/840Di/810D Description of Functions Special Functions (Part 3)

# Transformation Package Handling (TE4)

■

# 1 Brief Description

**Functionality**

The handling transformation package has been designed for use on **manipulators** and **robots**. This package is a type a modular system which enables the customer to configure the transformation for his particular machine by setting machine data (provided that the relevant kinematics are included in the handling transformation package).

**Structure of Chapter 2**

Chapter 2 (Detailed Description) deals with the following topics:

- Section 2.1 describes the environment for kinematic transformation.
- Section 2.2. provides an explanation of basic terms.
- Section 2.3 explains the machine data required to configure transformations.
- Section 2.4 uses configuring examples to illustrate the most commonly used 2-axis to 5-axis kinematics that can be configured with the handling transformation package.
- Sections 2.5 to 2.9 deal with the subject of programming, describing in detail how to program orientations, specify tool parameters and call transformations.

**Abbreviations**

| | |
|---|---|
| FL | Flange coordinate system |
| HP | Wrist point coordinate system |
| IRO | Internal robot coordinate system |
| $p_3$, $q_3$, $r_3$ | Coordinates of last basic axis |
| RO | Robot or base center point coordinate system |
| WS | Workpiece coordinate system |
| WZ | Tool coordinate system |
| $x_3$, $y_3$, $z_3$ | Coordinates of first hand axis |

■

## Notes

# Detailed Description 2

## 2.1 Kinematic transformation

**Task of a transformation**

The purpose of a transformation is to transform movements in the tool tip, which are programmed in a Cartesian coordinate system, into machine axis positions.

**Field of application**

The handling transformation package described here has been designed to cover the largest possible number of kinematic transformations implemented solely via parameter settings in machine data. The current package offers kinematics which include between 2 and 5 axes in the transformation, corresponding to up to five spatial degrees of freedom. In this case, a maximum of 3 degrees are available for translation and 2 degrees for orientation, allowing, for example, a tool (cutter, laser beam) on a 5-axis machine to assume any orientation in relation to the workpiece at any point in the machining space. The workpiece is always programmed in the rectangular workpiece coordinate system; any programmed or set frames rotate and shift this system in relation to the basic system. The kinematic transformation then converts this information into motion instructions for the real machine axes. The kinematic transformation requires information about the design (kinematics) of the machine; these are stored in machine data.

**Categories of kinematics**

The handling transformation package is divided into two categories of kinematics, which can be selected via MD 62600: TRAFO6_KINCLASS.

- STANDARD: This category includes the most commonly used kinematics.
- SPECIAL: Special kinematics.

# 2.2 Definition of terms

## 2.2.1 Units and directions

**Lengths and angles**

In the transformation machine data, all lengths are specified in millimeters or inches and, unless otherwise stated, all angles in degrees at intervals of [ –180°, 180° ].

**Direction of rotation**

In the case of angles, arrows in the drawings always indicate the mathematically positive direction of rotation.

## 2.2.2 Definition of positions and orientations using frames

In order to make a clear distinction from the term "frame" as it is used in the NC language, the following description explains the meaning of the term "frame" in relation to the handling transformation package.

**Frame**

A frame can be used to translate one coordinate system into another. In this respect, a distinction must be made between translation and rotation. "Translation" only effects an offset between the coordinate system and the reference system, while "Rotation" actually rotates the coordinate system in relation to the reference.

**Translation**

Coordinates X, Y and Z are used to describe the translation. They are defined such that the coordinate system is a right-handed system.
The translation is always specified in relation to the coordinate directions of the initial system. These directions are assigned to machine data as follows:

- X direction: ..._POS[0]
- Y direction: ..._POS[1]
- Z direction: ..._POS[2]

**Rotation**

The rotation is described by the RPY angles A, B and C (RPY stands for Roll Pitch Yaw). The positive direction of rotation is defined by the “right hand rule”, i.e. if the thumb on the right hand is pointing in the direction of the axis of rotation, then the fingers are pointing in the positive angular direction. In this respect, it must be noted that A and C are defined at intervals [–180; +180] and B at intervals [–90; +90].

The definitions of the RPY angles are as follows:

– Angle A: 1. rotation about the Z axis of the initial system
– Angle B: 2. rotation about the rotated Y axis
– Angle C: 3rd rotation about the twice rotated X axis

The RPY angles are assigned to machine data as follows:

– Angle A: ..._RPY[0]
– Angle B: ..._RPY[1]
– Angle C: ..._RPY[2]

Fig. 2-1 shows an example of rotation about the RPY angles. In this example, the initial coordinate system X1, Y1, Z1 is first rotated through angle A about axis Z1, then through angle B about axis Y2 and finally through angle C about axis X3.

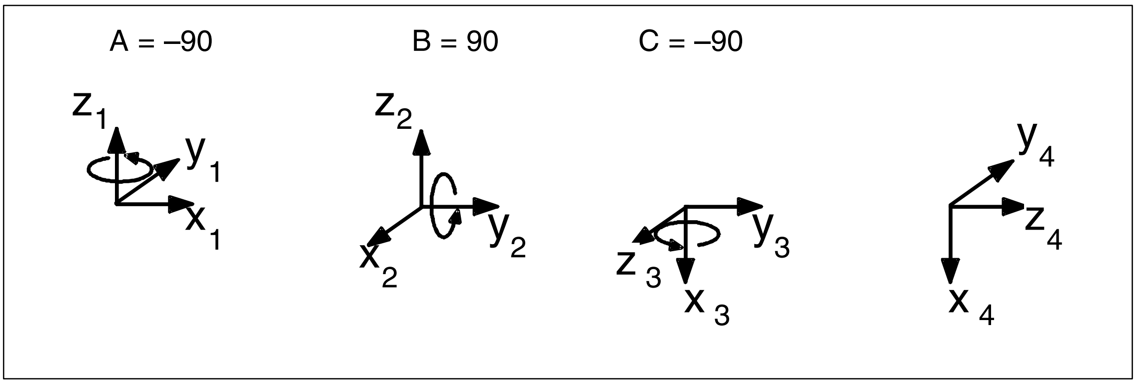



Fig. 2-1 Example of rotation through RPY angles

### 2.2.3 Definition of a joint

The term “joint” refers either to a translational axis or a rotary axis.
The basic axis identifiers are determined by the arrangement and sequence of individual joints. These are described by identifying letters (S, C, R, N) which are explained below.

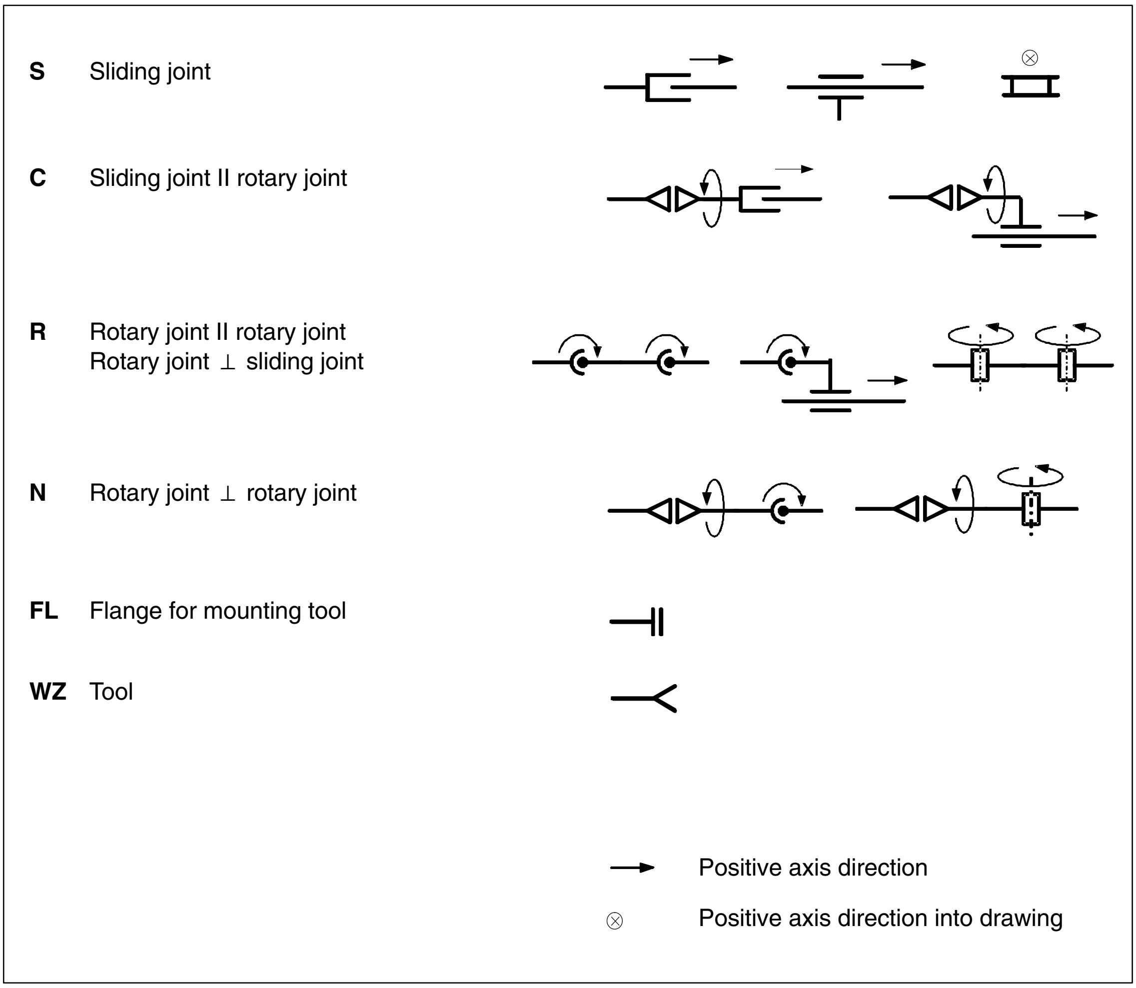



Fig. 2-2 Joint identifying letters

## 2.3 Configuration of a kinematic transformation

In order to ensure that the kinematic transformation can convert the programmed values to axis motions, it must have access to some information about the mechanical construction of the machine. This information is stored in machine data.

- Axis assignments
- Geometry information.

### 2.3.1 General machine data

**MD 24100**
**TRAFO_TYPE_1**

The value **4099** must be entered in this data for the handling transformation package.

**MD 24110**
**TRAFO_AXES_IN_1**

The axis assignment at the transformation input defines which transformation axis is mapped internally onto a channel axis. It is specified in MD 24110: TRAFO_AXES_IN_1.

There is a predetermined axis sequence for the handling transformation package, i.e. the first n channel axes must be assigned to the n transformation axes in ascending sequence:

- MD 24110: TRAFO_AXES_IN_1[0] = 1
- MD 24110: TRAFO_AXES_IN_1[1] = 2
- MD 24110: TRAFO_AXES_IN_1[2] = 3
- MD 24110: TRAFO_AXES_IN_1[3] = 4
- MD 24110: TRAFO_AXES_IN_1[4] = 5
- MD 24110: TRAFO_AXES_IN_1[5] = 6

**MD 24120**
**TRAFO_GEOAX_**
**ASSIGN_TAB_1**

The setting in MD 24120: TRAFO_GEOAX_ASSIGN_TAB_1 defines how many translational degrees of freedom are available for the transformation. The 3 geometry axes normally correspond to Cartesian axis directions X, Y and Z.

- MD 24120: TRAFO_GEO_AX_ASSIGN_TAB_1[0] = 1
- MD 24120: TRAFO_GEO_AX_ASSIGN_TAB_1[1] = 2
- MD 24120: TRAFO_GEO_AX_ASSIGN_TAB_1[2] = 3

## 2.3.2 Parameterization using geometry data

### Modular principle

The machine geometry is parameterized according to a type of modular principle. With this method, the machine is successively configured in geometry parameters from its base center point to the tool tip, thereby producing a closed kinematic loop. Frames (see Subsection 2.2.2) are used to describe the machine geometry. While the control is powering up, the configuration machine data are checked and alarms generated when necessary.
All axes in the mode group are made to follow, the alarms can only be reset by a power ON operation.

As shown in Fig. 2-3, the kinematic transformation effects a conversion of the tool operating point (tool coordinate system): $X_{WZ}$, $Y_{WZ}$, $Z_{WZ}$), that is specified in relation to the basic coordinate system (BCS = robot coordinate system: $X_{RO}$, $Y_{RO}$, $Z_{RO}$), into machine axis values (MCS positions: A1, A2, A3, ..). The operating point ($X_{WZ}$, $Y_{WZ}$, $Z_{WZ}$) is specified in the parts program in relation to the workpiece to be machined (workpiece coordinate system WCS: $X_{WS}$, $Y_{WS}$, $Z_{WS}$). The programmable frames make it possible to create an offset between the workpiece coordinate system WCS and the basic coordinate system BCS.

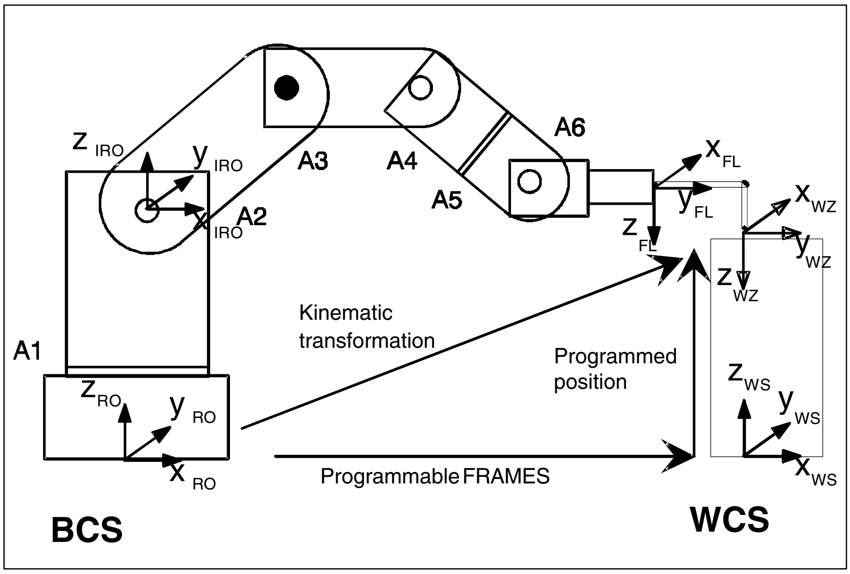



Fig. 2-3 Closed kinematic loop illustrated by the example of a robot

---

**Note**

For a more detailed explanation of coordinate systems, please refer to:
**References:** /PA/, Programming Guide Fundamentals

---

The following machine data are available for configuring kinematic transformations:

**MD 62612 TRAFO6_TIRORO_POS**
**MD 62613 TRAFO6_TIRORO_RPY**

The frame T_IRO_RO links the base center point of the machine (BCS = RO) with the first internal coordinate system (IRO) determined by the transformation.

**MD 62603 TRAFO6_MAIN_AXES**

The setting in MD 62603: TRAFO6_MAIN_AXES defines the type of basic axis arrangement. The "basic axes" are generally the first 3 axes to be included in the transformation.

**MD 62607 TRAFO6_MAIN_LENGTH_AB**

The setting in MD 62607: TRAFO6_MAIN_LENGTH_AB specifies the basic axis lengths A and B. As Fig. 2-4 illustrates, these are specially defined for each type of basic axis.

**MD 62606 TRAFO6_A4PAR**

The setting in MD 62606: TRAFO6_A4PAR specifies whether the 4th axis is mounted parallel, anti-parallel or perpendicular to the last rotary basic axis.

**MD 62608 TRAFO6_TX3P3_POS**
**MD 62609 TRAFO6_TX3P3_RPY**

Frame T_X3_P3 links the last coordinate system of the basic axes with the first hand coordinate system.

**MD 62604 ...**
**MD 62616 TRAFO6_DHPAR4_5..**

These parameters describe the hand geometry.

**MD 62604 TRAFO6_WRIST_AXES**

Machine data MD 62604: TRAFO6_WRIST_AXES specifies the hand type. The term "hand axes" generally refers to axes four to six.

**MD 62610 TRAFO6_TFLWP_POS**
**MD 62611 TRAFO6_TFLWP_RPY**

Frame T_FL_WP provides the link between the last hand coordinate system and the flange coordinate system.

These data are explained in more detail below.

## Basic axes included in every transformation

**MD 62603**
**TRAFO6_MAIN_AXES**

The first 3 axes which are included in the transformation are generally referred to as the “basic axes”. They must always be mutually parallel or perpendicular. Each of the following basic axis arrangements has its own special identifier (see Subsection 2.2.3). The basic axis identifier is entered in MD 62603: TRAFO6_MAIN_AXES.

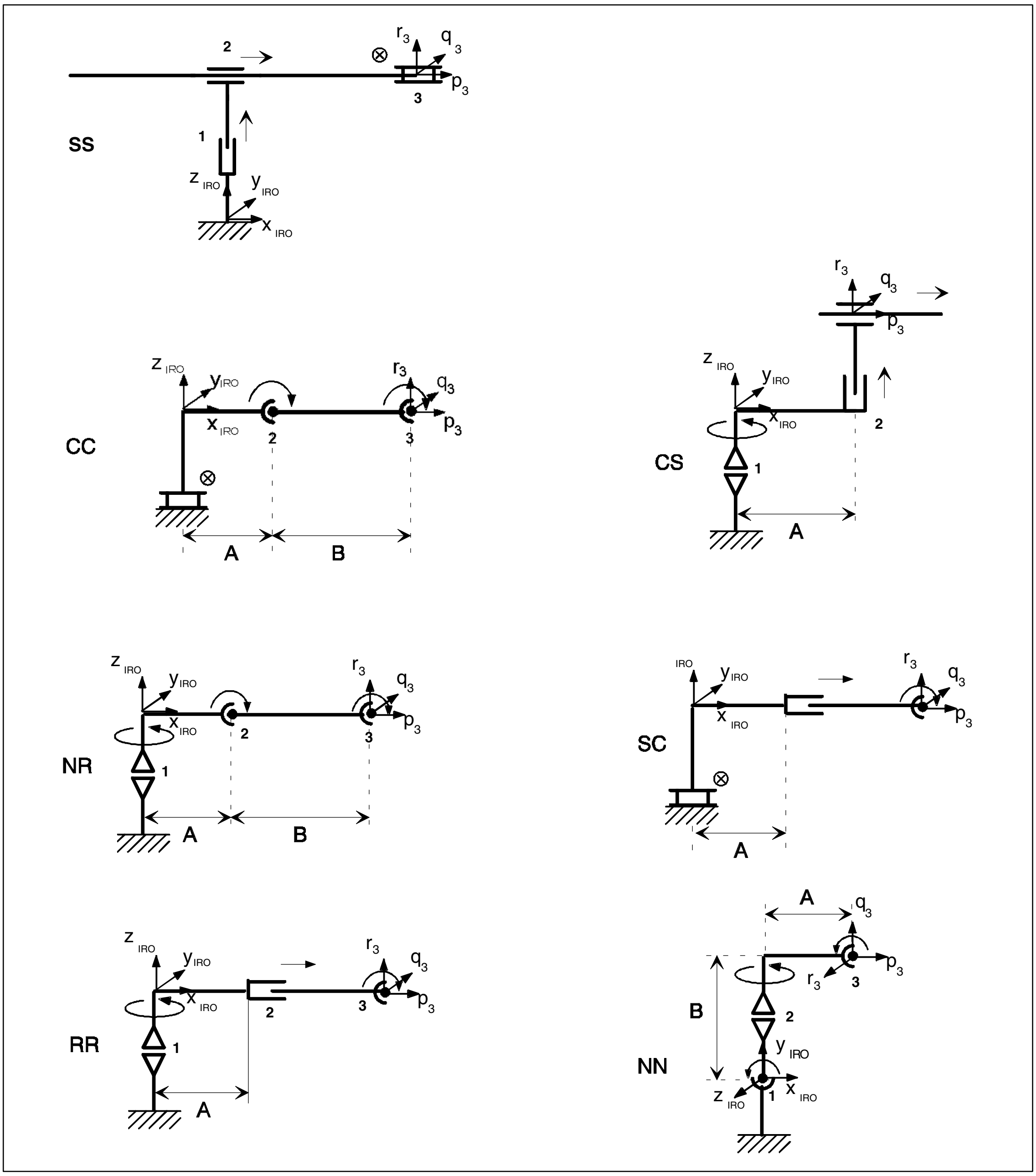



Fig. 2-4 Overview of basic axis configurations

The handling transformation package contains the following basic axis kinematics:

- SS: Gantry (3 linear axes, rectangular)
- CC: SCARA (1 linear axis, 2 rotary axes (in parallel))
- SC: SCARA (2 linear axes, 1 rotary axis (swivel axis))
- CS: SCARA (2 linear axes, 1 rotary axis) (axis of rotation)
- NR: Articulated arm (3 rotary axes (2 axes in parallel))
- NN: Articulated arm (3 rotary axes)
- RR: Articulated arm (1 linear axis, 2 rotary axes (perpendicular)).

## Hand axes included in every transformation

**MD 62604 TRAFO6_WRIST_ AXES**

The fourth axis and all further axes are generally referred to as “hand axes”. The handling transformation package can only identify hands with rotary axes. The hand axis identifier for three-axis hands is entered in MD 62603: TRAFO6_MAIN_AXES. In the case of hands with less than three axes, the identifier for a bevelled hand with elbow or a central hand is entered in MD 62603: TRAFO6_MAIN_AXES. The current software supports only hand axis types “bevelled hand with elbow” or “central hand”.

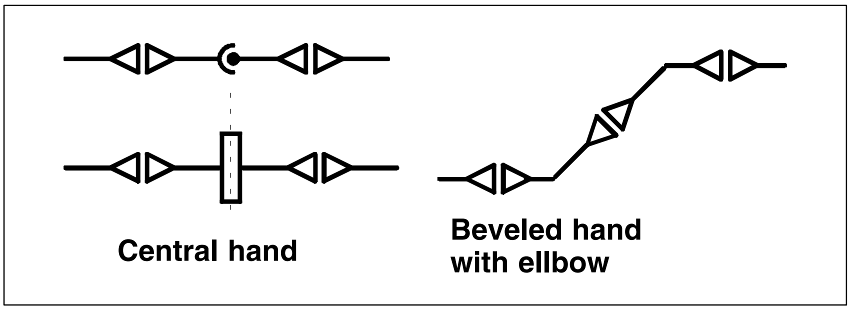



Fig. 2-5 Overview of hand axis configurations

## Parameterization of hand axes

**MD 62614 ... MD 62616 TRAFO6_DHPAR4_5..**

Hands are parameterized via machine data MD 62614: TRAFO6_DHPAR4_5A, MD 62615: TRAFO6_DHPAR4_5D and MD 62616: TRAFO6_DHPAR4_5ALPHA. These data are special types of frame which describe the relative positions of the coordinate systems in the hand. In this case, TRAFO6_DHPAR4_5A corresponds to .._POS[0] (x component), TRAFO6_DHPAR4_5D to .._POS[2] (z component) and TRAFO6_DHPAR4_5ALPHA to .._RPY[2] (C angle) of a frame (see Subsection 2.2.2). The other components of the frame are then zero.

**Central hand (CH)**

On a central hand, all hand axes intersect at one point. All parameters must be set as shown in Table 2-1.

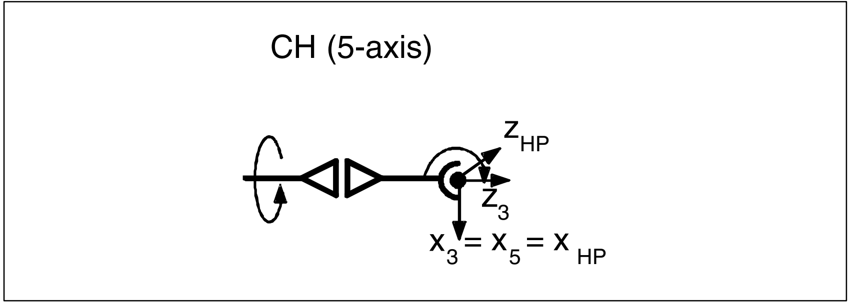



Fig. 2-6 Central hand

Table 2-1 Configuring data for a central hand

| Machine data | Value |
|---|---|
| MD 62604: TRAFO6_WRIST_AXES | 2 |
| MD 62614: TRAFO6_DHPAR4_5A | [0.0, 0.0] |
| MD 62615: TRAFO6_DHPAR4_5D | [0.0, 0.0] |
| MD 62616: TRAFO6_DHPAR4_5ALPHA | [-90.0, 90.0] |

**Beveled hand with elbow (BHE)**

The beveled hand with elbow differs from the central hand in two respects, i.e. the axes do not intersect nor are they mutually perpendicular. Parameters $a_4$, $d_5$, and $a_4$ are available for this type of hand, as shown in Table 2-1.

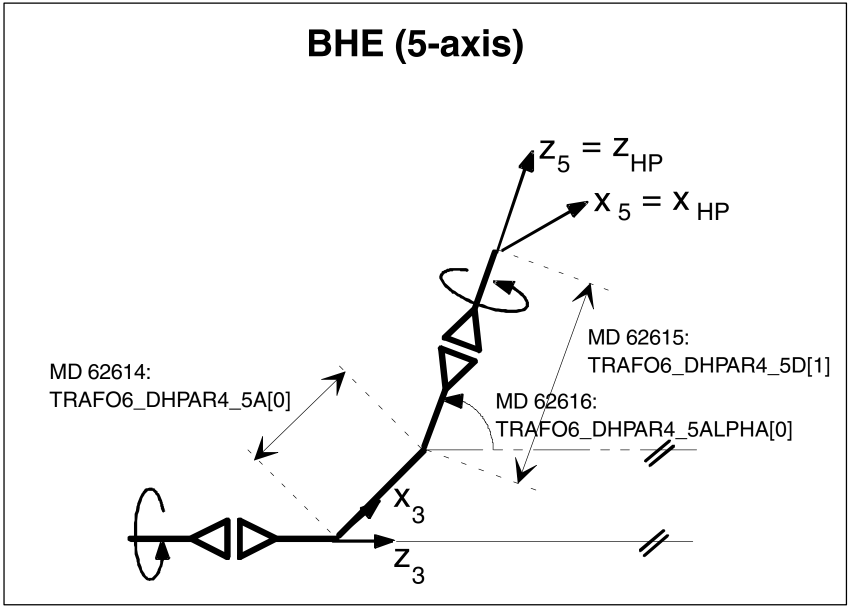



Fig. 2-7 Beveled hand with elbow (5-axis)

Table 2-2 Configuring data for a beveled hand with elbow (5-axis)

| Machine data | Value |
|---|---|
| MD 62604: TRAFO6_WRIST_AXES | 6 |
| MD 62614: TRAFO6_DHPAR4_5A | [$a_4$, 0.0] |
| MD 62615: TRAFO6_DHPAR4_5D | [0.0, $d_5$] |
| MD 62616: TRAFO6_DHPAR4_5ALPHA | [$\alpha_4$, 0.0] |

## Link frames

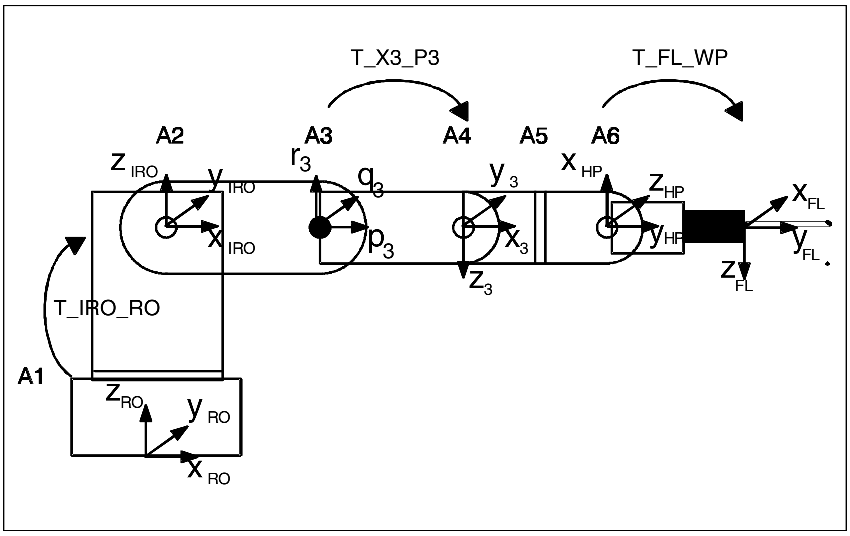



Fig. 2-8 Link frames

**T_IRO_RO**

Frame T_IRO_RO provides the link between the base center point coordinate system (RO) defined by the user and the internal robot coordinate system (IRO). The internal robot coordinate system is predefined in the handling transformation package for each basic axis type and included in the kinematic diagrams for the basic axis arrangements. The base center point system is in the Cartesian zero point of the machine, corresponding to the basic coordinate system. If no FRAMES are programmed, the basic coordinate system equals the workpiece coordinate system.

---

**Note**

For a more detailed explanation of FRAMES, please refer to:
**References:** /PA/, Programming Guide Fundamentals

---

Frame T_IRO_RO is not subject to any restrictions for 5-axis kinematics.

The following restrictions apply in relation to 4-axis kinematics:

- The first rotary axis must always be parallel/anti-parallel to one of the coordinate axes of the base center point coordinate system (RO).
- No further restrictions apply to type SS basic axes.
- In the case of type CC, CS or SC basic axes, no further restrictions apply provided that the 4th axis is parallel to the last rotary basic axis.
- With respect to all other basic axes, and basic axes of type CC, CS or SC if the 4th axis is perpendicular to the last rotary basic axis, the Z axis of RO must be parallel to the Z axis of IRO.

**T_X3_P3**

Frame T_X3_P3 describes the method used to attach the hand to the basic axes. Frame T_X3_P3 is used to link the coordinate system of the last basic axis (p3_q3_r3 coordinate system) with the coordinate system of the first hand axis (x3_y3_z3 coordinate system). The p3_q3_r3 coordinate system is shown in the kinematic diagrams for the basic axis arrangements.
The z3 axis is always positioned on the 4th axis.

Depending on the number of axes to be included in the transformation, frame T_X3_P3 is subject to certain restrictions relating to the hand and basic axes:

- For 5-axis kinematics, frame T_X3_P3 can be freely selected in the following cases:
  - If the basic axes are of the SS type.
  - If the basic axes are of the CC, CS or SC type, the transformation must either include a central hand (ZEH) or the 4th axis must be positioned in parallel to the last rotary basic axis.
  - If the basic axes are of the NR or RR type, the transformation must either include a central hand (ZEH) or the 4th axis must be positioned in parallel to the last basic rotary axis and an X flange must intersect the 5th axis.
  - If the basic axes are of the NN type, the transformation must include a central hand.
- With 4-axis kinematics, the z3 axis must always be parallel/anti-parallel or perpendicular to the last basic axis.

**T_FL_WP**

Frame T_FL_WP provides the link between the flange and the last internal coordinate system (wrist point coordinate system) predefined in the handling transformation package.
This frame is subject to certain restrictions for kinematics involving less than 6 axes. These restrictions are explained in the descriptions of the relevant kinematics.

## Other configuring data

**Number of transformed axes**

**MD 62605 TRAFO6_NUM_AXES**
Machine data MD 62605: TRAFO6_NUM_AXES is set to define how many axes must be included in the transformation. With the current software, the machine data can be set to between 2 and 5 transformed axes.

**Changing the axis sequence**

**MD 62620 TRAFO6_AXIS_SEQ**

!

**Important**

With certain types of kinematics, it is possible to transpose axes without changing the behavior of the kinematic transformation. MD 62620: TRAFO6_AXIS_SEQ is set to convert these kinematics into standard kinematics. The axes on the machine are numbered consecutively from 1 to 5 and must be entered in the internal sequence in MD 62620: TRAFO6_AXIS_SEQ[0] ...[4].
All other axis-specific machine data refer to the sequence of axes on the machine.

Table 2-3 Changing the axis sequence

| Basic axis kinematics | Options for changing axis sequence |
|---|---|
| SS, CC | Any |
| SC | 1 and 2 |
| CS | 2 and 3 |

Example 1

This example involves two kinematics such as those illustrated in Fig. 2-9. Kinematic 1 is directly included in the handling transformation package. It corresponds to a CC kinematic where a hand axis is positioned in parallel to the last basic rotary axis.
Kinematic 2 is equivalent to kinematic 1 since, as regards the final motion of the robot, it is irrelevant whether the translational axis is axis 1 or axis 4. In this instance, the data for kinematic 2 must be entered as follows in MD 62620: TRAFO6_AXIS_SEQ:

MD 62620: TRAFO6_AXIS_SEQ[ 0 ] = 4
MD 62620: TRAFO6_AXIS_SEQ[ 1 ] = 1
MD 62620: TRAFO6_AXIS_SEQ[ 2 ] = 2
MD 62620: TRAFO6_AXIS_SEQ[ 3 ] = 3

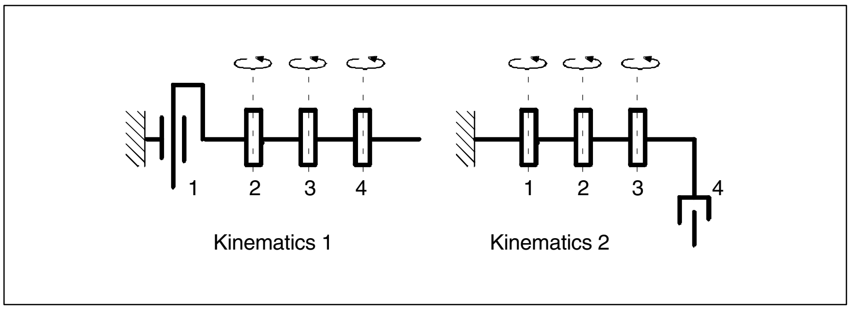



Fig. 2-9 Rearrangement of axes (example 1)

Example 2

This example involves a SCARA kinematic transformation as illustrated in Fig. 2-10, in which the axes can be freely transposed. Kinematic 1 is directly included in the handling transformation package. It corresponds to a CC kinematic. As regards the transposition of axes, it is irrelevant how many hand axes are involved in the transformation.

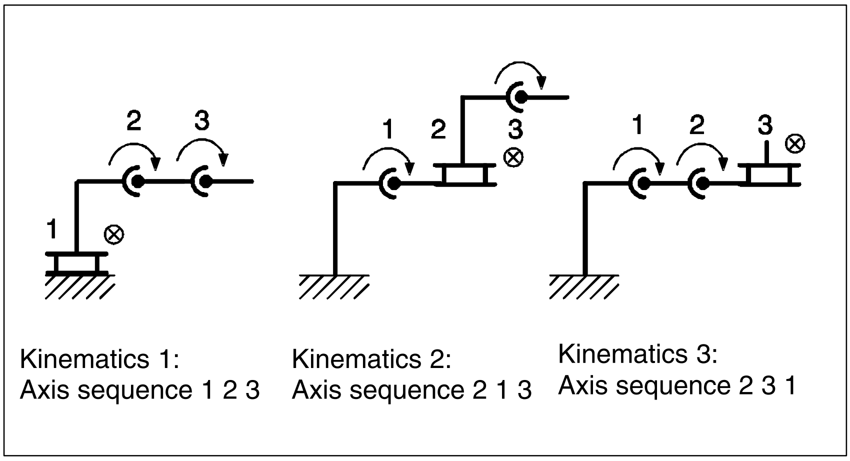



Fig. 2-10 Rearrangement of axes (example 2)

**Changing the directions of axes**

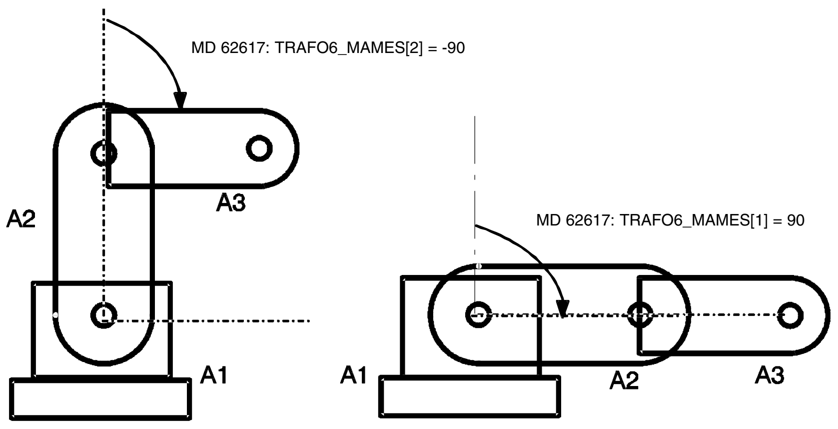
**MD 62618 TRAFO6_AXES_DIR**
A rotational or offset direction is preset for each axis in the handling transformation package. This direction is not necessarily the same as the corresponding direction on the machine. In order to match the directions, MD 62618: TRAFO6_AXES_DIR[ ] must be set to **–1** for the relevant axis if the direction is to be reversed, or otherwise to **+1**.

## Matching the zero points of axes

**MD 62617 TRAFO6_MAMES**
The mathematical zero points of axes are preset in the handling transformation package. However, the mathematical zero point does not always correspond to the mechanical zero point (calibration point) of axes. In order to match the mathematical and mechanical zero points, the difference between the mathematical value and the mechanical calibration point must be entered in MD 62617: TRAFO6_MAMES[ ] for each axis. The deviation to be entered corresponds to the difference between the mechanical zero point and the mathematically positive direction of rotation of the axis.

Example

The example (Fig. 2-11) shows an articulated arm kinematic. Axis 2 has a mathematical zero point of 90º. This value must be entered in MD 62617: TRAFO6_MAMES[1] for axis 2. Axis 3 is counted relative to axis 2 and therefore has a value of –90º as a mathematical zero point.




a) Mechanical zero position (adjustment position)    b) Mathematical zero position

Fig. 2-11 Matching mathematical and mechanical zero points

## Axis types

**MD 62601 TRAFO6_AXES_TYPE**
The relevant axis type is defined in MD 62601: TRAFO6_AXES_TYPE. The transformation package distinguishes between the following axis types:

- Linear axis
- Rotary axis

## Velocities and acceleration rates

For the traversal of axes with G00 and an active transformation, special velocities for the Cartesian motion components have been introduced.
If the axis is traversing with G01 or G02, the path velocity is determined by path feedrate F.

**MD 62629**
**TRAFO6_VELCP**

Machine data 62629: TRAFO6_VELCP[i] can be set to define the velocities for individual translational motion directions for axis traversal with G00.
Index i = 0 : X component of basic system
Index i = 1 : Y component of basic system
Index i = 2 : Z component of basic system

**MD 62630**
**TRAFO6_ACCCP**

Machine data MD 62630: TRAFO6_ACCCP[i] can be set to define the acceleration rates for individual translational motion directions for axis traversal with G00.
Index i = 0 : X component of basic system
Index i = 1 : Y component of basic system
Index i = 2 : Z component of basic system

**MD 62631**
**TRAFO6_VELORI**

Machine data MD 62631: TRAFO6_VELORI[i] can be set to define the velocities for individual directions of orientation for axis traversal with G00.
Index i = 0 : Angle A
Index i = 1 : Angle B
Index i = 2 : Angle C

**MD 62632**
**TRAFO6_ACCORI**

Machine data MD 62632: TRAFO6_ACCORI[i] can be set to define the acceleration rates for individual directions of orientation for axis traversal with G00.
Index i = 0 : Angle A
Index i = 1 : Angle B
Index i = 2 : Angle C

# 2.4 Descriptions of kinematics

The following descriptions of kinematics for transformations involving 2 to 5 axes explain the general configuring procedure first. They then describe how the machine data need to be configured, using a configuring example for each kinematic type. These examples do not include all possible lengths and offsets. The direction data refer to the positive directions of traversal and rotation for the transformation. The axis positions correspond to their zero position for the relevant transformation.

## 2.4.1 3-axis kinematics

3-axis kinematics normally possess 3 translational degrees of freedom, but no degree of freedom for orientation. In other words, they only include basic axes.

**Configuring**

The procedure for configuring a 3-axis kinematic is as follows:

1. Enter “Standard” kinematic category in MD 62600: TRAFO6_KINCLASS.
2. Set number of axes for transformation in MD 62605: TRAFO6_NUM_AXES = 3.
3. Compare basic axes with basic axes contained in the handling transformation package. –> Enter basic axis identifier in MD 62603: TRAFO6_MAIN_AXES.
4. If the axis sequence is not the same as the normal axis sequence, it must be corrected in MD 62620: TRAFO6_AXIS_SEQ.
5. MD 62604: TRAFO6_WRIST_AXES = 1 must be set as the hand axis identifier (i.e. no hand in this case).
6. Enter the axis types for the transformation in MD 62601: TRAFO6_AXES_TYPE.
7. Compare directions of rotation of axes with directions defined in the handling transformation package and correct in MD 62618: TRAFO6_AXES_DIR.
8. Enter mechanical zero offset in MD 62617: TRAFO6_MAMES.
9. Enter basic axis lengths in MD 62607: TRAFO6_MAIN_LENGTH_AB.
10. Define frame T_IRO_RO and enter the offset in MD 62612: TRAFO6_TIRORO_POS and the rotation in MD 62613: TRAFO6_TIRORO_RPY.
11. Determine the flange coordinate system. For this purpose, the p3_q3_r3 coordinate system must be regarded as the initial system. The offset must be entered in MD 62610: TRAFO6_TFLWP_POS and the rotation in MD 62611: TRAFO6_TFLWP_RPY.

## SCARA kinematics

SCARA kinematics are characterized by the fact that they possess both translational and rotary axes. The basic axes are divided into 3 categories depending on how they are mutually positioned.

- CC types
- CS types
- SC types (cf. Fig. 2-4).

**3-axis CC kinematic**

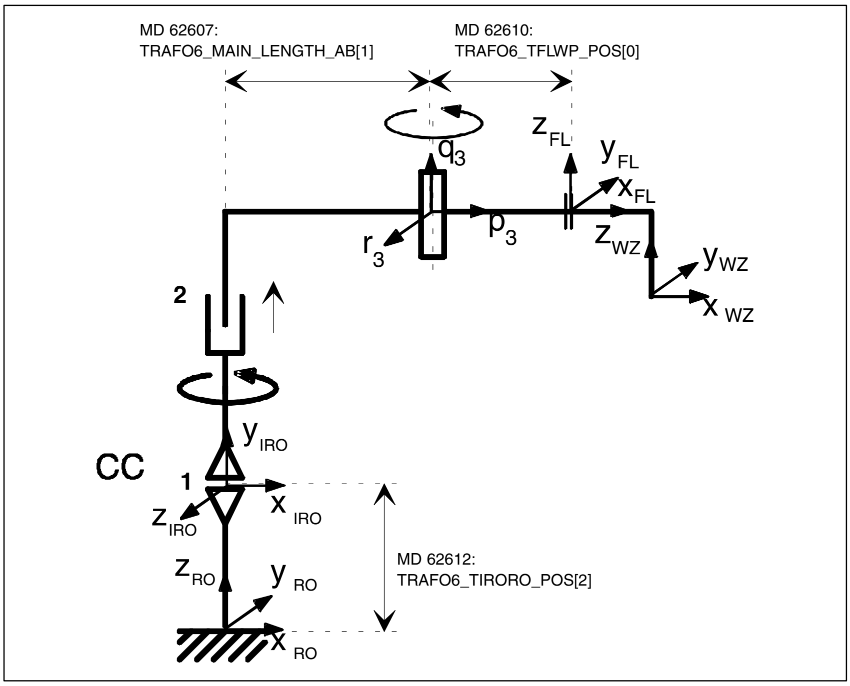



Fig. 2-12 3-axis CC kinematic

Table 2-4 Configuring data for a 3-axis CC kinematic

| Machine data | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 1 |
| MD 62605: TRAFO6_NUM_AXES | 3 |
| MD 62603: TRAFO6_MAIN_AXES | 2 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [3, 1, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [2, 1, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [0.0, 300.0] |
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 500.0] |

Table 2-4 Configuring data for a 3-axis CC kinematic

| Machine data | Value |
|---|---|
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 90.0] |
| MD 62608: TRAFO6_TX3P3_POS | [0.0, 0.0, 0.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 0.0, 0.0] |
| MD 62610: TRAFO6_TFLWP_POS | [200.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, 0.0, -90.0] |

**3-axis SC kinematic**

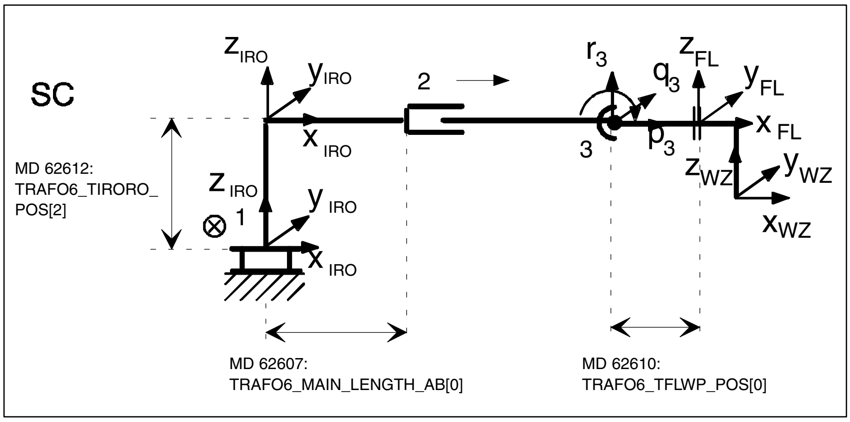



Fig. 2-13 3-axis SC kinematic

Table 2-5 Configuring data for a 3-axis SC kinematic

| Machine data | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 1 |
| MD 62605: TRAFO6_NUM_AXES | 3 |
| MD 62603: TRAFO6_MAIN_AXES | 4 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [1, 1, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [1, 2, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [500.0, 0.0] |
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 500.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 0.0] |
| MD 62608: TRAFO6_TX3P3_POS | [0.0, 0.0, 0.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 0.0, 0.0] |
| MD 62610: TRAFO6_TFLWP_POS | [300.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, 0.0, 0.0] |

**3-axis CS kinematic**

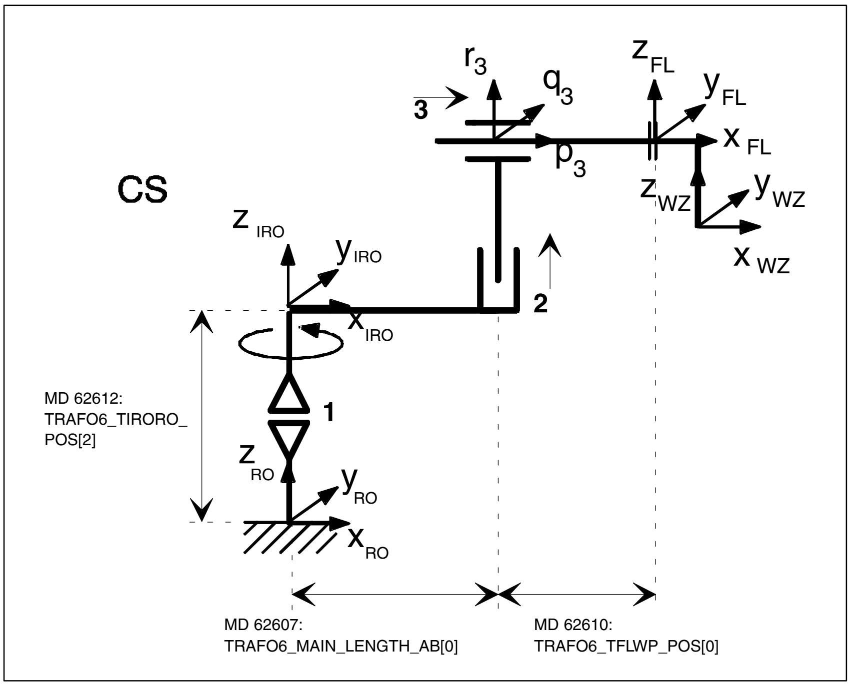



Fig. 2-14 3-axis CS kinematic

Table 2-6 Configuration data 3-axis CS kinematic

| Machine data | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 1 |
| MD 62605: TRAFO6_NUM_AXES | 3 |
| MD 62603: TRAFO6_MAIN_AXES | 6 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [3, 1, 1, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [1, 2, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [500.0, 0.0] |
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 500.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 0.0] |
| MD 62608: TRAFO6_TX3P3_POS | [0.0, 0.0, 0.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 0.0, 0.0] |
| MD 62610: TRAFO6_TFLWP_POS | [300.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, 0.0, 0.0] |

## Articulated-arm kinematics

**3-axis NR kinematic**

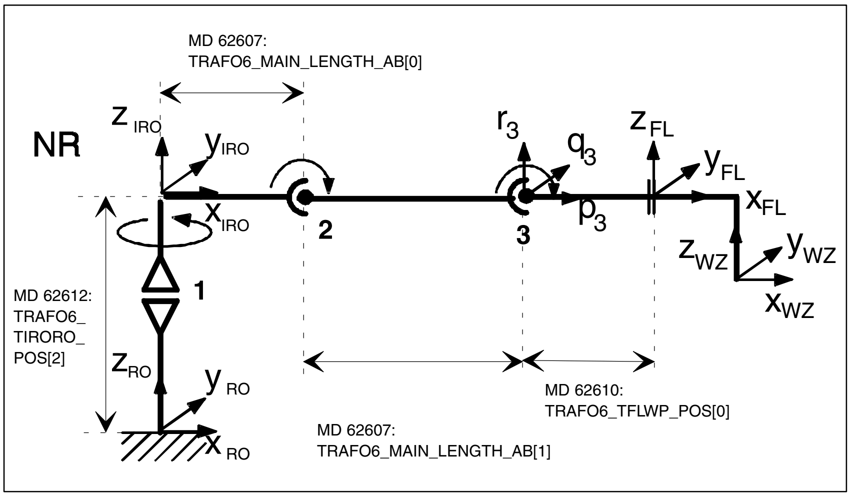



Fig. 2-15 3-axis NR kinematic

Table 2-7 Configuration data 3-axis NR kinematic

| Machine data | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 1 |
| MD 62605: TRAFO6_NUM_AXES | 3 |
| MD 62603: TRAFO6_MAIN_AXES | 3 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [3, 3, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [1, 2, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [300.0, 500.0] |
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 500.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 0.0] |
| MD 62608: TRAFO6_TX3P3_POS | [0.0, 0.0, 0.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 0.0, 0.0] |
| MD 62610: TRAFO6_TFLWP_POS | [300.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, 0.0, 0.0] |

**3-axis
RR kinematic**

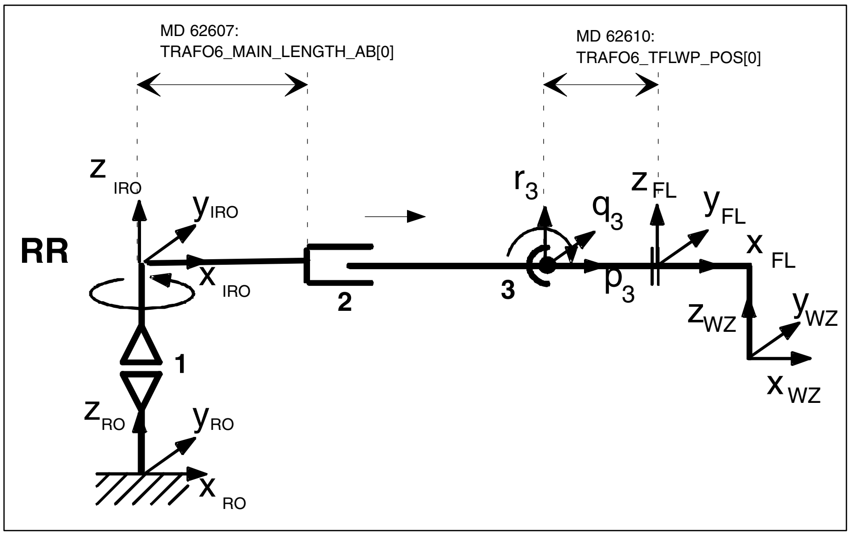



Fig. 2-16 3-axis RR kinematic

Table 2-8 Configuration data 3-axis RR kinematic

| Machine data | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 1 |
| MD 62605: TRAFO6_NUM_AXES | 3 |
| MD 62603: TRAFO6_MAIN_AXES | 5 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [3, 1, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [1, 2, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [0.0, 300.0] |
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 300.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 0.0] |
| MD 62608: TRAFO6_TX3P3_POS | [0.0, 0.0, 0.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 0.0, 0.0] |
| MD 62610: TRAFO6_TFLWP_POS | [200.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, 0.0, 0.0] |

**3-axis NN kinematic**

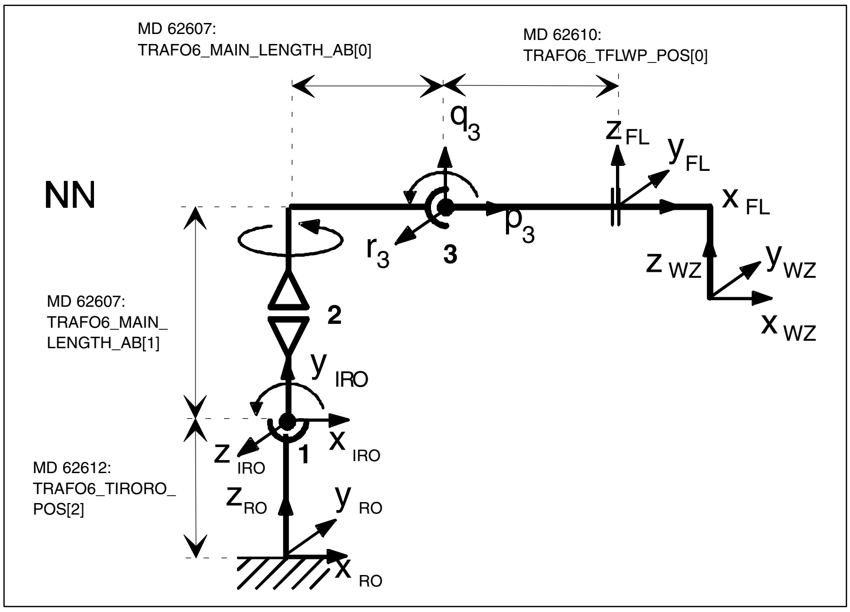



Fig. 2-17 3-axis NN kinematic

Table 2-9 Configuration data 3-axis NN kinematic

| Machine data | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 1 |
| MD 62605: TRAFO6_NUM_AXES | 3 |
| MD 62603: TRAFO6_MAIN_AXES | 7 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [3, 3, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [1, 2, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [300.0, 500.0] |
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 300.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 90.0] |
| MD 62608: TRAFO6_TX3P3_POS | [0.0, 0.0, 0.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 0.0, 0.0] |
| MD 62610: TRAFO6_TFLWP_POS | [400.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, 0.0, -90.0] |

## 2.4.2 4-axis kinematics

4-axis kinematics usually imply 3 translational degrees of freedom and one degree of freedom for orientation.

**Restrictions**

The following restrictions apply for 4-axis kinematics:

The frame T_FL_WP is subject to the following condition:

- MD 62611: TRAFO6_TFLWP_RPY = [ 0.0, 90.0, 0.0 ].
- X flange and X tool must be parallel to the 4th axis.
- Two successive basic axes must be parallel or orthogonal.
- The 4th axis must only be mounted in a parallel or orthogonal way to the last basic axis.

**Configuring**

The procedure for configuring a 4-axis kinematic is as follows:

1. Enter “Standard” kinematic category in MD 62600: TRAFO6_KINCLASS.
2. Set number of axes for transformation in MD 62605: TRAFO6_NUM_AXES = 4.
3. Compare basic axes with basic axes contained in the handling transformation package.
–> Enter basic axis identifier in MD 62603: TRAFO6_MAIN_AXES.
4. If the axis sequence is not the same as the normal axis sequence, it must be corrected in MD 62620: TRAFO6_AXIS_SEQ.
5. MD 62604: TRAFO6_WRIST_AXES = 1 must be set as the hand axis identifier (i.e. no hand in this case).
6. Enter in MD 62606: TRAFO6_A4PAR whether axis 4 parallel/anti-parallel to the last rotary basic axis.
7. Enter the axis types for the transformation in MD 62601: TRAFO6_AXES_TYPE.
8. Compare directions of rotation of axes with directions defined in the handling transformation package and correct in MD 62618: TRAFO6_AXES_DIR.
9. Enter mechanical zero offset in MD 62617: TRAFO6_MAMES.
10. Enter basic axis lengths in MD 62607: TRAFO6_MAIN_LENGTH_AB.
11. Define frame T_IRO_RO and enter the offset in MD 62612: TRAFO6_TIRORO_POS and the rotation in MD 62613: TRAFO6_TIRORO_RPY.
12. Specification of frame T_X3_P3 to attach hand. For this purpose, the p3_q3_r3 coordinate system must be regarded as the initial system. The offset must be entered in MD 62608: TRAFO6_TX3P3_POS and the rotation in MD 62609: TRAFO6_TX3P3_RPY.

13. Determine the flange coordinate system. For this purpose, the hand-point coordinate system must be regarded as the initial system. The offset must be entered in MD 62610: TRAFO6_TFLWP_POS and the rotation in MD 62611: TRAFO6_TFLWP_RPY.

## SCARA kinematics

**4-axis CC kinematic**

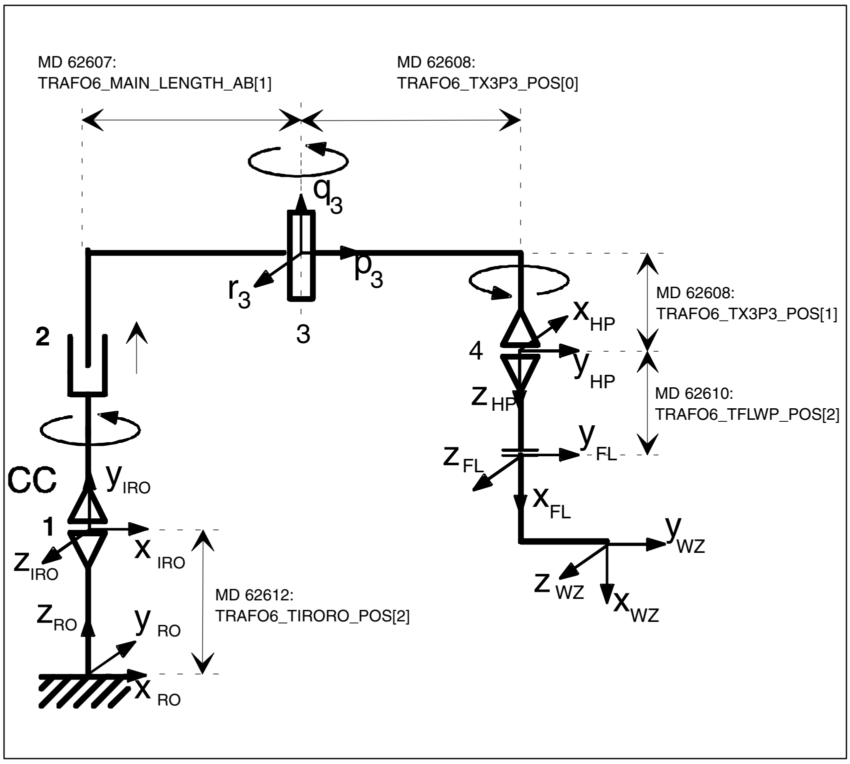



Fig. 2-18 4-axis CC kinematic

Table 2-10 Configuring data for a 4-axis CC kinematic

| Machine date | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 1 |
| MD 62605: TRAFO6_NUM_AXES | 4 |
| MD 62603: TRAFO6_MAIN_AXES | 2 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62606: TRAFO6_A4PAR | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [3, 3, 1, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [2, 1, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [0.0, 300.0] |

Table 2-10 Configuring data for a 4-axis CC kinematic

| Machine date | Value |
|---|---|
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 500.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 90.0] |
| MD 62608: TRAFO6_TX3P3_POS | [300.0, 0.0, -200.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [-90.0, 90.0, 0.0] |
| MD 62610: TRAFO6_TFLWP_POS | [200.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, -90.0, 0.0] |

**4-axis SC kinematic**

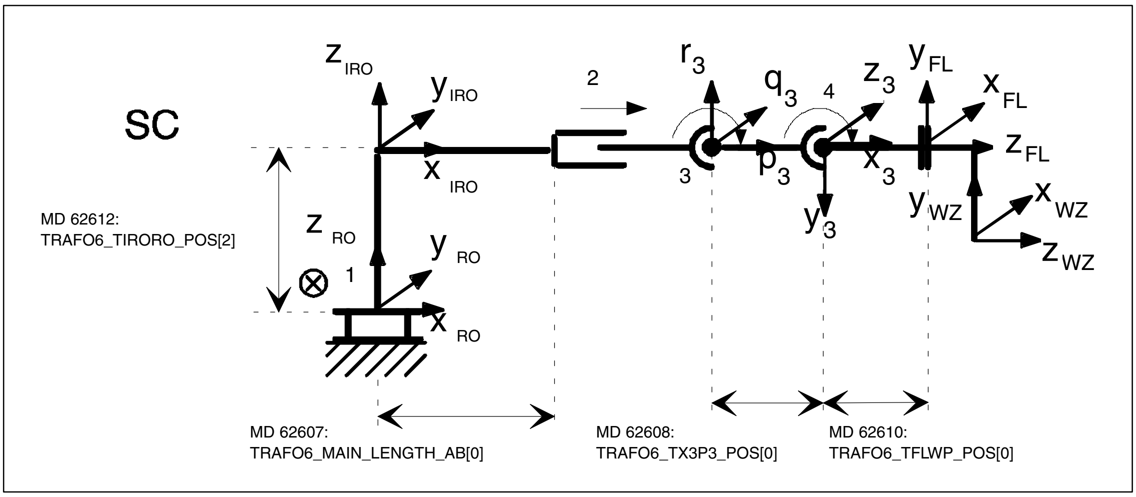



Fig. 2-19 4-axis SC kinematic

Table 2-11 Configuring data for a 4-axis SC kinematic

| Machine date | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 1 |
| MD 62605: TRAFO6_NUM_AXES | 4 |
| MD 62603: TRAFO6_MAIN_AXES | 4 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62606: TRAFO6_A4PAR | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [1, 3, 1, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [1, 2, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [0.0, 300.0] |
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 300.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 0.0] |
| MD 62608: TRAFO6_TX3P3_POS | [200.0, 0.0, 0.0] |

Table 2-11 Configuring data for a 4-axis SC kinematic

| Machine date | Value |
|---|---|
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 0.0, -90.0] |
| MD 62610: TRAFO6_TFLWP_POS | [200.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, -90.0, 180.0] |

**4-axis CS kinematic**

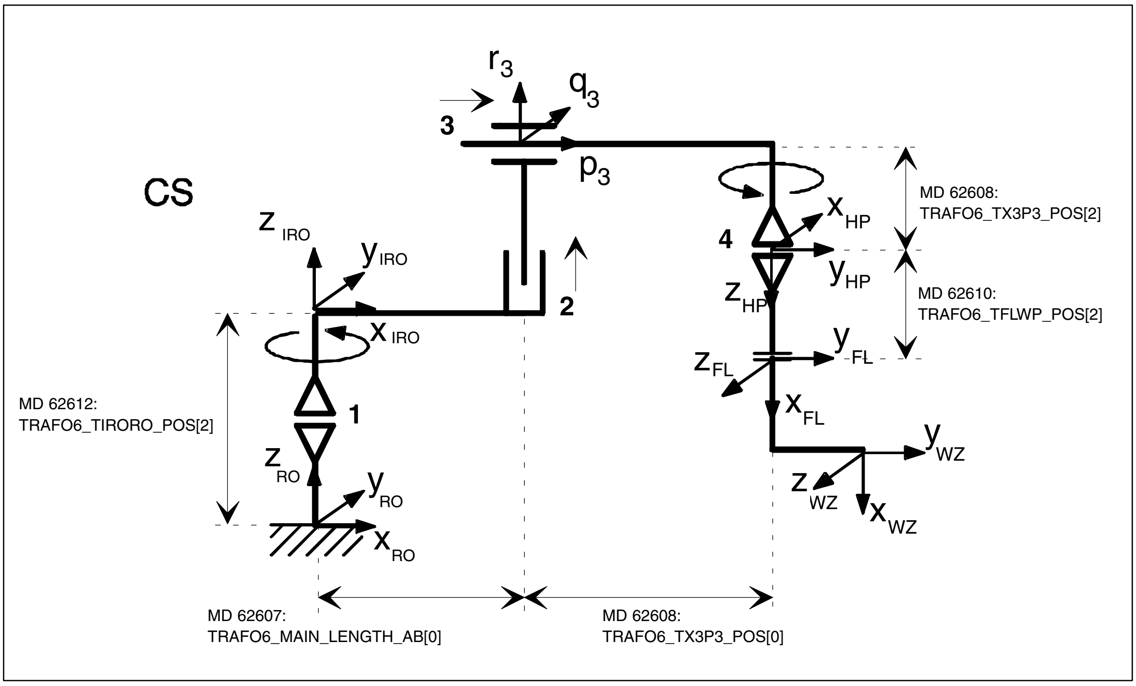



Fig. 2-20 4-axis CS kinematic

Table 2-12 Configuration data 4-axis CS kinematic

| Machine date | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 1 |
| MD 62605: TRAFO6_NUM_AXES | 4 |
| MD 62603: TRAFO6_MAIN_AXES | 6 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62606: TRAFO6_A4PAR | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [3, 1, 1, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [1, 2, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [400.0, 0.0] |
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 400.0] |

Table 2-12 Configuration data 4-axis CS kinematic

| Machine date | Value |
|---|---|
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 0.0] |
| MD 62608: TRAFO6_TX3P3_POS | [500.0, 0.0, -200.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [90.0, 0.0, 180.0] |
| MD 62610: TRAFO6_TFLWP_POS | [200.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, -90.0, 0.0] |

## Articulated-arm kinematics

### 4-axis NR kinematic

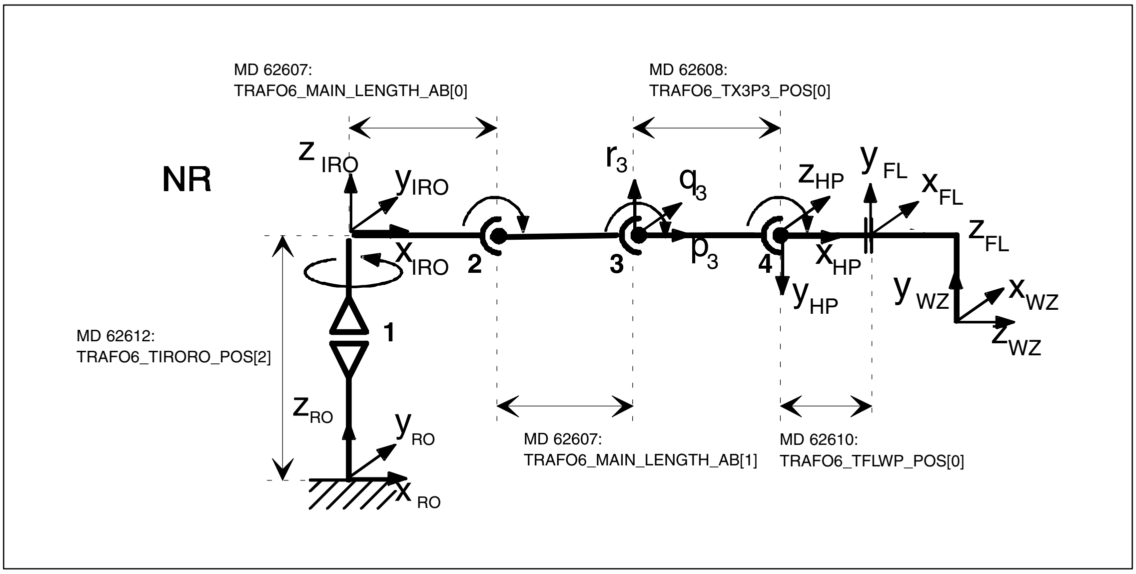



Fig. 2-21 4-axis NR kinematic

Table 2-13 Configuration data 4-axis NR kinematic

| Machine date | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 1 |
| MD 62605: TRAFO6_NUM_AXES | 4 |
| MD 62603: TRAFO6_MAIN_AXES | 3 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62606: TRAFO6_A4PAR | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [3, 3, 3, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [1, 2, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |

Table 2-13 Configuration data 4-axis NR kinematic

| Machine date | Value |
|---|---|
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [300.0, 300.0] |
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 500.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 0.0] |
| MD 62608: TRAFO6_TX3P3_POS | [300.0, 0.0, 0.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 0.0, -90.0] |
| MD 62610: TRAFO6_TFLWP_POS | [200.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, -90.0, 180.0] |

### 2.4.3 5-axis kinematics

5-axis kinematics usually imply 3 translational degrees of freedom and 2 degrees of freedom for orientation.

**Restrictions**

The following restrictions apply to 5-axis kinematics:

1. There are restrictions for the flange coordinate system because the X flange axis must intersect the 5th axis, nevertheless, it must not be parallel to it.
2. The frame T_FL_WP is subject to the following condition as far as 5-axis articulated-arm kinematics are concerned:
   – MD 62610: TRAFO6_TFLWP_POS = [0.0, 0.0, Z]
   – MD 62611: TRAFO6_TFLWP_RPY = [A, 0.0, 0.0]
3. There are restrictions for the tool as far as 5-axis articulated-arm kinematics are concerned:
   – 4th axis parallel to the 3rd axis: 2-dimensional tool is possible [X, 0.0, Z]
   – 4th axis perpendicular to the 3rd axis: only 1-dimensional tool is possible [X, 0.0, 0.0]
4. There are restrictions for the tool as far as 5-axis Scara kinematics are concerned:
   – 4th axis perpendicular to the 3rd axis: 1-dimensional tool is possible [X, 0.0, 0.0]
5. Two successive basic axes must be parallel or orthogonal.
6. The 4th axis must only be mounted in a parallel or orthogonal way to the last basic axis.

**Configuring**

The procedure for configuring a 5-axis kinematic is as follows:

1. Enter “Standard” kinematic category in MD 62600: TRAFO6_KINCLASS.
2. Set number of axes for transformation in MD 62605: TRAFO6_NUM_AXES = 5.
3. Compare basic axes with basic axes contained in the handling transformation package.
   –> Enter basic axis identifier in MD 62603: TRAFO6_MAIN_AXES.
4. If the axis sequence is not the same as the normal axis sequence, it must be corrected in MD 62620: TRAFO6_AXIS_SEQ.
5. ID specification for the hand axes. If axis 4 and 5 intersect, a central hand (ZEH) is present. In all other case, the ID for bevelled hand with elbow (WSH) must be entered in MD 62604: TRAFO6_WRIST_AXES.
6. Enter in MD 62606: TRAFO6_A4PAR whether axis 4 is parallel/anti-parallel to the last rotary basic axis.
7. Enter the axis types for the transformation in MD 62601: TRAFO6_AXES_TYPE.

8. Compare directions of rotation of axes with directions defined in the handling transformation package and correct in MD 62618: TRAFO6_AXES_DIR.
9. Enter mechanical zero offset in MD 62617: TRAFO6_MAMES.
10. Enter basic axis lengths in MD 62607: TRAFO6_MAIN_LENGTH_AB.
11. Define frame T_IRO_RO and enter the offset in MD 62612: TRAFO6_TIRORO_POS and the rotation in MD 62613: TRAFO6_TIRORO_RPY.
12. Specification of frame T_X3_P3 to attach hand. The offset must be entered in MD 62608: TRAFO6_TX3P3_POS and the rotation in MD 62609: TRAFO6_TX3P3_RPY.
13. Specification of hand axes parameters. For this purpose, only the parameters for axis 4 must be entered in MD 62614: TRAFO6_DHPAR4_5A[0] and MD 62616: TRAFO6_DHPAR4_5ALPHA[0]. all other parameters must be set to 0.0.
14. Determine the flange coordinate system. For this purpose, the hand-point coordinate system must be regard as the initial system. The offset must be entered in MD 62610: TRAFO6_TFLWP_POS and the rotation in MD 62611: TRAFO6_TFLWP_RPY.

## SCARA kinematics

**5-axis CC kinematic**

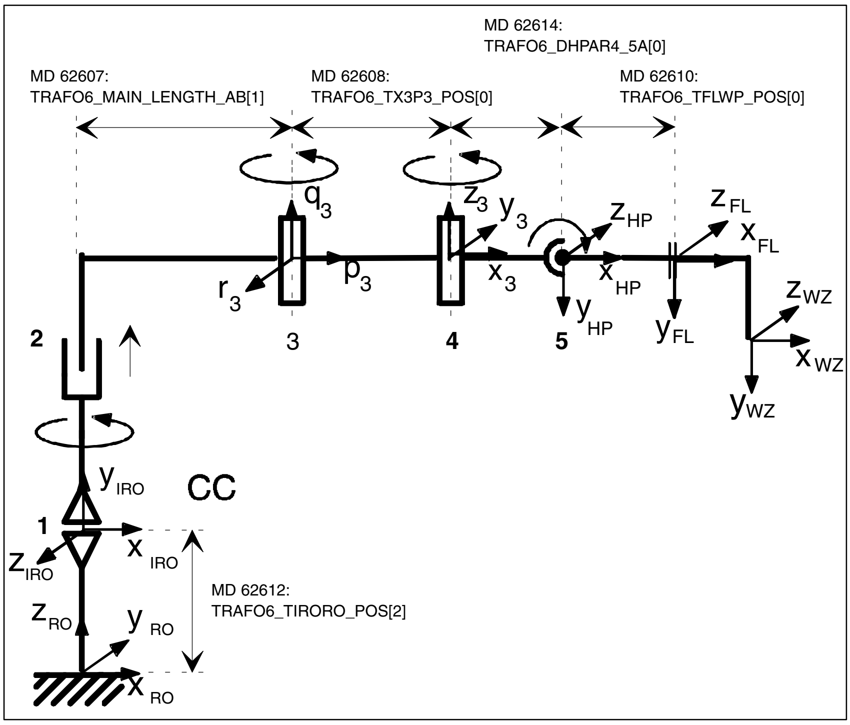



Fig. 2-22 5-axis CC kinematic

Table 2-14 Configuring data for a 5-axis CC kinematic

| Machine data | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 1 |
| MD 62605: TRAFO6_NUM_AXES | 5 |
| MD 62603: TRAFO6_MAIN_AXES | 2 |
| MD 62604: TRAFO6_WRIST_AXES | 5 |
| MD 62606: TRAFO6_A4PAR | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [3, 1, 3, 3, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [2, 1, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [500.0, 0.0] |
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 500.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 90.0] |
| MD 62608: TRAFO6_TX3P3_POS | [300.0, 0.0, -200.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 0.0, -90.0] |
| MD 62610: TRAFO6_TFLWP_POS | [200.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, 0.0, 0.0] |
| MD 62614: TRAFO6_DHPAR4_5A | [200.0, 0.0] |
| MD 62615: TRAFO6_DHPAR4_5D | [0.0, 0.0] |
| MD 62616: TRAFO6_DHPAR4_5ALPHA | [-90.0, 0.0] |

**5-axis**
**NR kinematic**

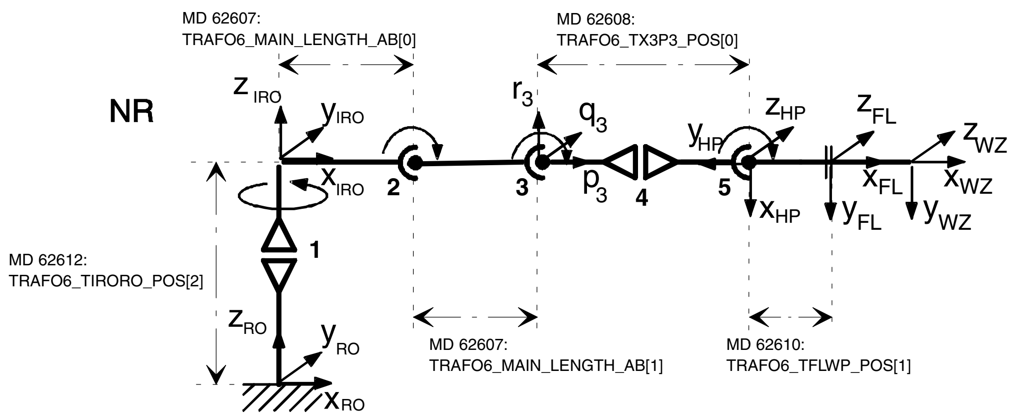



Fig. 2-23 5-axis NR kinematic

Table 2-15 Configuration data 5-axis NR kinematic

| Machine data | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 1 |
| MD 62605: TRAFO6_NUM_AXES | 5 |
| MD 62603: TRAFO6_MAIN_AXES | 3 |
| MD 62604: TRAFO6_WRIST_AXES | 2 |
| MD 62606: TRAFO6_A4PAR | 0 |
| MD 62601: TRAFO6_AXES_TYPE | [3, 3, 3, 3, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [1, 2, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [30.0, 300.0] |
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 500.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 0.0] |
| MD 62608: TRAFO6_TX3P3_POS | [0.0, 0.0, 500.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 90.0, 0.0] |
| MD 62610: TRAFO6_TFLWP_POS | [0.0, -300.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [-90.0, 0.0, 0.0] |
| MD 62614: TRAFO6_DHPAR4_5A | [0.0, 0.0] |
| MD 62615: TRAFO6_DHPAR4_5D | [0.0, 0.0] |
| MD 62616: TRAFO6_DHPAR4_5ALPHA | [-90.0, 0.0] |

### 2.4.4 6-axis kinematics

For SW 4.3, 6-axis kinematics have not yet been available.

### 2.4.5 Special kinematics

**MD 62602 TRAFO6_SPECIAL_KIN**

Special kinematics are kinematics that are not directly included in the building block system of the Handling transformation package. They are frequently missing a degree of freedom or are characterized by mechanical links between the axes or with the tool. MD 62600: TRAFO6_KINCLASS = 2 has to be set for these kinematics. MD 62602: TRAFO6_SPECIAL_KIN specifies the type of the special kinematics.

**2-axis SC special kinematic**

This special kinematic is characterized by the fact that the tool is always maintained in the same orientation via a mechanical linkage. It implies two Cartesian degrees of protection. The identifier for this kinematic is MD 62602: TRAFO6_SPECIAL_KIN = 3.

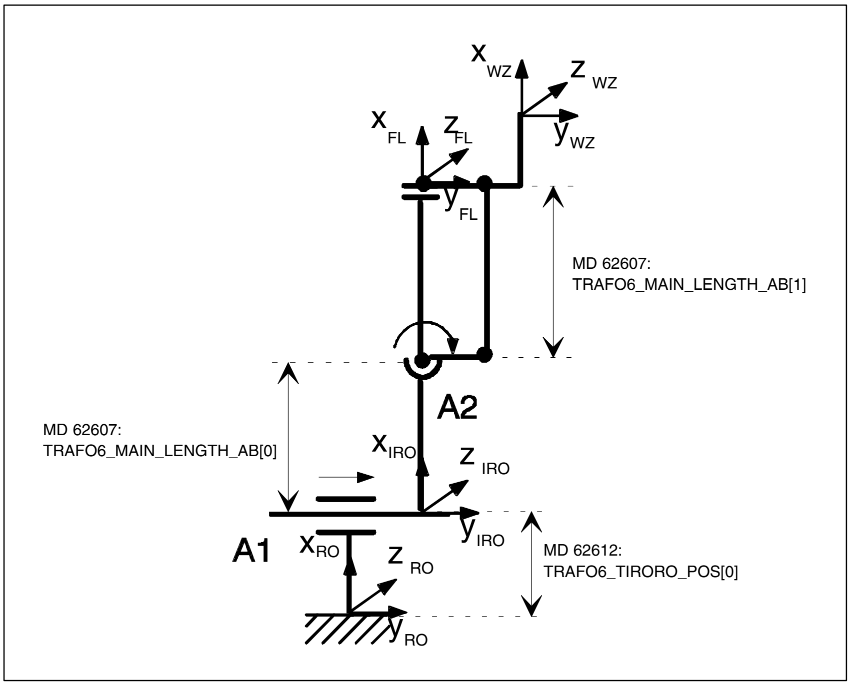



Fig. 2-24 Special 2-axis SC kinematic

Table 2-16 Configuring data for a special 2-axis SC kinematic

| Machine data | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 2 |
| MD 62602: TRAFO6_SPECIAL_KIN | 3 |
| MD 62605: TRAFO6_NUM_AXES | 2 |
| MD 62603: TRAFO6_MAIN_AXES | 2 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [1, 3, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [1, 2, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [400.0, 500.0] |
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 300.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 0.0] |
| MD 62608: TRAFO6_TX3P3_POS | [0.0, 0.0, 0.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 0.0, 0.0] |
| MD 62610: TRAFO6_TFLWP_POS | [0.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, 0.0, 0.0] |

**Special 3-axis SC kinematic**

The special kinematic has 2 Cartesian degrees of freedom and one degree of freedom for orientation. The identifier for this kinematic is MD 62602: TRAFO6_SPECIAL_KIN = 4.

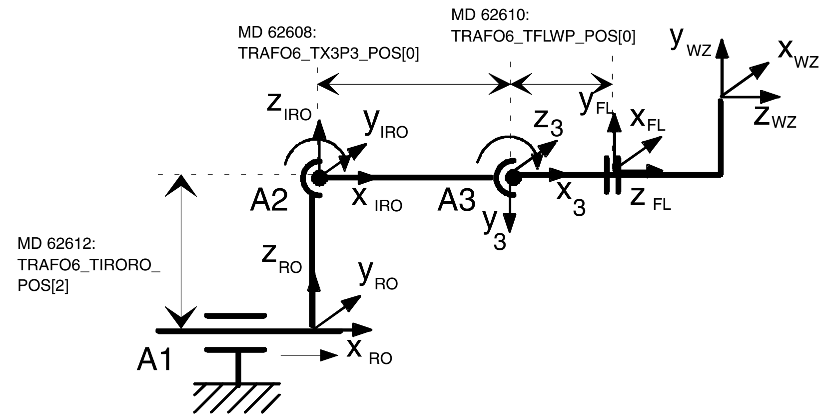



Fig. 2-25 Special 3-axis SC kinematic

Table 2-17 Configuring data for a special 3-axis SC kinematic

| Machine data | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 2 |
| MD 62602: TRAFO6_SPECIAL_KIN | 4 |
| MD 62605: TRAFO6_NUM_AXES | 3 |
| MD 62603: TRAFO6_MAIN_AXES | 2 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [1, 3, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [1, 2, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [0.0, 0.0] |
| MD 62612: TRAFO6_TIRORO_POS | [0.0, 0.0, 400.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 0.0] |
| MD 62608: TRAFO6_TX3P3_POS | [400.0, 0.0, 0.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 0.0, -90.0] |
| MD 62610: TRAFO6_TFLWP_POS | [200.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, -90.0, 180.0] |

**Special 4-axis SC kinematic**

This special kinematic is characterized by the fact that axis 1 and axis 2 are mechanically coupled. This coupling ensures that axis 2 is maintained at a constant angle when axis 1 is swiveled. This kinematic also guarantees that axes 3 and 4 always remain perpendicular, irrespective of the positions of axes 1 and 2. The identifier for this kinematic is MD 62602: TRAFO6_SPECIAL_KIN = 7.

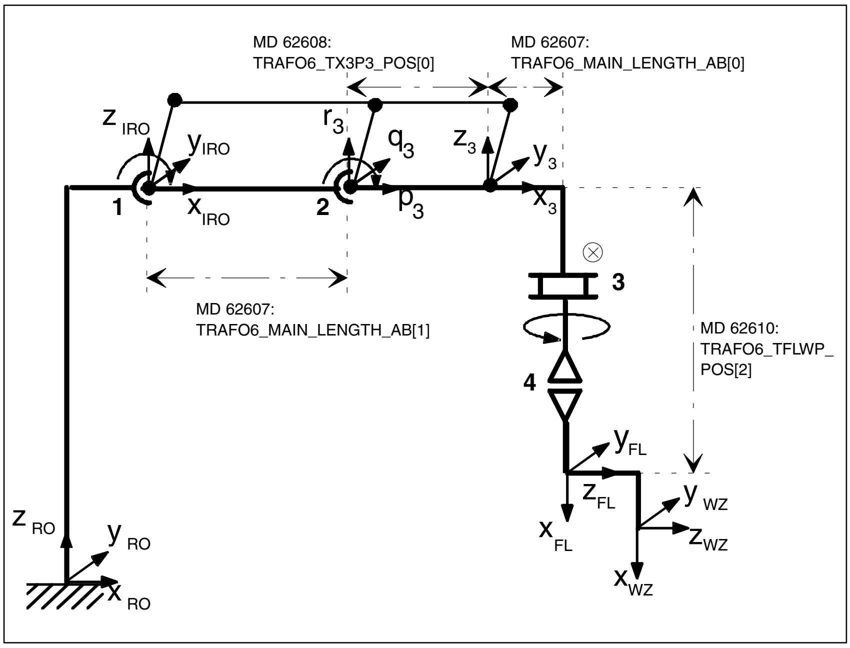



Fig. 2-26 Special 4-axis SC kinematic

Table 2-18 Configuring data for a special 4-axis SC kinematic

| Machine date | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 2 |
| MD 62602: TRAFO6_SPECIAL_KIN | 7 |
| MD 62605: TRAFO6_NUM_AXES | 4 |
| MD 62603: TRAFO6_MAIN_AXES | 2 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [3, 3, 1, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [1, 2, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [100.0, 400.0] |
| MD 62612: TRAFO6_TIRORO_POS | [100.0, 0.0, 1000.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, 0.0] |
| MD 62608: TRAFO6_TX3P3_POS | [300.0, 0.0, 0.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 0.0, 0.0] |
| MD 62610: TRAFO6_TFLWP_POS | [0.0, 0.0, -600.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, 90.0, 0.0] |

**Special 2-axis NR kinematic**

This special kinematic is characterized by the fact that axis 1 and axis 2 are mechanically coupled. Another special feature is the tool. With this kinematic, it maintains its orientation in space irrespective of the positions of the other axes. The identifier for this kinematic is MD 62602: TRAFO6_SPECIAL_KIN = 5.

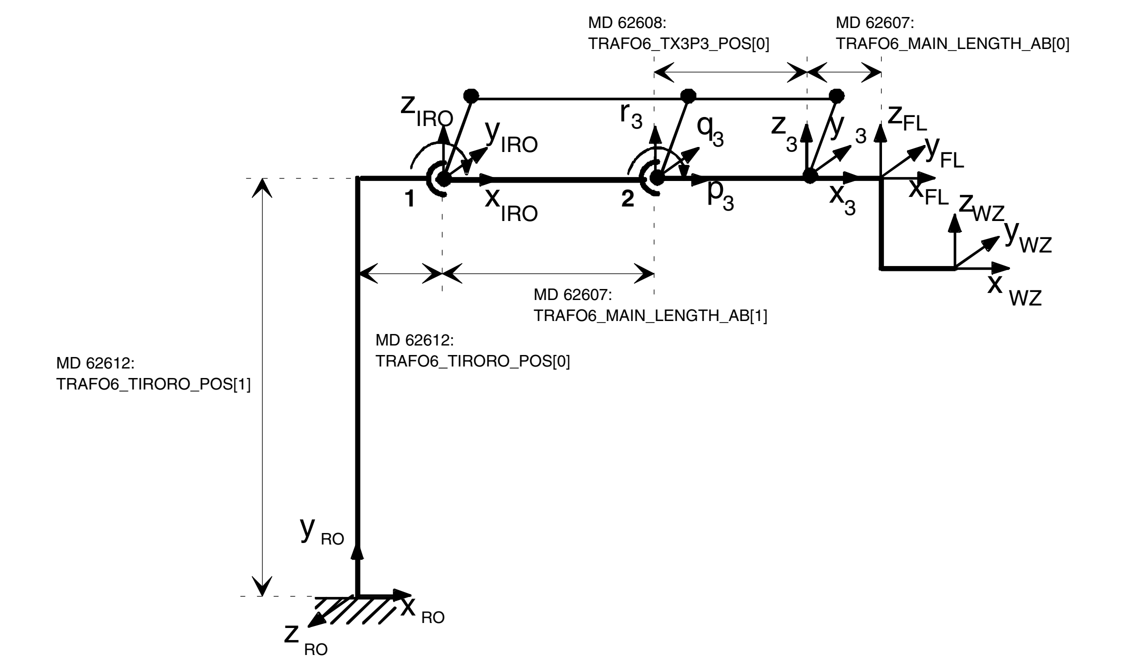



Fig. 2-27 Special 2-axis NR kinematic

Table 2-19 Configuring data for a special 2-axis NR kinematic

| Machine date | Value |
|---|---|
| MD 62600: TRAFO6_KINCLASS | 2 |
| MD 62602: TRAFO6_SPECIAL_KIN | 5 |
| MD 62605: TRAFO6_NUM_AXES | 2 |
| MD 62603: TRAFO6_MAIN_AXES | 3 |
| MD 62604: TRAFO6_WRIST_AXES | 1 |
| MD 62601: TRAFO6_AXES_TYPE | [3, 3, ...] |
| MD 62620: TRAFO6_AXIS_SEQ | [1, 2, 3, 4, 5, 6] |
| MD 62618: TRAFO6_AXES_DIR | [1, 1, 1, 1, 1, 1] |
| MD 62617: TRAFO6_MAMES | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |
| MD 62607: TRAFO6_MAIN_LENGTH_AB | [100.0, 400.0] |
| MD 62612: TRAFO6_TIRORO_POS | [100.0, 500.0, 0.0] |
| MD 62613: TRAFO6_TIRORO_RPY | [0.0, 0.0, -90.0] |
| MD 62608: TRAFO6_TX3P3_POS | [400.0, 0.0, 0.0] |
| MD 62609: TRAFO6_TX3P3_RPY | [0.0, 0.0, 0.0] |
| MD 62610: TRAFO6_TFLWP_POS | [0.0, 0.0, 0.0] |
| MD 62611: TRAFO6_TFLWP_RPY | [0.0, 0.0, 0.0] |

## 2.5 Tool orientation

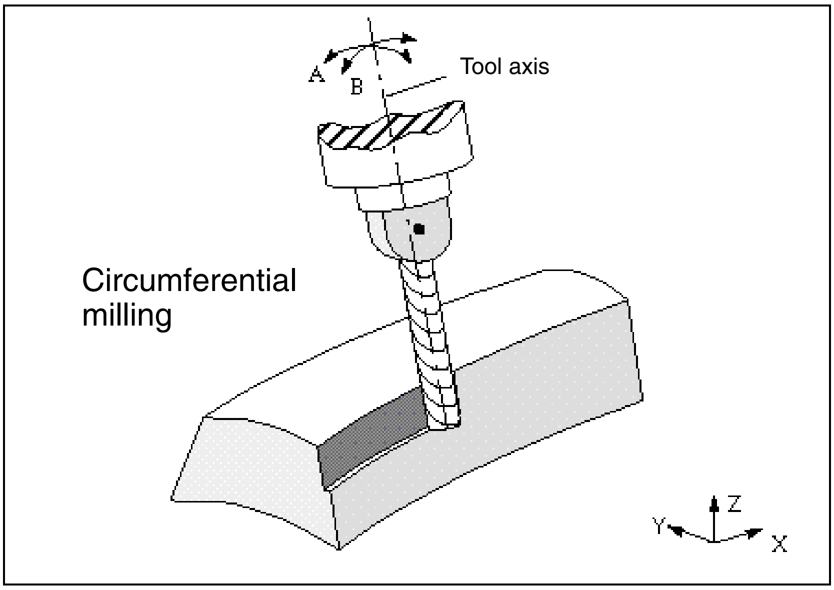



Fig. 2-28 Machining of workpieces with 5-axis transformation

**Programming**

Three possible methods can be used to program the orientation of the tool, i.e.

- directly as "orientation axes" A, B and C in degrees
- by means of Euler or RPY angles in degrees using A2, B2, C2
- using directional vectors A3, B3, C3

The designators for Euler angles or directional vectors can be set in machine data:

Euler angles in MD 10620: EULER_ANGLE_NAME_TAB
direction vector in MD10640: DIR_VECTOR_NAME_TAB

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

**Euler or RPY**

It is possible to switch between Euler and RPY by setting MD 21100: ORIENTATION_IS_EULER.

---

! **Important**

It is not possible to program using Euler angles, RPY angles or directional vectors for kinematics involving less than 5 axes. In such cases, only one degree of freedom is available for orientation. This orientation angle can only be programmed with “Orientation axis angle” “A”.

---

**Orientation reference**

A tool orientation at the block beginning can be transferred to an orientation at the block end only by means of the ORIWKS command in the workpiece coordinate system.

**ORIWKS command**

The tool orientation is programmed in the workpiece coordinate system (WCS) and is therefore not dependent on the machine kinematics. In the case of a change in orientation with the tool tip at a fixed point in space, the tool moves along a large arc on the plane stretching from the start vector to the end vector.

**ORIMKS command**

The tool orientation is programmed in the machine coordinate system and is thus dependent on the machine kinematics. In the case of a change in orientation with the tool tip at a fixed point in space, linear interpolation takes place between the rotary axis positions.

---

! **Important**

Transferring an orientation using ORIMKS is not allowed in the handling transformation package. With an active transformation, it is not the machine axis angles that are programmed and traversed, but “orientation angles” (RPY angles according to robotics definition, see Subsection 2.2.2).

---

The orientation is selected via NC language commands ORIWKS and ORIMKS.

The initial setting is ORIMKS (SW 2 and higher). The initial setting can be altered in MD 20150: GCODE_RESET_VALUES
GCODE_RESET_VALUES [24] = 1 ⇒ ORIWKS is initial setting
GCODE_RESET_VALUES [24] = 2 ⇒ ORIMKS is initial setting
GCODE_RESET_VALUES [24] = 3 ⇒ ORIPATH

When ORIPATH is active, the orientation is calculated from the lead and side angles relative to the path tangent and surface normal vector.

**Improper tool orientation**

If the tool orientation is programmed in conjunction with functions

- G04 Dwell time
- G33 Thread cutting with constant lead
- G74 Approach reference point
- G75 Approach fixed point
- REPOSL Repositioning
- REPOSQ Repositioning
- REPOSH Repositioning

then alarm 12130 "Illegal tool orientation" is output when Euler angles and directional vectors are selected. The NC program then stops (this alarm can also occur in connection with G331, G332 and G63). Alarm 17630 or 17620 is output for G74 and G75 if a transformation is active and the axes are involved in the transformation. This applies irrespective of orientation programming.

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

Alarm 14400 is output if the transformation is switched on or off when a tool offset is active.
In the reverse situation, i.e. a tool offset is selected or deselected when a transformation is active, no alarm message is output.

**Multiple input of tool orientation**

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

N50 A3=1 B3=0 C3=0

If the tool orientation is input several times, e.g. with directional vectors and Euler angles:

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

then error message 12240 "Channel X block Y tool orientation xx defined more than once" is displayed and the NC parts program stops.

### 2.5.1 Programming orientation for 4-axis kinematics

**Tool orientation with 4-axis kinematics**

4-axis kinematics possess only one degree of freedom for orientation. When the orientation is programmed using RPY angles, Euler angles or directional vectors, it is not generally possible to guarantee that the specified orientation can be approached. If used at all, this type of orientation programming is only suitable for certain types of kinematic, i.e. those which feature an invariance in orientation angles relative to the basic axes. This is true, for example, in the case of SCARA kinematics.
For this reason, the only method of orientation programming permitted for kinematics with 4 axes is the method that uses the “orientation angle” **A**. This angle corresponds to the RPY angle C according to the robotics definition, i.e. a rotation about the Z-RO axis, as illustrated in Fig. 2-29.

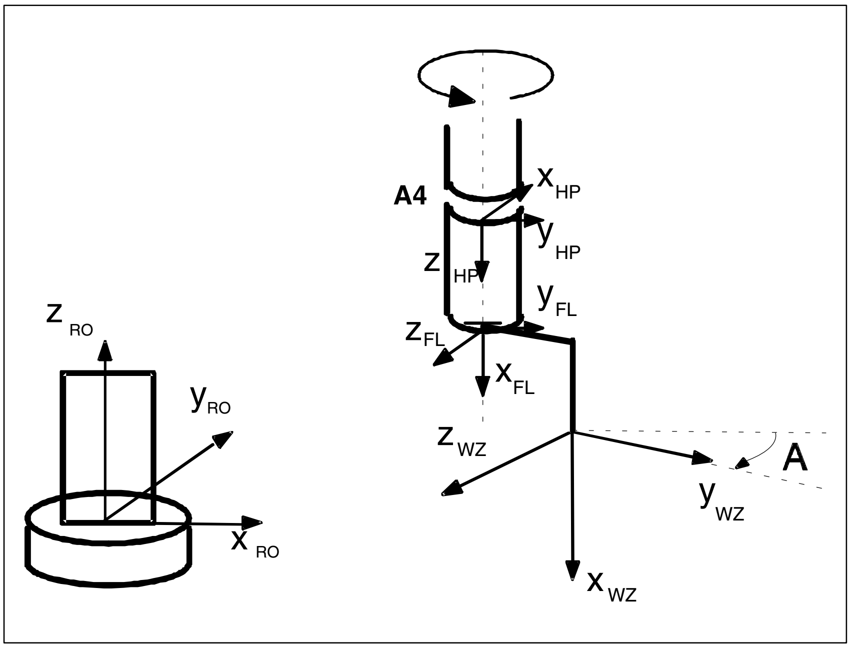



Fig. 2-29 Orientation angle for 4-axis kinematic

## 2.5.2 Programming orientation for 5-axis kinematics

**Tool orientation for 5-axis kinematic**

For 5-axis kinematics, when programming via orientation vector, it is assumed that the orientation vector corresponds to the x component of the tool.

When programming via orientation angle (RPY angle according to robotics definition), the x component of the tool is considered as the initial point for rotations.

For this purpose, the vector in the x tool direction, as shown in Fig. 2–30, is first rotated around the Z axis by the angle A and then around the rotated Y axis by the angle B. The rotation by the angle C is not possible for 5-axis kinematics because of the restricted degrees of freedom for the orientation.

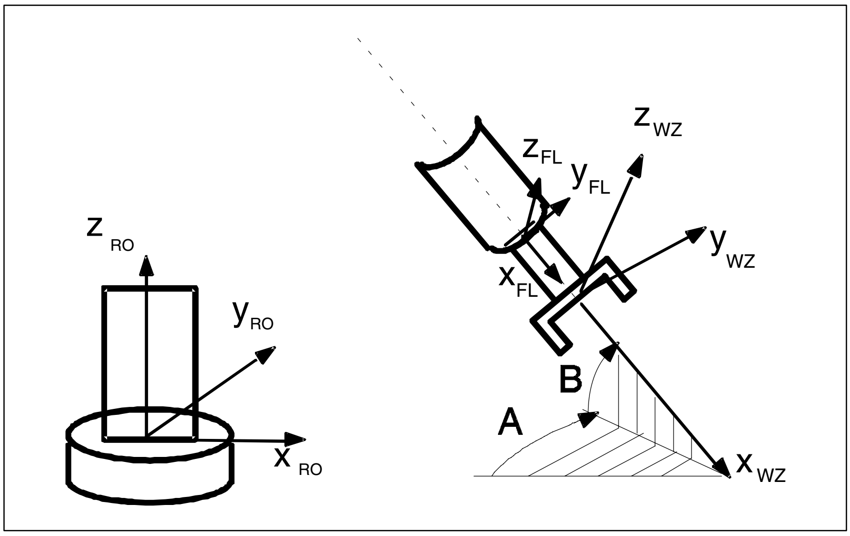



Fig. 2-30 Orientation angle for 5-axis kinematic

In SW 5.3 and higher, it is possible to define orientation axes for the handling transformation package.

!

**Important**

For further information refer to the Function Description Special Functions F2 (Part 3), Section 2.6 Orientation axes and the “Programming Guide Production Planning”, Subsection 7.1.4, “Orientation axes”.

## 2.6 Singular positions and their handling

The calculation of the machine axes to a preset position, i.e. position with orientation, is not always clear. Depending on the machine kinematic, there are positions with an infinite number of solutions. These positions are called "singular".

**Singular positions**

- A singular position is, for example, characterized by the fact that the **fifth axis** is positioned **at 0**°. The singular position does, in this case, not depend on a specified orientation. In this position, the fourth axis is not specified, that means that the fourth axis has no influence on the position or orientation.
- A singular position also exists for articulated arm and Scara kinematics if the **third axis** is positioned at **0**° or at **180**°. These positions are called **leveling/diffraction singularity**.
- Another singular position exists for articulated arm kinematics is the hand point is above the rotary axis of axis 1. This position is called **over-head singularity**.

**Extreme velocity increase**

If the path runs in the proximity of a pole (singularity), it is possible that one axis or several axes traverse at a very high velocity. In this case, alarm 10910 "Extreme axis velocity increase" is triggered.

**Behavior at the pole**

The unwanted behavior of fast compensating movements can be improved by reducing the velocity in the proximity of a pole. Traveling through the pole with active transformation is usually not possible.

## 2.7 Call and application of the transformation

**Activation**

The transformation is activated by means of the TRAORI(1) command.

If the TRAORI(1) command has been executed and the transformation has been activated, the IS “Transformation active” (DB21–28, DBX33.6) is set to “1”.

If the machine data have not been defined for an activated transformation grouping, the NC program stops and the control displays the alarm 14100 “Orientation transformation does not exist”.

For further information refer to the
**References:** /PGA/, Programming Guide Production Planning, Section “5-axis processing”

**Deactivation**

The currently active transformation is deactivated by means of TRAFOOF or TRAFOOF().

!

**Important**

When deactivating the transformation “Handling transformation package”, a preprocessing stop and a preprocessing synchronization is implicitly executed with the main run if MD 24100:TRAFO_TYPE_1 is set to 4099. If MD 24100:TRAFO_TYPE_1 is set to 4100, there is no implicit preprocessing stop.

**RESET/ end of program**

The control behavior concerning run-up, end of program or RESET depends on MD 20110: RESET_MODE_MASK.

Bit 7: Reset behavior of “Active kinematic transformation”

Bit 7 = 0 Here, the basic setting for the active transformation after the end of the parts program or RESET is defined by MD 20140: TRAFO_RESET_VALUE with the following significance:
- 0: After RESET no transformation is active.
- 1 to 8: The transformation preset in MD 24100: TRAFO_TYPE_1 to MD 24460: TRAFO_TYPE_8 is active.

Bit 7 = 1 The current setting for the active transformation remains unchanged after a RESET or end of parts program.

## 2.8 Actual-value display

**MCS machine coordinate system**

The machine axes are displayed in mm/inch and/or degrees in the MCS display mode.

**WCS workpiece coordinate system**

If the transformation is active, the tool tip (TCP) is specified in mm/inch and the orientation by the RPY angles A, B and C in the display mode WCS. The tool direction results from the fact that one vector is first rotated by A in Z direction around the Z axis, then by B around the new Y axis and, finally, by C around the new X axis.
If the transformation is deactivated, the axes are displayed with channel axis identifiers, otherwise, the geo axis identifiers are displayed.

## 2.9 Tool programming

The tool lengths are specified in relation to the flange coordinate system. Only 3-dimensional tool compensations are possible. Depending on the kinematic type, there are additional tool restrictions for 5-axis and 4-axis kinematics. Only a 1-dimensional tool with length in the x direction is possible for a kinematic shown in Fig. 2–23.
The tool direction depends on the basic position of the machine specified by the G codes G17, G18 and G19. The tool lengths refer to the zero position specified by G17. This zero position should not be modified in the program.

**Example**

In the following, the example of a 2-dimensional tool is described which is mounted on a 5-axis Scara (refer to Fig. 2.22). Type 100 (cutting tool) is specified as tool identifier. The tool lengths result from the specifications shown in Fig. 2–31. X-TOOL must be entered as tool length x and Y-TOOL must be entered as tool length y in the tool parameters.

```
$TC_DP1[1,1 ] = 100          ; type cutting tool
$TC_DP3[1,1 ] = 0.0          ; (z) length compensation vector
$TC_DP4[1,1 ] = Y-TOOL       ; (y) length compensation vector
$TC_DP5[1,1 ] = X-TOOL       ; (x) length compensation vector
```

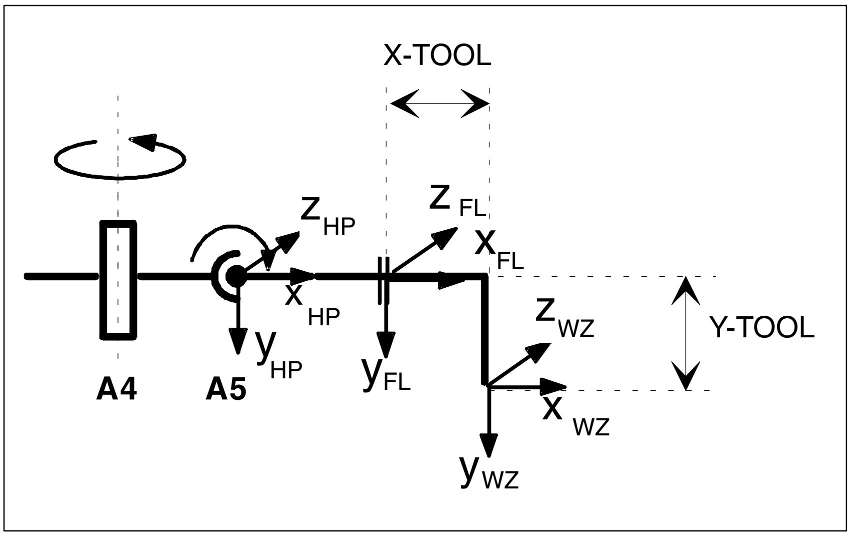



Fig. 2-31 Tool length programming

## 2.10 Cartesian PTP travel with handling transformation package

In software package 5.3 and higher it is possible to use the function Cartesian PTP travel with the handling transformation package. For this purpose, the MD 24100: TRAFO_TYPE_1 must be set to **4100**.

!

**Important**

For further information refer to the Function Description Special Functions F2 (Part 3) Section 2.7 “Cartesian PTP travel” and to the “Programming Guide Production Planning Subsection 7.1.5 Cartesian PTP travel”.

■

# Supplementary Conditions 3

## 3.1 Options

The “Handling Transformation Package” function is an option.

If this option is not implemented in the control and a transformation is called with the command TRAORI, the error message 12140 “5-axis transformation function not implemented” appears and the NC program stops.
If the transformation is not specified in MD 24100: TRAFO_TYPE_1, programming the TRAORI (1) command triggers alarm 14100 “Channel x block y orientation transformation not available”.
If MD 24100: TRAFO_TYPE_1 is set when the handling transformation package is not enabled, alarm 8040 “$MC_TRAFO_TYPE_n reset, corresponding option has not been set” appears when the control next powers up.

**SINUMERIK 840Di**

The compile cycle function of the SINUMERIK 840D is, for the time being, only available on request for the SINUMERIK 840Di.

## 3.2 Creating alarm texts

Add an entry for the alarm text files for the function described in the [TextFiles] section of the C:\OEM\MBDDE.INI file:

CZYK=C:\OEM\TF_

If file C:\OEM\MBDDE.INI does not exist, it must be set up, although only section [Text Files] is required.

Create language-specific text files
TF_xx.COM in directory

C:\OEM\

xx stands for the language code, e.g. GR for German and UK for English.

Enter the following alarm texts there:

in TF_GR.COM

**075200 0 0 “Channel %1, incorrect MD configuration, error: %2”**
**075210 0 0 “Channel %1 axis number/assignment inconsistent”**
**075250 0 0 “Channel %1 tool parameter error”**
**075255 0 0 “Channel %1 working space error”**
**075260 0 0 “Channel %1 block %2 tool parameter error”**
**075265 0 0 “Channel %1 block %2 working space error”**
**075270 0 0 “Channel %1 tool parameter error”**
**075275 0 0 “Channel %1 block %2 working space error”**

## 3.3 Limitations of function

**NCU 572.2**

The Handling Transformation Package can be utilized on NCU 572.2 hardware only on condition that is has been specifically enabled for the customer.

**Clearance control function**

Transformations from the “Handling transformation package” cannot be used in conjunction with the Clearance Control function (only in combination with 840D technology card), since the three basic axes are not generally perpendicular to one another.

**Travel to fixed stop function**

The handling transformation package cannot be operated in conjunction with the “Travel to fixed stop” function.

**Multi-channel systems**

The “Handling transformation package” is configured only for the first channel. It cannot therefore be activated in the second channel.

**Several transformations in one channel**

The “Handling transformation package” is available only once in each channel. It is not possible to switch between several transformations using the TRAFO_TYPE_n (n=1 to 8) command.

**Tool programming**

Tools can only be parameterized by specifying tool lengths. It is not possible to program an orientation for the tool.

**Orientation programming**

In the case of kinematics with less than 5 axes, the orientation can be programmed only by means of “orientation axis angles”. It is not possible to program an orientation using orientation vectors, Euler angles or RPY angles.

In the case of kinematics with 5 axes, the software only supports programming using “orientation axis angles” and orientation vectors.

**Axis assignment**

Channel axes must always be assigned to transformation axes in such a way that the first n channel axes are assigned in ascending sequence to the transformation axes in $MC_TRAFO_AXES_IN_1.

**Singular positions**

A pole cannot be crossed when a transformation is active. Axes may be overloaded at singular positions.

**Dynamic response of axes**

Depending on the type of kinematic, individual axes may be subjected to overloading at certain positions if they are traversing with active transformation. The feedrate is not automatically adjusted. For this reason, the user must reduce the feedrate appropriately at critical points.

■

# Notes

# Data Description (MD, SD) 4

## 4.1 Machine data of standard system

### 4.1.1 Channel-specific machine data

- MD 21100: ORIENTATION_IS_EULER
  Definition of angle for programming of orientation
- MD 24100: TRAFO_TYPE_1
  Definition of transformation
- MD 24110: TRAFO_AXES_IN_1[n]
  Axis assignment for transformation 1 [axis index]: 0 ... 5
- MD 24120: TRAFO_GEOAX_ASSIGN_TAB_1[n]
  Assignment between geometry axes and channel axes for transformation 1 [geometry axis number]: 0 ... 2.
- MD 24520: TRAFO5_ROT_SIGN_IS_PLUS_1[n]
  Sign of rotary axes 1/2 for 5-axis transformation 1 [axis no.]: 0 ... 1
  (not evaluated, see MD 62618: TRAFO6_AXES_DIR)

# 4.2 Machine data in the transformation standard set

## 4.2.1 Channel-specific machine data

| **62600**<br><br>MD number | **TRAFO6_KINCLASS**<br><br>Kinematic category | |
|---|---|---|
| Default setting: 1 | Min. input limit: 1 | Max. input limit: 2 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | | |
| Meaning: | The following categories of kinematic can be specified:<br>• Standard transformation: 1<br>• Special transformation: 2 | |
| Restriction: | See Section 2.1 | |

| **62601**<br><br>MD number | **TRAFO6_AXES_TYPE[n]**<br><br>Axis type for transformation [axis no.]: 0...5 | |
|---|---|---|
| Default setting:<br>1, 1, 1, 3, 3, 3 | Min. input limit: 1 | Max. input limit: 4 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | | |
| Meaning: | This machine data defines the type of axis used in the transformation.<br>The following axis types can be specified:<br>• Linear axis: 1<br>• Rotary axis: 3 (4) | |
| Restriction: | See Subsection 2.3.2 | |

| **62602**<br><br>MD number | **TRAFO6_SPECIAL_KIN**<br><br>Special kinematic type | |
|---|---|---|
| Default setting: 1 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines the type of special kinematic.<br>The following special kinematics are available:<br>• 5-axis articulated arm with coupling between axis 2 and axis 3: 1<br>• 2-axis SCARA with mechanical coupling to tool: 3<br>• 3-axis SCARA with degrees of freedom X, Y, A: 4<br>• 2-axis articulated arm with coupling between axis 1 and axis 2: 5<br>• 4-axis SCARA with coupling between axis 1 and axis 2: 7 | |
| Restriction: | See Subsection 2.4.7 | |
| Figure | See Subsection 2.4.7 | |

| 62603<br><br>MD number | TRAFO6_MAIN_AXES<br><br>Basic axis identifier | |
|---|---|---|
| Default setting: 1 | Min. input limit: 1 | Max. input limit: 12 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | | |
| Meaning: | This machine data defines the type of basic axis arrangement. The term “basic axes” normally refers to the first 3 axes.<br>The package contains the following basic axis arrangements:<br>• SS (gantry): 1<br>• CC (SCARA): 2<br>• NR (articulated arm): 3<br>• SC (SCARA): 4<br>• RR (articulated arm): 5<br>• CS (SCARA): 6<br>• NN (articulated arm): 7 | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| 62604<br><br>MD number | TRAFO6_WRIST_AXES<br><br>Hand axis identifier | |
|---|---|---|
| Default setting: 1 | Min. input limit: 1 | Max. input limit: 6 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | | |
| Meaning: | This machine data defines the type of robot hand. The term “robot hand” normally refers to axes 4 to 6.<br>The package contains the following hand types:<br>• No hand: 1<br>• Central hand: 2<br>• Bevelled hand with elbow: 6 | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| 62605<br><br>MD number | TRAFO6_NUM_AXES<br><br>Number of transformed axes | |
|---|---|---|
| Default setting: 3 | Min. input limit: 2 | Max. input limit: 5 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | | |
| Meaning: | This machine data defines the number of axes to be included in the transformation. SW package 2.3 (810D) or 4.3 (840D) supports kinematics involving a maximum of 5 axes. | |
| Restriction: | See Subsection 2.3.2 | |

| **62606**<br><br>MD number | **TRAFO6_A4PAR**<br><br>Axis 4 is parallel/anti-parallel to last basic axis | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 1 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | | |
| Meaning: | This machine data defines whether the 4th axis is parallel/anti-parallel to the last rotary basic axis.<br>This data is relevant only for kinematics with more than 3 axes.<br>• Axis 4 is parallel/anti-parallel: 1<br>• Axis 4 is not parallel: 0 | |
| Restriction: | See Subsection 2.3.2 | |

| **62607**<br><br>MD number | **TRAFO6_MAIN_LENGTH_AB[n]**<br><br>Basic axis lengths A and B, n = 0...1 | |
|---|---|---|
| Default setting: 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: mm/inches |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines the basic axis lengths A and B. These lengths are specially defined for each basic axis type.<br>• n = 0: Basic axis length A<br>• n = 1: Basic axis length B | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| **62608**<br><br>MD number | **TRAFO6_TX3P3_POS[n]**<br><br>Attachment of hand [position component], n = 0...2 | |
|---|---|---|
| Default setting:<br>0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: mm/inches |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines the position component of frame TX3P3, which provides the link between the basic axes and the hand.<br>• Index 0: x component<br>• Index 1: y component<br>• Index 2: y component | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| 62609<br><br>MD number | TRAFO6_TX3P3_RPY[n]<br><br>Attachment of hand [rotation component], n = 0...2 | |
|---|---|---|
| Default setting:<br>0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: Degrees |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines the orientation component of frame TX3P3, which provides the link between the basic axes and the hand.<br>• Index 0: Rotation through RPY angle A<br>• Index 1: Rotation through RPY angle B<br>• Index 2: Rotation through RPY angle C | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| 62610<br><br>MD number | TRAFO6_TFLWP_POS[n]<br><br>Frame between wrist point and flange coordinate system (position component), n = 0...2 | |
|---|---|---|
| Default setting:<br>0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: mm/inches |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines the position component of frame TFLWP which links:<br>• Index 0: x component<br>• Index 1: y component<br>• Index 2: y component | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| 62611<br><br>MD number | TRAFO6_TFLWP_RPY[n]<br><br>Frame between wrist point and flange coordinate system (rotation component), n = 0...2 | |
|---|---|---|
| Default setting:<br>0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: Degrees |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines the orientation component of frame TFLWP which links:<br>• Index 0: Rotation through RPY angle A<br>• Index 1: Rotation through RPY angle B<br>• Index 2: Rotation through RPY angle C | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| 62612<br><br>MD number | TRAFO6_TIRORO_POS[n]<br><br>Frame between base center point and internal coordinate system (position component), n = 0...2 | |
|---|---|---|
| Default setting:<br>0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: mm |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines the position component of frame TIRORO which links:<br>• Index 0: x component<br>• Index 1: y component<br>• Index 2: y component | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| 62613<br><br>MD number | TRAFO6_TIRORO_RPY[n]<br><br>Frame between base center point and internal coordinate system (rotation component), n = 0...2 | |
|---|---|---|
| Default setting:<br>0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: Degrees |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines the orientation component of frame TIRORO which links:<br>• Index 0: Rotation through RPY angle A<br>• Index 1: Rotation through RPY angle B<br>• Index 2: Rotation through RPY angle C | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| 62614<br><br>MD number | TRAFO6_DHPAR4_5A[n]<br><br>Parameter A for configuring the hand, n = 0...1 | |
|---|---|---|
| Default setting: 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: mm/inches |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines the length a.<br>• n = 0: Transition from axis 5 to 4<br>• n = 1: Transition from axis 5 to 6 | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| 62615 | TRAFO6_DHPAR4_5D[n] | |
|---|---|---|
| MD number | Parameter D for configuring the hand, n = 0...1 | |
| Default setting: 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: mm |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines the length d.<br>• n = 0: Transition from axis 5 to 4<br>• n = 1: Transition from axis 5 to 6 | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| 62616 | TRAFO6_DHPAR4_4ALPHA[n] | |
|---|---|---|
| MD number | Parameter ALPHA for configuring the hand, n = 0...1 | |
| Default setting:<br>-90.0, 90.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: Degrees |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines the angle a.<br>• n = 0: Transition from axis 5 to 4<br>• n = 1: Transition from axis 5 to 6 | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| 62617 | TRAFO6_MAMES[n] | |
|---|---|---|
| MD number | Offset between mathematical and mechanical zero point [axis no.]: 0...5 | |
| Default setting:<br>0.0, 0.0, 0.0, 0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: Degrees |
| Data type: DOUBLE | | |
| Meaning: | An offset can be entered in this data in order to match the mechanical zero point of a rotaryaxis and the mathematical zero point defined by the transformation. | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| 62618<br><br>MD number | TRAFO6_AXES_DIR[n]<br><br>Matching of physical and mathematical direction of rotation [axis no.]: 0...5 | |
|---|---|---|
| Default setting:<br>1, 1, 1, 1, 1, 1 | Min. input limit: -1 | Max. input limit: 1 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | | |
| Meaning: | This machine data is set to match the mathematical and physical directions of rotation of the axes.<br>• +1: Direction of rotation is identical<br>• –1: Direction of rotation is different | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | | |

| 62619<br><br>MD number | TRAFO6_DIS_WRP<br><br>Mean distance between wrist point and singularity | |
|---|---|---|
| Default setting: 10.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: mm/inches |
| Data type: DOUBLE | | |
| Meaning: | A limit value for the distance between the wrist point and a singularity can be entered in this machine data.<br><br>**Not functional!** | |
| Restriction: | | |
| Figure | | |

| 62620<br><br>MD number | TRAFO6_AXIS_SEQ<br><br>Rearrangement of axes | |
|---|---|---|
| Default setting:<br>1, 2, 3, 4, 5, 6 | Min. input limit: 1 | Max. input limit: 6 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DOUBLE | | |
| Meaning: | This machine data can be set to change the positions of axes in the axis sequence so as to convert a kinematic to a standard kinematic. | |
| Restriction: | See Subsection 2.3.2 | |
| Figure | See Subsection 2.3.2 | |

| | | |
|---|---|---|
| **62621**<br><br>MD number | **TRAFO6_SPIN_ON**<br><br>Configuration includes triangular or trapezoidal spindles | |
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 1 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | | |
| Meaning: | This machine data specifies whether the configuration includes triangular spindles or trapezoidal connections.<br>• 0: None included<br>• 1: Connections included<br><br>This function is not currently supported in the software.<br><br>**$MC_TRAFO6_SPIN_ON must be set to** 0**.**<br>**Machine data** 62622 **to** 62628 **are therefore not functional!** | |
| Restriction: | | |
| Figure | | |

| | | |
|---|---|---|
| **62622**<br><br>MD number | **TRAFO6_SPIND_AXIS[n]**<br><br>Axis controlled by triangular spindle, n = 0...2 | |
| Default setting: 0, 0, 0 | Min. input limit: 0 | Max. input limit: 5 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | | |
| Meaning: | This machine data specifies which axis is controlled by a triangular spindle. The configuration can include a maximum of 3 triangular spindles.<br>• n = 0: 1. triangular spindle<br>• n = 1: 2. triangular spindle<br>• n = 2: 3rd triangular spindle | |
| Restriction: | | |
| Figure | | |

| | | |
|---|---|---|
| **62623**<br><br>MD number | **TRAFO6_SPINDLE_RAD_G[n]**<br><br>Radius G for triangular spindle, n = 0...2 | |
| Default setting: 0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: mm/inches |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines the radius G for the nth triangular spindle. | |
| Restriction: | | |
| Figure | | |

| 62624<br><br>MD number | TRAFO6_SPINDLE_RAD_H[n]<br><br>Radius H for triangular spindle, n = 0...2 | |
|---|---|---|
| Default setting:<br>0.0, 0.0, 0.0 | Min. input limit: | Max. input limit: |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: mm/inches |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines the radius H for the nth triangular spindle. | |
| Restriction: | | |
| Figure | | |

| 62625<br><br>MD number | TRAFO6_SPINDLE_SIGN[n]<br><br>Sign for triangular spindle, n = 0...2 | |
|---|---|---|
| Default setting: 1, 1, 1 | Min. input limit: -1 | Max. input limit: 1 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | | |
| Meaning: | This machine data defines the sign for adapting the direction of rotation for the nth triangular spindle. | |
| Restriction: | | |
| Figure | | |

| 62626<br><br>MD number | TRAFO6_SPINDLE_BETA[n]<br><br>Angular offset for triangular spindle, n = 0...2 | |
|---|---|---|
| Default setting:<br>0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: Degrees |
| Data type: DOUBLE | | |
| Meaning: | This machine data defines offset angle b for adapting the zero point for the nth triangular spindle. | |
| Restriction: | | |
| Figure | | |

| 62627<br><br>MD number | TRAFO6_TRP_SPIND_AXIS[n]<br><br>Axes driven via trapezoidal spindle, n = 0...1 | |
|---|---|---|
| Default setting: 0.0 | Min. input limit: 0 | Max. input limit: 5 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | | |
| Meaning: | This machine data specifies which axes are driven via a trapezoidal connection.<br>• n = 0: Axis driven via trapezoid<br>• n = 1: Coupling axis | |
| Restriction: | | |
| Figure | | |

| **62628** | **TRAFO6_TRP_SPIND_LEN[n]** | |
|---|---|---|
| MD number | Trapezoid lengths, n = 0...3 | |
| Default setting:<br>0.0, 0.0, 0.0, 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: mm/inches |
| Data type: DOUBLE | | |
| Meaning: | This machine data specifies the lengths of the trapezoid connection. | |

| **62629** | **TRAFO6_VELCP[n]** | |
|---|---|---|
| MD number | Cartesian velocity [no.]: 0...2 | |
| Default setting:<br>10000.0, 10000.0, 10000.0 | Min. input limit: – | Max. input limit: – |
| Changes effective immediately | Protection level: 2 / 7 | Unit: mm/min, inch/min |
| Data type: DOUBLE | | |
| Meaning: | This machine data can be set to specify a velocity for Cartesian directions for traversing blocks containing G0.<br>• n = 0: Velocity in x direction<br>• n = 1: Velocity in y direction<br>• n = 2: Velocity in z direction | |
| Restriction: | See Subsection 2.3.2 | |

| **62630** | **TRAFO6_ACCCP[n]** | |
|---|---|---|
| MD number | Cartesian acceleration rates [no.]: 0...2 | |
| Default setting:<br>2.0, 2.0, 2.0 | Min. input limit: – | Max. input limit: – |
| Changes effective immediately | Protection level: 2 / 7 | Unit: $m/s^2$ |
| Data type: DOUBLE | | |
| Meaning: | This machine data can be set to specify an acceleration rate for Cartesian directions for traversing blocks containing G0.<br>• n = 0: Acceleration in x direction<br>• n = 1: Acceleration in y direction<br>• n = 2: Acceleration in z direction | |
| Restriction: | See Subsection 2.3.2 | |

| 62631 | TRAFO6_VELORI[n] | |
|---|---|---|
| MD number | Orientation angle velocities [no.]: 0...2 | |
| Default setting:<br>10.0, 10.0, 10.0 | Min. input limit: – | Max. input limit: – |
| Changes effective immediately | Protection level: 2 / 7 | Unit: rev/min |
| Data type: DOUBLE | | |
| Meaning: | This machine data can be set to specify a velocity for orientation angles for traversing blocks containing G0.<br>• n = 0: Velocity angle A<br>• n = 1: Velocity angle B<br>• n = 2: Velocity angle C | |
| Restriction: | See Subsection 2.3.2 | |

| 62632 | TRAFO6_ACCORI[n] | |
|---|---|---|
| MD number | Orientation angle acceleration rates [no.]: 0...2 | |
| Default setting:<br>1.0, 1.0, 1.0 | Min. input limit: – | Max. input limit: – |
| Changes effective immediately | Protection level: 2 / 7 | Unit: Degree/$s^2$ |
| Data type: DOUBLE | | |
| Meaning: | This machine data can be set to specify an acceleration rate for orientation angles for traversing blocks containing G0.<br>• n = 0: Acceleration angle A<br>• n = 1: Acceleration angle B<br>• n = 2: Acceleration angle C | |
| Restriction: | See Subsection 2.3.2 | |

| 62633 | TRAFO6_REDVELJOG[n] | |
|---|---|---|
| MD number | Reduction factor for Cartesian velocities in JOG [no.]: 0...2 | |
| Default setting: 10.0 | Min. input limit: – | Max. input limit: – |
| Changes effective immediately | Protection level: 2 / 7 | Unit: % |
| Data type: DOUBLE | | |
| Meaning: | **Not functional!** | |

■

# 5 Signal Descriptions

## 5.1 Channel-specific signals

| | |
|---|---|
| **DB21–DB28**<br>**DBB232**<br><br>Data block | Number of active F function of G function group 25 (reference tool orientation)<br><br><br>Signal(s) from channel (NCK -> PLC) |
| Edge evaluation: | Signal(s) updated: |
| Meaning 1 | ORIWKS: The tool orientation is implemented in a workpiece coordinate system and is thus not dependent on the machine kinematics. |
| Meaning 2 | ORIMKS: The tool orientation is implemented in a machine coordinate system and is thus dependent on the machine kinematics. This is the default setting with SW2.1 and higher. |
| Meaning 3 | ORIPATH: The tool orientation is implemented with the programmed lead and side angles relative to the path tangent and surface normal vector. |

| | |
|---|---|
| **DB21–DB28**<br>**DBX317.6**<br><br>Data block | PTP travel active<br><br><br>Signal(s) from channel (NCK -> PLC) |
| Edge evaluation: yes | Signal(s) updated: |
| Signal state 1<br>(or signal transition 0 –> 1) | PTP travel active |
| Signal state 0<br>(or signal transition 1 –> 0) | CP travel active |
| Signal irrelevant for ... | no handling transformations active |
| Further references | FB3 Special Functions, F2 |

| | |
|---|---|
| **DB21–DB28**<br>**DBX33.6**<br><br>Data block | Transformation active<br><br><br>Signal(s) from channel (NCK -> PLC) |
| Edge evaluation: yes | Signal(s) updated: |
| Signal state 1<br>(or signal transition 0 –> 1) | Active transformation |
| Signal state 0<br>(or signal transition 1 –> 0) | Transformation not (no longer) active |
| Signal irrelevant for ... | No transformation used |
| Further references | FB3 Special Functions, F2 |

| | |
|---|---|
| **DB21–DB28**<br>**DBX29.4**<br><br>Data block | Activate PTP travel<br><br><br>Signal(s) to channel (PLC –> NCK) |
| Edge evaluation: yes | Signal(s) updated: |
| Signal state 1<br>(or signal transition 0 –> 1) | Activate PTP travel |
| Signal state 0<br>(or signal transition 1 –> 0) | Activate CP travel |
| Signal irrelevant for ... | no handling transformations active |
| Further references | FB3 Special Functions, F2 |

■

# Examples 6

## 6.1 General information about start-up

---

**Note**

With SW 6.4, the compile cycles are supplied as loadable modules. The general procedure for installing such compile cycles can be found in TE0. The specific installation measures for this compile cycle can be found from Section 6.2 onwards.

---

The MMC software version must be 3.5 or later.
For **810D**, a NCK Jeida card with the kinematic transformation “Transformation package Handling” (as of P2.3) must be available and for **840D** a NCK-OEM Jeida card (as of technology card 2).

The following measures need only be taken for 840D controls, since the “Handling transformation package” is integrated as a standard feature in the 810D.

**1. Back up SRAM contents (840D only)**

The first step to be taken to install a compile cycle function on the **840D** is to replace the original card inserted in the NCU by the technology card.
This measure is identical to the procedure followed for upgrading the NCU to a later software version and likewise requires the static (battery-backed) control system memory to be erased. When this is done, all data stored in the SRAM are lost and they must therefore be saved beforehand. For a detailed description, please refer to the Manufacturer/Service Documentation "SINUMERIK 840D/SIMODRIVE 611D Installation and Start-Up Guide":

1. Enter the machine manufacturer password.
2. Switch to the "Services" operating area.
3. Press the "Series start-up" softkey.
4. Select "NC" and "PLC" as the areas to be saved and enter a name of your choice for the archive file to be created on the hard disk. Finish by pressing the RETURN key.
5. If the control system contains machine-specific compensation data, then these must be saved in a separate archive file:
   Press softkey "Data Out" and select under item "NC active data" the following data as required:
   "Measuring system compensations",
   "Sag/Angularity comp." and
   "Quadrant error compensation".
   Save these data by selecting softkey "Archive ..." and specify another file name for a second archive file.

Keep the archive files you have created in a safe place. They will allow you to restore original settings in your system.

**2. Insert the PC card (840D only)**

- Deactivate the control.
- Insert the PC card with the new firmware (technology card) in the PCMCIA slot of the NCU.
- Then proceed as follows:

1. Turn switch S3 on the front panel of the NCU to 1.
2. Switch the control system back on again.
3. During power-up, the firmware is copied from the PC card to the NCU memory.
4. Wait until number "6" is displayed on the NCU digital display
   (after approximately one minute).
5. Turn switch S3 back to zero.

---

**Caution**

If number "6" does not appear, then an error has occurred.
- Incorrect PC card (e.g. card for NCU2 in NCU3 hardware)
- Card hardware defective

---

**3. Copy back SRAM contents (840D only)**

In order to copy the SRAM contents back into the control, please proceed as described in Section “Data backup” (series start-up) in /IAD/, SINUMERIK 840D Installation and Start-Up Guide. Please read all information provided by the manufacturer about new software versions.

- Enter the machine manufacturer password.
- Select “Data In” and “Archive...”. Then load the archive with backup compensation data (if applicable).

## 6.2 Starting up a kinematic transformation

The next step necessary to start up the kinematic transformation is to activate the handling transformation package (option).

Set the option data for handling transformation package.

**Alarms**

Enter the alarm texts in language-specific MMC text files ALC_GR.COM and ALC_UK.COM.

Set option data for transformation.

**Configure the transformation**

1. Enter the transformation type **4099** or **4100** (*if PTP travel* is active) in MD 24100: TRAFO_TYPE_1.
2. Enter the assignment of the channel axes involved in the transformation in MD 24110: TRAFO_AXES_IN_1[0 to 5].Axis numbers start at 1.
3. Enter the geometry axes corresponding to the Cartesian degrees of freedom of the machine in MD 24120: TRAFO_GEOAX_ASSIGN_TAB_1[0 to 2].
4. Enter the kinematic identifier in MD 62600: TRAFO6_KINCLASS.
5. Enter the identifier for special kinematics in MD 62602: TRAFO6_SPECIAL_KIN if you have used a special kinematic.
6. Enter the number of axes in MD 62605: TRAFO6_NUM_AXES.
7. Change the default setting in MD 62618: $MC_TRAFO6_AXES_DIR[ ] if the traversing directions of the axes involved are not the same as the directions defined in the transformation package.
8. Enter the data which define the basic axes:
   – Basic axis identifier in MD 62603: TRAFO6_MAIN_AXES
   – Basic axis lengths in MD 62607: TRAFO6_MAIN_LENGTH_AB
9. Enter any changes to the axis sequence in MD 62620: TRAFO6_AXIS_SEQ.
10. Enter the data which define the hand:
   – Hand axis identifier in MD 62604: TRAFO6_WRIST_AXES
   – Parameters for hand in MD 62614: TRAFO6_DHPAR4_5A, MD 62615: TRAFO6_DHPAR4_5D and 62616: TRAFO6_DHPAR4_5ALPHA
   – MD 62606: TRAFO6_A4PAR
11. Enter the geometry parameters:
   – Frame T_IRO_RO
   – Frame T_X3_P3
   – Frame T_FL_WP
12. Enter the position in relation to the calibration point in MD MD 62617: TRAFO6_MAMES.
13. Enter the Cartesian velocities and acceleration rates.

■

# Data Fields, Lists 7

## 7.1 Interface signals

| DB number | Bit, byte | Name | Ref. |
|---|---|---|---|
| **Channel-specific** | | | |
| 21–28 | 33.6 | Transformation active | K1 |
| 21–28 | 232 | Number of active G function of G function group 25 (ORIWKS, ORIMKS, ORIPATH) | |
| 21–28 | 317.6 | PTP traversal active | F2 |
| 21–28 | 29.4 | Activate PTP travel | F2 |

## 7.2 NC machine data

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| **General ($MN_ ...)** | | | |
| 10620 | EULER_ANGLE_NAME_TAB[n] | Name of Euler angle | R1 |
| 19410 | TRAFO_TYPE_MASK, bit 4 | Option data for OEM transformation | A2 |
| 19600 | CC_EVENT_MASK | Enable “Handling transformation package” function | A3 |
| **Channel-specific ($MC_ ...)** | | | |
| 21100 | ORIENTATION_IS_EULER | Angle definition for orientation programming | F2 |
| 21110 | X_AXIS_IN_OLD_X_Z_PLANE | Coordinate system for automatic FRAME definition | F2 |
| 24100 | TRAFO_TYPE_1 | Definition of transformation | F2 |
| 24110 | TRAFO_AXES_IN_1 | Axis assignment for transformation 1 | F2 |
| 24120 | TRAFO_GEOAX_ASSIGN_TAB_1 | Assignment between geometry axes and channel axes | F2 |
| | | | |
| 62600 | TRAFO6_KINCLASS | Category of kinematic | Sect. 2.1 |
| 62601 | TRAFO6_AXES_TYPE | Axis type for transformation | Subs. 2.3.2 |
| 62602 | TRAFO6_SPECIAL_KIN | Special kinematic type | Subs. 2.4.7 |
| 62603 | TRAFO6_MAIN_AXES | Basic axis identifier | Subs. 2.3.2 |
| 62604 | TRAFO6_WRIST_AXES | Hand axis identifier | Subs. 2.3.2 |
| 62605 | TRAFO6_NUM_AXES | Number of transformed axes | Subs. 2.3.2 |
| 62606 | TRAFO6_A4PAR | Axis 4 parallel/anti-parallel to last basic axis | Subs. 2.3.2 |
| 62607 | TRAFO6_MAIN_LENGTH_AB | Basic axis lengths A and B | Subs. 2.3.2 |
| 62608 | TRAFO6_TX3P3_POS | Attachment of hand (position component) | Subs. 2.3.2 |
| 62609 | TRAFO6_TX3P3_RPY | Attachment of hand (rotation component) | Subs. 2.3.2 |
| 62610 | TRAFO6_TFLWP_POS | Frame between wrist point and flange (position component) | Subs. 2.3.2 |
| 62611 | TRAFO6_TFLWP_RPY | Frame between wrist point and flange (rotation component) | Subs. 2.3.2 |
| 62612 | TRAFO6_TIRORO_POS | Frame between base center point and internal system (position component) | Subs. 2.3.2 |

| | | | |
|---|---|---|---|
| 62613 | TRAFO6_TIRORO_RPY | Frame between base center point and internal system (rotation component) | Subs. 2.3.2 |
| 62614 | TRAFO6_DHPAR4_5A | Parameter A for configuring the hand | Subs. 2.3.2 |
| 62615 | TRAFO6_DHPAR4_5D | Parameter D for configuring the hand | Subs. 2.3.2 |
| 62616 | TRAFO6_DHPAR4_5ALPHA | Parameter ALPHA for configuring the hand | Subs. 2.3.2 |
| 62617 | TRAFO6_MAMES | Offset between mathematical and mechanical zero points | Subs. 2.3.2 |
| 62618 | TRAFO6_AXES_DIR | Matching of physical and mathematical directions of rotation | Subs. 2.3.2 |
| 62619 | TRAFO6_DIS_WRP | Mean distance between wrist point and singularity | |
| 62620 | TRAFO6_AXIS_SEQ | Rearrangement of axes | Subs. 2.3.2 |
| 62621 | TRAFO6_SPIN_ON | Configuration includes triangular or trapezoidal spindles | |
| 62622 | TRAFO6_SPIND_AXIS | Axis that is controlled by triangular spindle | |
| 62623 | TRAFO6_SPINDLE_RAD_G | Radius G for triangular spindle | |
| 62624 | TRAFO6_SPINDLE_RAD_H | Radius H for triangular spindle | |
| 62625 | TRAFO6_SPINDLE_SIGN | Sign for triangular spindle | |
| 62626 | TRAFO6_SPINDLE_BETA | Angular offset for triangular spindle | |
| 62627 | TRAFO6_TRP_SPIND_AXIS | Axes driven via trapezoidal connection | |
| 62628 | TRAFO6_TRP_SPIND_LEN | Trapezoid lengths | |
| 62629 | TRAFO6_VELCP | Cartesian velocities | Subs. 2.3.2 |
| 62630 | TRAFO6_ACCCP | Cartesian acceleration rates | Subs. 2.3.2 |
| 62631 | TRAFO6_VELORI | Orientation angle velocities | Subs. 2.3.2 |
| 62632 | TRAFO6_ACCORI | Orientation angle acceleration rates | Subs. 2.3.2 |
| 62633 | TRAFO6_REDVELJOG | Reduction factor for Cartesian velocities in JOG | |

# 7.3 Alarms

| **75200** | **Channel %1 incorrect MD configuration, %2 incorrect** |
|---|---|
| Explanation | A window in machine data %2 has been detected during power-up in the machine data of the handling transformation package. |
| Reaction | Cancellation of signal “Mode group ready”<br>VDI signal “Alarm is active”, DB10.DBB109 bit 0 |
| Remedy | Configure machine data |
| To continue program | Power ON |

| **75210** | **Channel %1 axis number/axis assignment inconsistent** |
|---|---|
| Explanation | The number of axes specified in MD TRAFO 6_NUM_AXES and the number of axes specified in MD TRAFO_AXES_ IN_1 is inconsistent or the assignment of axis in MD TRAFO_AXES_IN_1 is incorrect. This alarm is output if the transformation is selected via TRAORI. |
| Reaction | Interpreter stop, start interlock |
| Remedy | Correct machine data |
| Cancel criterion | RESET |
| Ref. | Section |

| 75250 | **Channel %1 tool parameter faulty** |
|---|---|
| Explanation | The tool parameters are not the same as the settings for the handling transformation package (checked in interpreter). |
| Reaction | Interpreter Stop |
| Remedy | Correct tool parameters |
| To continue program | RESET |

| 75255 | **Channel %1 working area error** |
|---|---|
| Explanation | The programmed point is not within the working range of the kinematic (checked in interpreter). |
| Reaction | Interpreter Stop |
| Remedy | Correct position |
| To continue program | RESET |

| 75260 | **Channel %1 block %2 tool parameter faulty** |
|---|---|
| Explanation | The tool parameters are not the same as the settings for the handling transformation package (checked during preprocessing run). |
| Reaction | Cancellation of signal “Mode group ready”<br>VDI signal “Alarm is active”, DB10.DBB109 bit 0 |
| Remedy | Correct tool parameters |
| To continue program | RESET |

| 75265 | **Channel %1 block %2 working area error** |
|---|---|
| Explanation | The programmed point is not within the working range of the kinematic (checked during preprocessing run). |
| Reaction | Interpreter Stop, start interlock,<br>Cancellation of signal “Mode group ready”<br>VDI signal “Alarm is active” DB10.DBB109 bit 0 |
| Remedy | Correct position |
| To continue program | RESET |

| 75270 | **Channel %1 tool parameter faulty** |
|---|---|
| Explanation | The tool parameters are not the same as the settings for the handling transformation package (checked in interpolation). |
| Reaction | Motion Stop, start interlock,<br>Cancellation of signal “Mode group ready”<br>VDI signal “Alarm is active” DB10.DBB109 bit 0 |
| Remedy | Correct tool parameters |
| To continue program | RESET |

| 75275 | **Channel %1 block %2 working area error** |
|---|---|
| Explanation | The programmed point is not within the working range of the kinematic (checked during main run). |
| Reaction | Interpreter Stop, start interlock,<br>Cancellation of signal “Mode group ready”<br>VDI signal “Alarm is active” DB10.DBB109 bit 0 |
| Remedy | Correct position |
| To continue program | RESET |

A detailed description of the alarms which may occur is given in
**References:** /DA/, Diagnostics Guide
and the online help of MMC 101/102/103 systems.

■

# SINUMERIK 840D/840Di/810D Description of Functions Special Functions (Part 3)

# Setpoint Exchange (TE5)



■

## Notes

# 1 Brief Description

The setpoint switchover function is used for milling machines with special mills, for which the spindle motor is used both to drive the tool and for mill head orientation. The spindle and the mill head axes are defined as separated axes but driven by only one motor.

**Availability**

The compile cycles are available from NCK **SW 4.4** for NCU 572.2 and NCU 573.2 on the technology PC card (not for export version).

■

## Notes

# Detailed Description 2

## 2.1 General

The term “setpoint” as used in this documentation always refers to the digital speed setpoint of an axis which is transferred from the position controller on the NCU via the 611D drive bus and applied as the input quantity for the speed controller of the appropriate 611D axis module.

The “Setpoint exchange” function is needed in cases where one motor is used to drive two mechanical axes/spindles such as, for example, on milling machines with special millheads. The spindle motor is operated as both a tool drive and a millhead orienting mechanism.

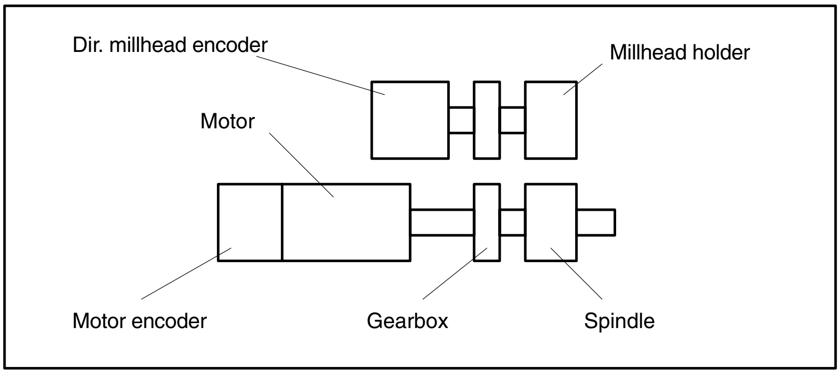



Fig. 2-1 Example 1

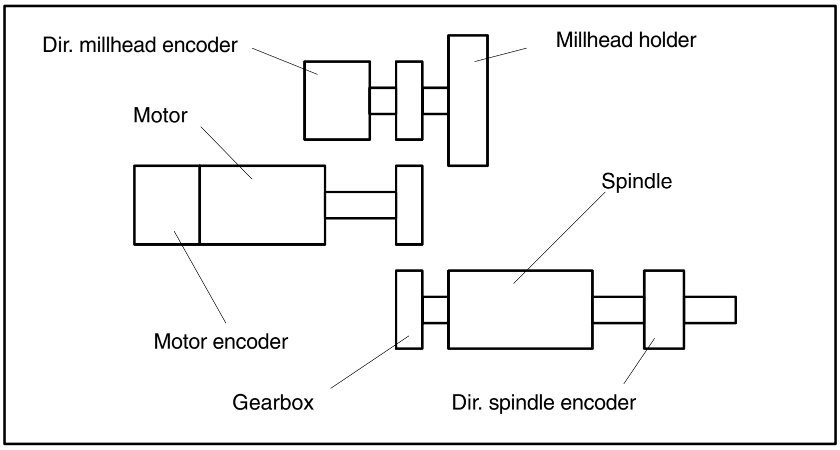



Fig. 2-2 Example 2

The direct spindle encoder is an optional feature. The motor encoder and direct millhead encoder rotate when the millhead axis is in operation. If a direct spindle encoder is fitted, it does not rotate with the other two. When the spindle is in operation, the motor encoder and direct spindle encoder (if installed) rotate. The encoder on the millhead axis does not rotate with the other two.

After coupling/decoupling with the control system switched on (e.g. parts program execution in automatic mode), the same motor alternately drives the spindle or orientates the millhead. Both the spindle and the millhead axis are defined as independent control system axes to which the available encoders, speed ratios, velocities, Hirth tooth system, etc. are allocated by means of machine data.

However, the setpoint output to the motor is permanently assigned to one axis (see standard machine data for setpoint definition: $MA_CTRLOUT_MODULE_NR, $MA_CTRLOUT_TYPE). A simulated setpoint output is assigned to the second axis (millhead axis).

The setpoint exchange function is only meaningful on digital 611D drives. Setpoints can be exchanged between axes only on condition that both axes are at standstill (DB3x.DB61.4). The user PLC program must set the necessary axis status (open position controller, follow-up mode) as a function of the application before the setpoints are switched.

It is possible to define up to three axes which can exchange the setpoint output with another specific axis, i.e. up to 4 axes can be switched onto a motor. This option is needed, for example, in applications involving a number of special millheads which need to be replaced online. The axes involved in the setpoint exchange configuration need not necessarily be assigned to the same channel or mode group.

## 2.2 Control structure

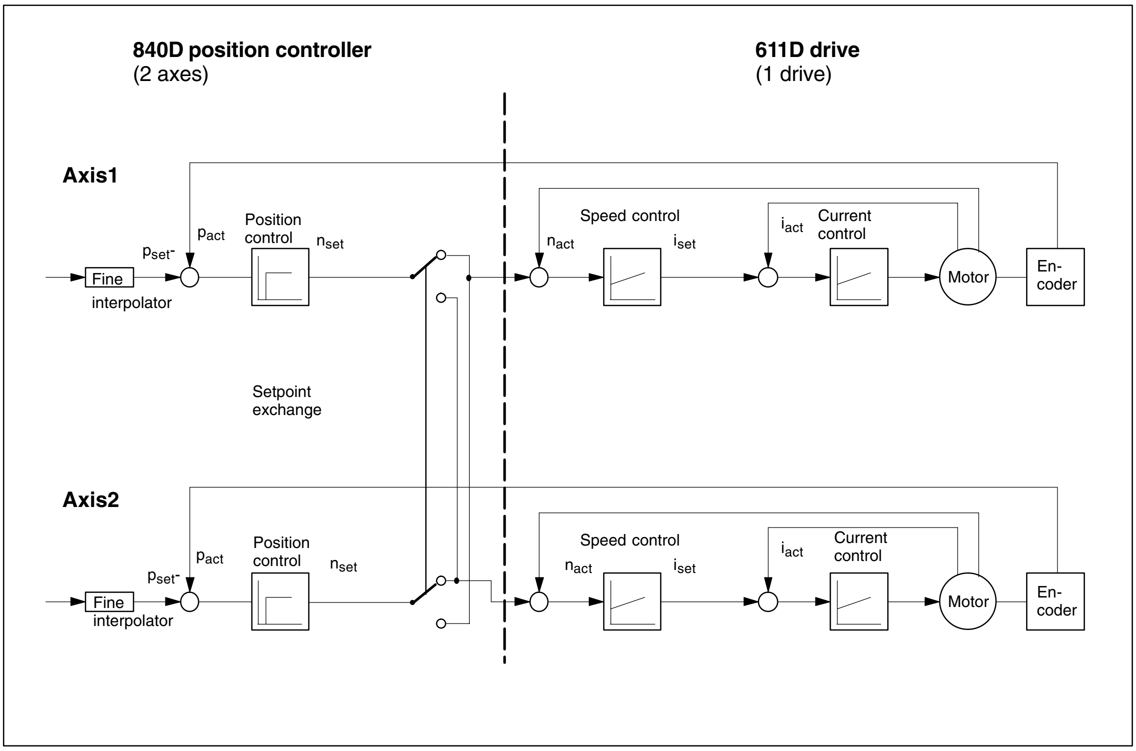



Fig. 2-3

The setpoints are switched after the position controller by exchanging the output drivers of two axes in the NCU. The machine data which define the load ratio (DRIVE_AX_RATIO_DENOM, DRIVE_AX_RATIO_NUMERA, LEADSCREW_PITCH) are permanently stored in the output drivers and updated only during power ON.

As a consequence, the power-ON-active machine data entered for the default axis are activated by the setpoint exchange function for the axis to which the setpoint output has been switched. To ensure accurate position control for the axis to which the setpoint output has been switched, the load ratio of this axis (and any others) must be set in one of the six possible position controller parameter sets in the machine data of the default axis.

This position controller parameter set must be activated for the relevant axis after setpoint output exchange by the PLC parameter set switchover function.

## 2.3 Configuration of a setpoint exchange

All axes to be involved in a “setpoint exchange group” must be defined as machine axes and assigned to an active channel.

Machine data: MD 63750 CTRLOUT_CHANGE_TAB[0..2] must be set to assign up to three axes to the axis which controls the drive in the basic configuration. These four axes can then exchange setpoint outputs.

**Example:**

Spindle “SPI” must exchange the setpoint with the A axis
(defined as the (defined as 6th machine axis in the system) and with the B axis (defined as the 7th machine axis of the system).

The following values must then be entered in the axis-specific MD of spindle SPI:

MD 63750: CTRLOUT_CHANGE_TAB[0]= 6

1st setpoint exchange between spindle and machine axis 6

MD 63750: CTRLOUT_CHANGE_TAB[1]= 7

2nd setpoint exchange between spindle and machine axis 7

MD 63750: CTRLOUT_CHANGE_TAB[2]= 0

3rd setpoint exchange option not utilized

**Display**

The axes involved in a setpoint exchange group are output in all axis displays on the MMC as either simulated or 611D axes depending on the basic configuration.

If an axis must not be output directly in the MMC basic display, then it must be configured as the last axis in the channel (MD 20070 AXCONF_MACHAX_USED), thus ensuring that it appears only when the axis display is actively scrolled.

Another possible option is to assign it to a different channel.

**Channels/mode groups**

The axes involved in a setpoint exchange group need not necessarily be assigned to the same channel. Cross-channel setpoint exchanges with axes from several active channels are possible.

**Axes with a simulated setpoint**

Machine data $MA_SIMU_AX_VDI_OUTPUT must be set to 1 for axes with a simulated setpoint ($MA_CTRLOUT_TYPE=0) in order to ensure that the output signals of the associated axial VDI interface are set.

**Computing time requirement**

The only significant additional CPU load on the cyclic levels are the clock cycles in which setpoints are actually exchanged.

**Direction of rotation**

The direction of rotation for an active coupling can be reversed via machine data 32100 AX_MOTION_DIR. (The machine data of the two axes must be set identically).

**Configuration alarms**

When the control system is powering up, machine data

MD 63750: CTRLOUT_CHANGE_TAB[0..2] is checked for all active machine axes. Alarm 70451 “Setpoint exchange incorrectly defined” is set if:

- The number of the machine axis itself is specified as the exchange axis number
- Gaps are defined or
- One of the exchange axis numbers addresses an inactive machine axis.

When this configuration alarm is generated, all axes in the mode group are switched to follow-up mode. The alarm can be cancelled only by correcting the machine data and then powering up the control again.

## 2.4 Activation and deactivation of a setpoint exchange

**Conditions**

A setpoint can be exchanged only when the following conditions are fulfilled:

- The axes involved in the setpoint exchange group are stationary (DB3x.DBB61.4).
- No alarm is active which has reset Mode Group Ready (DB1x.DBX6.3).
- None of the axes is already involved in any other active setpoint exchange group.
- The drive must have finished powering up (Drive Ready signal DB10.DBX108.6).

**Setpoint exchange via PLC signal**

Two "to axis" axis-specific PLC signals (i.e. axis which controls drive in basic configuration) request a setpoint exchange with one of the other three possible axes defined in machine data MD63750: CTRLOUT_CHANGE_TAB [0..2].

DB3x.DBX24.6–DBX24.5

```
"Exchange setpoint"
Bit     7  6 5  4 3 2 1 0
           | |
           | |
           00  Setpoint exchange deactivated
               Setpoint exchange in initial state<>00
->                   00 Deactivate setpoint exchange
==00 ->              01 Exchange setpoint with axis from
                           $MA_CTRLOUT_CHANGE_TAB[0]
                     10 Exchange setpoint with axis from
                           $MA_CTRLOUT_CHANGE_TAB[1]
                     11 Exchange setpoint with axis from
                           $MA_CTRLOUT_CHANGE_TAB[2]
```

It is not possible to switch directly between two setpoint exchange partners. The setpoint exchange for one axis must be deactivated first by resetting the PLC signals before a new axis can be selected.

If one of the conditions for setpoint exchange is not fulfilled, the last status remains valid. NC alarm "75452 axis %1 setpoint exchange not allowed" is output.

**PLC signal: Status of a setpoint exchange**

The status of a setpoint exchange is displayed in "from axis" axis-specific VDI signals. The signal is updated only for the axis for which a setpoint exchange is requested.

DB3x.DBX96.6–
DBX96.5

“Setpoint exchange status”

Bit 7 6 5 4 3 2 1 0

| |

| |

00 Setpoint exchange deactivated
Setpoint exchange in initial state

01 Setpoint exchanged with axis from
$MA_CTRLOUT_CHANGE_TAB[0]

10 Setpoint exchanged with axis from
$MA_CTRLOUT_CHANGE_TAB[1]

11 Setpoint exchanged with axis from
$MA_CTRLOUT_CHANGE_TAB[2]

**NC STOP, RESET, power ON**

Setpoint exchanges are executed via the PLC interface to ensure that the PLC program can close the mechanical couplings and switch the axes involved to the status required for a setpoint exchange.

To execute a setpoint exchange with a program command, a user auxiliary function (M, H, ..) can be defined which is then evaluated by the PLC program.

The setpoint exchange function remains in its current status on NC/STOP and RESET/M30.

If a defined status in response to NC STOP and RESET/M30 is required, the PLC program is responsible for setting the function to this status.

The status of setpoint assignments as defined in machine data is restored on power ON.

If a setpoint exchange function is to remain active beyond power ON, the PLC program must store the function status and activate it again after power ON.

**Start-up via MMC 103**

The MMC 103 start-up tool does not support the “Setpoint exchange” function. The millhead axis (exchange axis) has to be manually selected for optimization.

**Drive service display**

The drive service display does not support the “Setpoint exchange” function.

After a setpoint exchange, the name of the axis to which the motor is assigned in the initial state remains unaltered in the drive display.

## 2.5 System response to an active setpoint exchange

**PLC interface**

All axis-specific signals indicate the current status of the axis. The drive-specific signals (e.g.: DB3x.DBB93) are activated on the interface DB of the exchange axis after a setpoint exchange.

The user PLC program must process the interface of the exchange axis completely.

**Actual-value display**

The setpoint exchange has no effect on the actual-value display, i.e. the axes are displayed correctly.

In a configuration without direct spindle encoder (the motor encoder for the spindle/millhead axis rotates in millhead operation), the current orientation of the spindle is not displayed.

**Axis traversal after a setpoint exchange**

Axes involved in a setpoint exchange group can be traversed like normal NC axes.

If only the motor measuring system is installed for the spindle, and not a direct measuring system, then it should be set, for example, to follow-up mode in millhead operation to avoid the risk of alarm generation (zero speed monitoring, contour monitoring).

**Reference point approach**

The reference point can be approached in both JOG-Ref mode and from the parts program (G74).

Sequential referencing via MD 34110: REFP_CYCLE_NR is possible if the user PLC program can switch between axes as required during referencing.

■

# Supplementary Conditions 3

**NCU 572.2**

The Setpoint Exchange function can be utilized on NCU 572.2 hardware only on condition that is has been specifically enabled for the customer.

**Availability**

The compile cycles for the "Setpoint exchange" function are available on the technology PC card with NCK **SW 4.4** and higher for the NCU 572.2 and NCU 573.2 (not for export variant).

## 3.1 Limitations

- **Safety Integrated** is not approved for axes in a setpoint exchange group.
- It must be noted in connection with **star/delta changeover** that a request for switchover in the VDI interface of the exchanged axis may be output after the setpoint exchange. This response depends on the appropriate machine data settings on the axes involved in the setpoint exchange and on the current axis status.
- The **"Analog axis"** and **"Master/slave"** functions are not enabled for all axes participating in the setpoint exchange.
- As a result of the exchange of output drivers described in Section 2.2, a maximum of six load conditions (including spindle gear stages) in total can be defined for all the **axes/spindles involved in the setpoint exchange**, and that these must be entered both in the default and the exchange axis as part of the start-up process.
- If **linear axes** are also included in the setpoint exchange group, the spindle pitch of the default axis is also activated on the exchange axis.
- As the controller is located directly on the drive axis with **Dynamic Stiffness Control** functionality, its control response is superior. The setpoint exchange has a simulated axis and must not be used in conjunction with Dynamic Stiffness Control due to the differences in control response.

## 3.2 Creating alarm texts

Add an entry for the alarm text files for the function described in the [TextFiles] section of the C:\OEM\MBDDE.INI file:

CZYK=C:\OEM\TF_

If file C:\OEM\MBDDE.INI does not exist, it must be created, although only section [Text Files] is required.

Create language-specific text files TF_xx.COM in directory

C:\OEM\

xx stands for the language code, e.g. GR for German and UK for English.

Enter the following alarm texts in file TF_GR.COM:

- 70451 0 0 “Setpoint exchange incorrectly defined”
- 70452 0 0 “Axis %1, setpoint cannot be exchanged in this state”

■

# Data Descriptions (MD, SD) 4

## 4.1 Machine data of standard system

The “Setpoint exchange” function is implemented as a compile cycle application. In addition to the function-specific machine data, the following option data must be set.

**Warning**

Failure to take appropriate precautions **can** have undesirable consequences.

The functions activated by the option data trigger corresponding compile cycles. These cycles can have a significant effect on the behavior of the control system and may cause hazardous situations by accessing the NC.
Before a compile cycle is activated, the necessary safety precautions must be taken to protect operating personnel and machine (safety precautions against parameterization or programming errors in the compile cycles may need to be taken).

### 4.1.1 Machine data for setpoint exchange function

| | | |
|---|---|---|
| **63750**<br><br>MD number | **$MA_CTRLOUT_CHANGE_TAB[0..2]**<br><br>Configuration of a setpoint exchange axis group | |
| Default setting: 0 | Min. input limit: 0 | Max. input limit:<br>Number of axes in group |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: Machine axis number |
| Data type: INT | | |
| Meaning: | This machine data configures a setpoint exchange operation by defining the axes between which a setpoint output can be exchanged.<br>The machine axis number of the axis with which setpoints can be exchanged is specified.<br>Up to three “exchange” axes can be defined.These need not be assigned to the same channel.<br><br>Example:<br><br>The spindle (5th machine axis of the system) must exchange a setpoint with the A axis (defined as 6th machine axis in the system) and with the B axis (defined as the 7th machine axis in the system).<br><br>MD 63750: CTRLOUT_CHANGE_TAB[0] = 6<br>1. Setpoint exchange between spindle and machine axis 6<br>MD 63750: CTRLOUT_CHANGE_TAB[1] = 7<br>2. Setpoint exchange between spindle and machine axis 7<br>MD 63750: CTRLOUT_CHANGE_TAB[2] = 0<br>3. Setpoint exchange option not utilized<br><br>The setpoint exchange operation is activated via the PLC interface in the axis DB of the relevant machine axis (DB3x.DBB24, bit 6–5). | |

■

# 5 Signal Descriptions

## 5.1 Axis-specific signals

| **DB31 – DB38**<br>**DBX24.6-DBX24.5**<br>Data block | **Activate setpoint exchange**<br>Signal(s) to axis/spindle (PLC -> NCK) |
|---|---|
| Edge evaluation: yes | Signal(s) updated: Cyclically — Signal(s) valid from SW: 3.5 |
| Signal state ==0 or signal transition 00 –> 01<br>10<br>11 | Activate setpoint exchange.<br><br>The setpoint is exchanged with the axis defined in machine data:<br>MD 63750: CTRLOUT_CHANGE_TAB[0..2]:<br><br>Bit 7 65 4 3 2 1 0<br>\| \|<br>\| \|<br>==00 –> 01 Exchange setpoint with axis set in $MA_CTRLOUT_CHANGE_TAB[0]<br>10 Exchange setpoint with axis set in $MA_CTRLOUT_CHANGE_TAB[1]<br>11 Exchange setpoint with axis set in $MA_CTRLOUT_CHANGE_TAB[2] |
| Signal state 0 or signal transition <> 0 –> 0 | Setpoint exchange must be deactivated.<br><br>Bit 7 65 4 3 2 1 0<br>\| \|<br>\| \|<br><>0 –> 00 Setpoint exchange deactivated<br>00 Deactivate setpoint exchange<br><br>The following conditions must be fulfilled to activate or deactivate the exchange function:<br>– Both axes are stationary (DB3x.DBB61.4)<br>– No alarm is active which has reset Mode Group Ready (DB1x.DBX6.3)<br>– The drives must be operating in cyclic mode (DriveReady DB10.DBX108.6).<br>– The axis with which the setpoint is to be exchanged must not already be involved in another active exchange group.<br>– Machine data $MA_CTRLOUT_CHANGE_TAB[0.2] for the setpoint exchange to be activated must contain a valid machine axis number.<br><br>If one of these conditions is not satisfied, the coupling cannot be activated or deactivated. Alarm 70452 0 0 “Axis %1, setpoint cannot be exchanged in this state” is output. |
| Signal irrelevant for ...... | The signal is relevant only for those axes for which MD 63700 SET_POINT_CHANGE_AX has been set to value other than zero. |

| | | |
|---|---|---|
| **DB31 – DB38**<br>**DBBX96.6 – DBBX96.5**<br>Data block | **Status of setpoint exchange**<br>Signal(s) from axis spindle (NCK –> PLC) | |
| Edge evaluation: no | Signal(s) updated: Cyclically | Signal(s) valid from SW: 3.5 |
| Signal state 1 or signal transition 0 ––> 1 | “Setpoint exchange status”<br><br>Bit 7 65 4 3 2 1 0<br>           \| \|<br>==00 –> 01 Setpoint exchanged with axis set in $MA_CTRLOUT_CHANGE_TAB[0]<br>10 Setpoint exchanged with axis set in $MA_CTRLOUT_CHANGE_TAB[1]<br>11 Setpoint exchanged with axis set in $MA_CTRLOUT_CHANGE_TAB[2] | |
| Signal state 0 or signal transition 1 –> 0 | Setpoint exchange is not active.<br><br>Setpoint exchange must be deactivated.<br><br>Bit 7 65 4 3 2 1 0<br>           \| \|<br><>0 –> 00 Setpoint exchange deactivated (initial state) | |
| Signal irrelevant for ...... | This signal is relevant for the axis which has requested a setpoint exchange. | |

■

# 6 Examples

## 6.1 General start-up of a compile cycle function

**Note**

With SW 6.4, the compile cycles are supplied as loadable modules. The general procedure for installing such compile cycles can be found in TE0. The specific installation measures for this compile cycle can be found from Section 6.2 onwards.

**Precondition**

- The software version installed on the MMC must be 3.5 or higher.
- An NCK technology card with the “Setpoint exchange” function must be available.

**Saving SRAM contents**

As the first step in installing a compile cycle function, the original card inserted in the NCU must be replaced by the technology card.
This is identical to the procedure followed for upgrading the NCU to a later software version and likewise requires the static (battery-backed) control system memory to be erased. When this is done, all data stored in the SRAM are lost and they must therefore be saved beforehand.

Please proceed as follows:

1. Enter the machine manufacturer password.
2. Change to the “Services” operating area.
3. Press softkey “Series start-up”.
4. Select “NC” and “PLC” as the areas to be saved and enter a name of your choice for the archive file to be created on the hard disk. Finish by pressing the RETURN key.
5. If the control system contains machine-specific compensation data, then these must be saved in a separate archive file:
   - Press software “Data out”
   - Select from “NC active data” menu:
     – “Measuring system compensations”
     – “Sag/angularity comp.”
     – “Quadrant error compensation”.
   - Save these data by selecting softkey “Archive...” and
   - enter another file name for a 2nd archive file. archive file.

These archive files will enable you to restore the original status if required.

**References:** For a detailed description, please refer to the Manufacturer/Service Documentation “SINUMERIK 840D/SIMODRIVE 611D Installation and Start-Up Guide”.

**Insert the PC card**

- Switch off control system
- Insert the PC card with the new firmware (technology card) in the PCMCIA slot of the NCU
- Then proceed as follows:
  1. Turn switch S3 on the front panel of the NCU to 1
  2. Switch the control system back on again.
  3. When the system powers up, the firmware is copied from the

     PC card into the NCU memory.
  4. Wait until number “6” appears on the NCU digital display (after approximately 1 minute).
  5. Turn switch S3 back to zero.

---

**Note**

If number “6” does not appear, then the following errors might be the cause:
- Incorrect PC card (e.g. card for NCU2 in NCU3 hardware)
- Card hardware defective

---

**Copy back SRAM contents**

To copy the saved data back into the control system, proceed as described in Section 12.2 (series start-up). Please read all information provided by the manufacturer about new software versions.

- Enter the machine manufacturer password.
- Select “Data In” and “Archive...”.
- Then load the archive with backup compensation data (if applicable).

## 6.2 Start-up of a setpoint exchange

To start up the “Setpoint exchange” function, you next need to active the compile cycle.

**Option data for compile cycles**

To start up the “Setpoint exchange” function, you next need to activate the compile cycle.

To do so, proceed as follows:

- Set the option for the compile cycle application
- Run up the software again. Machine data 63750 will be displayed at the end of the axial machine data list (look for “CTRLOUT_CHANGE_TAB” or “63750”).

**Alarms**

Enter the alarm texts in language-specific text file TK1_GR.COM or TK1_UK.COM.

**Configuring the setpoint exchange**

Assign the exchange axes for the setpoint exchange to the default axis in machine data 63750.

## 6.3 Example of a setpoint exchange configuration

**Inputs**

**Typical milling machine:**
X, Y, Z, main spindle, each with its own drive and motor

**Millhead:**
Modulo rotary axis with Hirth tooth system:
A axis with direct measuring system, but no separate motor

**Extract of machine data**

$MN_DRIVE_IS_ACTIVE[0]=1

$MN_DRIVE_IS_ACTIVE[1]=1

$MN_DRIVE_IS_ACTIVE[2]=1

$MN_DRIVE_IS_ACTIVE[3]=1

$MN_DRIVE_IS_ACTIVE[4]=0

$MN_DRIVE_LOGIC_NR[0]=1

$MN_DRIVE_LOGIC_NR[1]=2

$MN_DRIVE_LOGIC_NR[2]=3

$MN_DRIVE_LOGIC_NR[3]=4

$MN_DRIVE_LOGIC_NR[4]=5

**X axis**

$MA_CTRLOUT_NR[0,AX1]=1

$MA_CTRLOUT_TYPE[0,AX1]=1

$MA_NUM_ENCS[AX1]=1

$MA_ENC_SEGMENT_NR[0,AX1]=1

$MA_ENC_SEGMENT_NR[1,AX1]=1

$MA_ENC_MODULE_NR[0,AX1]=1

$MA_ENC_MODULE_NR[1,AX1]=1

$MA_ENC_INPUT_NR[0,AX1]=1

$MA_ENC_INPUT_NR[1,AX1]=2

$MA_ENC_TYPE[0,AX1]=1

$MA_ENC_TYPE[1,AX1]=0

**Y axis**

$MA_CTRLOUT_NR[0,AX2]=2

$MA_CTRLOUT_TYPE[0,AX2]=1

$MA_NUM_ENCS[AX1]=1

$MA_ENC_MODULE_NR[0,AX2]=2

$MA_ENC_MODULE_NR[1,AX2]=2

$MA_ENC_INPUT_NR[0,AX2]=1

$MA_ENC_INPUT_NR[1,AX2]=2

```
$MA_ENC_TYPE[0,AX2]=1
$MA_ENC_TYPE[1,AX2]=0
```

**Z axis**

```
$MA_CTRLOUT_NR[0,AX3]=3
$MA_CTRLOUT_TYPE[0,AX3]=1
$MA_NUM_ENCS[AX3]=1
$MA_ENC_MODULE_NR[0,AX3]=3
$MA_ENC_MODULE_NR[1,AX3]=3
$MA_ENC_INPUT_NR[0,AX3]=1
$MA_ENC_INPUT_NR[1,AX3]=2
$MA_ENC_TYPE[0,AX3]=1
$MA_ENC_TYPE[1,AX3]=0
```

**Spindle**

```
$MA_CTRLOUT_NR[0,AX4]=4
$MA_CTRLOUT_TYPE[0,AX4]=1
$MA_NUM_ENCS[AX4]=1
$MA_ENC_MODULE_NR[0,AX4]=4
$MA_ENC_MODULE_NR[1,AX4]=4
$MA_ENC_INPUT_NR[0,AX4]=1
$MA_ENC_INPUT_NR[1,AX4]=2
$MA_ENC_TYPE[0,AX4]=1
$MA_ENC_TYPE[1,AX4]=0
$MA_IS_ROT_AX[AX4]=1
$MA_ROT_IS_MODULO[AX4]=1
$MA_DISPLAY_IS_MODULO[AX4]=1
$MA_SIMU_AX_VDI_OUTPUT[AX4]=0
$MA_DRIVE_AX_RATIO_DENUM[AX4,0]=1     // Spindle in axis mode
$MA_DRIVE_AX_RATIO_DENUM[AX4,1]=1     // 1st spindle gear stage
$MA_DRIVE_AX_RATIO_DENUM[AX4,2]=1     // 2nd spindle gear stage
$MA_DRIVE_AX_RATIO_DENUM[AX4,3]=1     // Not used
$MA_DRIVE_AX_RATIO_DENUM[AX4,4]=1     // Not used
$MA_DRIVE_AX_RATIO_DENUM[AX4,5]=12    // Used for A axis operation
$MA_DRIVE_AX_RATIO_DENUM[AX4,0]=1     // Spindle in axis mode
$MA_DRIVE_AX_RATIO_DENUM[AX4,1]=1     // 1st spindle gear stage
$MA_DRIVE_AX_RATIO_DENUM[AX4,2]=1     // 2nd spindle gear stage
$MA_DRIVE_AX_RATIO_DENUM[AX4,3]=1     // Not used
$MA_DRIVE_AX_RATIO_DENUM[AX4,4]=1     // Not used
$MA_DRIVE_AX_RATIO_DENUM[AX4,5]=13    // Used for A axis operation
```

| | |
|---|---|
| $MA_SPIND_ASSIGN_TO_MACHAX[AX4]=1 | |
| $MA_CTRLOUT_CHANGE_TAB[0] = 5 | // Spindle can exchange setpoint with 5th machine axis (A axis) |
| $MA_CTRLOUT_CHANGE_TAB[1] = 0 | |
| $MA_CTRLOUT_CHANGE_TAB[2] = 0 | |

**A axis**

The A axis has a direct encoder on the free measuring system input of the Z axis, but does not have its own motor, i.e. the setpoint output is simulated in the basic configuration.

| | |
|---|---|
| $MA_CTRLOUT_NR[0,AX5]=5 | |
| $MA_CTRLOUT_TYPE[0,AX5]=0 | |
| $MA_NUM_ENCS[AX5]=1 | |
| $MA_ENC_MODULE_NR[0,AX5]=3 | |
| $MA_ENC_MODULE_NR[1,AX5]=3 | |
| $MA_ENC_INPUT_NR[0,AX5]=2 | |
| $MA_ENC_INPUT_NR[1,AX5]=2 | |
| $MA_ENC_TYPE[0,AX5]=1 | |
| $MA_ENC_IS_DIRECT[1,AX5]=1 | |
| $MA_IS_ROT_AX[AX5]=1 | |
| $MA_ROT_IS_MODULO[AX5]=1 | |
| $MA_DISPLAY_IS_MODULO[AX5]=1 | |
| $MA_SIMU_AX_VDI_OUTPUT[AX5]=1 | // Important for updating the VDI interface |
| $MA_DRIVE_AX_RATIO_DENUM[AX5,0]=12 | // Not used |
| $MA_DRIVE_AX_RATIO_DENUM[AX5,1]=12 | // Not used |
| $MA_DRIVE_AX_RATIO_DENUM[AX5,2]=12 | // Not used |
| $MA_DRIVE_AX_RATIO_DENUM[AX5,3]=12 | // Not used |
| $MA_DRIVE_AX_RATIO_DENUM[AX5,4]=12 | // Not used |
| $MA_DRIVE_AX_RATIO_DENUM[AX5,5]=12 | // Used for A axis operation |
| $MA_DRIVE_AX_RATIO_DENUM[AX5,0]=13 | // Not used |
| $MA_DRIVE_AX_RATIO_DENUM[AX5,1]=13 | // Not used |
| $MA_DRIVE_AX_RATIO_DENUM[AX5,2]=13 | // Not used |
| $MA_DRIVE_AX_RATIO_DENUM[AX5,3]=13 | // Not used |
| $MA_DRIVE_AX_RATIO_DENUM[AX5,4]=13 | // Not used |
| $MA_DRIVE_AX_RATIO_DENUM[AX5,5]=13 | // Used for A axis operation |
| $MA_MODULO_RANGE[AX5]=360 | |
| $MA_INDEX_AX_ASSIGN_POS_TAB[AX5]=3 | |
| $MA_INDEX_AX_NUMERATOR[AX5]=360 | |

$MA_INDEX_AX_DENUMERATOR[AX5]=360

$MA_HIRTH_IS_ACITVE[AX5]=3

$MA_PARAMSET_CHANGE_ENABLE[AX5]=TRUE
// see Description of Functions A2, Parameter Set Switchover

**User PLC program:**

M81: Prepare millhead operation,
Set spindle to “Follow-up mode”,
mechanical switchover,
select setpoint exchange function
(DB34.DBX24.6–DBX24.5= 01),
set millhead axis to position control
(DB35.DBX2.1=1, DBX1.4=0),

M80: Millhead has been oriented,
switch back to spindle mode,
set millhead axis to “Follow-up mode”
(DB35.DBX2.1=0, DBX1.4=1),
deselect setpoint exchange function
(DB34.DBX24.6–DBX24.5= 00),
mechanical switchover,
activate spindle

After a setpoint output has been exchanged, the PLC user program must select the parameter set which contains the correct load condition (DB3x.DBB9). Checkback signal for selection DB3X.DBB69.

Before the spindle is activated again, the gear stage that was active prior to the setpoint exchange should be selected again by means of a parameter set switchover (current gear stage = parameter set: DB3x.DBB82).

**Extract of parts program**

```
M3 S1000          // Spindle mode
  ..              // Workpiece machining
  M5
  SPOS= 98        // Turn spindle to mechanical switchover position
M81               // Activate mechanical switchover
G1 C78,5 F1000    // Re-orientate millhead axis
M80               // Mechanical switchover to spindle mode
M3 S1000          // Workpiece machining
..
```

**PLC user program:**

**Activate setpoint exchange:**

Step 1: Store current spindle gear state, activate setpoint exchange on spindle interface DB (DB34.DBB24).

Step 2: Wait for “Setpoint exchanged” signal (DB34.DBB 96).

Step 3: Activate parameter set on exchange axis (A axis) (DB35.DBB9), see example: Parameter set 5.

Step 4: Wait for “Parameter set switched” checkback signal (DB35.DBB69).

Step 5: Enable exchange axis (controller enable).

**Deactivate setpoint exchange:**

Step 1: Activate the parameter set with stored spindle gear stage on the exchange axis (A axis) again using the parameter set switchover function (DB35.DBB9).

Step 2: Wait for “Parameter set switched” checkback signal (DB35.DBB69), cancel the controller enable for the axis.

Step 3: Deactivate the setpoint exchange (DB34.DBB24).

Step 4: Wait for checkback signal (DB34.DBB 96).

Step 5: Enable the spindle (controller enable).

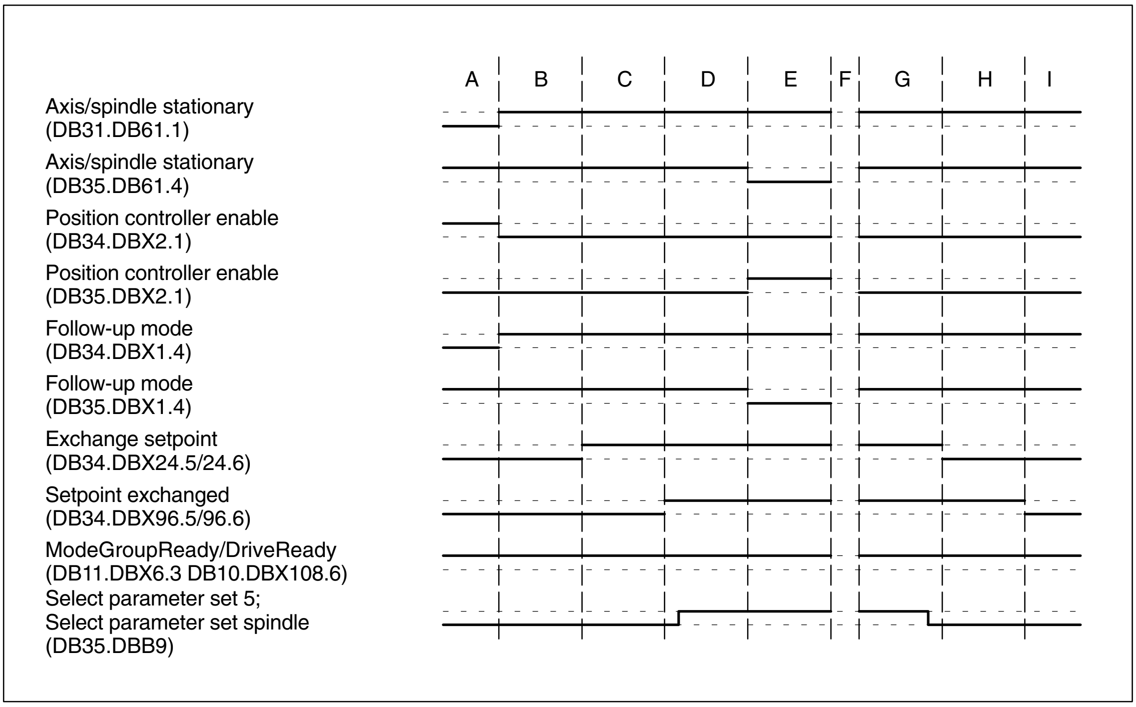



Fig. 6-1 Example of a PLC signal diagram

A: Setpoint exchange function is in the initial state, Axis 1 is in operation under position control.

B: Axes 1 and 2 are stationary and in “Follow-up mode” (if required by application).

C: Select setpoint exchange.

D: “Setpoint exchange activated” checkback.

D: Select parameter set for axis operation.

E: Axis 2 is switched to position control.

F: Axis 2 is active.

G: Axes 1 and 2 are stationary and in “Follow-up mode”
(if required by application).

G: Select parameter set on axis for spindle operation (gear stage).

H: Deactivate setpoint exchange.

I: “Setpoint exchange deactivated” checkback.

Axis 1 can now be traversed normally again.

■

# 7 Data Fields, Lists

## 7.1 Alarms

**75451** **Error in definition of a setpoint exchange**

Explanation: Exchange axis number and machine axis number are identical, gaps have been defined, or the machine axis number is the machine axis number of an inactive machine axis of the system.

Reaction: Axes switch to follow-up mode.

Remedy: Enter another axis number for the setpoint exchange in MD 63750.

Cancel criterion: Power ON

**75452** **Axis %1, setpoint cannot be exchanged in this state**

Explanation: %1 = axis number

- The axes included in the setpoint exchange group are not all stationary (DB3x.DBB61.4).
- No ModeGroupReady signal.
- One of the two axes in the setpoint exchange group is already configured in another active setpoint exchange.
- The machine data MD63750: CTRLOUT_CHANGE_TAB[0.2] for the setpoint exchange to be activated equals zero.
- A new exchange has been requested via PLC before the function has been returned to its initial state.

Reaction: VDI signal “Alarm is active” DB10.DBB109 bit 0

Remedy: Enter another axis number for the setpoint exchange in MD 63750.

Cancel criterion: RESET

## 7.2 Machine data

| Number | Identifier name | | Reference |
|---|---|---|---|
| **Axis/spindle-specific ($MA_ ...)** | | | |
| 34110 | REFP_CYC_NR | Axis sequence for referencing | **R1** |
| 30120 | CTRLOUT_NR | Setpoint assignment | G2 |
| 30130 | CTRLOUT_TYPE | Setpoint output mode | G2 |
| 30350 | SIMU_AX_VDI_OUTPUT | Output of axis signals for simulation. | G2 |
| 63750 | CTRLOUT_CHANGE_TAB[0..2] | Definition of setpoint exchange | TE |
| | | | |

## 7.3 Interface signals

| DB no. | Bit, byte name | Reference |
|---|---|---|
| **Axis/spindle-specific** | | |
| DB3x.DBX24 | Bit6, Bit5 “Activate setpoint exchange” | |
| DB3x.DBX96 | Bit6, Bit5 “Status of setpoint exchange” | |

■

# SINUMERIK 840D/840Di/810D Description of Functions Special Functions (Part 3)

# MCS Coupling (TE6)



■

## Notes

# 1 Brief Description

**MCS coupling**

A 1:1 coupling in the machine coordinate system (MCS coupling) has been introduced in the compile cycle application.

The axes involved in the coupling are defined in an axial machine data. The machine data is updated by RESET to allow new axis pairs to be defined in operation.

**CC_Master**
**CC_Slave**

There are CC_Master and CC_Slave axes. A CC_Master axis can have several CC_Slave axes, but a CC_Slave axis cannot be a CC_Master axis (error message).

The coupling between these pairs is activated and deactivated by means of an OEM-specific language command and can thus be active in all operating modes. If a CC_Slave axis is programmed in a parts program, either an alarm is output or a "GET" operation initiated (depending on MD30552: AUTO_GET_TYPE).

The following restrictions apply to CC-Slave axes:

- It cannot be made into a PLC axis.
- It cannot be made into a command axis.
- It cannot be operated separately from its CC_Master axis in JOG mode.

A tolerance window between the CC_Master and CC_Slave axes is specified via an axial machine data. When an MCS coupling is active, the actual values of the two axes must not leave this window.

**Collision protection**

To protect machining heads against collision in decoupled operation or in mirrored coupling mode, a collision protection can be set in a machine data. This is then activated either via a machine data or via the VDI-IN interface. The assignment of the protected pairs is not related to the CC_Master and CC_Slave pairs.

■

# Notes

# Detailed Description 2

## 2.1 General

If a machine tool has 2 or more mutually independent traversing machining heads (in this case K1 (Y/ Z/ C/ A/ W or K2 (Y2/ Z2/ C2/ A2/ W2)), and if a transformation needs to be activated for the machining operation, the orientation axes cannot be coupled by means of the standard coupling functions (COPON, TRAILON). The only coupling function currently available in the machine coordinate system (MCS) is the GANTRY function. However, this cannot be activated in a parts program and only permits 1:1 couplings.

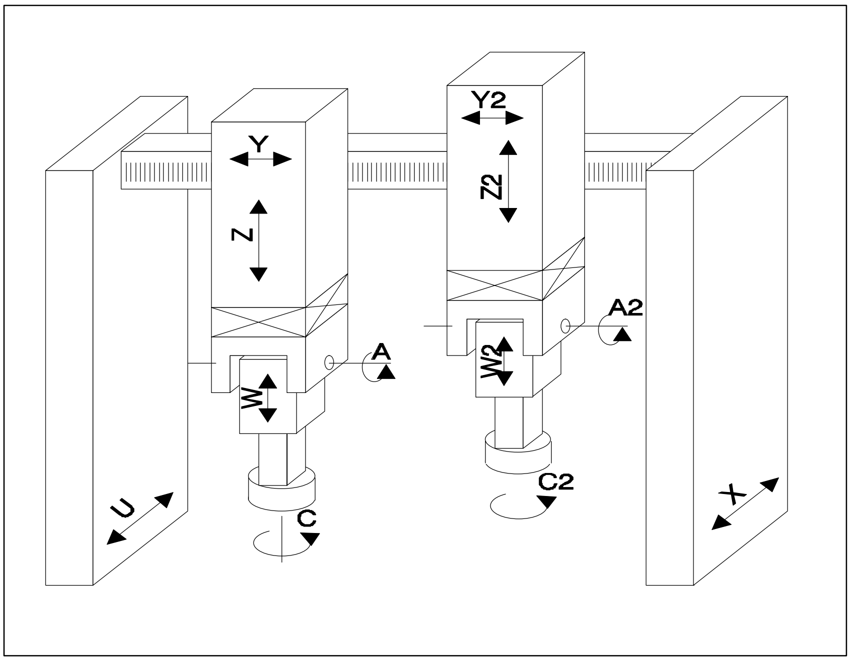



Fig. 2-1

The compile cycle function "MCS coupling" allows a 1:1 or 1:–1 coupling in the machine coordinate system to be switched ON and OFF by parts program commands.

## 2.2 Description of MCS coupling functions

### 2.2.1 Defining coupling pairs

A CC_Slave axis is assigned to its CC_Master axis by means of axial machine data **MD 63540: CC_MASTER_AXIS**. Axis assignments to a CC coupling can be altered only when the coupling is deactivated.

A CC_Slave axis is displayed in axial VDI-Out byte **DB3x DBB97 bit0**.

**Precondition**

- The CC_Master and CC_Slave axes must be either both rotary axes or both linear axes.
- Spindles cannot be coupled by this function.
- Neither the CC_Master nor CC_Slave axis may be an “exchange axis” ($MA_MASTER_CHAN[AXn]=0)

### 2.2.2 Switching the coupling ON/OFF

**CC_COPON()**

CC_COPON([A1][A2][A3][A4][A5])
Switch on the 1:1 coupling.
Tolerance window monitoring is active.

**CC_COPONM()**

CC_COPONM([A1][A2][A3][A4][A5])
Switch on the 1:–1 coupling (Mirror).
Tolerance window monitoring is not active.

A1–A5 are axis names. These can be used to program either the machine axis names, channel axis names or geometry axis names of the axis assigned to a coupling. In other words, either the CC_Master axes or the CC_Slave axes or both can be programmed at the same time. An alarm is output if an axis not involved in a coupling is programmed in A1–A5. All defined couplings are switched on with CC_COPON() or CC_COPONM(). An active coupling is displayed in axial VDI-Out byte **DB3x DBB97 bit1** for the CC-Slave axis. If mirroring is active, it is displayed additionally in **DB3xDBB97 bit2**.

The coupling can be suppressed in axial VDI-In byte **DB3x DBB24 bit2** for the CC_Slave axis. This does not generate an alarm.

**CC_COPOFF()**

CC_COPOFF([A1][A2][A3][A4][A5])

As CC_COPON or CC_COPONM() except for the fact that no alarm is generated if A1–A5 is used to program an axis that is not involved in a coupling.

An existing coupling can also be switched off via the axial VDI-In bit on the CC-Slave axis.

The coupling can be switched ON or OFF only if all axes involved are stationary.

### 2.2.3 Tolerance window

A monitoring window can be programmed via the axial machine data **MD 63541: CC_POSITION_TOL**. The absolute difference between the actual values of CC_Slave axis and CC_Master axis must never be greater than this value. Alarm 70010 is output if the tolerance window is violated.

The monitoring function is not active

- if the machine data is set to 0.
- if the coupling is switched off.
- if “Axis/spindle inhibit” is set for one of the axes.
- if an axis is in “Follow-up” mode.
- for the 1:–1 coupling.

If the offset stored at the instant of coupling activation changes when the 1:1 coupling is active, the change is indicated by the NC => PLC VDI IS **DB3x DBB97 bit 3**.

---

**Note**

The offset might change

- if the SW limit monitor was active for one axis during the main run.
- if one axis has been switched to follow-up mode.
- if the collision protection was active for one axis.

---

# 2.3 Description of collision protection functions

## 2.3.1 Defining protection pairs

A ProtecMaster (PMaster) is assigned to its ProtectSlave axis (PSlave) in axial machine data **MD 63542: CC_PROTECT_MASTER**. The protection pairs can thus be defined independently of the coupling pairs.

A PSlave axis may act as the PMaster axis for another axis.

The axes must be either both rotary axes or both linear axes.

## 2.3.2 Switching the collision protection ON/OFF

The minimum clearance between PSlave and PMaster is programmed in axial machine data **MD 63544: CC_COLLISION_WIN** for the PSlave axis. No collision protection is implemented if the value entered here is less than 0. The offset of the 0 position between PSlave and PMaster is entered in axial machine data **MD 63545 CC_OFFSET_MASTER** (PSlave axis).

The monitoring function for each individual axis must be enabled in machine data **MD 63543: CC_PROTECT_OPTIONS** before the collision protection is switched on. In the same machine data for the PSlave axis, a setting is entered to specify whether the collision protection must be active continuously or whether it is activated via VDI interface signal (PLC => NC) **DB3x DBB24 bit3**.

If collision protection is active, the setpoint positions of the PSlave and PMaster in the next IPO cycle are extrapolated and monitored in the IPO clock cycle using the current setpoint position and current velocity.
If the axes violate the minimum clearance, they are braked at the configured maximum acceleration rate (MD 32300: MAX_AX_ACCEL) or at a 20% faster acceleration rate (defined in MD 63543: CC_PROTECT_OPTIONS). An alarm is output as soon as the axes reach zero speed.

---

**Warning**

If the axes are forced to brake, the positions displayed in the workpiece coordinate system are incorrect!
These are not re-synchronized again until a system RESET.

---

If the axes are already violating the minimum clearance when collision protection is activated, they can only be traversed in one direction (retraction direction). The retraction direction is programmed in MD 63543: CC_PROTECT_OPTIONS.

The collision protection status is optionally displayed in axial VDI-Out byte DB3x DBB66 bit0 of the PSlave.

- DB3x DBB66 Bit0=1 => collision protection activated
- DB3x DBB66 Bit0=0 => collision protection deactivated

The output is activated via bit7 in MD 63543: CC_PROTECT_OPTIONS of the PSlave axis.

### 2.3.3 Configuring example

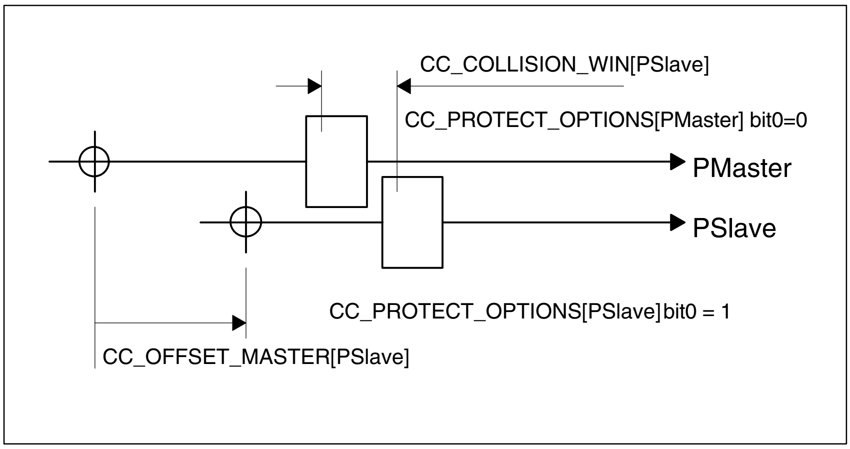



Fig. 2-2 Configuring example

---

**Note**

Since the collision protection function extrapolates the target positions from the "current velocity + maximum acceleration (or +20%)", the monitoring alarm may be activated unexpectedly at reduced acceleration rates:

**Example**:
PMaster = X, PSlave = X2, $MA_CC_COLLISION_WIN = 10 mm
Starting point in parts program: X=0.0 X2=20.0

N50 G0 X100 X2=90 ; the monitoring alarm is activated
because X and X2 are interpolating together: For this reason, the acceleration rate of X2 is less than maximum acceleration.

**Remedy**:

- N50 G0 POS[X]=100 POS[X2]=90 or
- switch the monitoring function off.

---

## 2.4 User-specific configurations

**Parking the machining head**

In this context, “parking” means that the relevant machining head is not involved in workpiece machining. All axes are operating under position control and positioned at exact stop.

The coupling should be active even if only one head is being used in the machining operation! This applies in particular if only the second head (Y2...) is in use. “Axis/spindle inhibit” must then be set axially (PLC –> NCK) for the “parked” head.

---

**Note**

When an axis/spindle inhibit is active, a parts program can be executed if this axis is not operating under position control.

---

**Spindle functionalities**

Since an MCS coupling cannot be activated for spindles, other types of solution should be configured for these.

- Position spindle (SPOS= .....)
  A cycle is called instead of SPOS.
  SPOS is called for all active spindles in this cycle.
- Speed input
  Speed and direction of rotation inputs can be detected via synchronous actions or PLC and passed on to all other active spindles.
- Synchronous spindle function.

## 2.5 Special operating states

**RESET**

The couplings can remain active after a RESET.

**Reorg**

No non-standard functionalities.

**Block search**

During a block search, the last block containing an OEM-specific language command is always stored and then output with the last action block. This feature is illustrated in the following examples.
The output positions of the axes are always 0.

**Example 1:**

N01 M3 S1000

N02 G01 F1000 X10 Y10

N03 CC_COPON( X, Y)

TARGET:

If this program is started normally, axes X and Z traverse to X10 Z10 in the decoupled state. After block search to TARGET: Axes X and Y traverse to this position in the coupled state!

**Example 2:**

N01 M3 S1000

N02 CC_COPON( X)

N03 G01 F1000 X100 Y50

N04 CC_COPOFF( X)

N05 CC_COPON( Y)

N06 Y100

N10 CC_COPOFF()

TARGET:

After block search to TARGET: The axes traverse to X100 Y100 in the decoupled state.

**Example 3:**

N01 CC_COPON( X, Y, Z)

N02 ...

...

N10 CC_COPOFF( Z)

TARGET:

After block search to TARGET: If no coupling is active!

**Single block**

There are no nonstandard functionalities.

■

# Supplementary Conditions 3

**Validity**

The function is configured only for the first channel.

**Compile cycle no.**

The function is assigned internal compile cycle no. 7.

**NCU 572.2**

The MCS Coupling function can be utilized on NCU 572.2 hardware only on condition that is has been specifically enabled for the customer.

**Braking behavior**

**Braking behavior of path axes at SW limit**
The programmable acceleration factor ACC for deceleration at the SW limit refers to path axes.

The axes in an MCS coupling are principal axes that are referred to as geometry axes due to their geometric arrangement.

**Braking geometry axes using synchronized actions**
The faster deceleration capacity as required for path axes can be implemented for geometry axes as follows using a synchronized action.

ACC[x2] = 190

■

# Notes

# Data Descriptions (MD, SD)

# 4

## 4.1 General machine data

The MCS coupling function is implemented as a compile cycle application. In addition to the function-specific machine data, the following option data must be set.

---

**Warning**

Failure to take appropriate precautions **can** have undesirable consequences.

The functions activated by the option data trigger corresponding compile cycles. These cycles can have a significant effect on the behavior of the control system and may cause hazardous situations by accessing the NC.
Before a compile cycle is activated, the necessary safety precautions must be taken to protect operating personnel and machine (safety precautions against parameterization or programming errors in the compile cycles may need to be taken).

---

## 4.2 Channel-specific machine data

| | | |
|---|---|---|
| **28090**<br>MD number | **NUM_CC_BLOCK_ELEMENTS**<br>Number of block elements for compile cycles. | |
| Default setting: 0 | Min. input limit: 0 | Max. input limit: |
| Changes effective after power ON | Protection level: | Unit: – |
| Data type: DWORD | | |
| Meaning: | Number of block elements for compile cycles.<br>Dynamic memory is reserved.<br>MN_NUM_CC_BLOCK_ELEMENTS = 1 (or higher) | |

| | | |
|---|---|---|
| **28100**<br>MD number | **NUM_CC_BLOCK_USER_MEM**<br>Total size of usable block memory for compile cycles | |
| Default setting: 256 | Min. input limit: 0 | Max. input limit: 256 |
| Changes effective after power ON | Protection level: | Unit: – |
| Data type: DWORD | | |
| Meaning: | Total size of block memory for compile cycles available to user in KB.<br>Dynamic memory is reserved.<br>The memory is allocated in blocks of 128 bytes. | |

## 4.3 Axis-specific OEM machine data

| **63540**<br>MD number | **CC_MASTER_AXIS**<br>Specifies the CC_Master axis assigned to a CC_Slave axis | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 8 |
| Changes effective after RESET | Protection level: | Unit: – |
| Data type: INT | | |
| Meaning: | With a value (n) of higher than 0, the axis is a CC_Slave axis. This machine data specifies the associated CC_Master axis.<br><br>The machine axis and axis name can be determined from channel-specific machine data<br>20070 MC_AXCONF_MACHAX_USED[ n–1 ] and<br>20080 MC_CHANAX_NAME_TAB[n–1].<br><br>**Caution:**<br><br>CC_Master and CC_Slave must be of the same axis type<br>(i.e. both linear or both rotary).<br><br>CC_Master and CC_Slave must not be a spindle.<br>CC_Master and CC_Slave must not be an exchange axis.<br><br>If the two axes have different dynamic responses, it is advisable to make the weaker of the two the CC_Master axis.<br><br>The machine data may be altered only when the coupling is switched off. | |

| **63541**<br>MD number | **CC_POSITION_TOL**<br>Monitoring window (valid only for CC_Slave axes) | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: ∞ |
| Changes effective after RESET | Protection level: | Unit: – |
| Data type: DOUBLE | | |
| Meaning: | Monitoring window (valid only for CC_Slave axes)<br><br>The difference between the actual values of CC_Slave axis and CC_Master axis must never leave the monitoring window or else an alarm will be generated.<br><br>The following equation applies:<br><br>d = I act [CC_Master ] – (act[CC_Slave] + Offset) I d <= MD63001<br><br>Offset = difference in position between CC_Master and CC_Slave when coupling is activated.<br><br>A setting of 0 deactivates the monitoring function. | |

| 63542<br><br>MD number | CC_PROTEC_MASTER<br><br>Specifies the PMaster axis assigned to a PSlave axis. | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 8 |
| Changes effective after RESET | Protection level: | Unit: – |
| Data type: INT | | |
| Meaning: | With a value (n) of higher than 0, the axis is a PSlave axis.<br>The machine data specifies the associated PMaster axis.<br>The machine axis and axis name can be defined in channel-specific machine data MD 20070: MC_AXCONF_MACHAX_USED[ n–1 ] and<br>MD 20080: MC_CHANAX_NAME_TAB[n–1].<br><br>**Caution:**<br>PMaster and PSlave must be of the same axis type<br>(i.e. both linear or both rotary). | |

| 63543<br><br>MD number | CC_PROTEC_OPTIONS | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 7 |
| Changes effective after RESET | Protection level: | Unit: – |
| Data type: INT | | |
| Meaning: | **Bit0 – bit3 for PMaster and PSlave**<br>Bit0 = 1 Retract in PLUS<br>Bit1 = 1 Factor 1.2 for maximum braking acceleration rate<br>Bit2 = 1 Monitoring can be activated even if axis is unreferenced.<br>Bit Spare<br><br>**Bit4 – bit7 for PSlave only**<br>Bit4 = 1 Monitoring continuously active.<br>(otherwise switch ON/OFF by PLC)<br>Bit Spare<br>Bit Spare<br>Bit Display active protection in DBx DBB66 bit0. | |

| 63544<br><br>MD number | CC_COLLISION_WIN<br><br>Collision protection window | |
|---|---|---|
| Default setting: 1.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after RESET | Protection level: | Unit: – |
| Data type: DOUBLE | | |
| Meaning: | Minimum clearance between this (PSlave) axis and the programmed PMaster axis. The monitoring function cannot be activated if setting value is 0. Only the value set for the PSlave is applied. | |

| | | |
|---|---|---|
| **63545**<br><br>MD number | **CC_OFFSET_MASTER** | |
| Default setting: 0.0 | Min. input limit: – | Max. input limit: – |
| Changes effective after RESET | Protection level: | Unit: – |
| Data type: INT | | |
| Meaning: | Zero point offset between PSlave and PMaster.<br>Only the value for the PSlave axis is applied. | |

■

# Notes

# Signal Descriptions 5

## 5.1 Axis-specific VDI OUT signals

| DB31 - 61<br>**DBX66.0**<br>Data block | **Activate monitor** | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: | Signal(s) valid from SW: 5.1 |
| Signal state 1 | Monitoring is active.<br>This display must be activated in MD 65543: CC_PROTECT_OPTIONS for the PSlave axis.<br>**Note**: Conflicts may occur in connection with customer-specific compile cycles. | |
| Signal state 0 | Monitor is not active. | |

| DB31 - 61<br>**DBX97.0**<br>Data block | **Axis is a slave axis** | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: | Signal(s) valid from SW: 5.1 |
| Signal state 1 | Axis is a CC_Slave axis.<br>The associated CC_Master axis can be found in the machine data. | |
| Signal state 0 | Axis is not a CC_Slave axis. | |

| DB31 - 61<br>**DBX97.1**<br>Data block | **Activate coupling** | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: | Signal(s) valid from SW: 5.1 |
| Signal state 1 | Coupling active | |
| Signal state 0 | Coupling not active | |
| Signal irrelevant for ...... | | |
| Application | Displayed only for the CC_Slave axis. | |

| DB31 - 61<br>**DBX97.2**<br>Data block | **Activate mirroring** | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: | Signal(s) valid from SW: 5.1 |
| Signal state 1 | Mirroring active (1:–1) | |
| Signal state 0 | 1:1 coupling active | |
| Signal irrelevant for ...... | Relevant only if coupling is active (DBB97.1 = 1) | |
| Application | Displayed only for the CC_Slave axis. | |

| DB31 - 61<br>**DBX97.3**<br>Data block | **Offset after point of activation** | |
|---|---|---|
| Edge evaluation: yes | Signal(s) updated: | Signal(s) valid from SW: 5.1 |
| Signal state 1 | New offset after point of activation<br><br>This bit is set to 1 if a particular event (SW/HW limit switch on CC_Slave axis) causes a change in the offset between CC_Master and CC_Slave which was stored when the coupling was activated. | |
| Signal state 0 | No new offset since activation | |
| Signal irrelevant for ...... | The bit is not set in the RESET phase. | |
| Application | Displayed only for the CC_Slave axis. | |
| Further references | | |

## 5.2 Axis-specific VDI IN signals

| DB31 - 61<br>**DBX24.2**<br>Data block | **Deactivate or disable coupling** | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: | Signal(s) valid from SW: 5.1 |
| Signal state 1 | An active coupling is not deactivated until the relevant axes are stationary. If CC_COPON is programmed for this axis, **no** error message is generated. | |
| Signal state 0 | Coupling may be activated | |
| Signal irrelevant for ...... | | |
| Application | Evaluated only on the CC_Slave axis. | |

| DB31 - 61<br>**DBX24.3**<br>Data block | **Switch on collision protection** | |
|---|---|---|
| Edge evaluation: yes | Signal(s) updated: | Signal(s) valid from SW: 5.1 |
| Signal state 1 | Collision protection ON | |
| Signal state 0 | Collision protection OFF | |
| Signal irrelevant for ...... | This signal is processed only if collision protection is not activated in a machine data (MD 65543: CC_PROTECT_OPTIONS). | |
| Application | Evaluated only on the PSlave axis. | |

■

# Examples 6

## 6.1 General start-up of a compile cycle function

**Note**

With SW 6.4, the compile cycles are supplied as loadable modules. The general procedure for installing such compile cycles can be found in TE0. You will find the specific extensions of this compile cycle from Section 6.2 onwards.

**Requirement**

The MMC software version must be 3.5 or higher.

**Saving SRAM contents**

As the first step in installing a compile cycle function, the original card inserted in the NCU must be replaced by the technology card.
This measure is identical to the procedure followed for upgrading the NCU to a later software version and likewise requires the static (battery-backed) control system memory to be erased. When this is done, all data stored in the SRAM are lost and they must therefore be saved beforehand.

Please proceed as follows:

1. Enter the machine manufacturer password.
2. Switch to the “Services” operating area.
3. Press the “Series start-up” softkey.
4. Select “NC” and “PLC” as the areas to be saved and enter a name of your choice for the archive file to be created on the hard disk. Finish by pressing the RETURN key.
5. If the control system contains machine-specific compensation data, then these must be saved in a separate archive file:
   - Press software “Data out”
   - Select from “NC active data” menu:
     – “Measuring system compensations”
     – “Sag/angularity comp.”
     – “Quadrant error compensation”.
   - Save these data by selecting softkey “Archive...”.
   - Enter another file name for a second archive file.

These archive files will enable you to restore the original status if required.

**References:** For a detailed description, please refer to the Manufacturer/Service Documentation “SINUMERIK 840D/SIMODRIVE 611D Installation and Start-Up Guide”.

**Insert the PC card**

- Switch off control system
- Insert the PC card with the new firmware (technology card) in the PCMCIA slot of the NCU
- Then proceed as follows:
  1. Turn switch S3 on the front panel of the NCU to 1
  2. Switch the control system back on again.
  3. When the system powers up, the firmware is copied from the PC card into the NCU memory.
  4. Wait until number “6” appears on the NCU digital display (after approximately 1 minute).
  5. Turn switch S3 back to zero.

---

**Note**

If number “6” does not appear, then the following errors might be the cause:
– Incorrect PC card (e.g. card for NCU2 in NCU3 hardware)
– Card hardware defective

---

**Copy back SRAM contents**

To copy the saved data back into the control system, proceed as described in Section 12.2 (series start-up). Please read all information provided by the manufacturer about new software versions.

- Enter the machine manufacturer password.
- Select “Data In” and “Archive...”.
- Load the archive with the backed up compensation data (if applicable).

## 6.2 Update of NCKOEM_CC_0013_01.02.00

- Extended functionality CC_COPONM and collision protection.
- Relocated machine data numbers, alarm numbers, VDI bytes and bits and new compile cycle no. (function can also be supplied on a technology card).

■

# Data Fields, Lists

7

## 7.1 Machine data

| Number | Identifier | Name | Ref. |
|---|---|---|---|
| **General ($MN_ ...)** | | | |
| 19600 | N_CC_EVENT_MASK[n] | Enable CC events for possible CC applications | |
| **Axis/spindle-specific ($MA_ ...)** | | | |
| 28090 | NUM_CC_BLOCK_ELEMENTS | Number of block elements for compile cycles. | |
| 28100 | NUM_CC_BLOCK_USER_MEM | Total size of usable block memory for compile cycles | |
| 63540 | CC_MASTER_AXIS | Specifies the CC_Master axis assigned to a CC_Slave axis | |
| 63541 | CC_POSITION_TOL | Monitoring window | |
| 63542 | CC_PROTEC_MASTER | Specifies the PMaster axis assigned to a PSlave axis. | |
| 63543 | CC_PROTEC_OPTIONS | | |
| 63544 | CC_COLLISION_WIN | Collision protection window | |
| 63545 | CC_OFFSET_MASTER | Zero point offset between PSlave and PMaster | |

## 7.2 Alarms

Add an entry for the alarm text files for the described function in the [TextFiles] section of the C:\OEM\MBDDE.INI file:

CZYK=C:\OEM\TF_

If file C:\OEM\MBDDE.INI does not exist, it must be set up, although only section [Text Files] is required.

Create language-specific text files
TF_xx.COM in directory

C:\OEM\

xx stands for the language code, e.g. GR for German and UK for English.

Enter the following alarm texts there:

in TF_GR.COM:

075050 0 0 “Channel %1, incorrect MD configuration. Error no. %2”

075051 0 0 “Channel %1 CC_COPON CC_COPOFF error no. %2”

075060 0 0 “Channel %1 tolerance window exceeded on axis %2”

075061 0 0 “Channel %1 coupling active on axis %2“

075062 0 0 “Channel %1 Channel %1 axes not at standstill axis %2”

075070 0 0 “Channel %1 incorrect machine data for collision protection”

075071 0 0 “Channel %1 collision monitoring axis %2”

**Alarm text** The complete alarm description is as follows:

**75050** **Channel %1 incorrect MD configuration. Error No. %2**

Explanation: Incorrect configuration in MD $MA_CC_MASTER_AXIS

%2 = 2 This or the CC_Master axis is a spindle.
%2 = 4 No coupling betw. rotary and linear axes.
%2 = 8 Axes must not be exchange axes.

Reaction:

Remedy: Check machine data.

Reset criterion: RESET

**75051** **Channel %1 CC_COPON CC_COPOFF error no. %2**

Explanation: Error in the interpretation of CC_COPON or CC_COPOFF

%2 = 1 Wrong argument programmed
%2 = 10 An axis which is not involved in a defined coupling has been programmed in CC_COPON(x).
%2 = 20 Too many arguments
%2 = 100 Internal error
%2 = 200 Internal error

Reaction: Interpreter stop

Remedy: Correct parts program.

Reset criterion: RESET

**75060** **Channel %1 tolerance window exceeded on axis %2**

Explanation: The actual value difference between the CC_Slave axis %2 and its CC_Master axis is outside the configured tolerance window.

Reaction: Axes brake along braking ramp.

Remedy: Check configured tolerance window.
Compare dynamic response settings of coupled axes.
Check mechanical components of axes.

Reset criterion: RESET

**75061** **Channel %1 coupling active axis %2**

Explanation: Machine data MD 63000: CC_MASTER_AXIS has been changed when the coupling was active.

Reaction: Axes brake along braking ramp.

| | |
|---|---|
| Remedy | Reset machine data to its old value, switch off the coupling and then enter the new value. |
| Reset criterion | RESET |

| | |
|---|---|
| **75062** | **Channel %1 axes not at standstill axis %2** |
| Explanation | The CC_Master and/or CC_Slave axis(es) were not at standstill when the coupling was switched on. |
| Reaction | Coupling cannot be activated. |
| Remedy | Input G601 for path axes or enter a STOPRE before the CC_COPON command. |
| Reset criterion | RESET |

| | |
|---|---|
| **75070** | **Channel %1 incorrect machine data for collision protection %2** |
| Explanation | Incorrect machine data for collision protection. |
| Reaction | Interpreter stop |
| Remedy | Correct machine data.<br>The axes must be either both rotary axes or both linear axes! |
| Reset criterion | RESET |

| | |
|---|---|
| **75071** | **Channel %1 collision monitoring axis %2** |
| Explanation | Collision monitor has responded. |
| Reaction | Axes brake at maximum acceleration or at a 20% higher acceleration rate. |
| Remedy | |
| Reset criterion | RESET |

■

## Notes

# SINUMERIK 840D/840Di/810D
# Description of Functions Special Functions (Part 3)

# Retrace Support (TE7)

■

# 1 Brief Description

Flat bed cutting processes (e.g. laser, oxygen or water jet cutting) require a solution which enables the machine operator to return to the so-called program continuation point (*damage point*) after an interruption in machining (e.g. cutting process failure) so as to remachine the workpiece from that point, without the need for a block-by-block knowledge of the parts program.

The “Retrace Support” is practical if the machine operator does not notice a failure until several blocks after the interruption. In other words, by the time the machine operator notices the machining interruption (*damage*), the head has already moved on and must be reversed back along the contour again. When the operator notices this type of disturbance, he stops the machining operation with NC STOP. If he now hits the “Reverse/Forward” button, the head reverses along the original contour the next time he presses NC START. Once the head has reached the damage point, the operator presses NC STOP. The head then remains stationary at its current position. If he now presses the “Start program rerun” button, then a block search is executed in the original program for the “continuation block” (at the current head position). The operator now needs to press NC START twice, as for a standard block search. Machining is then resumed at the damage point.

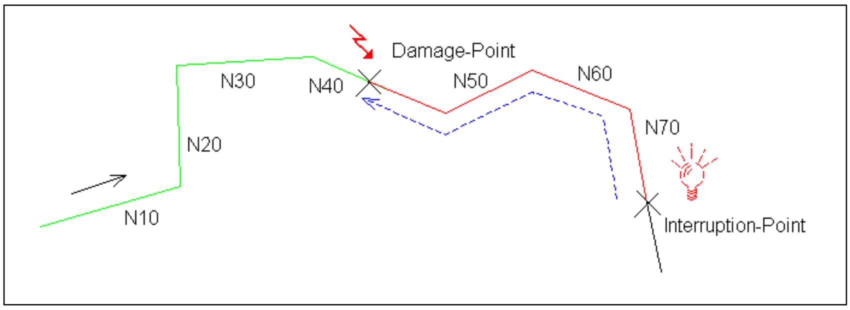



Fig. 1-1 Sequence of an operation requiring retrace and program rerun

**Example**

A parts program with blocks N10 to N70 is executed. The machining quality in block N40 becomes so poor that the workpiece must be remachined. The operator notices the problem when block N70 is reached and stops the machining operation. He now wants to reverse the head from the interruption point (at N70) along the original contour to the damage point (at N40) and continue machining there with all the supplementary conditions defined in the original program (synchronized actions, M functions, etc.).

---

**Note**

With SW 6.4, compile cycles TE1 – TE8 are no longer supplied on PC cards. They are available as loadable software blocks. Installation instructions are provided in the general description in TE0.

---

■

# Detailed Description 2

## 2.1 General

The “Retrace Support – function Retrace Support” is implemented using a compile cycle. This cycle is available on a standard technology card with SW 5.3 and higher under the name “Retrace/Continue Machining Support” (abbreviated to RESU).

### General principle

The compile cycle has an internal circular buffer containing geometric information about traversed blocks as well as data about the subprogram call level. If the user wishes to return the head along the contour, the stored geometric information is written to a new parts program (cc_resu.mpf). This parts program is automatically selected internally. cc_resu.mpf is executed when NC START is next actuated. Cc_resu.mpf contains the geometric information in the reverse order as compared to the original program, allowing the user to return along the original contour.

It is also possible to travel back and forth repeatedly along the original contour. parts program cc_resu.mpf is overwritten each time. The contents of the internal circular buffer remain unaltered.

If the machine operator wishes to rerun the program, the original parts program is selected internally and a block search executed for the “continuation block”. The “damage” point is not restricted to block limits, but can also be located within a block. The program continuation point corresponds the current position of the machining head, i.e. the machine operator moves the head backwards along the contour until it reaches exactly the point at which the program must be continued (rerun). The machine operator then starts the continue machining process.

---

**Note**

The head can only reverse along straight lines (G0, G1) and circles (G2, G3, CIP) as such. All other programmed movements (e.g. splines, ADIS roundings) are traversed as straight lines.

“Retrace Support” is implemented for 2D applications only, i.e. only movements of the first two geometry axes (typically designated X and Y) are logged. Consequently, only an XY path can be retraced.

---

## 2.2 Logical sequence of operations

If the machine operator decides that a particular cut section along the contour needs to be repeated, he stops the machine by NC STOP. He then hits the "Reverse/Forward" button. Parts program cc_resu.mpf is then generated and selected internally. When the operator next presses NC START, the head reverses along the contour according to the selected cc_resu.mpf program. If the machine operator hits NC STOP to halt the reverse movement, he has various options for continuing the machining operation:

- He can hit NC START to continue reversing along the contour.
- He can start the "continue machining" process at the current position and thus continue machining according to the original program.
- He can press the "Reverse/Forward" button again. The reversing program cc_resu.mpf is then overwritten. If he then starts cc_resu.mpf, the head traverses forwards along the original contour from its current position. The operator can change the direction of travel during reverse positioning in this way. He can do this repeatedly until the head reaches exactly the position at which he wishes to continue the program.

Once the head has reached the program continuation point, the operator hits NC STOP and then button "Start program rerun". The head then remains stationary at its current position. An internal block search in the original program for the continuation block (at current position) is then performed. The operator now needs to press NC START twice, as for a standard block search. Machining is then continued at the "program continuation" point.

---

**Note**

Retrace Support is not possible if you do not travel in reverse beforehand, i.e. you must be in Retrace mode in order to continue machining (see Chapter 7).

---

For detailed information about the sequence of operations, please see Section 3.1 "Requirements of the PLC user program" and Section 2.6 "Chronological sequence".

### 2.2.1 Actuation of NC RESET

An NC RESET deactivates the Retrace functionality.

Continue machining and reversal along contour (retrace) are possible only in the NC STOP state.

As a result, the following applies:

- If the machine operator activates a RESET while the original program is running, he cannot reverse the head along the contour. In this case, the operator only has the option of restarting the original program again.
- If the machine operator activates a RESET while the head is reversing, it cannot reverse any further (reversal in the opposite direction) or remachine the contour. The original program is selected every time RESET is activated in Retrace mode. As a result, the operator can only restart the original program and start the machining operation from the beginning, or use a standard block search.

### 2.2.2 Response to power OFF

Like NC RESET, a power OFF deactivates the Retrace functionality. If the power is switched off while the head is reversing (the reversal program cc_resu.mpf is selected), the original program is not selected after the next power ON.

By default, no parts program is selected ("/MPF0") after the next power ON, because the reversal program cc_resu.mpf is stored in the dynamic NC program memory (DRAM) and is thus no longer available after the next power ON (see also Section 3.2).

If the reversal program cc_resu.mpf is stored in buffered memory (SRAM, see Section ), it is still available and selected after power OFF –> power ON. The internal circular buffer is erased, however. The control system is not in Retrace mode. The selected reversal program cc_resu.mpf is thus treated like an original program.

## 2.3 Programming

The Retrace/Continue Machining function is activated and deactivated via NC language command CC_PREPRE() as a function of the passed parameters. If the function is activated without language command CC_PREPRE(), it is not possible to reverse the head while the program is running. PREPRE stands for "prepare retrace".

**Syntax**

The CC_PREPRE(n) command is implemented as a procedure call, i.e. it must be programmed in a separate NC block.

**CC_PREPRE(1)**

Starts storage of information. The information required from all executed blocks is stored from this command onwards. This is essentially the geometric information of geometry axes 1 and 2 of the first channel. Geometry axes 1 and 2 refer to channel-specific machine data $MC_AXCONF_GEOAX_ASSIGN_TAB[0..1] or, if a transformation is active, to machine data $MC_TRAFO_GEOAX_ASSIGN_TAB_n[0..1]. The information is written to an internal circular buffer of the compile cycle.

**CC_PREPRE(0)**

Interrupts storage of information. This command can be used to parenthesize passages in the parts program that are not relevant for reversal.

**CC_PREPRE(–1)**

Deactivates the function and empties the buffer. The stored information is not written to the reversal program cc_resu.mpf. CC_PREPRE(–1) has the same effect for the Retrace functionality as an NC RESET or M30 in the parts program.

**Error messages**

Call parameters other than those described above are rejected with alarm 75601. Standard alarm 12340 is output if more than one parameter is specified in the CC_PREPRE(n) call. If the "Retrace/Continue Machining Support" function is not available on the PC card or has not been activated via machine data, the activation command is rejected in the same way as other unknown commands with standard alarm 12550.

## 2.4 Retraceable area

The area within which a contour can be retraced is restricted by:

1. The geometric position in the main program at which CC_PREPRE(1) is programmed for the first time (see Section 2.3).
2. The position at which the retrace was started (actuation of button “Reverse/Forward”) or the geometric position in the main program at which CC_PREPRE(0) is last programmed (see Section 2.3).

---

**Note**

The internal circular buffer is re-initialized after every continue machining operation. Storage of travel information commences again from this point onwards. As a result, the operator can only ever reverse the head as far as the last program continuation point.

---

## 2.5 Adaptation options for machine manufacturers

**cc_resu.mpf**

Machine manufacturers can incorporate machine-specific adaptations at the beginning and end of the reversal sequence. Subprogram cc_resu_ini.spf is called at the beginning of reversal program cc_resu.mpf and subprogram cc_resu_end.spf at the end. Machine-specific or machining-specific actions can be implemented in both of these subprograms.

Both subprograms must be available and loaded. If they are not available, they are generated by the control during booting in the user cycles directory (_N_CUS_DIR) with the defaults specified below. If the subroutines already exist when the control boots, they are not overwritten. The machine manufacturer can therefore alter the two subprograms and, as such, is responsible for ensuring that they are correctly programmed and contain the necessary G functions.

**cc_resu_ini.spf**

cc_resu_ini.spf must contain at least the following:

```
PROC CC_RESU_INI
G71          ;input system metric
G90          ;position programming, absolute
G500         ;disable all active zero offsets
T0           ;disable all active tool offsets
G40          ;disable tool radius compensation
Fxxxx        ;programmed feedrate (default: F200)
M17
```

---

**Note**

Only the X and Y positions of the contour are taken into consideration since this compile cycle is designed for the field of 2D applications. A programmed Z axis relevant to the contour is ignored while the head is reversing. If a motion towards the Z axis needs to be executed as the head is reversing, the Z axis must approach a safe position first. This position for Z can be programmed in subroutine cc_resu_ini.spf.

---

**cc_resu_end.spf**

cc_resu_end.spf must contain at least the following:

PROC CC_RESU_END

M17

In its default state, cc_resu_end.spf contains an "M0" block before block "M17". As a result, every reverse movement is stopped at the end of cc_resu_end.spf without the system switching to the RESET state. The RESET state would preclude the possibility of reversing further along the contour or rerunning the program. Both these actions are possible only in the STOP state (for further information, see Subsection 2.2.1).

**cc_resu_bs_asub.spf**

When the NC boots, an ASUB named cc_resu_asub.spf is created in the user cycles directory. This ASUB is triggered at the end of the Retrace, or to be more precise, at the end of the final block search, after the first NC START has been executed and signal DB21.DBX32.6 "Last action block active" is active (see Section 2.6).

This ASUB is responsible for continuing machining at the current position in the block on account of the RMN REPOS type ('N' stands for "next"). This ASUB can also be modified by the machine manufacturer if necessary. Additional blocks should only be inserted in front of the "RMN" block.

The default version of cc_resu_bs_asub.spf contains the following blocks:

PROC CC_RESU_BS_ASUB SAVE

RMN

REPOSA

**cc_resu_asub.spf**

When the NC boots, a further ASUB is created in the user cycles directory. The name of the ASUB is cc_resu_asub.spf. For internal reasons, this ASUB is executed any time the machine operator wants to reverse the head, i.e. every time NC STOP is actuated in the original program and the "Backwards/ Forwards" button is pressed. This ASUB should not be changed, because it is required purely for internal reasons.

The default version of cc_resu_asub.spf contains the following blocks:

```
PROC CC_RESU_ASUB

; siemens system asub – do not change

G4 F0.001

M0

REPOSA
```

The RESU subprogams cc_resu_ini.spf, cc_resu_end.spf, cc_resu_asub.spf and cc_resu_bs_asub.spf are stored by default in the user cycles directory (_N_CUS_DIR). However, it is also possible to specify in machine data that the RESU subprograms are created in the machine manufacturer cycles directory (_N_CMA_DIR). See Section 3.3 for more information.

---

**Note**

Subprograms cc_resu_ini.spf, cc_resu_end.spf, cc_resu_bs_asub.spf, cc_resu_asub.spf and their subprograms may not contain block numbers. The block numbers in the automatically generated reversing program cc_resu.mpf have a specific internal meaning. In order to prevent blocks from the specified subprograms from being interpreted incorrectly, they may not contain block numbers (see also Subsection 6.3.2, Alarm 75604).

Subroutines cc_resu_ini.spf, cc_resu_end.spf and cc_resu_asub.spf must not contain a CC_PREPRE(n) command. CC_PREPRE(n) may only be programmed in the original program or in ASUB cc_resu_bs_asub.spf.

---

## 2.6 Chronological sequence

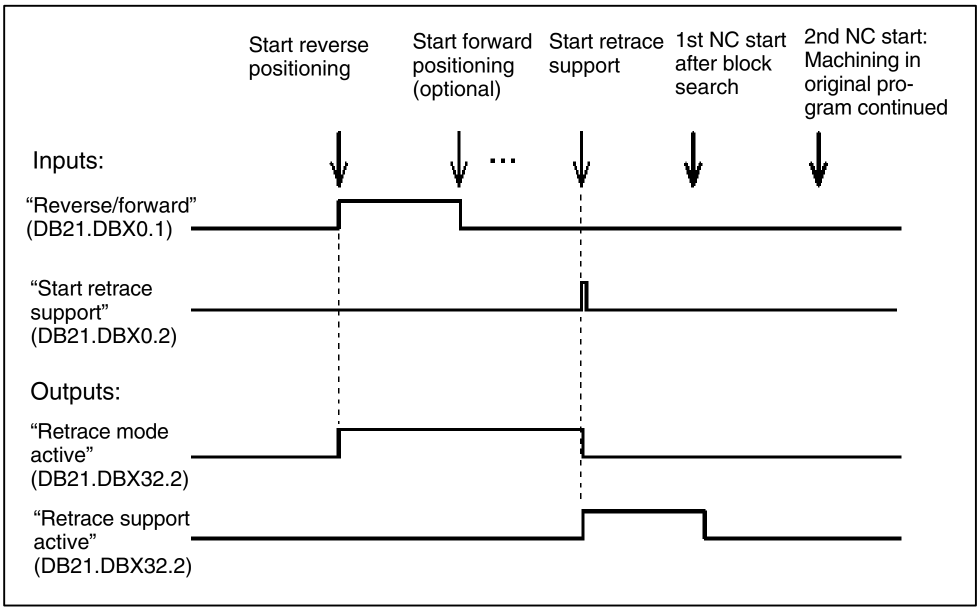



Fig. 2-1 Signal time chart, total retrace/continue machining process

The diagram shows an example of a possible retrace/continue machining process sequence with user interaction and input/output signals: Reverse, travel forwards, continue machining.

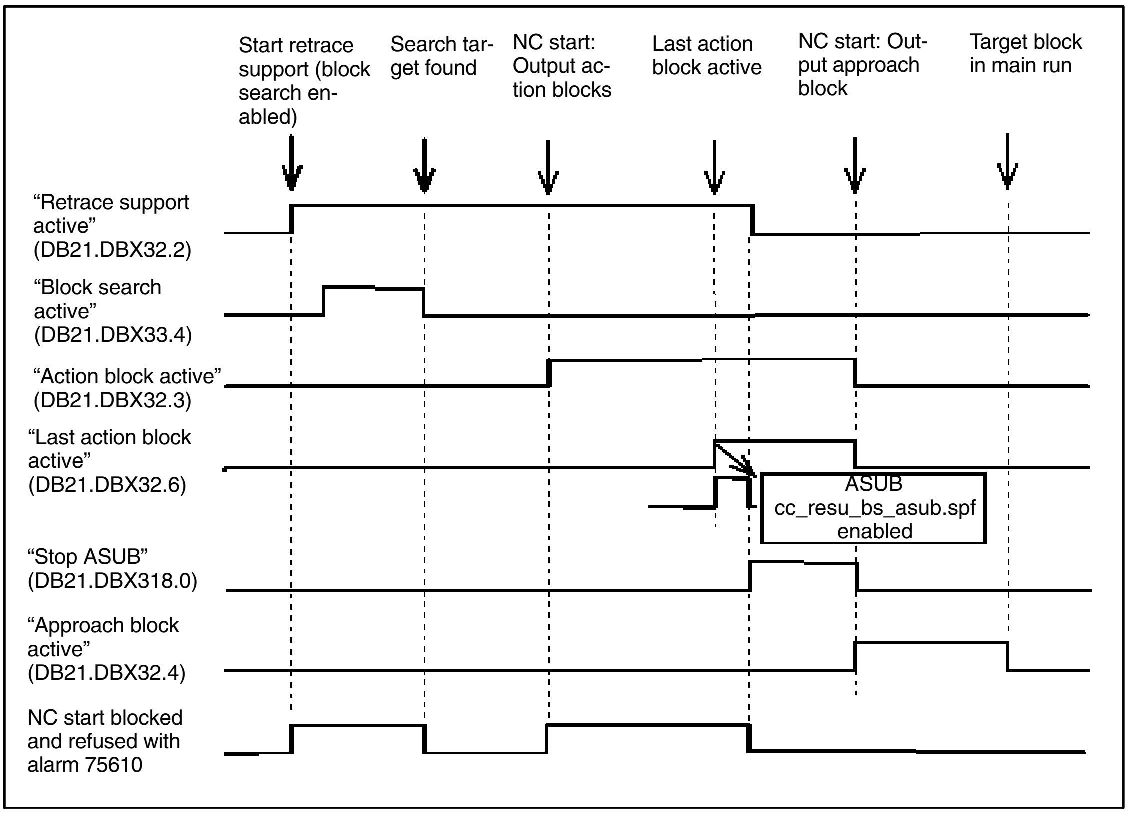



Fig. 2-2 Chronological sequence during continue machining

The continue machining process is examined in greater detail here: For example, you can see when the cc_resu_bs_asub.spf ASUB was triggered and when NC START was blocked explicitly by the application. It is also possible to see when an NC START is allowed:

- The first NC START after block search should be executed only when signal DB21.DBX33.4 ("Block search active") is no longer active (LOW).
- The second NC START after block search should be executed only when signal DB21.DBX318.0 ("ASUB stopped") is active (HIGH).

## 2.7 Block search from last main block

In the case of very long parts programs (containing several hundred or several thousand blocks) the retrace procedure can take a relatively long time (up to several minutes). This happens in particular if you want to continue machining near the end of the original program. The delay is caused by the block search triggered during the continue machining process.

If such wait times cause significant delays in the production process, you can use “block search from last main block”. This block search type, which is only available for the “Retrace Support” function, is intended to accelerate the continue machining process. It is associated with certain secondary conditions and restrictions, however.

**Operating principle**

An internal “block search without calculation” is executed initally at the last main block before the interruption block, followed by a “block search with contour calculation” at the interruption block. For this reason, it is vital that the original program contains main blocks.

**Main blocks**

In order to use the block search from last main block, the original parts program must contain main blocks (typically in the main program). These main blocks must always contain all the necessary modal G functions and an absolute position for all axes used. The main block number is designated with “:” (colon) instead of the letter “N” for all other block numbers (subblocks). If no block number is programmed, the block is not interpreted as a main block.

Such a main block might look like this:

:10 G1 G17 G90 X10 Y10 Z5 F1000 G71 G55 G64 G41

**Activation**

The block search from last main block is activated with bit 0 of MD $MC_RESU_SPECIAL_FEATURE_MASK_2 (see Subsection 4.2.1).

Bit 0 = 0

Retrace Support uses a standard block search (block search with contour calculation). Default setting.

Bit 1 = 0

Retrace Support uses block search from last main block.

## Restrictions

**Note**

Block search from last main block is only possible if a main block was logged during execution of the original program. The original program must therefore contain at least one main block after CC_PREPRE(1).

All of the supplementary conditions defined in the original program (synchronized actions, M functions, etc.), which appear before the last main block in the program, are not active after the retrace process. This behavior arises from the fact that the block search without calculation has not collected any information up to the last main block. All necessary supplementary conditions (synchronized actions, M functions, etc.) must therefore be activated/defined in the RESU ASUB cc_resu_bs_asub.spf. This ASUB is called at the end of the second block search just before program execution is resumed in the original program (see Section 2.5 and Section 2.6).

The activation command CC_PREPRE(1) (see Section 2.3) should also appear in the cc_resu_bs_asub.spf ASUB when using the block search from last main block, since this command is not collected by the block search without calculation.

**cc_resu_bs_asub.spf**

cc_resu_bs_asub.spf should contain at least the following when using the block search from last main block:

```
PROC CC_RESU_BS_ASUB SAVE
CC_PREPRE(1)
RMN
REPOSA
```

**Note**

It is recommended to use the block search from last main block only when absolutely necessary. In case of doubt, the normal block search should always be used (MD $MC_RESU_SPECIAL_FEATURE_MASK_2 bit 0 = 0).

■

# Notes

# Supplementary Conditions 3

## 3.1 Requirements of the PLC program

The following signals at the VDI interface must be evaluated or supplied by the PLC user program:

Signals from PLC to NCK channel:

- DB2x.DBX0.1 “Reverse/Forward”
- DB2x.DBX0.2 “Start Continue Machining”

Signals from NCK channel to PLC:

- DB2x.DBX32.1 “Retrace Mode Active”
- DB2x.DBX32.2 “Continue Machining Active”

### 3.1.1 Meaning of the individual signals

**“Reverse/ Forward”**

Reverse travel applies when the “Reverse/Forward” signal is HIGH. Forward travel applies when the “Reverse/Forward” signal is LOW.

The “Reverse/Forward” signal is only evaluated when the control is in the NC STOP state.

**Example sequence:**

1. Stop original program execution (NC STOP)
2. Press “Reverse/Forward” key (DB2x.DBX0.1 = HIGH)
3. –> Reverse travel is initiated
4. Start reverse travel (NC START)
5. Stop reverse travel (NC STOP)
6. Press “Reverse/Forward” key again (DB2x.DBX0.1 = LOW)
7. –> Reverse travel in forward direction is initiated
8. Start forward travel (NC START)
9. Stop forward travel (NC STOP)
10. Press “Reverse/Forward” key (DB2x.DBX0.1 = HIGH)
11. Start reverse travel (NC START)
12. ... (see continuation below)

**"Start Continue Machining"**

A Continue Machining operation is triggered when the "Start Continue Machining" signal is HIGH. Reverse travel is required beforehand. The "Start Continue Machining" signal is only evaluated when the control is in the NC STOP state.

**Example sequence (continued):**

13. Stop reverse travel (NC STOP)
14. Press "Start Continue Machining" key (DB2x.DBX0.2 = HIGH)
15. –> Continue Machining is initiated
16. –> A block search is triggered implicitly by the Continue Machining process.
17. Press NC START: Action blocks are output.
18. Press NC START again: Program execution is resumed at the current position in the original program.

**"Retrace Mode Active"**

The "Retrace Mode Active" signal is active as long as the control is in Retrace mode. This is the case from initial activation of the "Reverse/Forward" signal until activation of the "Start Continue Machining" signal. In the above example, this would mean: from step 2 to step 14.

**"Continue Machining Active"**

The "Continue Machining Active" signal is set when the HIGH state is detected at the "Start Continue Machining" signal. The "Continue machining active" signal is reset when the program has been continued,

The Continue Machining process is completed after the first NC START at the end of the final block search in the original program (–> action blocks are output, the last action block is active). In the above example, this would mean: from step 14 to step 17.

**Graphical representation**

A graphical representation of the individual signal sequences is provided in Section 2.6.

**User keys**

The "Reverse/Forward" user key can have two states:

| | |
|---|---|
| LOW: | Forward travel |
| HIGH: | Backward travel |

When the user key is pressed, the "Reverse/Forward" signal changes state. The current state of the user key (of this signal) should be indicated to the machine operator. The operator can then see the current status of the function.

Only one state is relevant for the "Start Continue Machining" key:

| | |
|---|---|
| HIGH: | Continue Machining is initiated |

Pressing of the "Start Continue Machining" key should trigger a HIGH edge at the PLC signal "Start Continue Machining". The "Start Continue Machining" signal should be reset again as soon as the "Continue Machining Active" response signal is activated.

### 3.1.2 Minimum requirements for the PLC user program

The PLC user program must ensure the following:

- When the "Continue Machining Active" signal is active (signal from NCK channel to PLC), both signals "Reverse/Forward" and "Start Continue Machining" should be returned to their default state (LOW).
- Whenever the machine operator initiates a RESET (channel or mode group RESET), both signals ("Reverse/Forward" and "Start Continue Machining") must be reset.
- For safety, the "Reverse/Forward" signal should be reset after the "Start Continue Machining" signal is set. The "Reverse/Forward" signal should not be reset at the same time as the "Start Continue Machining" signal is set, but delayed by at least one cycle.
- For safety, the "Reverse/Forward" signal should be reset after the "Start Continue Machining" signal is set.

**Sample program**

At least the following PLC program extract should appear in every PLC user program on machines which use RESU:

```
U     DB21.DBX    32.2
R     DB21.DBX    0.2
R     DB21.DBX    0.1
O     DB11.DBX    0.7
O     DB21.DBX    7.7
R     DB21.DBX    0.1
R     DB21.DBX    0.2
U     DB21.DBX    0.2
R     DB21.DBX    0.1
U     DB21.DBX    0.1
R     DB21.DBX    0.2
```

## 3.2 NC program memory for cc_resu.mpf

The standard version of the reverse travel program cc_resu.mpf does not need a buffered NC program memory, i.e. SRAM (static memory). It is created by default in DRAM (dynamic NC program memory). The reverse travel program cc_resu.mpf is generated on each reversing operation, and so it is not necessary to back up this parts program across a power ON.

The type of memory for the reverse travel program can be selected explicitly in machine data $MC_RESU_SPECIAL_FEATURE_MASK, bit 1 (see also Subsection 4.2.1):

**Bit 1 = 0**

cc_resu.mpf is stored in DRAM (default setting)

**Bit 1 = 1**

cc_resu.mpf is stored in SRAM (not recommended)

Regardless of which memory is used to store reverse travel program cc_resu.mpf, the remaining NC program memory may, in certain circumstances, be exhausted while cc_resu.mpf is being generated. In this case, the reverse travel program is generated anyway. However, it does not contain the complete reversing information. If this happens, alarm 75608 “NC memory limit reached” is displayed (see Subsection 6.3.2). When buffered memory (SRAM) is used for cc_resu.mpf system alarm 6500 (see Subsection 6.3.1) appears concurrently with alarm 75608. If alarm 75608 occurs, you should increase the available NC program memory:

- Cc_resu.mpf is stored in DRAM:

The available memory is expanded by increasing machine data $MN_MM_DRAM_FILE_MEM_SIZE.

- Cc_resu.mpf is stored in SRAM:
  The available memory in the buffered NC program memory (SRAM) can be expanded either by unloading parts programs which are not required or by increasing the overall NC program memory (machine data $MN_MM_USER_MEM_BUFFERED).

If the reversing information is insufficient under these conditions, you should reverse as far as possible and then change the Retrace direction twice. You can now move back in the original traversing direction with additional reversing information. In other words, the complete reversing information remains stored in the internal circular buffer. This reversing information is still available even after the direction of travel has changed several times (see Section 2.2).

## 3.3 RESU subprogams in user or machine manufacturer cycle directory

The RESU subprogams cc_resu_ini.spf, cc_resu_end.spf, cc_resu_asub.spf and cc_resu_bs_asub.spf (see also Section 2.5) can be created either in the user (/_N_CUS_DIR) or machine manufacturer cycle directory (/_N_CMA_DIR). The behavior can be controlled via bits 2 and 3 of machine data $MC_RESU_SPECIAL_FEATURE_MASK (see Subsection 4.2.1).

If files cc_resu_ini.spf, cc_resu_end.spf, cc_resu_asub.spf and cc_resu_bs_asub.spf are not yet stored in the specified directory (see below), they are created there.

---

**Note**

The behavior set in machine data (bits 2 and 3 in MD $MC_RESU_SPECIAL_FEATURE_MASK) with reference to the storage location of RESU subprogams cc_resu_ini.spf, cc_resu_end.spf, cc_resu_asub.spf and cc_resu_bs_asub.spf is binding. The RESU subprogams can be changed or modified but must not be moved somewhere else.

---

### 3.3.1 User cycle directory

To create RESU subprogams cc_resu_ini.spf, cc_resu_end.spf, cc_resu_asub.spf and cc_resu_bs_asub.spf in the user cycle directory (/_N_CUS_DIR), **bit 2** of MD **$MC_RESU_SPECIAL_FEATURE_MASK** must be set equal to **zero**.

In this case, bit 3 in MD $MC_RESU_SPECIAL_FEATURE_MASK is ignored.

This is also the default setting of the function.

### 3.3.2 Machine manufacturer cycle directory

To create RESU subprogams cc_resu_ini.spf, cc_resu_end.spf, cc_resu_asub.spf and cc_resu_bs_asub.spf in the machine manufacturer cycle directory (/_N_CMA_DIR), **bit 2** of MD **$MC_RESU_SPECIAL_FEATURE_MASK** must be set equal to **one**.

If you want to store the RESU subprogams in the machine manufacturer cycle directory (/_N_CMA_DIR), please note the following for installation and start-up (initial and series machine start-up):

The first time the system is booted after a general reset, the RESU subprogams with the default contents (see Section 2.5) are created in the user cycle directory (/_N_CUS_DIR), because bit 2 of MD $MC_RESU_SPECIAL_FEATURE_MASK is set to zero by default. If bit 2 of MD $MC_RESU_SPECIAL_FEATURE_MASK is set later to one, the RESU subprograms are created again in the machine manufacturer cycle directory (/_N_CMA_DIR). The RESU subprograms which were created initially in the user cycle directory remain stored there and thus conflict with the RESU subprograms in the machine manufacturer cycle directory. In this case, it is vital to delete the existing RESU subprograms from the user cycle directory. This procedure can be automated via bit 3 of MD $MC_RESU_SPECIAL_FEATURE_MASK:

If **bit 3** of MD **$MC_RESU_SPECIAL_FEATURE_MASK** is equal to **one** and RESU subprogams (cc_resu_ini.spf, cc_resu_end.spf, cc_resu_asub.spf and cc_resu_bs_asub.spf) exist in the user cycle directory, these are automatically deleted the next time the control is booted.

This behavior is intended to support series machine start-up. If there were no means of deleting the RESU subprograms automatically from the user cycle directory on the control, the commissioning engineer would have to do this manually for each machine commissioned.

---

**Note**

When bit 3 of MD $MC_RESU_SPECIAL_FEATURE_MASK is set, **files** with the following names are **deleted** automatically **without prompting** on each power ON:

/_N_CUS_DIR/_N_CC_RESU_INI_SPF
/_N_CUS_DIR/_N_CC_RESU_END_SPF
/_N_CUS_DIR/_N_CC_RESU_ASUB_SPF
/_N_CUS_DIR/_N_CC_RESU_BS_ASUB_SPF

---

The following setting is recommended:

Set bit 2 and bit 3 of MD $MC_RESU_SPECIAL_FEATURE_MASK (hex: 0x0c). Please read the notice above.

## 3.4 Programming loops

The user has the option of programming loops (LOOP-ENDLOOP, FOR-ENDFOR, WHILE-ENDWHILE, REPEAT-UNTIL) and jumps (GOTO) in the parts program. Program reruns within structures of this type is only possible subject to certain conditions:

The program is always continued in the first loop pass.

## 3.5 Programming subroutines

Subroutines can be programmed. All information from executed subroutines is always stored in the internal circular buffer. The program can be continued even if the continuation block is located in a subroutine.

## 3.6 G641, G642, RND, TRC blocks

Blocks which contain RND or in which G641 (ADIS), G642 or a tool radius compensation (TRC) is operative are split internally by the NCK system into a number of short blocks. The additionally generated blocks contain curved contour sections. Like spline blocks, these “curved” blocks are reversed as linear blocks.

## 3.7 Ambiguity as regards full circles

With full circles the geometric block start is identical to the geometric block end. If the program must be continued at this point, it is always continued at the start of the block. In other words, the full circle is executed as the first block when machining is continued. The program cannot be continued at exactly the block end point on a full circle. This ambiguity as regards continuing the program at the block end point can be overcome by continuing the program just before the block end point on a full circle.

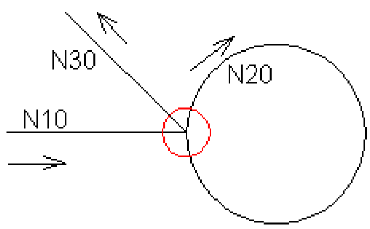



Fig. 3-1 Ambiguity as regards continuing a program on full circles

Since the block start and block end points on a full circle are identical, this position is ambiguous. If the program must be continued at this position, it is always continued at the start of the block. It cannot be continued exactly at the block end point.

## 3.8 Axes, channels, mode groups

Only the movement of the first two geometry axes is monitored. The first two geometry axes must be configured.

The “Retrace/Continue Machining Support” function can only be activated in the first mode group in the first channel.

## 3.9 Geo axis replacement

No geo axis replacement may be programmed in the original program while CC_PREPRE is active. CC_PREPRE is active if it is already programmed with 1 or 0. CC_PREPRE is not active if it is not programmed or if it has been programmed with –1.

## 3.10 Block search

If a block search causes the program to branch to a block positioned after the switch-on command CC_ PREPRE(1), then the programmed contour logging function is activated. That means the compile cycle saves the last NC command found and evaluates it before the first actual traversing block at the end of the block search. The compile cycle only logs real movements, i.e. the first block in the buffer is the one at the end of the block search, i.e. the “approach block”.

If several language commands CC_PREPRE(n) with different parameters are found during a search, the last programmed command will be operative after the search.

This mechanism works only in conjunction with block search variants “With calculation on contour” and “With calculation at block end”. With search variant “Without calculation” the programmed language command CC_PREPRE(n) is not detected and cannot therefore be evaluated.

## 3.11 Block numbers

It is not necessary to program unique block numbers for the "Retrace/Continue Machining Support" compile cycle function.

## 3.12 Transformations, compensations, tool compensation, frames

The following boundary conditions apply:

- Transformations:
  The function works independently of transformations, i.e. it can function when a transformation is active. The BCS coordinates of the first two geometry axes (X and Y) are stored. The transformation may need to be activated in cc_resu_ini.spf (see Section 2). The transformation must not be switched on or off while CC_PREPRE is active. In other words, the transformation must always be activated before CC_PREPRE.
- Compensations (CEC, temperature compensation, leadscrew pitch):
  The function works independently of compensations, i.e. it does not affect them.
- Tool compensation:
  The offset resulting from a tool compensation (e.g. tool radius compensation) is taken into account by the continue machining function In other words, the tool offset must be deactivated while continue machining is active (T0, G40, see Section 2) The tool offset is already added to the absolute axis positions in cc_resu.mpf. In certain situations, the generated reversal program may deviate slightly from the original program with tool radius compensation (e.g. DISC).
- Frames:
  The offset resulting from a frame compensation (e.g. zero offset) is taken into account by the continue machining function. In other words, the frame offset must be deactivated while continue machining is active (G500, see Section 2) The frame offset is already added to the absolute axis positions in cc_resu.mpf.

## 3.13 Compatibility with other functions

There are no known conflicts between the Retrace Support function and other compile cycle functions on the SINUMERIK 840D technology card.

■

# Notes

# Data Descriptions (MD, SD) 4

## 4.1 Machine data of standard system

The ”Retrace/Continue Machining Support” function is implemented as a compile cycle application. The following standard machine data must be set:

19600 $ON_CC_EVENT_MASK[9] = ’HFF’

activates compile cycle application 10. If this data is set to zero, the application is not executed at all. The system is then identical to the standard system.

19340 $ON_PROG_MASK = ’H04’

These are option data. Please observe the respective conditions associated with option data.

The following machine data are also required:

28090 $MC_MM_NUM_CC_BLOCK_ELEMENTS = 1

28100 $MC_MM_NUM_CC_BLOCK_USER_MEM = 2

If other compile cycles are also active (e.g. CLC), the above values must be applied additively.

11602 $MN_ASUB_START_MASK = ’H01’

11604 $MN_ASUB_START_PRIO_LEVEL = 1

28105 $MC_MM_NUM_CC_HEAP_MEM = x (see Section 5.3)

18351 $MN_MM_DRAM_FILE_MEM_SIZE = 100

# 4.2 Machine data for continue machining function

## 4.2.1 Channel-specific machine data

| 62571<br>MD number | $MC_RESU_RING_BUFFER_SIZE<br>Size of circular buffer | |
|---|---|---|
| Default setting: 1000 | Min. input limit: 10 | Max. input limit: 1000000 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | Applies from SW: | |
| Meaning: | The circular buffer contains the geometric block information from the original parts program. The size of the circular buffer corresponds to the number of blocks that can be logged. It is therefore directly proportional to the number of retraceable blocks. | |

| 62572<br>MD number | $MC_RESU_S$MC_RESU_SHARE_OF_CC_HEAP_MEM<br>Size of circular buffer | |
|---|---|---|
| Default setting: 100.0 | Min. input limit: 1.0 | Max. input limit: 100.0 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DOUBLE | Applies from SW: | |
| Meaning: | Machine data $MC_MM_NUM_CC_HEAP_MEM defines the amount of heap memory available for all the active compile cycles. Machine data $MC_RESU_SHARE_OF_CC_HEAP_MEM can be set to define the share of heap memory which will be available for the “Retrace/Continue Machining Support” function. For example, the data can be set to allocate 50% of the heap memory to the ”Retrace/Continue Machining Support” function. If no other compile cycles that require heap memory are active, then this machine data should always be set to 100.0%. | |

| 62573<br>MD number | $MC_RESU_INFO_SA_VAR_INDEX[1]<br>Indices of the synchronized action variables | |
|---|---|---|
| Default setting: –1 | Min. input limit: –1 | Max. input limit: 10000 |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | Applies from SW: 6.4 | |
| Meaning: | Reserved. Machine data may not be used. | |

| 62574<br>MD number | $MC_RESU_SPECIAL_FEATURE_MASK<br>Additional properties of the “Retrace/Continue Machining Support” function which can be enabled. Power ON – effective properties. | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 0F (hex) |
| Changes effective after power ON | Protection level: 2 / 7 | Unit: – |

| 62574<br>MD number | **$MC_RESU_SPECIAL_FEATURE_MASK**<br>Additional properties of the “Retrace/Continue Machining Support” function which can be enabled. Power ON – effective properties. |
|---|---|
| Data type: DWORD | Applies from SW: |
| Meaning:<br>Evaluation bit by bit | Bit 0: Reserved. May not be used.<br>Bit 1=0: Default setting.<br>The reverse travel program cc_resu.mpf is created in dynamic NC program memory (DRAM, see Section 3.2). Recommended setting.<br>Bit 1=1: The reverse travel program cc_resu.mpf is created in buffered NC program memory (SRAM, see Section 3.2).<br>Bit 2 = 0: Default setting.<br>The RESU subprogams cc_resu_ini.spf, cc_resu_end.spf, cc_resu_asub.spf and cc_resu_bs_asub.spf are created in the user cycle directory (/_N_CUS_DIR, see Subsection 3.3.1).<br>Bit 2=1: The RESU subprogams cc_resu_ini.spf, cc_resu_end.spf, cc_resu_asub.spf and cc_resu_bs_asub.spf are created in the machine manufacturer cycle directory (/_N_CMA_DIR, see Subsection 3.3.2). Recommended setting.<br>Bit 3 = 0: Default setting.<br>No effect. (see Subsection 3.3.1)<br>Bit 3=1: If files with the names cc_resu_ini.spf, cc_resu_end.spf, cc_resu_asub.spf or cc_resu_bs_asub.spf exist in the user cycle directory (/_N_CUS_DIR), they are deleted automatically without prompting each time the control is booted (see Subsection 3.3.2).<br>Recommended setting if bit 2 = 1. Bit 3 has no effect if bit 2 = 0. |

| 62575<br>MD number | **$MC_RESU_SPECIAL_FEATURE_MASK_2**<br>Additional properties of the “Retrace/Continue Machining Support” function which can be enabled. RESET – effective properties. | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 01 (hex) |
| Changes effective after RESET | Protection level: 2 / 7 | Unit: – |
| Data type: DWORD | Applies from SW: | |
| Meaning: | Bit 0=0: Default setting.<br>A standard block search is used for the Continue Machining process (see Section 2.7). Recommended setting.<br><br>Bit 0=1: The RESU block search type is used for the Continue Machining process (block search from last main block, see Section 2.7). | |

■

## Notes

# Start-Up 5

## 5.1 General start-up of a compile cycle function

---

**Note**

With SW 6.4, the compile cycles are supplied as loadable modules. The general procedure for installing such compile cycles can be found in TE0. The specific installation measures for this compile cycle can be found from Section 5.2 onwards.

---

The MMC software version (SW 5.1. or higher) must match the relevant NCK version. An NCK PCMCIA card containing the "Retrace/Continue Machining Support" function must be available.

**Saving SRAM contents**

As the first step in installing a compile cycle function, the original card inserted in the NCU must be replaced by the special card. This step is identical to the procedure followed for upgrading the NCU to a more recent software version and likewise requires the static (battery-backed) control system memory to be erased.

When this is done, all data stored in the SRAM are lost and they must therefore be saved beforehand. For a detailed description, please refer to the Manufacturer/Service Documentation "SINUMERIK 840D/SIMODRIVE 611D Installation and Start-Up Guide":

- Enter the machine manufacturer password.
- Change to the "Services" operating area.
- Press softkey "Series start-up".
- Select "NC" and "PLC" as the areas to be saved and enter a name of your choice for the archive file to be created on the hard disk. Finish by pressing the RETURN key.
- If the control system contains machine-specific compensation data, then these must also be saved:

Keep the archive files you have created in a safe place. They will allow you to restore original settings in your system.

**Insert the PC card**

- Deactivate the control.
- Insert the PC card with the new firmware in the PCMCIA slot of the NCU.
- Then proceed as follows:

1. Turn switch S3 on the front panel of the NCU to 1
2. Switch the control system back on again.
3. During power-up, the firmware is copied from the PC card to the NCU memory.
4. Wait until number “6” is displayed on the NCU digital display (after approximately one minute).
5. Turn switch S3 back to zero.

---

**Note**

If number “6” does not appear, then an error has occurred.

– Incorrect PC card (e.g. card for NCU2 in NCU3 hardware)

– Card hardware defective.

---

**Copy back SRAM contents**

To copy the saved data back into the control system, proceed as described in Section 12.2 (series start-up) in the Installation and Start-Up Guide. Please read all information provided by the manufacturer about new software versions.

- Enter the machine manufacturer password.
- Select “Data In” and “Archive...”.

## 5.2 Starting up the Remachining function

To start up the “Retrace/Continue Machining Support” function, you next need to activate the compile cycle.

**Option data for compile cycles**

- Set the option for the compile cycle application 10

Machine data 19600 $ON_CC_EVENT_MASK[9] = ’HFF’

- Set bit 2 in option data $ON_PROG_MASK.

Machine data 19340 $ON_PROG_MASK = ’H04’

If this data is already set to a value higher than 0, then OR the existing value with 0x04 and set data MD $ON_PROG_MASK to the resulting value.

**Other MD**

- Set MD 28090 $MC_MM_NUM_CC_BLOCK_ELEMENTS = 1
- Set MD 28100 $MC_MM_NUM_CC_BLOCK_USER_MEM = 2

If these two machine data are already set to values greater than zero for other compile cycles, the settings increase accordingly.

- Set MD 11602 $MN_ASUB_START_MASK = 'H01'
- Set MD 11604 $MN_ASUB_START_PRIO_LEVEL = 1

If these two machine data are already set to values greater than zero, please note the following:

- Do not alter MD $MN_ASUB_START_PRIO_LEVEL.
- MD $MN_ASUB_START_MASK is evaluated as a binary value. For this reason, OR the existing value with 0x01 and set MD $MN_ASUB_START_MASK to the resulting value.

The two machine data $MN_ASUB_START_MASK and $MN_ASUB_START_PRIO_LEVEL must be set as described above because the ASUB cc_resu_asub.spf must be started in the NC STOP state (see Section 2.5). If these two machine data do not contain the appropriate values, alarm 75600 appears (see Subsection 6.3.1).

- Set MD 62571 $MC_RESU_RING_BUFFER_SIZE
  Recommended default setting for start-up: 1000
- Set MD 62572 $MC_RESU_SHARE_OF_CC_HEAP_MEM
  Recommended default setting for start-up: 100
- Set MD 28105 $MC_MM_NUM_CC_HEAP_MEM
  Recommended default setting for start-up: 50
  When optimizing these 3 machine data ($MC_RESU_RING_BUFFER_SIZE, $MC_RESU_SHARE_OF_CC_HEAP_MEM and $MC_MM_NUM_CC_HEAP_MEM), please read the information in Section 5.3.
- Set MD 18351 $MN_MM_DRAM_FILE_MEM_SIZE
  Recommended default setting for start-up: 100
  If this machine data is already set to a value greater than 0, the setting increases accordingly.

  Machine data $MN_MM_DRAM_FILE_MEM_SIZE is needed if the reverse travel program cc_resu.mpf is to be stored in the dynamic NC program memory and not in the buffered NC program memory (see Section 3.2). If cc_resu.mpf is to be stored in the buffered memory, it is not necessary to reserve space in the dynamic memory for reverse travel program cc_resu.mpf. In this case, MD $MN_MM_DRAM_FILE_MEM_SIZE should not be changed. If the value of MD $MN_MM_DRAM_FILE_MEM_SIZE is equal to zero but reverse travel program cc_resu.mpf is to be stored in the dynamic NC program memory (DRAM), alarm 75604 appears (see Subsection 6.3.2).
- Set MD 62574 $MC_RESU_SPECIAL_FEATURE_MASK
  Recommended default setting for first start-up: 0C (hex)

  For further information on this machine data, please refer to Subsection 4.2.1.

**Run up the software again.**

**Alarms**

Enter the alarm texts in language-specific text file TF_GR.COM or TF_UK.COM (see Section 5.4).

## 5.3 Calculation of memory requirements

This section explains how to calculate the requirement for compile cycle heap memory ($MC_MM_NUM_CC_HEAP_MEM).

The “Retrace/Continue Machining Support” compile cycle has 2 internal buffers. Both of these are set up in the compile cycle heap memory. Machine data $MC_MM_NUM_CC_HEAP_MEM must therefore be set to reserve the appropriate memory space. The calculation method recommended for determining how much compile cycle heap memory needs to be reserved at the start-up stage is detailed below. The memory requirement is dependent on the sizes of the buffers.

The first buffer is a circular buffer. Each memory cell contains the geometric information from one program block. One cell in this buffer requires 32 bytes. The size of the circular buffer is defined via machine data $MC_ RESU_RING_BUFFER_SIZE. If you want, for example, to be able to reverse the last 1000 blocks at any given time, then you need to set this data to 1000. This also means that the circular buffer will require 32000 bytes of heap memory.

The block search information is stored in the second buffer. A new cell is added to the buffer every time a subroutine is called. The dimension of this buffer can be varied according to whether the parts programs subsequently generated will contain many or just a few subroutine calls. Each cell in this buffer requires 180 bytes of heap memory. Its size can only be defined indirectly. Machine data $MC_MM_NUM_CC_HEAP_MEM can be set to define the total size of the compile cycle heap memory (e.g. 50 KB). Machine data $MC_ RESU_RING_BUFFER_SIZE is set to dimension the size of the circular buffer (e.g. 1000 cells equals 32000 bytes). Hence, 51200 Byte – 32000 bytes = 19200 bytes remain free for the block search buffer. Consequently, the buffer for the block search information in our example can accommodate up to 19200 bytes / 180 bytes = 106 elements. If this is not enough, the compile cycle heap memory ($MC_MM_NUM_CC_HEAP_MEM) can either be increased or the circular buffer size ($MC_RESU_RING_BUFFER_SIZE) reduced.

Memory for at least 12 elements must be available for the block search buffer (12 * 180 bytes = 2160 bytes), otherwise alarm 75600 appears (see Subsection 6.3.2). If the size of the buffer for block search information should not be sufficient while the function is running, alarm 75606 appears (see Section 6). Alarm 75606 does not interrupt the current machining operation, but results in a reduction in the length of the retraceable contour. If alarm 75606 is generated when parts programs are running, then machine data $MC_MM_NUM_CC_HEAP_MEM and $MC_ RESU_RING_BUFFER_SIZE must be adjusted to increase the available memory for the block search buffer.

**Size of the block search buffer**

The size of the buffer for the block search information can only be defined indirectly, as described above.

The actual size of this buffer can be read from the channel-specific GUD CC_RESU_LENGTH_BS_BUFFER. This GUD variable is intended to provide feedback or confirmation for the commissioning engineer. It can be viewed on the operator panel in the “Parameter” – “User Data” display area.

Before you can view the GUD, it has to be created. For a detailed description of the procedure to be followed, please refer to Section “File and program management” in the document “SINUMERIK 840D/810D/FM-NC Programming Guide Advanced”.

Proceed as follows:

- Create the file sgud.def on the MMC in the definition directory (_N_DEF_DIR) and enter the following lines:

DEF CHAN REAL CC_RESU_LENGTH_BS_BUFFER ; RESU variable

M30

- Press the appropriate softkey to activate the GUD.
- The next time the NCK is booted, it recognizes the GUD and you can read the values.

**Allocation of CC heap memory**

If other compile cycles requiring a share of the heap memory are active at the same time as the “Retrace/Continue Machining Support” function, then the amount of memory allocated to the latter must be limited. This can be done via machine data $MC_RESU_SHARE_OF_CC_HEAP_MEM (see Subsection 4.2.1).

## 5.4 Creating alarm texts

---

**Note**

With SW 6.4, compile cycles TE1 – TE8 are no longer supplied on PC cards. They are available as loadable software blocks. Installation instructions are provided in the general description in TE0.

---

Add an entry for the alarm text files for the function described in the [TextFiles] section of the C:\OEM\MBDDE.INI file:

CZYK=C:\OEM\TF_

If file C:\OEM\MBDDE.INI does not exist, it must be set up, although only section [Text Files] is required.

Create language-specific text files
TF_xx.COM in directory

C:\OEM\

xx stands for the language code, e.g. GR for German and UK for English.

Enter the following alarm texts there:

in TF_GR.COM:

075600 0 0 “Channel %1 Retrace Support: incorrect MD configuration, error no. %2”

075601 0 0 “Channel %1 block %2 Invalid argument forCC_PREPRE()”

075604 0 0 “Channel %1 reverse travel not possible, error no. %2”

075605 0 0 “Channel %1 Retrace Support: Internal error, error no. %2”

075606 0 0 “Channel %1 Retraceable contour shortened”

075607 0 0 “Channel %1 Program cannot be continued”

075608 0 0 “Channel %1 NC memory limit reached, RAM type %2”

075609 0 0 “Channel %1 RESU axis, wrong axis config., axis type %2, block %3”

075610 0 0 “Channel %1 RESU, NC START not possible” or in TF_UK.COM:

075600 0 0 “Channel %1 Retrace Support: invalid MD configuration, error no. %2”

075601 0 0 “Channel %1 block %2 invalid argument of CC_PREPRE()”

075604 0 0 “Channel %1 retracing not possible, error no. %2”

075605 0 0 “Channel %1 Retrace Support: Internal error, error no. %2”

075606 0 0 “Channel %1 Retraceable contour shortened”

075607 0 0 “Channel %1 Program cannot be continued”

075608 0 0 “Channel %1 NC memory is full, RAM type %2”

075609 0 0 “ Channel %1 RESU axis, wrong axis config., axis type %2, block %3”

075610 0 0 “ Channel %1 RESU, NC START not possible”

■

# Data Fields, Lists 6

## 6.1 Interface signals

| **DB21– DB28**<br>**DBX0.1**<br>Data block | **Reverse/Forward**<br>Signal(s) to channel (PLC–>NCK) | |
|---|---|---|
| Edge evaluation: yes | Signal(s) updated: | Signal(s) valid from SW 5.3 |
| Signal state 1 or<br>signal transition 0 –> 1 | Activate reverse travel. –> Switch control to Retrace mode. The head is reversed along the original contour when NC Start is next actuated. | |
| Signal transition 1 –> 0 | Activate forward travel. Precondition: The control system is in Retrace mode. On the next NC START, the head is moved forwards along the original contour; the reverse travel program cc_resu.mpf, and not the original program, is active. | |
| Signal state 0 | No meaning | |
| Signal irrelevant for ..... | “Retrace/Continue Machining Support” compile cycle function not active | |
| Further references | Sections 2.2, 2.6 and 3.1 | |

| **DB21– DB28**<br>**DBX0.2**<br>Data block | **Start program rerun**<br>Signal(s) to channel (PLC–>NCK) | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: | Signal(s) valid from SW 5.3 |
| Signal state 1 | Start Continue Machining: Original program is selected and a block search to the program continuation point is performed. | |
| Signal state 0 | No meaning | |
| Signal irrelevant for ..... | Control is not in Retrace mode (no previous reverse travel) or “Retrace/Continue Machining Support” compile cycle function is not active | |
| Further references | Sections 2.2, 2.6 and 3.1 | |

| **DB21– DB28**<br>**DBX32.1**<br>Data block | **Retrace mode active**<br>**Signal(s) from channel (NCK->PLC)** | |
|---|---|---|
| Edge evaluation: no | Signal(s) updated: | Signal(s) valid from SW 5.3 |
| Signal state 1 | The “Retrace mode active” signal is active as long as the control is in Retrace mode. This is the case from initial activation of the “Reverse/Forward” signal until activation of the “Start Continue Machining” signal. | |
| Signal state 0 | Original program is executed. The control system is not in Retrace mode. | |
| Signal irrelevant for ..... | “Retrace/Continue Machining Support” compile cycle function not active | |
| Further references | Sections 2.2, 2.6 and 3.1 | |

| | | |
|---|---|---|
| **DB21– DB28**<br>**DBX32.2**<br>Data block | **Continue machining active**<br>Signal(s) from channel (NCK->PLC) | |
| Edge evaluation: no | Signal(s) updated: | Signal(s) valid from SW 5.3 |
| Signal state 1 | The "Continue Machining Active" signal is set when the HIGH state is detected at the "Start Continue Machining" signal. The "Continue machining active" signal is reset when the program has been continued, The Continue Machining process is completed after the first NC START at the end of the final block search in the original program. | |
| Signal state 0 | Continue Machining not active | |
| Signal irrelevant for ..... | "Retrace/Continue Machining Support" compile cycle function not active | |
| Further references | Sections 2.2, 2.6 and 3.1 | |

## 6.2 NC machine data

| Number | Identifier | Name | Reference |
|---|---|---|---|
| **General ($MN_...)** | | | |
| 11602 | ASUB_START_MASK | Ignore stop reasons if an ASUB is running | 5.2 |
| 11604 | ASUB_START_PRIO_LEVEL | Defines the ASUB priority level from which ASUB_START_MASK is operative. | 5.2 |
| 18351 | MM_DRAM_FILE_MEM_SIZE | Size of the memory for files in the DRAM of the passive file system (in KB) | 5.2, 3.2 |
| 19600 | CC_EVENT_MASK | Enable CC events | 5.2 |

| Number | Identifier | Name | Reference |
|---|---|---|---|
| **Channel-specific ($MC_ ...)** | | | |
| 20050 | AXCONF_GEOAX_ASSIGN_TAB | Assignment geometry – channel axis | 2.3 |
| 24120 | TRAFO_GEOAX_ASSIGN_TAB_1 | GEO/channel axis assignment of Transformation 1 | 2.3 |
| 28090 | MM_NUM_CC_BLOCK_ELEMENTS | Number of block elements for CC | 5.2 |
| 28100 | MM_NUM_CC_BLOCK_USER_MEM | Size of block memory for CC in KB | 5.2 |
| 28105 | MM_NUM_CC_HEAP_MEM | Heap memory in KB for compile cycle applications (DRAM) | 5.2 |
| 62571 | RESU_RING_BUFFER_SIZE | Size of circular buffer | 4.2.1 |
| 62572 | RESU_SHARE_OF_CC_HEAP_MEM | Share of total CC_HEAP_MEM | 4.2.1 |
| 62573 | RESU_INFO_SA_VAR_INDEX | Indices of the synchronized action variables | 4.2.1 |
| 62574 | RESU_SPECIAL_FEATURE_MASK | Additional properties of the "Retrace/Continue Machining Support" function which can be enabled.<br>power ON – effective. | 4.2.1 |
| 62575 | RESU_SPECIAL_FEATURE_MASK_2 | Additional properties of the "Retrace/Continue Machining Support" function which can be enabled.<br>RESET – effective. | 4.2.1 |

# 6.3 Alarms

## 6.3.1 Alarms of standard system

| 6500 | NC memory limit reached |
|---|---|
| Explanation | This alarm can occur while file cc_resu.mpf is being written if there is insufficient buffer memory available. |
| Reaction | Alarm display<br>File cc_resu.mpf is generated for as long as there is sufficient memory available and correctly terminated, i.e. a subroutine call CC_RESU_END is inserted with parts program terminator M30 at the end. Although the file is smaller, it is still usable. |
| Remedy | Adjust the size of the buffer memory ($MN_MM_USER_MEM_BUFFERED) or increase the available space in the buffer memory, e.g. by unloading parts programs that are no longer needed, or reduce the size of the circular buffer (see also Subsection 4.2.1, MD $MC_RESU_RING_BUFFER_SIZE).<br>Or create the file in the dynamic NC program memory (DRAM, see Section 3.2). In this case, no memory space is needed for cc_resu.mpf in the buffered NC program memory. |
| To continue program | Clear the alarm with the cancel key. No further operator action required. |

| 12340 | Channel %1 block %2 too many arguments %3 |
|---|---|
| Explanation | %1 = channel number<br>%2 = block number, label<br>%3 = source symbol (in this case CC_PREPRE)<br>Too many arguments are specified in the displayed function call. |
| Reaction | Alarm display, interruption of parts program interpretation, alarm signal in PLC interface |
| Remedy | Correct function call according to function definition. In other words, program CC_PREPRE() with one parameter. |
| To continue program | Clear the alarm with the RESET key. Restart the parts program. |

| 12550 | Channel %1 block %2 name %3 not defined or option not installed |
|---|---|
| Explanation | %1 = channel number<br>%2 = block number, label<br>%3 = source symbol (in this case CC_PREPRE)<br>If the “Retrace Support/Continue Machining” function is not available on the PC card or has not yet been activated via machine data, the corresponding parts program command (CC_PREPRE) is rejected as invalid syntax. |
| Reaction | Alarm display, interruption of parts program interpretation, alarm signal in PLC interface |
| Remedy | Use technology card and install function (see Section 5.2). |
| To continue program | Clear the alarm with the RESET key. Restart the parts program. |

### 6.3.2 Alarms associated with continue machining function

| 75600 | Channel %1 Retrace Support: incorrect MD configuration, error no. %2” |
|---|---|
| Explanation | The following error has been detected in the machine data for the continue machining function during booting:<br>Error no. = 4<br>Machine data $MC_MM_NUM_CC_BLOCK_ELEMENTS or $MC_MM_NUM_CC_BLOCK_USER_MEM must be increased (see Section 5.2).<br>Error no. = 5<br>Too little compile cycle heap memory is available. Adjust machine data $MC_RESU_RING_BUFFER_SIZE, $MC_RESU_SHARE_OF_CC_HEAP_MEM and $MC_MM_NUM_CC_HEAP_MEM (see Section 5.3).<br>Error no. = 6<br>Machine data $MN_ASUB_START_MASK and $MN_ASUB_START_PRIO_LEVEL are not correctly set (see Section 5.2).<br>Error no. = 11<br>Values have not been assigned to machine data $MC_AX-CONF_GEOAX_NAME_TAB[n], $MN_INTERMEDIATE_POINT_NAME_TAB[n] and $MN_IPO_PARAM_NAME_TAB[n]. This is necessary, however.<br>Error no. = 13<br>Bit 2 = 0 of MD $MC_RESU_SPECIAL_FEATURE_MASK specifies that the reverse travel program cc_resu.mpf is to be stored in the DRAM NC program memory. However, no DRAM NC program memory was requested via MD $MN_MM_DRAM_FILE_MEM_SIZE. Remedy: Either set MD $MN_MM_DRAM_FILE_MEM_SIZE to a value not equal to zero or set bit 2 of MD $MC_RESU_SPECIAL_FEATURE_MASK equal to one. See also Sections 3.2 and 5.2. |
| Reaction | Alarm display, mode group not ready, motion stop, no NC START possible, alarm signal at PLC interface |
| Remedy | Correct the machine data or assign values. |
| To continue program | Switch the control OFF – ON. |

| 75601 | Channel %1 block %2 invalid argument for CC_PREPRE() |
|---|---|
| Explanation | Only values –1, 0, 1 are valid arguments. |
| Reaction | Alarm display, interpreter stop, alarm signal at PLC interface |
| Remedy | Correct the parts program |
| To continue program | Clear the alarm with the RESET key. Restart the parts program. |

| 75604 | **Channel %1 reverse travel not possible, error no. %2** |
|---|---|
| Explanation | Reverse travel is not possible because the following error was detected:<br>Error no. = 1<br>The current reversal block for reverse travel is probably a block from cc_resu_ini.spf or cc_resu_end.spf, programmed with a block number. The cc_resu_ini.spf and cc_resu_end.spf subprograms may not contain block numbers, because these have an internal meaning (see Section 2.5).<br>Error no. = 2<br>cc_resu.mpf cannot be created, because no DRAM is available (see Section 5.2 and Section 3.2).<br>Error no. = 4<br>The selected Continue Machining block is probably a block from cc_resu_ini.spf or cc_resu_end.spf, programmed with a block number. The cc_resu_ini.spf and cc_resu_end.spf subprograms may not contain block numbers, because these have an internal meaning (see Section 2.5). |
| Reaction | Alarm display, alarm signal at PLC interface, no NC START possible |
| Remedy | Error no. = 1<br>Remove all block numbers from subprograms cc_resu_ini.spf and cc_resu_end.spf and their subprogams.<br>Error no. = 2<br>Assign the desired value to machine data $MN_MM_DRAM_FILE_MEM_SIZE (see Section 5.2).<br>Error no. = 4<br>Remove all block numbers from subprograms cc_resu_ini.spf and cc_resu_end.spf and their subprogams. |
| To continue program | Clear the alarm with the RESET key. Restart the parts program. |

| 75605 | **Channel %1 Retrace Support: Internal error, error no. %2** |
|---|---|
| Explanation | RESU-internal error states are displayed with this alarm. An error number is also specified to provide further details about the cause and location of the error. This alarm should never occur during normal operation. |
| Reaction | Alarm display, no NC START possible, alarm signal at PLC interface |
| Remedy | If this alarm should occur, please note the error number and contact the SINUMERIK hotline at SIEMENS AG. |
| To continue program | Clear the alarm with the RESET key. Restart the parts program. |

| 75606 | **Channel %1 Retraceable contour shortened** |
|---|---|
| Explanation | The block search buffer is full. The retraceable contour needed to be shortened as a result (see Section 2). |
| Reaction | Alarm display |
| Remedy | This alarm has no effect on current machining operations.<br>However, the cause should be remedied if the alarm occurs frequently:<br>Adjust machine data $MC_RESU_RING_BUFFER_SIZE, $MC_RESU_SHARE_OF_CC_HEAP_MEM and $MC_MM_NUM_CC_HEAP_MEM (see Section 5.3). |
| To continue program | Clear the alarm with the cancel key. No further operator action required. |

| 75607 | **Channel %1 Machining cannot be continued** |
|---|---|
| Explanation | The block search triggered by the compile cycle has been terminated with an error. This may have the following cause: The control is not in the correct operating mode, e.g. JOG-AUTO instead of AUTO. |
| Reaction | Alarm display, alarm signal at PLC interface |
| Remedy | Switch control to AUTO mode. |
| To continue program | Clear the alarm with the cancel key. Start Continue Machining again. |

| 75608 | **Channel %1 NC memory limit reached, RAM type %2** |
|---|---|
| Explanation | This alarm can occur while file cc_resu.mpf is being written if there is insufficient memory available.<br>RAM type = 1:<br>The reverse travel program cc_resu.mpf is created in buffered memory (SRAM). The buffered memory is full. If buffered memory is used and alarm 75608 occurs with RAM type 1, system alarm 6500 occurs at the same time as this alarm (see Subsection 6.3.1).<br>RAM type = 2:<br>The cc_resu.mpf file is created in dynamic memory (DRAM NC program memory). The DRAM NC program memory is full. |
| Reaction | Alarm display<br>File cc_resu.mpf is generated for as long as there is sufficient memory available and correctly terminated, i.e. a subroutine call CC_RESU_END is inserted with parts program terminator M30 at the end. The file is smaller but can be used nonetheless (see also Section 3.2). |
| Remedy | This alarm has no effect on current machining operations.<br>However, the cause should be remedied if the alarm occurs frequently:<br>RAM type = 1:<br>Adjust the size of the buffer memory ($MN_MM_USER_MEM_BUFFERED) or increase the available space in the buffer memory, e.g. by unloading parts programs that are no longer needed, or reduce the size of the circular buffer (see also Subsection 4.2.1, MD $MC_RESU_RING_BUFFER_SIZE).<br>RAM type = 2:<br>Adjust the size of the DRAM NC program memory (MD $MN_MM_DRAM_FILE_MEM_SIZE – see Section 5.2) or reduce the size of the circular buffer (see Subsection 4.2.1, MD $MC_RESU_RING_BUFFER_SIZE). |
| To continue program | Clear the alarm with the cancel key. No further operator action required. |

| 75609 | **Channel %1 RESU axis, wrong axis config., axis type %2, block %3** |
|---|---|
| Explanation | A RESU axis (geometry axis one or two) is not traversed as a geometry axis in the block displayed. For example, the following was programmed:<br>N20 POS[X] = 10<br>Axis type: value not relevant |
| Reaction | Alarm display, interpreter stop, no NC START possible, motion stop, alarm signal at PLC interface |

| | |
|---|---|
| Remedy | Correct parts program, e.g.:<br>1. N20 X = 10<br>2. N19 CC_PREPRE(0)<br>N20 POS[X] = 10 ; The X axis is to be traversed as<br>; POS axis. Therefore do not log this block.<br>N21 CC_PREPRE(1)<br>N22 G1 X5<br>If this remedy is insufficient or if the error message specifies a block in which the relevant axis was not programmed, an incorrect axis type may be saved internally for the relevant axis. You can correct this by programming the following:<br>N30 X=IC(0)<br>Programming an incremental distance of 0 mm has no effect on the programmed contour. However, it ensures that POS axis X is used as a geometry axis again. |
| To continue program | Clear the alarm with the RESET key. Restart the parts program. |

| **75610** | **Channel %1 RESU, NC START not possible.** |
|---|---|
| Explanation | For internal reasons, an NC START cannot be triggered in certain situations while RESU is active. If an NC START is triggered, it is blocked and alarm 75610 is displayed. Such situations occur:<br>• if reverse travel is requested: the operator presses the “Reverse/Forward” key in the NC STOP state. NC START is blocked while the reverse travel program cc_resu.mpf is being generated and selected.<br>• if Continue Machining is initiated: the operator presses the “Start Continue Machining” key in the NC STOP state (in Retrace mode). NC START is blocked while the block search is being initiated and executed and while the ASUB cc_resu_bs_asub.spf is running at the end of the block search. See also Section 2.6. |
| Reaction | Alarm display, alarm signal at PLC interface |
| Remedy | Press NC START again. |
| To continue program | Wait briefly, then cancel the alarm with NC START and continue machining. |

■

# Explanation of terms 7

**Damage point**

The point on the contour at which the cutting quality is visibly substandard or at which the cutting process is interrupted. The machine operator would like to repeat the machining process at this point. In other words, the actual position of the damage can be further back, since it is typically necessary to restart the cutting process before the actual damage point.

**Interruption point**

The point on the contour at which the operator notices the damage and stops the machine.

**Program continuation point**

The machine operator initiates the Continue Machining process at the program continuation point. The original program is selected automatically and machining resumes at the current position in the original program. The program continuation point is normally the same as the damage point.

**Retraceable contour**

Contour stored in the circular buffer along which the head can be reversed. See also Section LEERER MERKER.

**Retrace mode**

While the head is reversing along the contour, the control is in Retrace mode. This is the case from initial activation of the “Reverse/Forward” signal until activation of the “Start Continue Machining” signal. In figure 7-1 (see below), the control system is in Retrace mode in steps two to five.

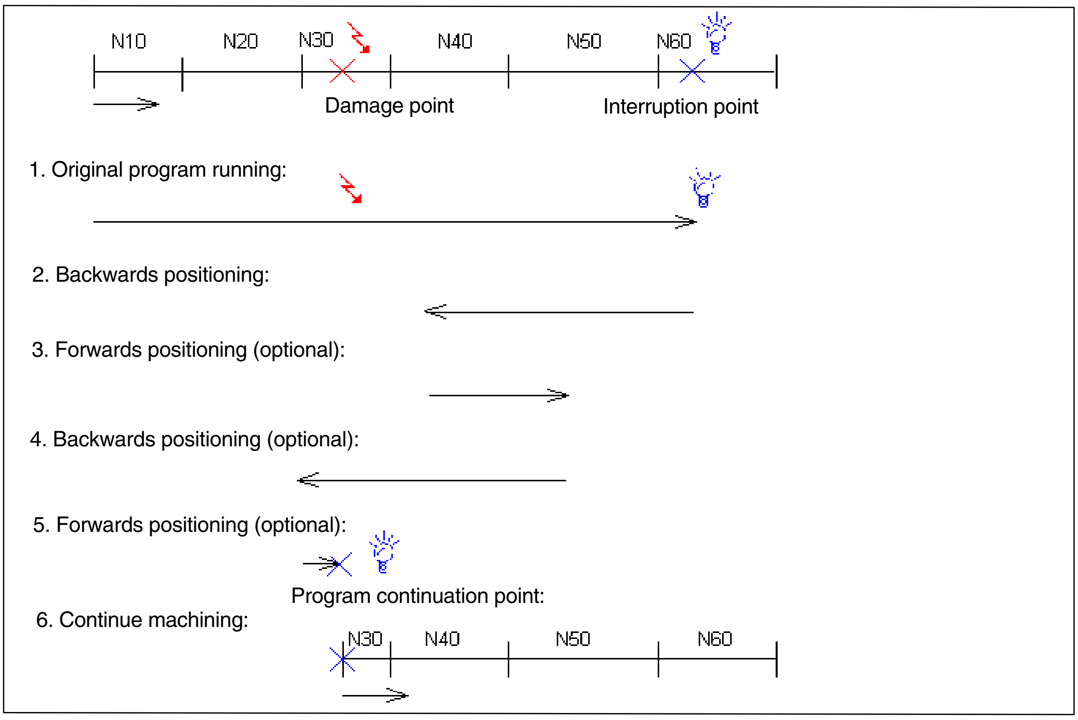



Fig. 7-1 Example program execution and use of the RESU functionality

**Example**

This diagram shows an example of a possible sequence for reversing and subsequent program continuation. The diagram shows the execution of a parts program with blocks N10 to N60 and the path traversed in steps one to six. The machine operator positions the head in reverse and forward several times in steps two to five until he has placed it exactly on the damage point in step five, and thus exactly on the program continuation point. He starts the Continue Machining process in step six and then continues execution of the original program at the program continuation point.

■

# SINUMERIK 840D/840Di/810D Description of Functions Special Functions (Part 3)

# Cycle-Clock-Independent Path-Synchronous Signal Output (TE8)

# Notes

# 1 Brief Description

A solution is required for rapid laser machining that allows the laser to be switched on and off independent of the position, e.g. for cutting perforations in steel plates. On the assumption that all motions requiring the laser to be switched off are traversed with G0, it is possible to combine the switching signal for the laser with the rising and falling edges of G0. As an alternative, a specific threshold of the programmed feedrate can be applied as the switching signal: If lower feedrate values are programmed, the laser is switched on and, conversely, switched off at higher feedrate values. The threshold value itself also causes activation of the signal.

**Basis for calculation**

The difference between the current position and setpoint position for switching is calculated in every position controller cycle. Taking into account the current velocity and acceleration, the time required until the next switching operation is calculated. If this time falls within 1.5 position controller cycles, a hardware timer set to the calculated time is started. This outputs the switching signal independently of the position control cycle on timeout.

■

## Notes

# Detailed Description 2

## 2.1 General

When an edge (positive or negative) of G0 is detected, or the feedrate exceeds or falls below the selected threshold, the setpoint position at which the laser signal must next be switched is stored. In the simplest case, this is the programmed setpoint in the last block, i.e. the position on block change.
The stored setpoint is now compared with the actual (encoder) value in the position control cycle. If the actual value approaches so close to the stored setpoint that the laser signal must be switched within the next 1.5 position controller cycles, the hardware timer is started with the calculated time value. The switching operation is then performed asynchronously to all cycles in the system.

If, for example, the setpoint cannot be reached at all due to a change in direction, then the laser is switched as soon as the difference between the setpoint and actual value has reached a minimum tolerance window.
If the variant using a feedrate threshold is selected, the laser is still switch off when G0 is detected (the programmed feedrate does not change with G0). This is designed specially to prevent a G0 block at the beginning of a machining operation (programmed feedrate = 0) from switching on the laser. The laser can only be switched back on again by a feedrate value that is lower than the selected threshold value.

An offset can be specified (programmed) additionally in distance units for both switching operations, i.e. switch-on and switch-off can be displaced with respect to the setpoint position by a particular differential distance. In this case, a negative value denotes an offset ahead of the setpoint (lead) and a positive number an offset behind the setpoint (lag). The offset refers to the path; for purposes of simplicity, the calculation presupposes a linear motion, ignoring any curvature.

If an excessively large lead value is programmed, i.e. the setpoint is already exceeded by the time the edge is detected, the signal is switched immediately.

---

**Note**

Owing to the traversing logic used, programmed lead values are inoperative in SBL mode and with G60!

---

When the program is interrupted (NC Stop) and the machine switched to JOG mode, the function is deactivated until the system reverts to Automatic mode and the programmed continued with NC Start.

## 2.2 Programming

As regards switching the laser itself on and off, there is no need to take any special programming measures since this operation is directly linked to the programmed G functions or the programmed feedrate:

- Falling edge of G0, i.e. all other G functions in the 1st G group, or feedrate drops below the feedrate threshold value:
  –> Laser is switched on
- Rising edge of G0 or feedrate exceeds the feedrate threshold value:
  –> Laser is switched off

An enabling command must be programed for the operation as a whole, generally at the beginning of the program; the two offset values and, if applicable, a velocity threshold value, are specified simultaneously with this enabling command:

**Programming**

```
DEF REAL DIFFON        = –0.08
DEF REAL DIFFOFF       = 0.08
DEF REAL FEEDTOSWITCH = 20000
CC_FASTON(DIFFON, DIFFOFF, FEEDTOSWITCH)
```

**Meaning of parameters**

Parameter “DIFFON” contains the offset value for the falling edge of G0 (i.e. for the laser switch-on signal), parameter “DIFFOFF” contains the offset for the falling edge (i.e. for the switch-off operation).
The applicable unit is the setting in MD “$MN_SCALING_SYSTEM_IS_METRIC” (mm or inch).

Parameter FEEDTOSWITCH is optional; if it is not programmed, the switching operation is combined with the rising or falling edge of G0. If a feedrate value is programmed, this is applied subsequently as a threshold value for the signal.

If the offset values need to be altered within the program, command CC_FASTON can be programmed again at any time (even if the function is already active). The two switching criteria (G function or feedrate) can be changed at any time.

As soon as CC_FASTON is programmed, a laser signal is output. The level of this signal depends on the current status in this block (G0 or not, which feedrate).

The complete function can be deactivated again via a CC_FASTOFF() command; a Reset also switches the function off. The two program commands are implemented as procedures, i.e. they must be programmed alone in the block, but do not cause a drop in velocity on a contour programmed with constant tool path velocity (G64, G641,..).

## 2.3 Configuration of I/O devices

Only one of the onboard outputs (1 – 4) can be used as an output for the laser switching signal output. The number is configured via a channel-specific machine data $MC_FASTON_NUM_DIG_OUTPUT.
The number of the output bit (1 .. 4) is specified in the same way as digital NCK I/O devices are addressed via $A_OUT[nr] in the parts program.
The configured bit can be overwritten from the parts program if necessary. It can also be disabled from the PLC. It is not possible to overwrite the bit from the PLC.

If 0 is entered as the bit number for the laser output, the function is effectively deactivated. This does not cause output of an alarm, but the function cannot be activated, i.e. programmed commands CC_FASTON and CC_FASTOFF are ignored.

---

**Note**

The signal output at the three NCU outputs not used by CC_FASTON is suppressed for 2 interpolation cycles with every timer-controlled switching edge, i.e. a signal output triggered, for example, by synchronized actions, is only actually transferred to the hardware after a delay of 2 interpolation cycles.

---

■

# Notes

# Supplementary Conditions 3

## 3.1 Axes, channels

Only the movement of geometry axes is monitored. If no geometry axes are configured, a configuration alarm is output.
The function can only be activated in the 1st channel.

## 3.2 Block search

If, during a block search, the program branches to a block positioned after the switch-on command CC_FASTON, then the switching signal output behaves differently to in normal program operation. The function is activated with the first NC Start after a block search (provided the switch-on command CC_FASTON has already been executed), i.e. every subsequent movement (except for G0 motions) causes the laser to be switched on, in particular the movement towards the “damage point”, although this may have nothing to do with the original contour.
It is possible to bypass this effect by overstoring a movement to the damage point with G0.

**Example**

```
N1 G0 X0 Y0
N2 G1 F1000 X10 Y10
N3 G0 X50
N4 CC_FASTON(-0.4, 0.4)
N5 Y50
N6 G1 X100 Y100
```

With a normal program run, the laser is switched on once at the beginning of block N6.

Block search to end of block N6 (starting point: X0 Y0):
Both axes move for 0 to 100 with laser switched on.

## 3.3 Transformations, compensation, TRC, ADIS

The following boundary conditions apply:

- Transformations:
  The function can only operate with deactivated transformation. However, this is not monitored!
- Compensations:
  Compensations are generally taken into account.
- TRC:
  Tool radius compensations are also taken into account if possible. Please note, however, that additional blocks inserted as a result of the radius compensation (e.g. at outside corners) are added to the new block. This means that if there is also an edge change of G0 at this type of place on the contour, then the switching signal is gated with the block end point of the last programmed block.
- ADIS:
  A block inserted by ADIS is added to the old block, i.e. the laser signal in this case would only be switched when the new block began.

## 3.4 Compatibility with other functions

Since the hardware timer is also used for the “Software cams” function, it is not possible to utilize this at the same time as the “Clock-independent switching signal output” function. These two functions are monitored by a configuration alarm.

Otherwise there are no conflicts with other functions on the SINUMERIK 840D.

## 3.5 Creating alarm texts

Add an entry for the alarm text files for the function described in the [TextFiles] section of the C:\OEM\MBDDE.INI file:

CZYK=C:\OEM\TF_

If file C:\OEM\MBDDE.INI does not exist, it must be set up, although only section [Text Files] is required.

Create language-specific text files TF_xx.COM in directory
C:\OEM\
xx stands for the language code, e.g. GR for German and UK for English.

Enter the following alarm text there:
in TF_GR.COM:

075500 0 0 “Channel %1 Clock-independent switching signal output function incorrectly configured”.

■

# Notes

# 4 Data Descriptions (MD, SD)

## 4.1 Machine data of standard system

The “Clock-independent switching signal output” function is implemented as a compile cycle application. In addition to the function-specific machine data, the following standard machine data must therefore be set also:

- This means:

---

**Warning**

Failure to take appropriate precautions **can** have undesirable consequences.

The functions activated by the option data trigger corresponding compile cycles. These cycles can have a significant effect on the behavior of the control system and may cause hazardous situations by accessing the NC.
Before a compile cycle is activated, the necessary safety precautions must be taken to protect operating personnel and machine (safety precautions against parameterization or programming errors in the compile cycles may need to be taken).

---

The following machine data are also required:

- 28090: $MC_MM_NUM_CC_BLOCK_ELEMENTS = 1
- 28100: $MC_MM_NUM_CC_BLOCK_USER_MEM >= 10

If other compile cycles are also active (e.g. CLC), the above values must be applied additively.

| **62560**<br>MD number | **$MC_FASTON_NUM_DIG_OUTPUT**<br>Configuration of the NCK output | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 4 |
| Changes effective after power ON | Protection level: 2/7 | Unit: – |
| Data type: Byte | | |
| Meaning: | This machine data configures the number of the output bit for the laser signal.<br>Value 0 deactivates the function. | |

■

## Notes

# Example 5

## 5.1 General start-up of a compile cycle function

---

**Note**

With SW 6.4, the compile cycles are supplied as loadable modules. The general procedure for installing such compile cycles can be found in TE0. The specific installation measures for this compile cycle can be found from Section 5.2 onwards.

---

The MMC software version must match the NCK version.

An NCK PCMCIA card containing the “Clock-independent switching signal output” function must be available.

**Saving SRAM contents**

As the first step in installing a compile cycle function, the original card inserted in the NCU must be replaced by the special card.
This is identical to the procedure followed for upgrading the NCU to a later software version and likewise requires the static (battery-backed) control system memory to be erased. When this is done, all data stored in the SRAM are lost and they must therefore be saved beforehand. For a detailed description, please refer to the Manufacturer/Service Documentation “SINUMERIK 840D/SIMODRIVE 611D Installation and Start-Up Guide”:

- Enter the machine manufacturer password.
- Change to the “Services” operating area.
- Press softkey “Series start-up”.
- Select “NC” and “PLC” as the areas to be saved and enter a name of your choice for the archive file to be created on the hard disk. Finish by pressing the RETURN key.
- If the control system contains machine-specific compensation data, then these must also be saved:

Keep the archive files you have created in a safe place. They will allow you to restore original settings in your system.

**Insert the PC card**

- Deactivate the control.
- Insert the PC card with the new firmware in the PCMCIA slot of the NCU.
- Then proceed as follows:
  1. Turn switch S3 on the front panel of the NCU to 1
  2. Switch the control system back on again.
  3. During power-up, the firmware is copied from the PC card to the NCU memory.
  4. Wait until number “6” is displayed on the NCU digital display (after approximately one minute).
  5. Turn switch S3 back to zero.

---

**Note**

If number “6” does not appear, then an error has occurred.
– Incorrect PC card (e.g. card for NCU2 in NCU3 hardware)
– Card hardware defective

---

**Copy back SRAM contents**

To copy the saved data back into the control system, proceed as described in Section 12.2 (series start-up). Please read all information provided by the manufacturer about new software versions.

- Enter the machine manufacturer password.
- Select “Data In” and “Archive...”.

## 5.2 Starting up the function

To start up the “Clock-independent switching signal output” function, you next need to activate the compile cycle.

**Option data for compile cycles**

Set the option for compile cycle function 9

**Other MD**

Set the

- MD 28090 MM_NUM_CC_BLOCK_ELEMENTS = 1
- MD 28100 MM_NUM_CC_BLOCK_USER_MEM >= 10

If these two machine data are already set to values > 0 for other compile cycles, then the settings increase accordingly.

- **Run up the software again.** Machine data 62560 will then be displayed at the end of the channel-specific machine data list (look for “FASTON” or “62560”).

**Alarms**

Enter the alarm texts in language-specific text file TK1_GR.COM or TK1_UK.COM.

**I/O devices**

Enter the bit number for the laser signal in channel-specific MD 62560.

■

# Notes

# Data Fields, Lists

# 6

## 6.1 Interface signals

None

## 6.2 NC machine data

| Number | MD identifier | Name | Ref. |
|---|---|---|---|
| **General ($MN_ ...)** | | | |
| 10360 | FASTO_NUM_DIG_OUTPUTS | Number of digital output bytes | A4 |
| **Channel-specific ($MC_...)** | | | |
| 28090 | MM_NUM_CC_BLOCK_ELEMENTS | Number of block elements for CC | |
| 28100 | MM_NUM_CC_BLOCK_USER_MEM | Size of block memory for CC | |
| 28090 | FASTON_NUM_DIG_OUTPUT | Bit number for output signal for laser | |

## 6.3 Alarms

| Alarm No. | |
|---|---|
| 75500 | **Channel % 1, Configuring error** |
| Explanation | %1 = channel number<br>There are 2 causes for this alarm:<br>– No geometry axis is defined<br>– The “Software cams” option is set |
| Reaction | The function cannot be activated |
| Remedy | Change the configuration |
| To continue program | RESET |

■

## Notes

# SINUMERIK 840D/840Di/810D Description of Functions Special Functions (Part 3)

# Preprocessing (V2)



■

# Notes

# Brief Description 1

**Preprocessing**

The programs stored in the directories for standard and user cycles can be preprocessed to reduce runtimes.

Preprocessing is activated via machine data.

Standard and user cycles are preprocessed when the power is switched on, i.e. as an internal control function, the parts program is translated (compiled) into a binary intermediate code optimized for processing purposes.

All program errors that can be corrected by means of a compensation block are detected during preprocessing. In addition, when the program includes branches and check structures, a check is made to ensure that the branch destinations are present and that check structures are nested correctly.

The full scope of control functionality is available:

- Override control.
- Reactions to data and signals that are input by the PLC or the operator.
- Current block display.
- The programs can be processed in single block mode (SBL1 and SBL2). Block searches can be executed. The compilation cannot be archived; it is concealed from the user and regenerated every time the power is switched on.

Preprocessing can be used for

- optimizing the runtimes of parts programs with high-level language components (branches, check structures, motion synchronous actions).
- CPU time intensive parts programs (e.g. stock removal cycles).
- faster processing of time-critical sections (e.g. program continuation after preprocessing stop during rapid deletion of distance to go, or return stroke, or in the tool change cycle).

■

# Notes

# Detailed Description 2

## 2.1 General functionality

### General notes

- **SW 3.2 and higher**
  Standard and user cycles can be preprocessed.
  The processing time of parts programs can then be reduced without restricting the control functionality.
  If machine data MD 10700: PREPROCESSING_LEVEL is set, standard and user cycles are preprocessed.
  Preprocessing is implemented on a program-specific basis. It is possible to mix preprocessed parts programs and those that are interpreted in ASCII format.
  The purpose of preprocessing is to reduce downtimes.
- **SW 3.5 and higher**
  Memory is required for preprocessing cycles. You can optimize your memory in two ways:
  - The program to be executed can be reduced with the command DISPLOF (display off).
  - MD 10700: PREPROCESSING_LEVEL, Bits 2 and 3 have been added. These can be set to select cycle preprocessing for individual directories (e.g. user cycles).
- **As of SW 5**
  MD 10700: PREPROCESSING_LEVEL, Bit 4 has been added. This allows you to select the preprocessing for user cycles from the directory _N_CMA_DIR.
- **SW 6.4 and higher**
  Bit 45 has been added to MD 10700: PREPROCESSING_LEVEL. This allows selective preprocessing of the individual user cycles that have the ID PREPRO after the PROC instruction.

### Functionality

The programs stored in the directories for standard and user cycles are preprocessed when the power is switched on, i.e. the parts program is translated (compiled) into an intermediate binary code optimized for processing purposes. The compilation is processed when called.

### Runtime optimization

The preprocessing function is primarily suited for optimizing the runtimes of parts programs with high-level language components (branches, check structures, motion synchronous actions).

While branches and check structures are invalidated by a search through all blocks (block start) when parts programs are interpreted in ASCII format (active as default), a branch is made directly to the destination block in a preprocessed parts program.

The runtime differences between branches and check structures are thus eliminated.

Example of runtime of preprocessed program:
Runtime reduction by 30% with active compressor.

```
DEF INT COUNTER
Destination: G1 G91 COMPON
G1 X0.001 Y0.001 Z0.001 F100000
COUNTER=COUNTER +1
COUNTER=COUNTER –1
COUNTER=COUNTER +1
IF COUNTER<= 100000 GOTOB TARGET
```

CPU time intensive programs and programs with symbolic names are processed faster.

Runtime-critical sections (e.g. continuation of processing after deletion of distance to go or preprocessing stop in cycles) can be processed faster.

If the interrupt routine is available as a preprocessed cycle, processing can be continued more rapidly after the program interrupt.

## 2.2 Program handling

**Activation/ Deactivation**

Cycles are preprocessed on power ON if machine data **MD 10700: PREPROCESSING_LEVEL**, Bit 1 is set..

0: No preprocessing

Bit 0=0: Call description of cycles is not known as standard.
Like normal subroutines, cycles must be declared as external before the cycle is called.
This setting is meaningful if no cycles with call parameters are used.

Bit 0=1: The call description of the cycles is generated during control power-up. All user cycles (directory _N_CUS_DIR) and Simians cycles (directory _N_CST_DIR) with transfer parameters can be called without external declaration.Changes to the cycle call interface do not take effect until power ON.

The following machine data need to be set:
$MN_MM_NUM_MAX_FUNC_NAMES
$MN_MM_NUM_MAX_FUNC_PARAM

Bit 1=1: During control power-up, all cycles are preprocessed into a compilation optimized for processing.
All user cycles (directory _N_CUS_DIR) and
standard cycles (directory _N_CST_DIR) are processed rapidly.
Changes to the cycle programs do not take effect until the next power ON.

**SW 3.5 and higher**

Bit 2=1: During control power-up the standard cycles in directory _N_CST_DIR are preprocessed in a compilation optimized for processing.

Bit 3=1: During control power-up, the user cycles in directory _N_CUS_DIR are preprocessed in a compilation optimized for processing.

Bit 4=1: Preprocessing of user cycles from the directory _N_CMA_DIR

Bit 5=1: Preprocessing of user cycles with the ID PREPRO in the PROC statement line. No preprocessing is carried out for files that are not marked (SW 6.4 and higher)
If the bit is Bit 0, preprocessing is controlled exclusively according to Bits 0–4.

Bit combinations are permissible.

**Compilation**

The subroutines (extension _SPF) stored in the directories for standard cycles: _N_CST_DIR, _N_CMA_DIR and user cycles: _N_CUS_DIR and if the necessary the subroutines marked PREPRO are compiled. The name of the compilation corresponds to the original cycle with extension _CYC.

---

**Note**

Program changes to precompiled programs do not take effect until the next power ON.

---

**Access authorization**

The preprocessed program can only be executed, but not read or written. The compilation cannot be modified or archived. The original cycles _SPF files are not deleted.

The compilation is not changed when the ASCII cycle is altered, i.e. changes do not take effect until after the next power ON.

**Memory required**

The memory required for compiled cycles is approximately factor 2 in addition to the ASCII parts programs.

The memory required for variables defined in the parts programs is defined via the following machine data:

MD 28020 $MC_MM_NUM_LUD_NAMES_TOTAL
MD 28010 $MC_MM_NUM_REORG_LUD_MODULES
MD 28040 $MC_MM_LUD_VALUES_MEM
MD 18242 $MC_MM_MAX_SIZE_OF_LUD_VALUE

**References:** /FB/, S7, "Memory Configuration"

While preprocessing is in progress, the amount of memory required is the same as if the preprocessed program were called on the first subprogram level.

When programs are preprocessed after power ON, a name is counted for each branch destination/label as if it were a variable. These names must be taken into account in **MD 28020: MM_NUM_LUD_NAMES_TOTAL**.

Example:

```
PROC NAMES                    ; 1 name
DEF INT VARIABLE, FIELD[2]    ; 2 names
BEGINNING:                    ; 1 name, for preprocessing only
FOR VARIABLE = 1 TO 9         ; 1 name, for preprocessing only
G1 F10 X=VARIABLE*10–56/86EX4+4*SIN(VARIABLE/3)
ENDFOR                        ; 1 name, for preprocessing only
M17
```

In order to execute this program normally, machine data $MC_MM_NUM_LUD_NAMES_TOTAL must specify at least 3 names.

6 names are required to compile this program after power ON.

## 2.3 Program call

### Overview

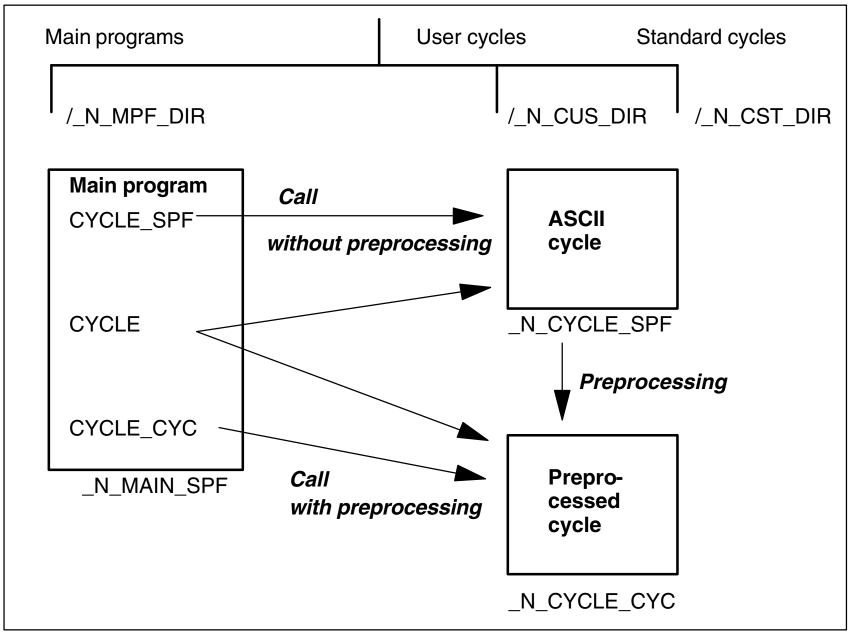



Fig. 2-1 Generation and call of preprocessed cycles without parameter

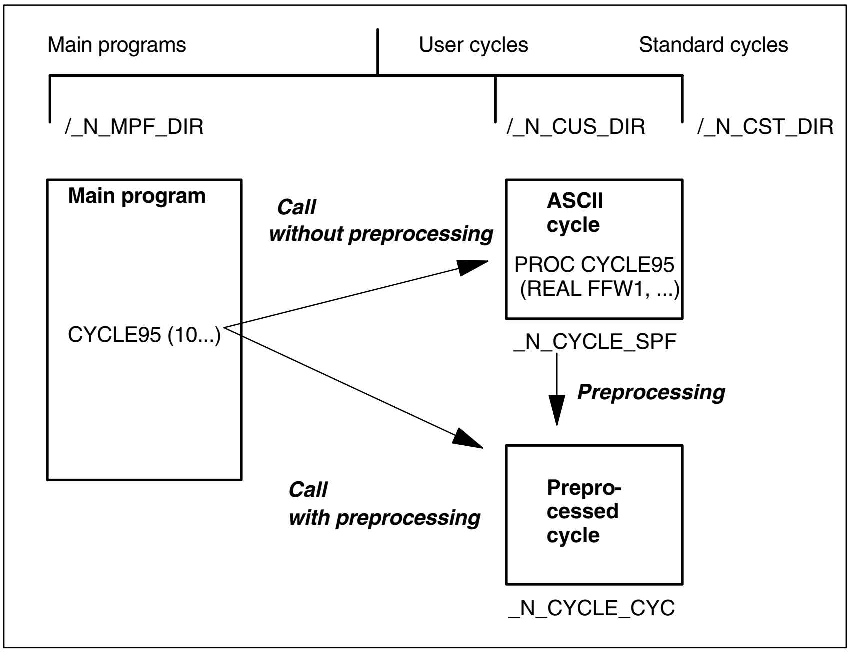



Fig. 2-2 Generation and call of preprocessed cycles with parameter

**Call**

- Compiled cycle
  A compiled cycle is called in exactly the same way as a normal subprogram.

  Example: CYCLE

- Preprocessing is activated:
  The compiled cycle is called instead of the ASCII cycle.
  - If the subprogram is called explicitly with extension _SPF, then the ASCII cycle is called even if a compilation is available.

    Example: CYCLE_SPF ;ASCII cycle call
  - If the subprogram is called explicitly with extension _CYC, then the preprocessed cycle is called if available. An error message is output if no compilation is available.

    Example: CYCLE_CYC ;Preprocessed cycle call
  - If Bit 5 is set and a file that is not marked with PREPRO called explicitly with the extension _CYC, an error message is issued with Alarm 14011.
- If a subprogram is called without explicit extension, an attempt is first made to load the program. If this is not possible (not marked with PREPRO), an attempt is made to load the SPF program.
- The change to an external language mode with G291 is rejected and an alarm issued. When the pre-compiled cycle is called, an explicit change is made to the Siemens language mode.
- When the subprogram is called, it is checked whether the compiled file is older than the cycle. If so, the compile file is deleted and an alarm issued. The user must pre/compile the cycles again.

---

**Note**

Only cycles without parameters may be called with the extension _SPF or _CYC (see Fig. 2-1).

Do not use PUDs in cycles that are preprocessed. The PUDs are created in the calling main program. At the time of compilation after power-on, these data are not known to the cycles.

---

The current program display shows whether the current ASCII cycle or the compilation has been called (extension _SPF or _CYC).

**Call condition**

All cycles in the cycle directories must be compiled before preprocessing is activated. Non-compiled cycles in _N_CUS_DIR and _N_CST_DIR which were only loaded, for example, after power ON, can only be called through explicit specification of extension _SPF.

If preprocessing is active and Bit 5 set, all programs that do not start with the PROC statement are not precompiled.

**Syntax check**

All program errors that can be corrected by means of a compensation block are detected during preprocessing. In addition, when the program includes branches and check structures, a check is made to ensure that the branch destinations are present and that check structures are nested correctly.

Branch destinations/labels must be unique in the program.

After the errors detected during preprocessing have been corrected, preprocessing must be started again by means of an NCK power ON.

## 2.4 Supplementary Conditions

**Vocabulary**

The full vocabulary of the NC language is available in the parts program.

There are no restrictions on the calculation of measured process variables and in the reaction to signals from the process and other channels (override, deletion of distance to go, motion synchronous actions, channel coordination, interrupt processing, etc.).

**Axis identifiers**

Parts programs are compiled independently of channels. For this reason, the geometry and channel identifiers set via MD$MC_AXCONF_GEOAX_NAME_TAB and $MC_AXCONF_CHANAX_NAME_TAB must be identical in all channels **if they are used directly in the precompiled cycles.**

Generally speaking, axis identifiers are not used directly in machining cycles since cycles are written

- independently of channel and
- independently of the axis identifiers defined on the machine.

The axes to be traversed are addressed indirectly via machine data or transferred as parameters:

- Indirect axis programming:
  – IF $AA_IM[**AXNAME($MC_AXCONF_CHANAX_NAME_TAB[4])**] > 5
    ; This branch is executed if the actual value of the 5th channel axis
    ; referred to the machine coordinate system is greater than 5.
  – G1 **AX[AXNAME($MC–AXCONF–GEOAX–NAME–TAB[0])**] = 10
    F1000 G90
    ; Traverse 1st geometry axis to the value 10.

  ENDIF
- Transfer of axis to be traversed from the main program:
  – Cycle definition
    PROC DRILL(AXIS DRILLING AXIS)
    WHILE $AA_IW[**DRILLING AXIS**] > –10
        G1 G91 F250 **AX[DRILLING AXIS]** = –1
    ENDWHILE
  – Call from main program
    DRILL(Z)

■

# Supplementary Conditions 3

**Availability of the "preprocessing" function**

The function is an option and available for

- SINUMERIK 840D, SW 3 and higher.

■

# Data Descriptions (MD, SD) 4

## 4.1 Description of machine data

### 4.1.1 General machine data

Preprocessing of cycles can be activated or predefined from SW 3.2 and SW 3.5 in more detail on a file level (SW 6.4 and higher) with the following existing machine data:

<table>
<tr><td colspan="2">10700<br>MD number</td><td colspan="4">PREPROCESSING_LEVEL<br>Program preprocessing level</td></tr>
<tr><td colspan="2">Default setting: 1</td><td colspan="2">Min. input limit: 0</td><td colspan="2">Max. input limit: 15</td></tr>
<tr><td colspan="3">Changes effective after power ON</td><td colspan="2">Protection level: 2/2</td><td>Unit: –</td></tr>
<tr><td colspan="4">Data type: BYTE</td><td colspan="2">Applies from SW: 3.2</td></tr>
<tr><td>Meaning:</td><td colspan="5">The term “preprocessing” refers to the declaration of cycles which can then be used without an additional EXTERN declaration in the parts program.<br><br>Several levels of processing are possible:<br>Bit 0= 0: No preprocessing<br>Bit 0= 1: The cycle call description is generated during control power up, i.e. all the programs in directories _N_CUS_DIR and _N_CST_DIR can be called in the parts program without an EXTERN declaration.<br>Any changes made to the parameter interface of a cycle in the control do not take effect until the next power ON.<br><br>Bit 1=1: During control power-up, all cycles are preprocessed into a compilation optimized for processing. All user cycles (directory _N_CUS_DIR) and standard cycles (directory _N_CST_DIR) are processed rapidly. Program changes to cycle programs do not take effect until the next power ON.<br><br>Bit 2=1: During control power-up, the Siemens cycles in directory _N_CST_DIR are preprocessed in a compilation optimized for processing (SW 3.5 and higher).<br><br>Bit 3=1: During control power-up, the user cycles in directory _N_CUS_DIR are preprocessed in a compilation optimized for processing (SW 3.5 and higher).<br><br>Bit 4=1: Preprocessing of user cycles from directory _N_CMA_DIR (SW 5 and higher)<br><br>Bit 5=1: Preprocessing of user cycles only by marking with PREPRO in the PROC statement (SW 6.4 and higher).<br><br>Memory is required for preprocessing cycles. Better language utilization can be achieved through selective setting of the preprocessing function:<br>– Runtime-critical cycles are combined in a directory.<br>– The other cycles are located in a different directory.<br>– Use Bit 5 and mark only critical cycles.0>!</td></tr>
<tr><td>Further references</td><td colspan="5">/PA/, “Programming Guide Fundamentals” (EXTERN Declaration)</td></tr>
</table>

## 4.1.2 Channel-specific machine data

**Memory requirements**

The memory space required for variables defined in the parts programs is defined via the following machine data:

- MD 28010: MM_NUM_REORG_LUD_MODULES
- MD 28020: MM_NUM_LUD_NAMES_TOTAL
- MD 28040: MM_LUD_VALUES_MEM

While preprocessing is in progress, the amount of memory required is the same as if the preprocessed program were called on the first subprogram level.

**References**: /FB/, S7, “Memory Configuration”

The memory configuration set via machine data
MD 28010: MM_NUM_REORG_LUD_MODULES,
MD 28040: MM_LUD_VALUES_MEM and
MD 18242: MM_MAX_SIZE_OF_LUD_VALUE is relevant at the time when the subroutine is called and remains unchanged compared to the ASCII interpretation of the subroutines.

**Name**

When programs are preprocessed after power ON, a name is counted for each branch destination/label as if it were a variable. These names must be taken into account in the following machine data:

- MD 28020: MM_NUM_LUD_NAMES_TOTAL

■

# Notes

# 5 Signal Descriptions

None

■

# Example 6

## 6.1 Preprocessing individual files

```
PROC PREPRO1 PREPRO                    ; Preprocessing if Bit 5= 1
                                       ; in PREPROCESSING_LEVEL
N1000 DEF INT COUNTER
N1010 TARGET: G1 G91 COMPON
N1020 G1 X0.001 Y0.001 Z0.001 F100000
N1030 COUNTER=COUNTER+1
N1040 COUNTER=COUNTER–1
N1050 COUNTER=COUNTER+1
N1060 IF COUNTER <=10 GOTOB TARGET
N1070 M30

PROC PREPRO2
N2000 DEF INT VARIABLE, FELD[2]
N2010 IF $AN_NCK_Version < 3.4
N2020 SETAL(61000)
N2030 ENDIF
N2040 START:
N2050 FOR VARIABLE = 1 TO 5
N2060 G1 F1000 X=VARIABLE*10–56/86EX4+4*SIN(VARIABLE/3)
N2070 ENDFOR
N2080 M17

PROC MAIN
N10 G0 X0 Y0 Z0
N20 PREPRO1
N30 G0 X10 Y10 Z10
N40 PREPRO2
N50 G0 X100 Y100
N60 PREPRO3
N70 G0 X10 Y10
N80 M30
```

**Sample constellations:**
a) Bit 5 = 1
$MN_PREPROCESSING_LEVEL=45 ; Bit 0, 2, 3,5

The PREPRO2 subroutine is pretranslated; however, the call description is generated.
The PREPRO2 subroutine is not pretranslated; however, the call description is generated.

b) Bit 5 = 0
$MN_PREPROCESSING_LEVEL=13 ; Bit 0, 2, 3,

Both subroutines are pretranslated; the call description is generated.

c)
Example of an **invalid** subroutine for activated compilation:

```
PROC SUB1 PREPRO

G291                ; <—— Alarm during compilation, G291 not possible
G0 X0 Y0 Z0
M17
```

■

# Data Fields, Lists

7

## 7.1 Machine data

| Number | Identifier | Name | Reference |
|---|---|---|---|
| **General ($MN_ ...)** | | | |
| 10700 | PREPROCESSING_LEVEL | Program preprocessing level | V2 |
| 18242 | MM_MAX_SIZE_OF_LUD_VALUE | Maximum field size of LUD variables | S7 |
| **Channel-specific ($MC_ ...)** | | | |
| 28010 | MM_NUM_REORG_LUD_MODULES | Number of modules for local user variables with REORG (DRAM) | S7 |
| 28020 | MM_NUM_LUD_NAMES_PER_PROG | Number of local user variables (DRAM) | S7 |
| 28040 | MM_LUD_VALUES_MEM | Memory size for local user variables (DRAM) | S7 |

## 7.2 Alarms

A detailed description of the alarms which may occur is given in
**References:** /DA/, Diagnostics Guide
or in the online help in systems with MMC 101/102/103/HMI Advanced/HMI Embedded.

■

# SINUMERIK 840D/840Di/810D Description of Functions Special Functions (Part 3)

# 3D Tool Radius Compensation (W5)



■

# Notes

# Brief Description 1

**Why 3D TRC?**

The 3D tool radius compensation capability is used to machine contours with tools that can be controlled in their orientation independently of the tool path and shape.

---

**Note**

This description is based on the specifications for 2D tool radius compensation.

**References:** /FB/, W1, "Tool Compensation"

---

**Difference $2^1/_2$ D–3D TRC**

- With **$2^1/_2$D TRC**, it is assumed that the tool is always space-bound. Tools with constant orientation (cylindrical tools) are used for peripheral milling operations.

  While the orientation of the machining surface is not constant when other tools are used, it is determined by the contour and cannot thus be controlled independently of it.

- With **3D TRC**, surfaces with variable orientation are generated.

  The precondition for peripheral milling is that the tool orientation can be changed, i.e. in addition to the 3 degrees of freedom needed to position the tool (normally 3 linear axes), a further two degrees of freedom (2 rotary axes) are required to set the tool orientation (5-axis machining).

  End faces can be milled with 3 or 5 degrees of freedom.

**Peripheral milling, face milling**

The following diagram (Fig. 1-1) shows the differences between $2^1/_2$ D TRC and 3D TRC with respect to peripheral milling operations.

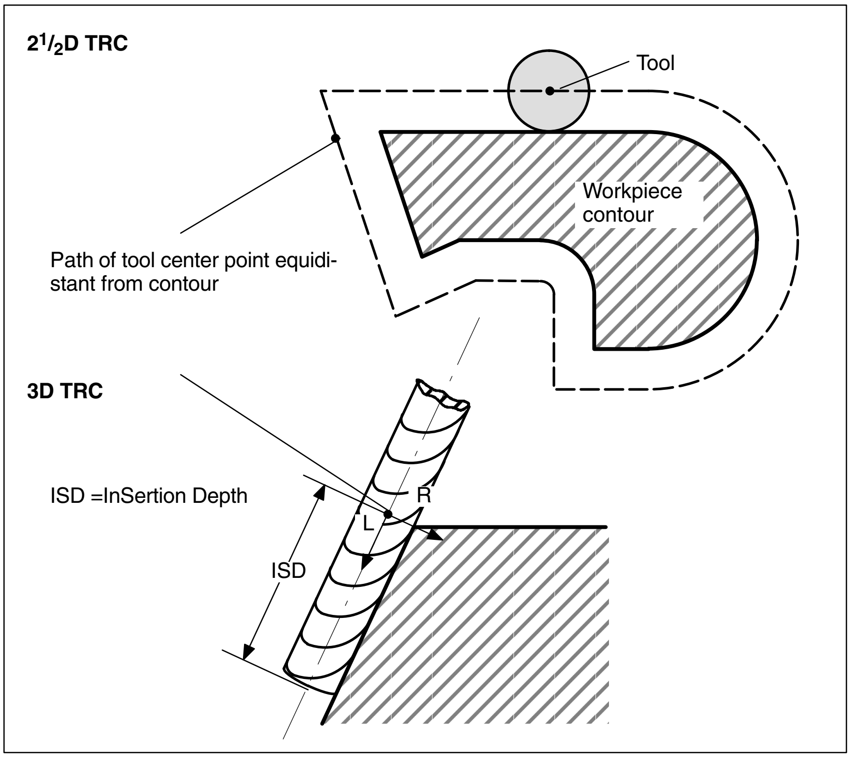



Fig. 1-1 $2^1/_2$ D and 3D tool radius compensation

The parameters for the operation shown in Fig. 1-2 “Face milling” are described in detail in Section 2.2.

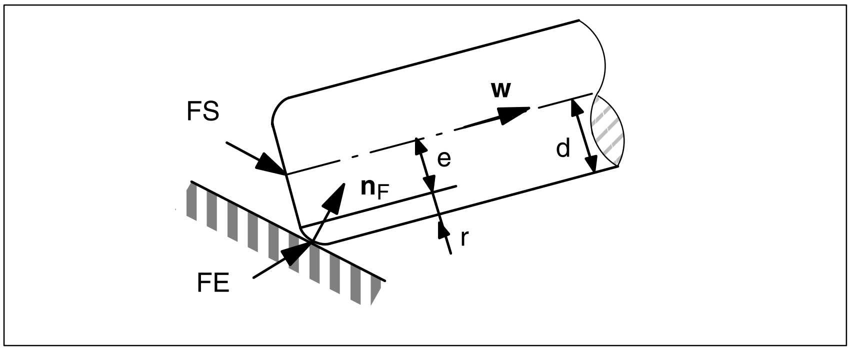



Fig. 1-2 Face milling

**Orientation**

Tools with different orientation characteristics may be used in 3D TRC, i.e.

- tools with space-bound orientation.
- tools with variable orientation.

## 1.1 Machining modes

There are two modes for milling spatial contours:

- Peripheral milling
- Face milling

Peripheral milling mode is provided for machining so-called ruled surfaces (e.g. taper, cylinder, etc.) while face milling is used to machine curved (sculptured) surfaces.

**Peripheral milling**

Tools with

- space-bound orientation ($2^1/_2$D TRC) and
- variable orientation (3D TRC)

are used in peripheral milling operations.

A 3D TRC can therefore be applied in peripheral milling only if the tool orientation is variable.

Intermediate blocks that are required from non-tangential transitions for mathematical reasons can be avoided using the intersection procedure. In these cases, the two curves in question are extended; the intersection of both extended curves is approached.

**Face milling**

Tools of both types, i.e. with constant or variable orientation, can be used for face milling operations.

Tools with variable orientation offer the following advantages:

- Better approximation of end contour.
- Greater cutting capability.
- Wider selection of tool shapes.
- Wider range of surfaces can be machined (relief cuts).

■

## Notes

# 2 Detailed Description

The following section provides a detailed function description of 3D tool radius compensation with respect to

- peripheral milling and
- face milling.

**Tool orientation**

The term “tool orientation” describes the geometric alignment of the tool in space. The tool orientation on a 5-axis machine tool can be set by means of program commands.

**References:** /PA/, Programming Guide

## 2.1 Peripheral milling

**Peripheral milling**

The variant of peripheral milling used here is implemented through the definition of a path (directrix) and the associated orientation. In this machining mode, the tool shape is irrelevant on the path and at the outer corners. The only decisive factor is the radius at the tool contact point.

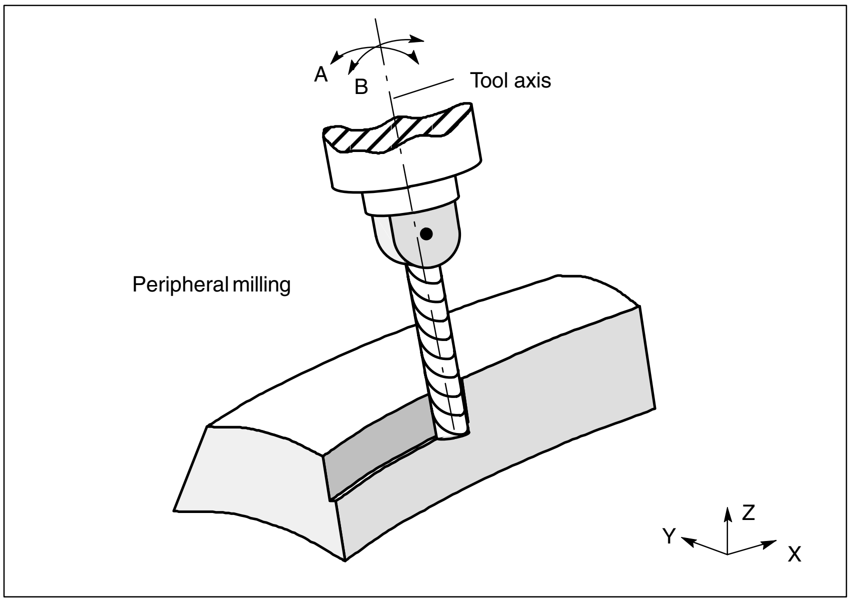



Fig. 2-1 Peripheral milling

**Insertion depth ISD**

The insertion depth of the tool for peripheral milling is programmed with program command ISD (InSertion Depth). It is therefore possible to change the position of the machining point on the peripheral surface of the tool.

ISD defines the distance between cutter tip FS and cutter construction point FH. Point FH is obtained by projecting the programmed machining point onto the tool axis. ISD is evaluated only when 3D TRC is active.

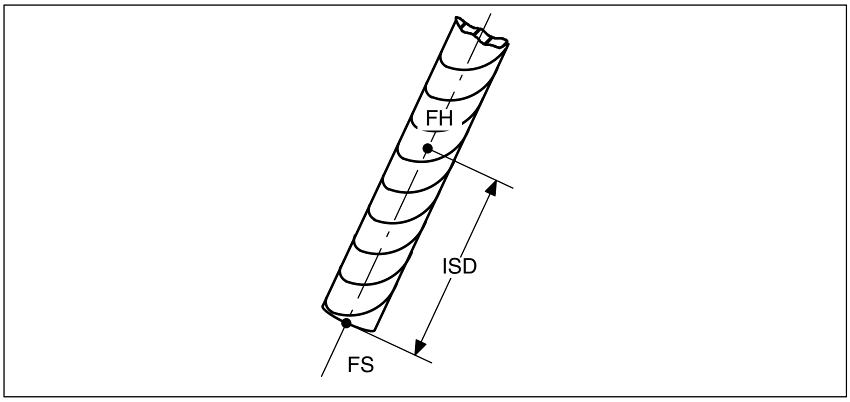



Fig. 2-2 Insertion depth

## 2.1.1 Corners for peripheral milling

**Outer corners/ inner corners**

Outer corners and inner corners must be treated separately. The terms “inner corner” and “outer corner” are dependent on the tool orientation. When the orientation changes at a corner, for example, the corner type may change **while** machining is in progress. Whenever this occurs, the machining operation is aborted with an error message.

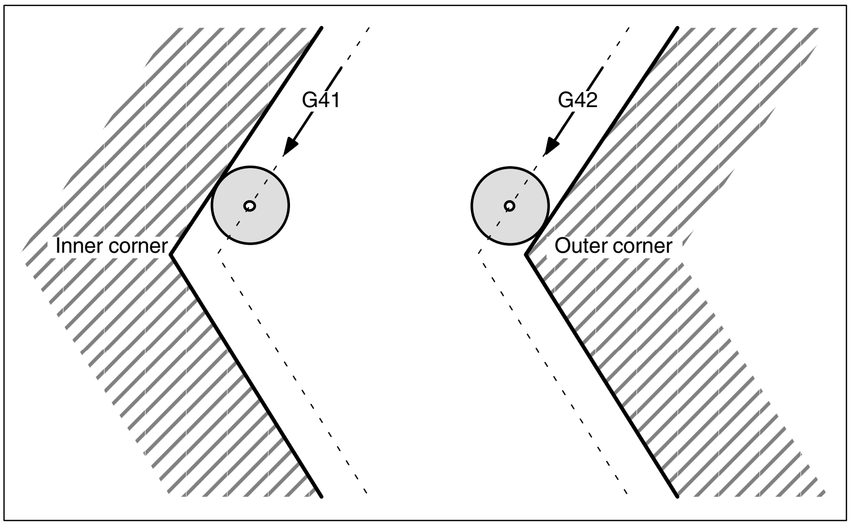



Fig. 2-3 Corner type

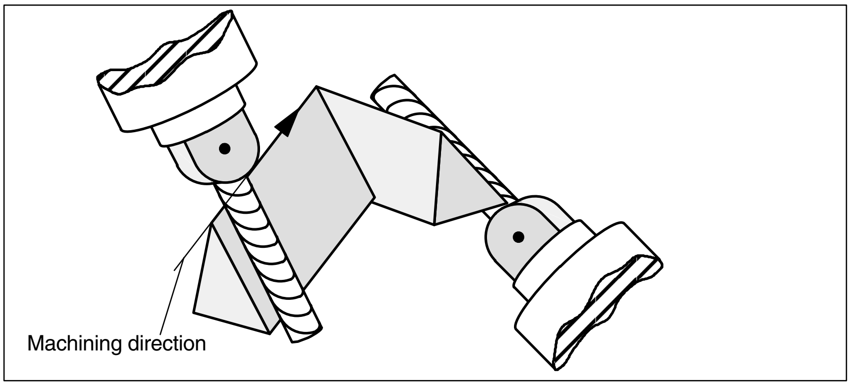



Fig. 2-4 Change of corner type during machining

## 2.1.2 Behavior at outer corners

Similarly to what happens for 21/2 D tool radius compensation, on outside corners a circle is inserted with G450 and the intersection of the offset curves approached with G451.
At transitions that are almost tangential, even if G450 is active, the behavior is as with G451 (limit angle can be set via MD). Conversely, a circle is inserted with active G451 too (behavior as with G450), if there is no intersection or when the corner angle exceeds a specific value (MD).
If there is a change in orientation between the two traversing blocks, a circle is always inserted.

**G450**

Outside corners are treated as if they were circles with a 0 radius. The tool radius compensation acts on these circles in the same way as on any other programmed path.

The circle plane extends from the final tangent of the first block to the start tangent of the second block.

The orientation can be changed during block transition.

A change in orientation between two programmed blocks is executed either before the circle block or in parallel to it. Circles are always inserted. The command DISC is not evaluated.

**Programming**

- **ORIC** Orientation change and path motion in parallel (**ORI**entation Change **C**ontinuously)
- **ORID** Orientation change and path motion in succession (**ORI**entation Change **D**iscontinuously)

Program commands ORIC and ORID are used to determine whether changes in orientation programmed between two blocks are executed before the inserted circle block is processed or at the same time.

When the orientation needs to be changed at outer corners, the change can be implemented in parallel to interpolation or separately from the path motion. When ORID is programmed, the inserted blocks are executed first without a path motion (blocks with orientation changes, auxiliary function outputs, etc.). The circle block is inserted immediately in front of the second of the two traversing blocks which form the corner.

**ORIC**

If ORIC is active and there are two or more blocks with orientation changes (e.g. A2= B2= C2=) programmed between the traversing blocks, then the inserted circle block is distributed among these intermediate blocks according to the absolute changes in angle.

**Change in orientation**

The method by which the orientation is changed at an outer corner is determined by the program command that is active in the first traversing block of an outer corner.

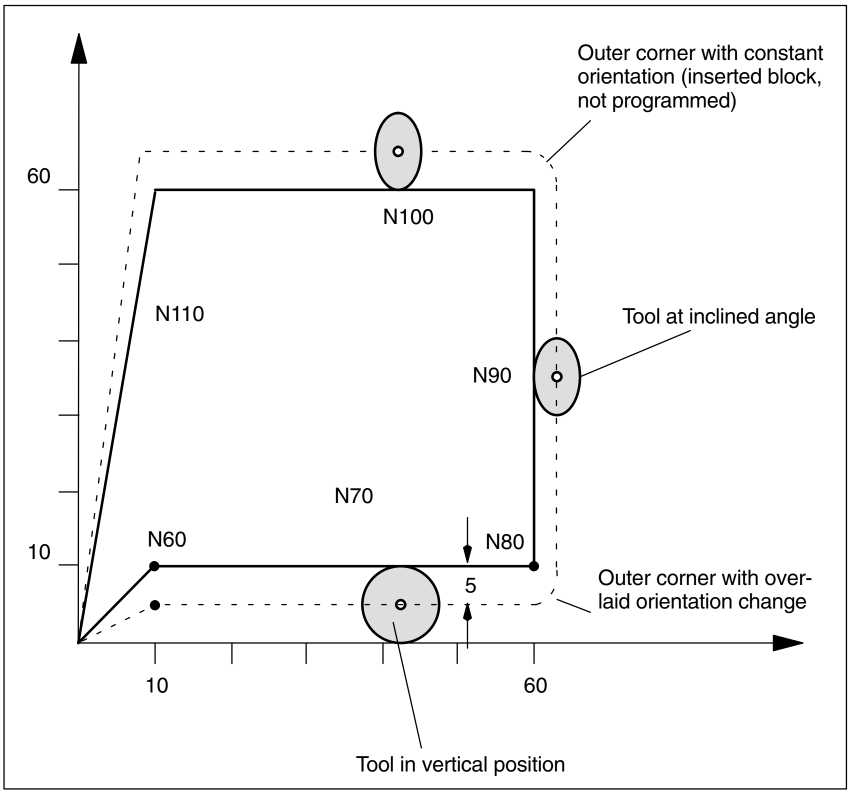



Fig. 2-5 **ORIC** Change in orientation and path motion in parallel

**Example:**

```
N10 A0 B0 X0 Y0 Z0 F5000
N20 T1 D1                     ;Radius=5
N30 TRAORI(1)                 ;Transformation selection
N40 CUT3DC                    ;3D TRC selection
N50 ORIC
N60 G42 X10 Y10               :TRC selection
N70 X60
N80 A3=1 B3=0 C3=1            ;Change in orientation at outer corner
N90 Y60                       ;formed by N70 and N90
N100 X10
N110 G40 X0 Y0
N120 M30
```

The circular motion and change in orientation are executed in parallel in block N80 (ORIC active).

**Exception**

Intermediate blocks without traversing and orientation motions are executed at the programmed positions, e.g. auxiliary functions.

**Example:**

```
...
N70 X60
N75 M20                       ;Auxiliary function call
N80 A3=1 B3=0 C3=1            ;Change in orientation at outer corner
N90 Y60                       ;formed by N70 and N90
...
```

Blocks N75 and N80 are executed after N70. The circle block is then executed with the current orientation.

**ORID**

If ORID is active, then all blocks between the two traversing blocks are executed at the end of the first traversing block. The circle block with constant orientation is executed immediately before the second traversing block.

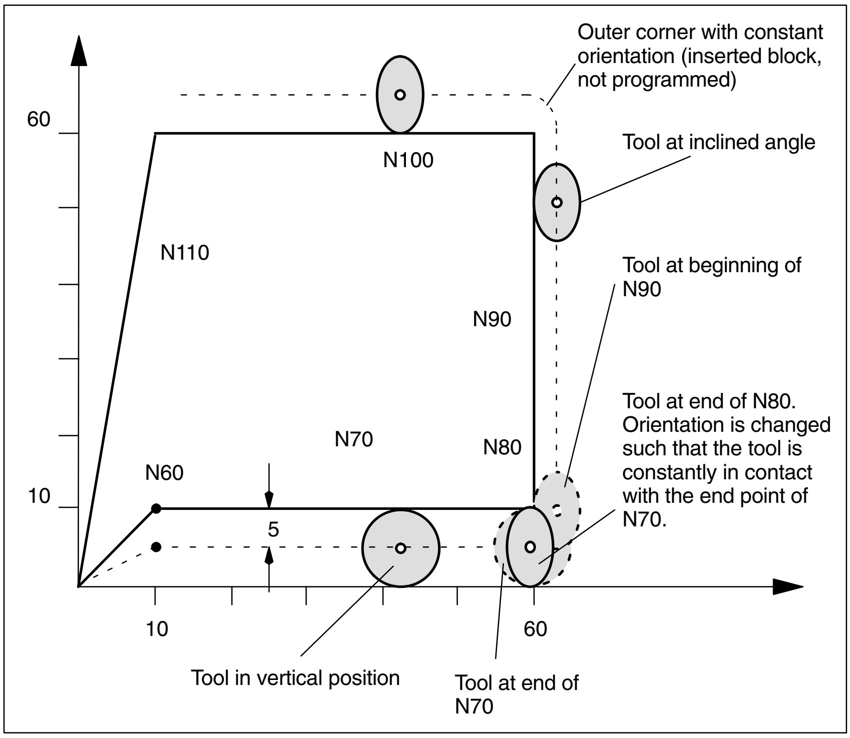



Fig. 2-6 **ORID**, change in orientation and path motion as successive actions

**Example:**

```
N10 A0 B0 X0 Y0 Z0 F5000
N20 T1 D1                      ;Radius=5
N30 TRAORI(1)                  ;Transformation selection
N40 CUT3DC                     ;3D TRC selection
N50 ORID
N60 G42 X10 Y10                :TRC selection
N70 X60
N80 A3=1 B3=0 C3=1             ;Change in orientation at outer corner
N90 Y60                        ;formed by N70 and N90
N100 X10
N110 G40 X0 Y0
N120 M30
```

**Note**

The command DISC is not evaluated.

**G451**
**(SW 5 and higher)**

The intersection is determined by extending the offset curves of the two participating blocks and defining the intersection of the two blocks at the corner in the plane perpendicular to the tool orientation. If no such intersection is available, a circle is inserted.

If an intersection is found in the plane perpendicular to the tool, this does not mean that the curves also intersect in space. Rather the curves in the direction of the tool longitudinal axis are considered, which are generally a certain distance apart. The positional offset is eliminated over the entire block length in direction of the tool.

The way this offset is processed in tool direction at outside corners is the same as for inside corners.

**No intersection procedure**

The intersection procedure is not used when at least one block containing a change to the tool orientation was inserted between the traversing blocks in question.
In this case a circle is always inserted at the corner.

**Blocks without traversing information**

Blocks without relevant traversing information (neither tool orientation nor position of geometry axes are changed) are permissible. The intersection procedure is applied to the adjacent blocks as if these intermediate blocks did not exist. In the same manner, tool direction motions in the tool direction may also be programmed in intermediate blocks.

### 2.1.3 Behavior at inner corners

**Collision monitoring**

With the 3D compensation function, only adjacent traversing blocks are taken into account in the calculation of intersections.
Path segments must be sufficiently long to ensure that the contact points of the tool do not cross the block limits into other blocks when the orientation changes at an inner corner.

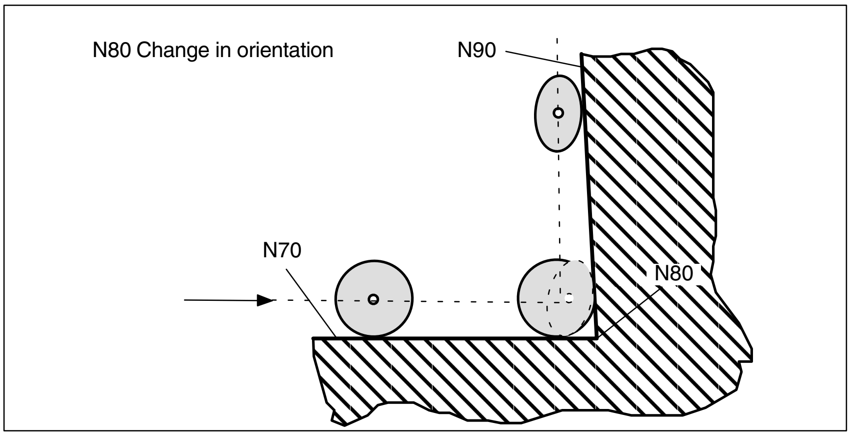



Fig. 2-7 The contact points of the tool must not cross the limits of block N70 or N90 as a result of the orientation change in block N80

**Example:**

```
N10 A0 B0 X0 Y0 Z0 F5000
N20 T1 D1                      ;Radius=5
N30 TRAORI(1)                  ;Transformation selection
N40 CUT3DC                     ;3D TRC selection
N50 ORID
N60 G42 X10 Y10                :TRC selection
N70 X60
N80 A3=1 B3=0 C3=1             ;Change in orientation at outer corner
                               ;formed by N70 and N90
N90 X10
N100 G40 X0 Y0
N120 M30
```

**Without change in orientation**

If the orientation is not changed at the block limit, then the contour need only be considered in the plane vertical to the tool axis. In this case, the tool cross-section is a circle which touches the two contours. The geometric relations in this plane are identical to those for $2^1/_2$D compensation.

**With change in orientation**

If the orientation changes on a block transition, the tool moves in the inner corner in such a way that it is constantly in contact with the two blocks forming the corner.

When the orientation changes in a block that is one of the two blocks forming the inner corner, then it is no longer possible to adhere to the programmed relationship between path position and associated orientation. This is because the orientation must reach its end value even though the path end position is not reached. This response is identical to the response of synchronous axes with $2^1/_2$ D tool radius compensation.

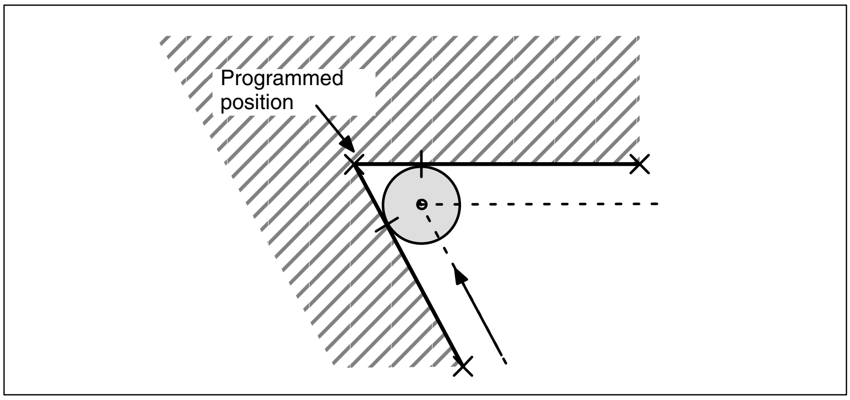



Fig. 2-8 Path end position and change in orientation at inner corners

**Change in insertion depth**

Generally speaking, the contour elements that form an inner corner are not positioned on the plane perpendicular to the tool. This means that the contact points between the two blocks and the tool are at different distances from the tool tip.

As a consequence, the insertion depth (ISD) changes abruptly from the 1st to the 2nd block at an inner corner.

To ensure that this difference in depth is not an abrupt step change, it is distributed continuously among the blocks involved during interpolation. The depth-compensating motion is executed in the current tool direction.

This solution prevents the contour from being violated by cylindrical tools if the tool is so long that the cutter contact point on the lateral surface of the cutter does not leave the range in which machining is possible.

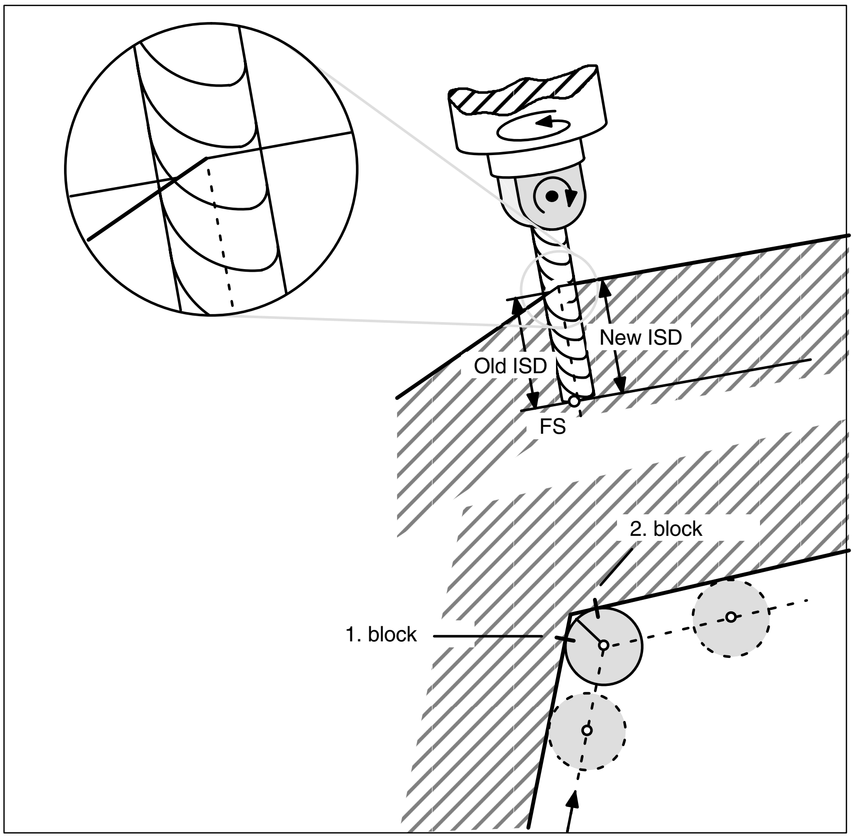



Fig. 2-9 Change in insertion depth

**Example of inner corners**

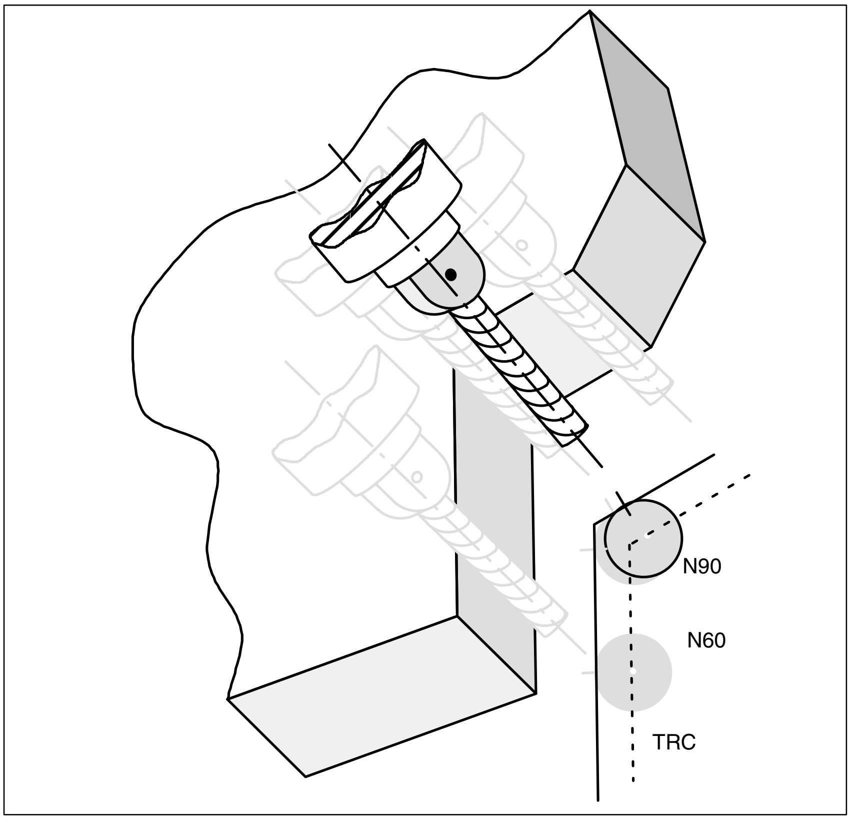



Fig. 2-10 Change in orientation at an inner corner

**Example:**

```
N10 A0 B0 X0 Y0 Z0 F5000
N20 T1 D1                      ;Radius=5
N30 TRAORI(1)                  ;Transformation selection
N40 CUT3DC                     ;3D TRC selection
N50 ORID
N60 G42 X10 Y10                 G451              :TRC selection
N70 Y60
N80 A3=1 B3=0 C3=1             ;Change in orientation at inner corner
                               ;formed by N70 and N90
N100 G40 X... Y...
...
N190 CDOF
N200 M30
```

## 2.2 Face milling

The face milling function allows surfaces with any degree or form of curvature to be machined. In this case, the longitudinal axis of the tool and the surface normal vector are more or less parallel. In contrast, the longitudinal axis and the surface normal vector of the surface to be machined in a peripheral milling operation are at right angles to one another.

Information about the surfaces to be machined is absolutely essential for face milling operations, i.e. a description of the linear path in space is not sufficient. The tool shape must also be known in order to implement the tool offset (the term “Tool radius compensation” is not appropriate in this case).

The relations in face milling are shown in Fig. 2-11.

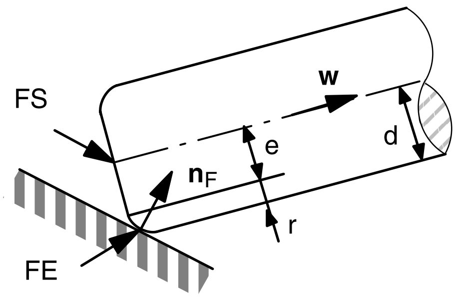



Fig. 2-11 Face milling with a torus

### 2.2.1 Cutter shapes

The following table lists the possible tool shapes that may be used for face milling. They are shown in Fig. 2-11 with their dimensions.

Table 2-1 Tool shapes for face milling

| Cutter type | Tool No. | d | r | a |
|---|---|---|---|---|
| Ball end mill (cylindrical die sinker) | 110 | >0 | X | X |
| Ball end mill (tapered die sinker) | 111 | >0 | >d | X |
| End milling cutter without corner rounding | 120, 130 | >0 | X | X |
| End mill with corner rounding (torus) | 121, 131 | >r | >0 | X |
| Bevel cutter without corner rounding | 155 | >0 | X | >0 |
| Bevel cutter with corner rounding | 156 | >r | >0 | >0 |

If a tool number other than any of those specified in the table above is used in the NC program, then the tool type is assumed to be a ball end mill (tool type 110). Tool parameters marked with an X in the tool table are not evaluated. A value other than zero is meaningless for the tool offset for face milling.

An alarm is output if tool data are programmed that violate the limits specified in the table above.

The shaft characteristics are not taken into account on any of the tool types. For this reason, the two tool types 120 (end mill) and 155 (truncated cone mill), for example, have an identical machining action since only the section at the tool tip is taken into account. The only difference between these tools is that the tool shape is represented differently (dimensions).

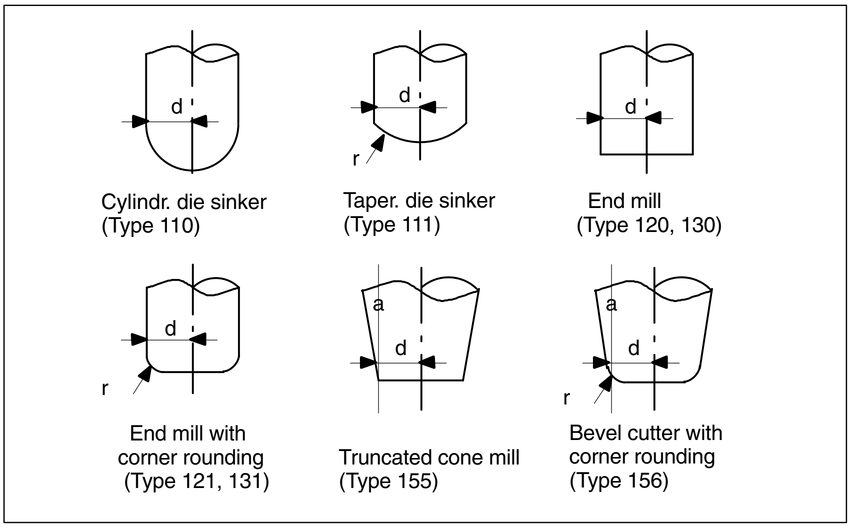



Fig. 2-12 Tool types for face milling

The tool data are stored under the following tool parameter numbers:

Table 2-2 Tool parameter numbers for tool data

| Tool data | Geometry | Wear |
|---|---|---|
| d | $TC_DP6 | $TC_DP15 |
| r | $TC_DP7 | $TC_DP16 |
| a | $TC_DP11 | $TC_DP20 |

**Note**

The geometry and wear values of a tool data are added.

The reference point for tool length compensation (also referred to as tool tip or Tool Center Point – TCP) on all tool types is the point at which the longitudinal axis of the tool penetrates the surface.

A new tool with different dimensions may be programmed only when the tool compensation is activated for the first time (i.e. on transition from G40 to G41 or G42) or, if the compensation is already active, only when G41 or G42 are reprogrammed.

In contrast to peripheral milling, therefore, there are no variable tool dimensions in one block.

This restriction applies only to the tool shape (tool type, dimensions d, r and a).

A change in tool involving only a change in other tool data (e.g. tool length) is permitted provided that no other restrictions apply. An alarm is output if a tool is changed illegally.

### 2.2.2 Orientation

The options for programming the orientation have been extended for 3D face milling.

The tool offset for face milling cannot be calculated simply by specifying the path (e.g. a line in space). The surface to be machined must also be known. The control is supplied with the information it requires about this surface by the surface normal vector.

The surface normal vector at the block beginning is programmed with A4, B4 and C4 and the vector at the block end with A5, B5 and C5. Components of the surface normal vector that are not programmed are set to zero. The length of a vector programmed in this way is irrelevant. A vector of zero length (all three components are zero) is ignored, i.e. the direction programmed beforehand remains valid, no alarm is generated.

If only the start vector is programmed (A4, B4, C4) in a block, then the programmed surface normal vector remains constant over the entire block. If only the end vector is programmed (A5, B5, C5), then large-circle interpolation is used to interpolate between the end value of the preceding block and the programmed end value. If both the start and end vectors are programmed, then interpolation takes place between both directions using the large-circle interpolation method. The fact that the start vector may be reprogrammed in a block means that the direction of the surface normal vector can change irregularly on a block transition. Irregular transitions of the surface normal vector always occur in cases where there is no tangential transition between the surfaces (planes) involved, i.e. if they form an edge.

Once a surface normal vector has been programmed, it remains valid until another vector is programmed. In the basic setting, the surface normal vector is set to the same values as the vector in the z direction. This basic setting direction is independent of the active plane (G17–G19). If ORIWKS is active, surface normal vectors refer to the active frame, i.e. when the frame is rotated, the vectors rotate simultaneously. This applies both to programmed orientations as well as to those derived from the active plane. If ORIWKS is active, the surface normal vectors are adjusted when a new frame becomes active. An orientation modified as the result of frame rotations is not returned to its original state on switchover from ORIWKS to ORIMKS.

It must be noted that the programmed surface normal vectors may not necessarily be the same as those used internally. This always applies when the programmed surface normal vector is not perpendicular to the path tangent. A new surface normal vector is then generated which is positioned in the plane extending from the path tangent to the programmed surface normal vector, but which is at right angles to the path tangent vector. This orthogonalization is necessary because the path tangent vector and surface normal vector for a real surface must always be perpendicular to one another. However, since the two values can be programmed independently, they may contain mutually contradictory information. Orthogonalization ensures that the information contained in the path tangent vector has priority over the data in the surface normal vector. An alarm is output if the angle between the path tangent vector and the programmed surface normal vector is smaller than the limit value programmed in machine data MC_CUTCOM_PLANENORMAL_PATH_LIMIT.

If a block is shortened (inner corner), then the interpolation range of the surface normal vector is reduced accordingly, i.e. the end value of the surface normal vector is not reached as it would be with other interpolation quantities such as, for example, the position of an additional synchronous axis.

In addition to the usual methods of programming orientation, it is also possible to refer the tool orientation to the surface normal vector and path tangent vector using the addresses LEAD (lead or camber angle) and TILT (side angle). The lead angle is the angle between the tool orientation and the surface normal vector. The side angle is the angle between the path tangent and the projection of the tool vector into the surface to be machined. Specification of the angle relative to the surface normal is merely an additional option for programming tool orientation at the block end. It does not imply that the lead and side angles reach their programmed values before the path end point is reached.

The final tool orientation is calculated from the path tangent, surface normal vector, lead angle and side angle at the block end. This orientation is always implemented by the end of the block, particularly in cases where the block is shortened (at an inner corner). If the omitted path section is not a straight line in a plane, the lead and side angles generally deviate from their programmed values at the path end point. This is because the orientation has changed relative to the surface normal vector or path tangent vector when the absolute orientation of the tool is the same as at the original path end point.

### 2.2.3 Compensation on path

**Tool longitudinal axis parallel to surface normal**

A special case must be examined with respect to face milling operations, i.e. that the machining point on the tool surface moves around. This may be the case on a torus cutter whenever surface normal vector $\mathbf{n}_F$ and tool vector $\mathbf{w}$ become collinear (i.e. the tool is at exact right angles to the surface) since it is not a single point on the tool that corresponds to this direction, but the entire circular surface on the tool end face. The contact point is not, therefore, defined with this type of orientation. A path point in which tool longitudinal axis and surface normal are parallel is therefore referred to below as a “singular point” or a “singularity”.

The above case is also meaningful in practical terms, e.g. in cases where a convex surface, which may have a vertical surface normal (e.g. hemisphere), must be machined with a perpendicular tool (e.g. face milling with constant orientation). The machining point on the contour remains fixed, but the machine must be moved to bring the machining point from one side of the tool to the other.

The problem described is only a borderline case (lead angle $\beta = 0$ and side angle $\gamma = 0$). If the lead angle $\beta = 0$ and the side angle $\gamma$ has a low value, then the tool must be moved very rapidly (in borderline case in steps) to keep the machining point resulting from the milling conditions close to the arc-line forming the end face, see Fig. 2-13.

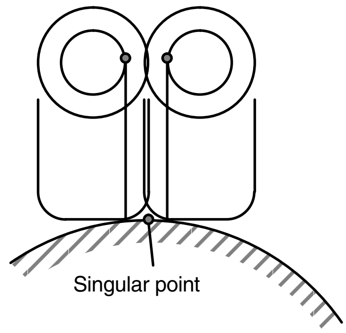



Fig. 2-13 Change in the machining point on the tool surface close to a point in which surface normal vector and tool orientation are parallel.

The problem is basically solved as follows: If the angle $\delta$ between the surface normal vector $\mathbf{n}_F$ and tool orientation $\mathbf{w}$ is smaller than a limit value (machine data) $\delta_{min}$, then the side angle $\gamma$ on tools with a flat end face (e.g. torus cutter or cylindrical mill) must be 0. This restriction does not apply to tool types with a spherical end face (e.g. ball end mill, die sinker) since angular changes close to the singular point do not lead to abrupt changes in the machining point on the surface of such tools. If $\delta$ now becomes 0, i.e. the sign of lead angle $\beta$ changes, the machining point moves from its current position to the opposite side of the tool. This movement is executed in an inserted linear block.

The machining operation is aborted with an alarm if an attempt is made to machine within the illegal angular range for side angle $\gamma$ (i.e. $\delta < \delta_{min}$ and $\gamma \neq 0$).

The insertion of linear blocks makes it necessary to split the original blocks at the singular points. The partial blocks created in this way are treated as if they were original which means, for example, that a concave path containing a “singularity” is treated like an inner corner, i.e. there is no contour violation. Each new partial block must contain at least one tool contact point since this is always calculated on the basis of adjacent traversing blocks.

“Singularities” do not just occur at isolated points, but along whole curves. This is the case, for example, if the curve to be interpolated is a plane curve (i.e. a curve with a constant osculating plane) and the tool is constantly aligned in parallel to the binormal vector, i.e. perpendicular to the osculating plane. A simple example is a circular arc in the x-y plane that is machined by a tool aligned in parallel to the z axis. On paths of this type, the tool offset is reduced to a tool length compensation, i.e. the tool is moved such that its tip FS is positioned on the programmed path.

On transition between singular and non-singular curves, linear blocks must be inserted in the same way as for isolated singular points such that the machining point on the tool can move from the tool tip FS to the periphery (on outer corners and convex surfaces) or the paths must be shortened to avoid contour violations (on inner corners and concave surfaces).

### 2.2.4 Corners for face milling

Two surfaces which do not merge tangentially form an edge. The paths defined on the surfaces make a corner. This corner is a point on the edge.

The corner type (inner or outer corner) is determined by the surface normal of the surfaces involved and by the paths defined on them.

The surface normals of the two surfaces forming the edge may point in opposite directions of the overall surface (the front edge of one surface is continued on the rear edge of the second surface), see also Fig. 2-14. Such transitions are not permissible and are rejected with an alarm.

The scalar product of the surface normal vector and (possibly variable) tool orientation on one corner/path must be positive at each point, i.e. it is not permissible to machine from the rear face of the surface. Failure to observe this rule results in an alarm. The permissible ranges of validity of tool orientation for inner and outer corners are illustrated in Fig. 2-14. These ranges are further restricted by the condition that the angle between the surfaces to be machined and the “steepest” surface line of the tool surface must not be lower than a particular machine data setting. The “steepest” surface line is a line at angle a to the tool longitudinal axis (this line is in the same direction as the tool longitudinal axis on cylindrical tools). This restriction must be imposed to ensure that the contact point on the tool does not leave the permissible range.

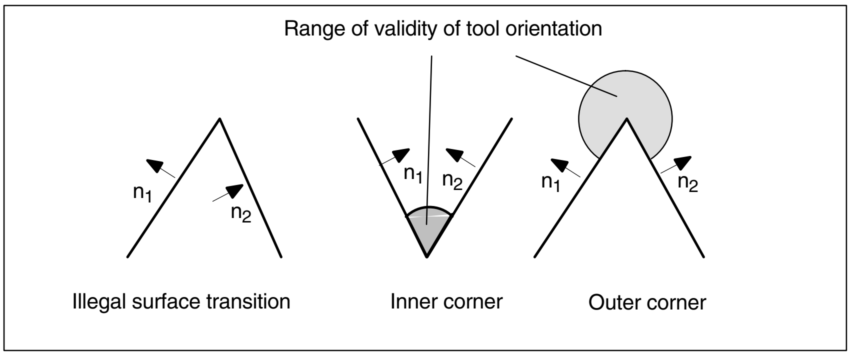



Fig. 2-14 Corners for face milling

It is possible to insert blocks that contain no motion commands (e.g. auxiliary function outputs) and/or that include motions of axes not involved in the path between two blocks which contain a path definition. Changes in orientation can also be programmed in such intermediate blocks. The only exception applies to the activation and deactivation of the 3D tool radius compensation function, i.e. intermediate blocks with orientation changes may not be inserted between the activation block and the first corrected block or between the last corrected block and the deactivation block. Other intermediate blocks are, however, permitted.

### 2.2.5 Behavior at outer corners

Outer corners are treated as if they were circles with a 0 radius. The tool radius compensation acts on these circles in the same way as on any other programmed path.

The circle plane extends from the final tangent of the first block to the start tangent of the second block.

The orientation can be changed during block transition.

A circle block is always inserted at an outer corner.

A change in orientation between two programmed blocks is executed either before the circle block or in parallel to it.

**Programming**

- **ORIC** Orientation change and path motion in parallel (**ORI**entation Change **C**ontinuously)
- **ORID** Orientation change and path motion in succession (**ORI**entation Change **D**iscontinuously)

Program commands ORIC and ORID are used to determine whether changes in orientation programmed between two blocks are executed before the inserted circle block is processed or at the same time.

When the orientation needs to be changed at outer corners, the change can be implemented in parallel to interpolation or separately from the path motion. When ORID is programmed, the inserted blocks are executed first without a path motion (blocks with orientation changes, auxiliary function outputs, etc.). The circle block is inserted immediately in front of the second of the two traversing blocks which form the corner.

**ORIC**

If ORIC is active and there are two or more blocks with orientation changes (e.g. A2= B2= C2=) programmed between the traversing blocks, then the inserted circle block is distributed among these intermediate blocks according to the absolute changes in angle.

**Change in orientation**

The method by which the orientation is changed at an outer corner is determined by the program command that is active in the first traversing block of an outer corner.

If the tool orientation at an outer corner is not constant, then the change in orientation is implemented in exactly the same way as described in Subsection 2.1.2 for peripheral milling operations.

## 2.2.6 Behavior at inner corners

The position of the tool in which it is in contact with the two surfaces forming the corner must be determined at an inner corner. The contact points must be on the paths defined on both surfaces. It is not usually possible to solve this problem exactly since, when the tool is moved along the path on the first surface, it normally touches a point on the second surface which is not on the path.

For this reason, the tool is not moved along the path on the first surface, but deviates from the path in such a way as to ensure that the distance between the contact points and the relevant contours in the position in which the tool contacts both surfaces is minimal, see also Fig. 2-15.

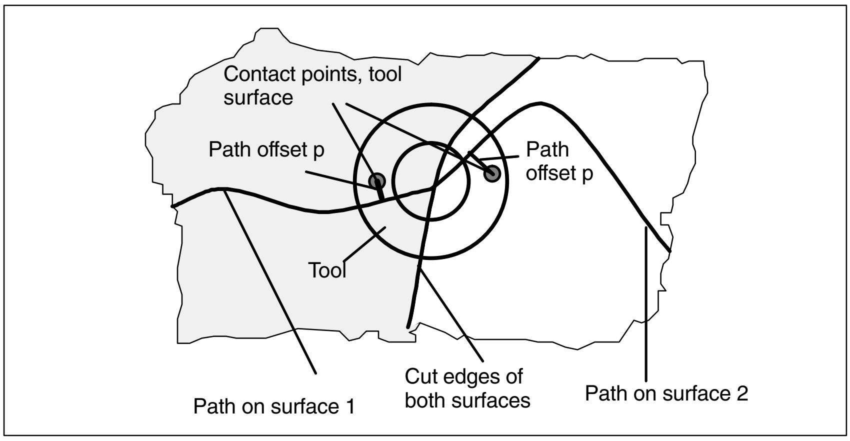



Fig. 2-15 Inner corner with face milling (view in direction of longitudinal axis of tool)

**Note**

The amount by which the contact points deviate from the programmed contour will generally be small since the explanatory example shown in Fig. 2-15 in which the machining point “changes” cutter side at an inner corner (the angular difference $\varphi$ about the tool longitudinal axis between the two contact points on the tool surface is about $180°$), is more likely to be an exception (see also Fig. 2-16 right). The angle $\varphi$ will normally stay almost constant so that there will be relatively small distance between the contact points on the tool surface (see also Fig. 2-16 left).

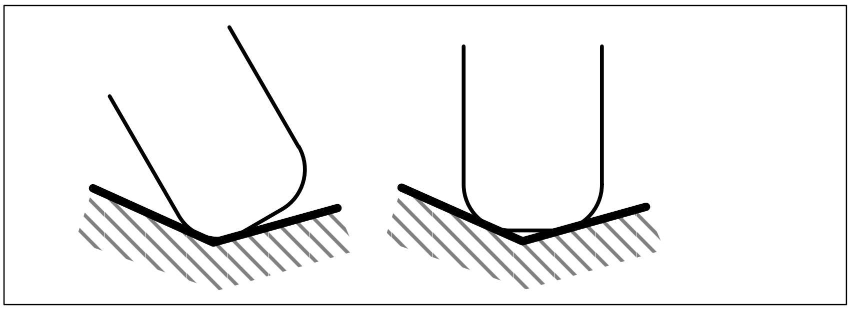


Fig. 2-16 Machining at inner corners

The difference between the programmed point on the path and the point actually to be approached (path offset p) is eliminated linearly over the entire block length. Differences resulting from inner corners at the block start and block end are overlaid. The current difference in a path point is always perpendicular to the path and in the surface defined by the surface normal vector.

If the tool orientation at an inner corner is not constant, the change in orientation is implemented in the same way as described in Subsection 2.1.3 for 3D peripheral milling, i.e. the tool is moved in the corner such that it contacts the two adjacent surfaces at the block start, block end and at two points $^1/_3$ and $^2/_3$ of the orientation change. A 3rd-degree polynomial is used to interpolate between these 4 points.

A variable tool orientation in a block that is shortened owing to an inner corner is also treated in the same way as described in Subsection 2.1.3 for 3D milling, i.e. the entire change in orientation is executed in the shortened block. Consequently, the functional relationship between path tangent, surface normal and tool orientation also changes. This results in new, previously nonexistent singularities or impermissible side angles (at points which are virtually singular) occurring in the shortened block. If this type of situation is detected during processing of an inner corner, the machining operation is aborted with an alarm. No block division takes place at the singular points since the compensatory motions this would involve frequently cause contour violations and the change in machining side on the tool is not generally intended or even foreseen by the user. The alarm is also output during examination of an inner corner if the singularity occurs in the second of the two blocks without the transition to the next block being considered. The system does not therefore detect that a block of this type will form an inner corner in conjunction with the following block and that the singularity would be eliminated again by the second block reduction.

The surface normal vector $\mathbf{n}_F$ is not affected by the reduction of a block. This means that in contrast to the tool orientation, the change in orientation that may need to be executed for this vector will not be imaged onto the reduced traversing interval. This is necessary because a surface other than that programmed would be machined. Unlike the tool orientation, no problems arise as the result of an abrupt change in the surface normal vector at a block transition since it does not reflect any axis motions.

Analogously to 3D peripheral milling, (see Subsection 2.1.3), the two traversing blocks that form an inner corner must contain contact points. There is no evaluation of several traversing blocks (i.e. no bottleneck detection), CDON/CDOF are not evaluated. If no contact point can be found, the machining operation is aborted with an alarm (risk of collision).

### 2.2.7 Monitoring of path curvature

The path curvature is not monitored, i.e. the system does not usually detect any attempt to machine a concave surface that is curved to such a degree that the tool currently in use is not capable of performing the machining operation. A possible exception are blocks that are split owing to a singularity. The transition between the two partial blocks created after the split is then treated like an inner corner. Except for such special cases, the user is responsible for ensuring that only tools that can machine along the entire contour without violating it are used.

## 2.3 Selection/deselection of 3D TRC

The following commands are used to select/deselect 3D tool radius compensation for peripheral milling or face milling.

- CUT3DC (peripheral milling)
- CUT3DFS (face milling)
- CUT3DFF (face milling)
- CUT3DF (face milling).

### 2.3.1 Selection of 3D TRC

**CUT3DC**

3D radius compensation for peripheral milling (only when 5-axis transformation is active).

**CUT3DFS**

3D tool offset for face milling with constant orientation. The tool orientation is defined by G17–G19 and is not affected by frames.

**CUT3DFF**

3D tool offset for face milling with constant orientation. The tool orientation is the direction defined by G17–G19 and, in some case, rotated by a frame.

**CUT3DF**

This programming command selects the 3D tool offset for face milling with change in orientation (only when 5-axis transformation is active).

**TRC selection**

The program commands used to select 3D TRC are the same as those for 2D TRC. G41/G42 specify the compensation on the left or right in the direction of motion (the response on selection of G41 and G42 for 3D face milling is identical). Tool radius compensation is deactivated with G40. The approach response is always controlled with NORM. The activation command must always be included in a linear block.

**Example:**

```
N10 A0 B0 X0 Y0 Z0 F5000
N20 T1 D1                   ;Radius=5
N30 TRAORI(1)               ;Transformation selection
N40 CUT3DC                  ;3D TRC selection (peripheral milling)
N50 G42 X10 Y10             ;TRC selection
N60 X60
N70 ....
```

**Intermediate blocks**

Intermediate blocks are permitted when 3D TRC is active. The specifications for 2D TRC apply equally to 3D TRC.

## 2.3.2 Deselection of 3D TRC

**Deselection**

The 3D tool radius compensation function is deselected in a linear block G0/G1 with geometry axes by means of
**G40**

**Example:**

```
N10 A0 B0 X0 Y0 Z0 F5000
N20 T1 D1                   ;Radius=5
N30 TRAORI(1)               ;Transformation selection
N40 CUT3DC                  ;3D TRC selection
N50 G42 X10 Y10             ;TRC selection
N60 X60
N70 G40 X100 Y0 Z20         ;3D TRC deselection
N80 ...
```

---

**Note**

The TRC is not deselected if D0 is programmed when the tool radius compensation is active.
The TRC is not deselected if the block containing the deselection command does not include any geometry axes in the current plane.

---

■

# Notes

# Supplementary Conditions

3

**Availability of "3D tool radius compensation" function**

This function is an option and available for

- SINUMERIK 840D with NCU 572/573, with SW 3.1 and higher (peripheral milling) and SW 3.2 (face milling).

■

# Data Descriptions (MD, SD)

4

## 4.1 Channel-specific machine data

| | | | | | |
|---|---|---|---|---|---|
| **21080**<br>MD number | **CUTCOM_PARALLEL_ORI_LIMIT**<br>Limit angle between path tangent and tool orientation for 3D tool radius compensation | | | | |
| Default setting: 3 | | Min. input limit: 1.0 | | Max. input limit: 89 | |
| Changes effective after RESET | | | Protection level: 2/7 | | Unit: Degrees |
| Data type: DOUBLE | | | | Applies from SW: 3.1 | |
| Meaning: | With 3D tool radius compensation, the angle between the path tangent and the tool orientation may not drop below a certain limit angle. This machine data specifies this angle (in degrees).<br>Generally speaking, the lower the value entered in this machine data, the greater the computing capacity required to check that the above conditions are fulfilled.Linear blocks with constant orientation are an exception. | | | | |

| | | |
|---|---|---|
| **21082**<br>MD number | **CUTCOM_PLANE_ORI_LIMIT**<br>Minimum angle between surface normal and tool orientation with side angle $\neq$ 0. | |
| Default setting: 3 | Min. input limit: 1.0 | Max. input limit: 89.0 |
| Changes effective after RESET | Protection level: 2/7 | Unit: Degrees |
| Data type: DOUBLE | Applies from SW: 3.2 | |
| Meaning: | This machine data applies to 3D face milling operations and specifies the minimum angle that must exist between the surface normal vector and the tool orientation on every point of the path if the applied side angle is not equal to zero and the tool is not a ball mill. Machining is otherwise aborted with an alarm if the angle is smaller than the value set here.<br>Generally speaking, the lower the value entered in this machine data, the greater the computing capacity required to check that the above conditions are fulfilled.<br>This data has no effect in linear blocks with constant orientation. The angle between the surface normal vector and tool orientation may be as small as desired in such cases, even if the side angle is not equal to zero. | |

| | | |
|---|---|---|
| **21084**<br>MD number | **CUTCOM_PLANE_PATH_LIMIT**<br>Minimum angle between surface normal vector and path tangent vector, for 3D face milling | |
| Default setting: 3 | Min. input limit: 1.0 | Max. input limit: 89.0 |
| Changes effective after RESET | Protection level: 2/7 | Unit: Degrees |
| Data type: DOUBLE | Applies from SW: 3.2 | |
| Meaning: | This machine data applies to 3D face milling operations and specifies the minimum angle that must exist between the surface normal vector and the path tangent vector on every point of the path. Machining is otherwise aborted with an alarm if the angle is smaller than the value set here.<br>Generally speaking, the lower the value entered in this machine data, the greater the computing capacity required to check that the above conditions are fulfilled. | |

■

# Signal Descriptions 5

None

# Example 6

### Example program for 3D peripheral milling:

```
; Definition of tool D1
$TC_DP1[1,1]=120                        ; Type (end mill)
$TC_DP3[1,1]= 20.                       ; Length compensation vector
$TC_DP6[1,1]=  8.                       ; Radius

N10 X0 Y0 Z0 T1 D1 F12000               ; Selection of tool
N20 TRAORI(1)                           ; Activation of transformation
N30 G42 ORIC ISD=10 CUT3DC G64 X30      ; Activation of 3D peripheral milling,
                                        ; Change in orientation at outer corners
                                        ; constant, insertion depth 10mm
N40 ORIWKS A30 B15                      ; Change in orientation at a corner
                                        ; through specification of axis positions
N50 Y20 A3=1 C3=1                       ; Traversing block with change in
                                        ; orientation
                                        ; Specification of orientation with
                                        ; direction vector
N60 X50 Y30                             ; Traversing block with constant
                                        ; orientation
N70 Y50 A3=0.5 B3=1 C3=5                ; Traversing block with change in
                                        ; orientation
N80 M63                                 ; Block without traversing information
N90 X0 ISD=20                           ; Traversing block with change in
                                        ; insertion depth
N100 G40 Y0                             ; Deactivation of tool radius compensation
N110 M30
```

### Example program for 3D face milling:

```
N10 ; Definition of tool d1
N20 $TC_DP1[1,1]=121                         ; Tool type (torus)
N30 $TC_DP3[1,1]=20.                         ; Length compensation
N40 $TC_DP6[1,1]=5.                          ; Radius
N50 $TC_DP7[1,1]=3.                          ; Smoothing radius
N60
N70
N80 X0 Y0 Z0 A0 B0 C0 G17 T1 D1 F12000       ; Tool selection
```

```
N90 TRAORI(1)
N100 B4=-1 C4=1                          ; Level definition
N110 G41 ORID CUT3DF G64 X10 Y0 Z0       ; Activate tool compensation
N120 X30
N130 Y20 A4=1 C4=1                       ; Outside corner, level redefinition
N140 B3=1 C3=5                           ; Orientation change with ORID
N150 B3=1 C3=1                           ; Orientation change with ORID
N160 X-10 A5=1 C5=2 ORIC
N170 A3=-2 C3=1                          ; Orientation change with ORIC
N180 A3=-1 C3=1                          ; Orientation change with ORIC
N190 Y-10 A4=-1 C4=3                     ; Level redefinition
N200 X-20 Y-20 Z10                       ; Inside corner with previous block
N210 X-30 Y10 A4=1 C4=1                  ; Inside corner, level redefinition
N220 A3=1 B3=0.5 C3=1.7                  ; Orientation change with ORIC
N230 X-20 Y30 A4=1 B4=-2 C4=3 ORID
N240 A3 = 0.5 B3=-0.5 C3=1               ; Orientation change
N250 X0 Y30 C4=1                         ; Path movement, new level,
                                         ; Orientation
                                         ; with relative programming
N260 BSPLINE X20 Z15                     ; spline start, relative programming
N270 X30 Y25 Z18                         ; Orientation remains active
N280 X40 Y20 Z13                         ; during spline.
N290 X45 Y0 PW=2 Z8
N300 Y-20
N310 G2 ORIMKS A30 B45 i-20 X25 Y-40 Z0    ; Helix, orientation with axis progr.
N320 G1 X0 A3=-0.123 B3=0.456 C3 =2.789 B4=-1 C4=5 B5=-1 C5=2 ; Path motion,
                                         ; Orientation, variable plane
N330 X-20 G40                            ; Deactivation of tool offset
N340 M30
```

■

# 7 Data Fields, Lists

## 7.1 Machine data

| Number | Identifier | Name | Reference |
|---|---|---|---|
| **General ($MN_ ...)** | | | |
| 18094 | MM_NUM_CC_TDA_PARAM | Number of TDA data | /FBW/ /S7/ |
| 18096 | MM_NUM_CC_TOA_PARAM | Number of TOA data that can be set up per tool and evaluated by the CC | /FBW/ /S7/ |
| 18100 | MM_NUM_CUTTING_EDGES_IN_TOA | Tool offsets per TOA module | S7 |
| 18110 | MM_NUM_TOA_MODULES | Number of TOA modules | S7 |
| **Channel-specific ($MC_ ...)** | | | |
| 20110 | RESET_MODE_MASK | Definition of control initial setting after power-up and RESET/parts program end | K2 |
| 20120 | TOOL_RESET_VALUE | Definition of tool for which tool length compensation is selected during power-up or on RESET or at parts program end as a function of MD 20110. | K2 |
| 20130 | CUTTING_EDGE_RESET_VALUE | Definition of tool edge for which tool length compensation is selected during power-up or on RESET or at parts program end as a function of MD 20110. | K2 |
| 20140 | TRAFO_RESET_VALUE | Definition of transformation block that is selected during power-up or on RESET or at parts program end as a function of MD 20110. | K2 |
| 20210 | CUTCOM_CORNER_LIMIT | Maximum angle for compensatory blocks with TRC | W1 |
| 20220 | CUTCOM_MAX_DISC | Response of TRC at outer corners | W1 |
| 20230 | CUTCOM_CURVE_INSERT_LIMIT | Minimum value for intersection calculation with TRC | W1 |
| 20240 | CUTCOM_MAXNUM_CHECK_BLOCKS | Blocks for predictive contour calculation with TRC | W1 |
| 20250 | CUTCOM_MAXNUM_DUMMY_BLOCKS | Number of blocks without traversing motion with TRC | W1 |
| 20270 | CUTTING_EDGE_DEFAULT | Selected tool edge after tool change | W1 |
| 20610 | ADD_MOVE_ACCEL_RESERVE | Acceleration reserve for superimposed motions | K1 |
| 21080 | CUTCOM_PARALLEL_ORI_LIMIT | Limit angle between path tangent and tool orientation with 3D tool radius compensation | |

| Channel-specific ($MC_ ...) | | | |
|---|---|---|---|
| 21082 | CUTCOM_PLANE_ORI_LIMIT | Minimum angle between surface normal and tool orientation for a side angle of ≠ 0 | |
| 21084 | CUTCOM_PLANE_PATH_LIMIT | Minimum angle between surface normal vector and path tangent vector for 3D face milling | |
| 22550 | TOOL_CHANGE_MODE | New tool with M function | W1 |
| 22560 | TOOL_CHANGE_M_CODE | M function for tool change | W1 |

## 7.2 Alarms

A detailed description of the alarms which may occur is given in
**References:** /DA/, Diagnostics Guide
or in the online help in systems with MMC 101/102/103.

■

# Index

**Cross-reference** Indicates the following:
Part of Description of Functions / Manual / Chapter / Section / Subsection / Page

## Numbers



## A



## B



## C

## D



## E

## F



## G



## H



## I



## K



## L



## M

## N



## O



## P



## R



## S

## T



## U



## V

# Notes

| | |
|---|---|
| To<br>SIEMENS AG<br>A&D MC BMS<br>P.O. Box 3180<br>D-91050 Erlangen, Germany<br>(Phone: ++49-(0)180-5050-222 [Hotline]<br>Fax: ++49-(0)9131-98-2176 [Documentation]<br>email: motioncontrol.docu@erlf.siemens.de) | **Suggestions**<br>**Corrections**<br>For Publication/Manual:<br>SINUMERIK 840D/840Di/810D<br>Special Functions (Part 3)<br>Manufacturer/Service Documentation |
| **From**<br>Name<br>Company/Dept.<br>Address<br><br>Phone: /<br>Fax: / | Description of Functions<br>Order No.: 6FC5 297-6AC80-0BP2<br>Edition: 11.02<br>Should you come across any printing errors when reading this publication, please notify us on this sheet.<br>Suggestions for improvement are also welcome. |

**Suggestions and/or corrections**

# Overview of SINUMERIK 840D/840Di/810D Documentation (11.2002)

## General Documentation

- *SINUMERIK* 840D/810D — Brochure
- *SINUMERIK* 840D/840Di/ 810D/ — Catalog Ordering Info. NC 60 ***)**
- *SINUMERIK SIROTEC SIMODRIVE* Accessories — Catalog Accessories NC-Z

## User Documentation

- *SINUMERIK* 840D/810D/ FM-NC — **AutoTurn**
  – Short Guide
  – Programming/ Setup
- *SINUMERIK* 840D/840Di/ 810D — Operator's Guide **– HT 6**
- *SINUMERIK* 840D/840Di/ 810D — **Diagnostics Guide *)**
- *SINUMERIK* 840D/840Di/ 810D — **Operator's Guide *)**
  – Short Guide
  – HMI Embedded
  – HMI Advanced

## User Documentation

- *SINUMERIK* 840D/840Di/ 810D — **Program. Guide**
  – Short Guide
  – Fundamentals ***)**
  – Advanced ***)**
  – Cycles
  – Measuring Cycles
  – ISO Turning/Milling
- *SINUMERIK* 840D/810D — Operator's Guide
  – **ManualTurn**
  – Short Guide ManualTurn
  – **ShopMill**
  – Short Guide ShopMill
  – **ShopTurn**
  – Short Guide ShopTurn
- *SINUMERIK* 840Di — **System Overview**

## Manufacturer/Service Documentation

- *SINUMERIK* — **Configuring** (HW) ***)**
  – 810D
  – 840D
- *SINUMERIK* 840D/840Di/ 810D — **Operator Components** (HW) ***)**
- *SINUMERIK* 840D/810D — Description of Functions
  **– ManualTurn**
  **– ShopMill**
  **– ShopTurn**
- *SINUMERIK* 840D/840Di/ 810D — Description of Functions **Synchronized Actions**

## Manufacturer/Service Documentation

- *SINUMERIK SIMODRIVE* 611D 840D/810D — Description of Functions **Drive Functions *)**
- *SINUMERIK* 840D/840Di/ 810D — **Description of Functions**
  – Basic Machine ***)**
  – Extended Functions
  – Special Functions
- *SINUMERIK* 840D/840Di/ 810D — Description of Functions **Tool Manage- ment**
- *SINUMERIK* 840D/810D — **Configuring Kit** HMI Embedded
- *SINUMERIK* 840D/810D — Description of Functions Operator Interface **OP 030**
- *SINUMERIK* 840D/810D — **IT Solutions**
  – Computer Link
  – Tool Data Information System
  – NC Data Management
  – NC Data Transfer
  – Tool Data Communication

## Manufacturer/Service Documentation

- *SINUMERIK SIMODRIVE* — Description of Functions SINUMERIK **Safety Integrated**
- *SINUMERIK* 840D — Description of Functions **Digitizing**
- *SINUMERIK SIMODRIVE* — **Installation & Start-Up Guide *)**
  – 810D
  – 840D/611D
  – HMI
- *SINUMERIK SIMODRIVE* 840D/840Di 810D 611D — **Lists *)**
- *SINUMERIK SIMODRIVE* 840D 611D — Description of Functions **Linear Motor**
- *SINUMERIK SIMODRIVE* 840D 611D — Description of Functions
  **– Hydraulics Module**
  **– Analog Module**
- *SINUMERIK SIMODRIVE SIROTEC* — **EMC** Guidelines

## Electronic Documentation

- *SINUMERIK SIMODRIVE* 840D/840Di/ 810D 611, Motors — **DOC ON CD *)** The SINUMERIK System

## Manufacturer/Service Documentation

- *SINUMERIK* 840D/840Di/ 810D — Description of Functions **ISO Dialects** for SINUMERIK
- *SINUMERIK* 840Di — **Manual** (HW + Installation and Start-Up)
- *SINUMERIK* 840D/810D — Description of Functions **Remote Diagnosis**
- *SINUMERIK* 840D/840Di/ 810D — **Manual @ Event**

***)** These documents are a minimum requirement

**Siemens AG**
Automation & Drives
Motion Control Systems
P.O. Box 3180, D-91050 Erlangen
Germany

www.ad.siemens.de


Order No: 6FC5297-6AC80-0BP2

Printed in Germany