SIEMENS

SINUMERIK 840D sl/840D/840Di
SINUMERIK 810D

Special Functions (Part 3)

Description of Functions

Valid for

Control	*Software Version*
SINUMERIK 840D sl/840DE sl	1.3
SINUMERIK 840D powerline	7.3
SINUMERIK 840DE powerline (export variant)	7.3
SINUMERIK 840Di	2.3
SINUMERIK 840DiE (export variant)	2.3
SINUMERIK 810D powerline	7.3
SINUMERIK 810DE powerline (export variant)	7.3
SINAMICS	2.3

08.05 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" columns.

Status code in the "Remarks" column:

A . . . . . New documentation.
B . . . . . Unrevised reprint with new Order No.
C . . . . . Revised edition with new status.
If the technical subject matter shown on the page has changed compared to the previous edition status, this is indicated by the changed edition status in the header of the respective page.

Edition	Order No.	Comments
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**
03.04	6FC5 297-7AC80-0BP0	**C**
08.04	6FC5 297-7AC80-0BP1	**C**
10.04	6FC5 297-7AC80-0BP2	**C**
08.05	6FC5 397-2BP10-0BA0	**A**

Further information is available in the Internet under:
http://www.siemens.com/motioncontrol

This publication was produced with Interleaf V 7

Order No. 6FC5 397-2BP10-0BA0
Printed in Germany

Siemens Aktiengesellschaft

Preface

SINUMERIK Documentation

The SINUMERIK documentation is subdivided into parts:

- General documentation
- User documentation
- Manufacturer/service documentation

Please contact your local Siemens office for more detailed information about other SINUMERIK 840D sl/840D/840Di/810D publications and publications that apply to all SINUMERIK controls (e.g. universal interface, measuring cycles, etc.).

An overview of publications, which is updated monthly and also provides information about the language versions available, can be found on the Internet at:

http://www.siemens.com/motioncontrol

Follow menu items "Support" → "Technical Documentation" → "Overview of Documents".

The Internet version of DOConCD (DOConWEB) is available at:

http://www.automation.siemens.com/doconweb

Target audience

This document is designed for machine tool manufacturers. It contains a detailed description of the functions offered by SINUMERIK controls.

Standard version

This Programming Guide describes the functionality afforded by standard functions. Extensions or changes made by the machine tool manufacturer are documented by the machine tool manufacturer.

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

Hotline

If you have any questions on the control, please get in touch with our hotline:

A&D Technical Support Phone: +49 (180) 5050-222
Fax: +49 (180) 5050-223
E-mail: mailto:adsupport@siemens.com
Internet: http://www.siemens.com/automation/support-request

If you have any questions about the documentation (suggestions for improvement, corrections), please send a fax to the following number:
Fax: +49 (9131) 98-63315
E-mail: mailto:motioncontrol.docu@siemens.com

Fax form: Refer to the reply form at the end of this manual

Internet address

http://www.siemens.com/motioncontrol

Objective

The Descriptions of Functions provide the information required for configuration and installation.

Target groups

The information contained in the Descriptions of Functions is designed for:

- Design engineers
- PLC programmers creating the PLC user program with the signals listed
- Startup engineers once the system has been configured and set up
- Maintenance personnel inspecting and interpreting status signals and alarms

Structure of the manual

This Function Manual is structured as follows:

- General contents
- Descriptions of Functions in alphabetical order in accordance with description of function codes
- Appendix with keyword index

Note

In addition to the keyword index, the **Basic Machine** Description of Functions (Part 1) also contains a list of abbreviations and terms.

The following information is provided on each page:
Part of Description of Functions / Publication / Chapter – Page

---

If you require information about a function, you will find the function and the code under which it is classified in the inside cover title of the manual.

If you need information about a certain term, please go to the section headed Index in the Appendix and look for the term concerned. The Description of Functions code, the chapter number and the number of the page on which you can find the information you need are listed in this section.

Chapters 4 and 5 of each Description of Functions contain definitions for “Activation, data format, input limits”, etc. for the various signals and data.

Safety information

This manual contains information which you should observe in order to ensure your own personal safety, as well to avoid material damage. Notes relating to your personal safety are highlighted in the manual by means of a warning triangle, no warning triangle appears in conjunction with notes that relate to property damage. The warnings appear in decreasing order of risk as given below.

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.

Correct usage

Please note the following:

Warning

The unit may be used only for the applications described in the catalog or the technical description, and only in combination with the equipment, components and devices of other manufacturers where recommended or permitted by Siemens. Correct transport, storage, installation and assembly, as well as careful operation and maintenance, are required to ensure that the product operates safely and without faults.

Further information

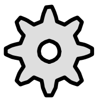
!

Important

This notice indicates important facts that must be taken into consideration.

Note

This symbol always appears in this documentation where further, explanatory information is provided.



Machine Manufacturer

This pictorial symbol appears in this document to indicate that the machine manufacturer can control or modify the function described. See machine manufacturer’s specifications.

Ordering Data Option

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

Technical information

Notations

The following notations and abbreviations are used in this document:

- PLC interface signals –> IS “signal name” (signal data)
  E.g.: – IS “MMC-CPU1 ready” (DB10, DBX108.2) i.e. the signal is stored in data block 10, data byte 108, bit 2.
  – IS “Feedrate/Spindle speed override” (DB31–48, DBB0) i.e. the signals for each axis/spindle are stored in data blocks 31 to 48, data block byte 0.
- Machine data –> MD: MD_NAME (German name)
- Setting data –> SD: SD_NAME (German name)
- The symbol “≙” means “corresponds to”.
- NEW_CONF (cf) – Reconfiguration of the PLC interface
  – “RESET” key on control unit, or
- RESET (re) “RESET” key on control unit or
- Immediately (im) after the value has been entered

Data types

The following data types are used in the control:

- DOUBLE
  Real values or integers
  input limits from +/–4.19*10^{-307} to +/–1.67*10^{308}
- DWORD
  Integers
  input limits from –2.147*10^{9} to +2.147*10^{9}
- BOOLEAN
  Possible input values: true or false/0 or 1
- BYTE
  Integers from –128 to +127
- STRING
  Comprising a max. of 16 ASCII characters (upper case letters, numbers and underscores)

Quantity framework

The explanations of the PLC interface in the individual Descriptions of Functions assume a theoretical maximum number of components:

- Mode groups (associated signals stored in DB11)
- Channels (associated signals stored in DB21, ...)
- Axes (associated signals stored in DB31, ...)

For details of the actual number of components which can be implemented with each software version, please refer to
References: /BU/, “Order Document”, Catalog NC 60

■

Notes

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

3-Axis to 5-Axis Transformation (F2)

Notes

1 Brief Description

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 traversing blocks.
The real-time transformation performs the calculation of the resulting motion of all 5 axes. The generated machining programs are therefore not machine specific. Kinematic-specific post-processors are not used for the 5-axis machining operation.
A selection of various transformations is available for adapting the control to various machine kinematics. Part program commands can be issued in operation to switch over between two transformations parameterized during start-up.

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

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

The calculation also includes 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 referred to as TCP-programming.

System variables for orientation

Part 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 see Sections 1.2 to 1.4; for a more detailed description, see 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, 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.

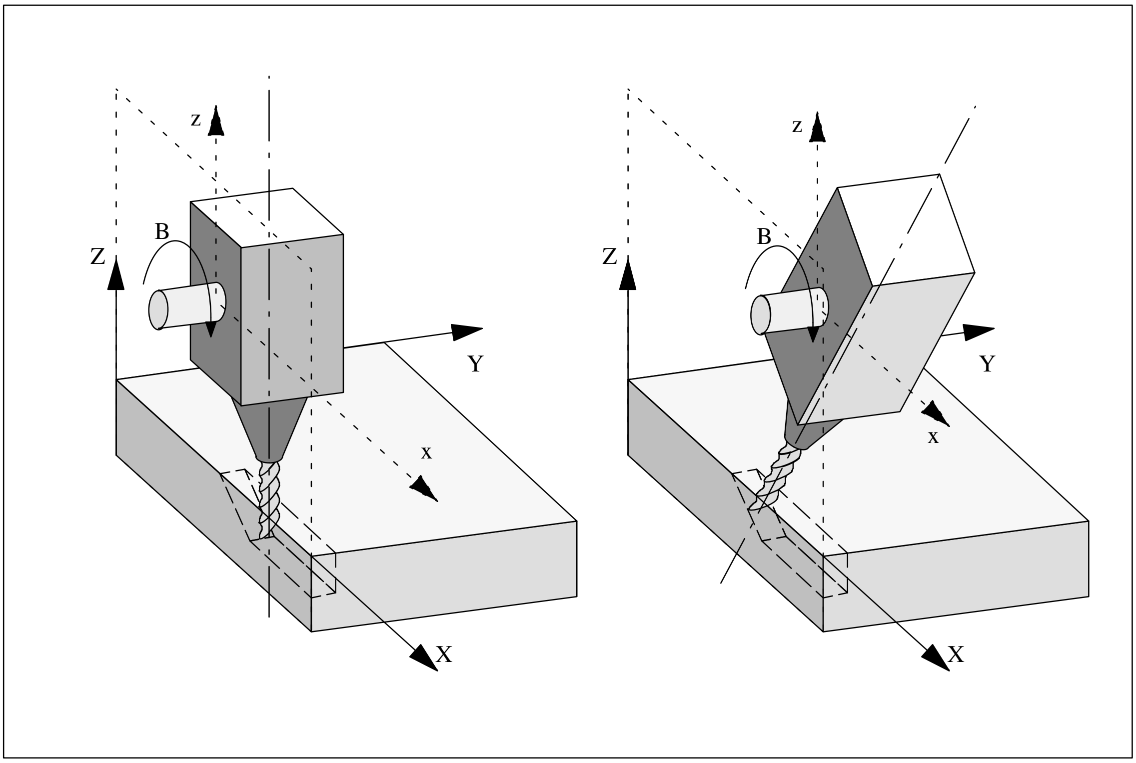


Fig. 1-1 Schematic diagram of 3-axis transformation

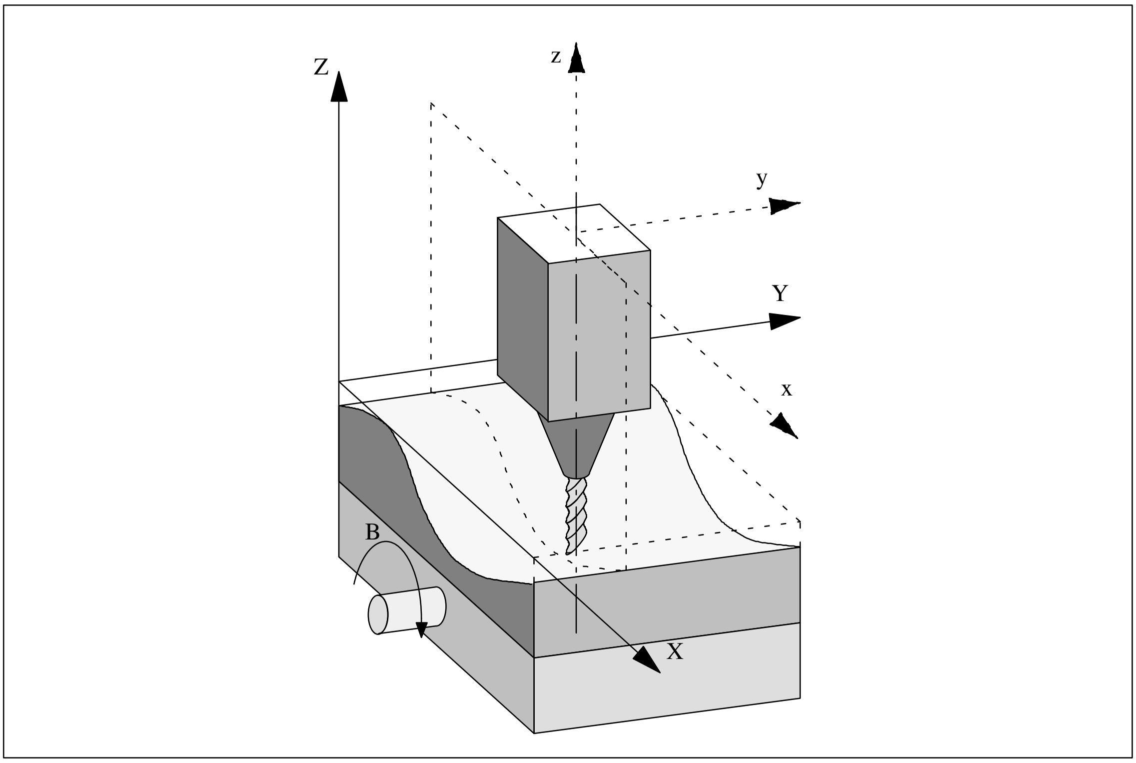


Fig. 1-2 Schematic diagram of 4-axis transformation with movable workpiece

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 transformation is possible.

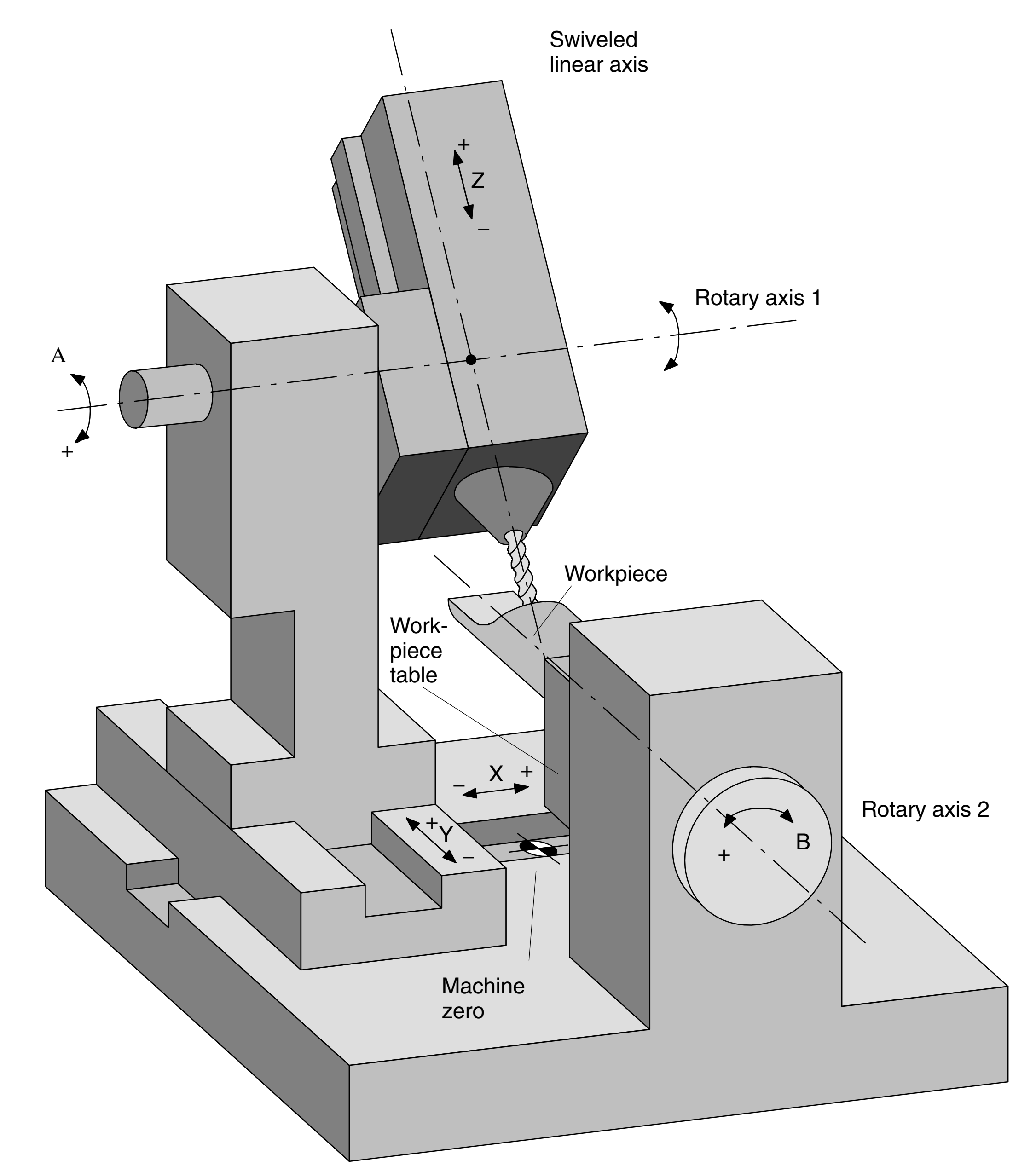


Fig. 1-3 Schematic diagram of a machine with swivelinglinear axis

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 arranged at a configurable angle (usually 45 degrees) allow the tool to swivel to positions in space that are limited to a half sphere in a 45-degree 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 transformation, 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 operating 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:

(for JOG motion)	basic coordinate system (BCS),
	workpiece coordinate system (WCS),
and the	tool coordinate system (TCS)

separately as reference system both for translation and for 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 insofar as it is no longer restricted with respect to the directions of rotary axes.

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

Detailed description given in Section 2.6.

1.9 Online 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 part program/softkey (SW 5.2 and higher)

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

In SW 5.2 and higher, you can also activate transformations MDs via the part 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 SW 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 in SW 6.3 and higher

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 1.12.

■

Notes

Detailed Description 2

Note

The transformations described below require that individual names are assigned to machine axes, channels and geometry axes when the transformation is active. Compare
MD 10000: AXCONF_MACHAX_NAME_TAB,
MD 20080: AXCONF_CHANAX_NAME_TAB,
MD 20060: AXCONF_GEOAX_NAME_TAB.
This is the only method of ensuring unique assignments.

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 means of 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 as well as three linear axes X, Y and Z also have 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. Initial tool position:
   – In negative Z direction.

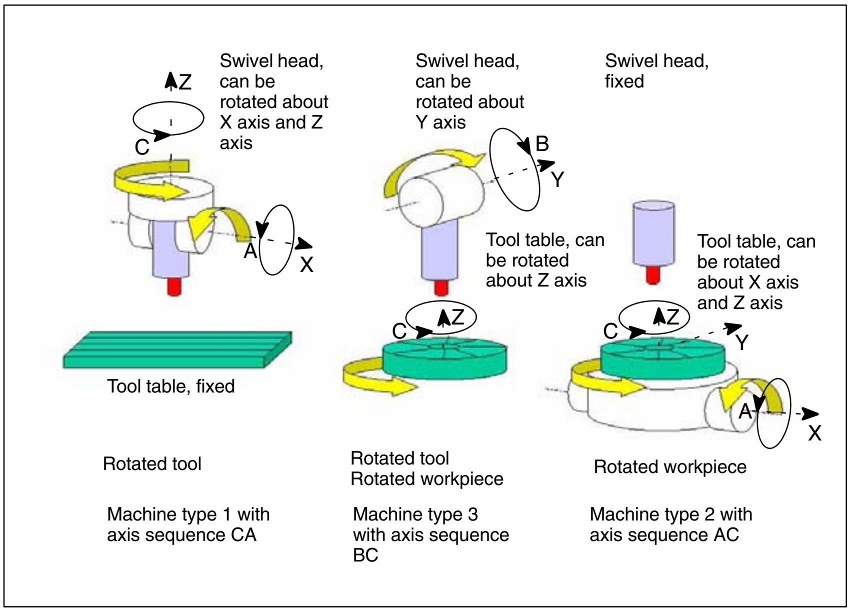


Fig. 2-1 Machine types for 5-axis transformation
Machine type 1 Two-axis swivel head
Machine type 2 Two-axis rotary table
Machine type 3 Single-axis swivel head and Single-axis rotary table

---

Note

Transformations that do not meet all of the above conditions are described in dedicated subsections:
3-axis and 4-axis transformation in 2.2
Swiveling linear axis in 2.3
Universal milling head in 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_10

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			
OB	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.
It is defined in MD: $MC_TRAFO_AXES_IN_1 ... $MC_TRAFO_AXES_IN_10.

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

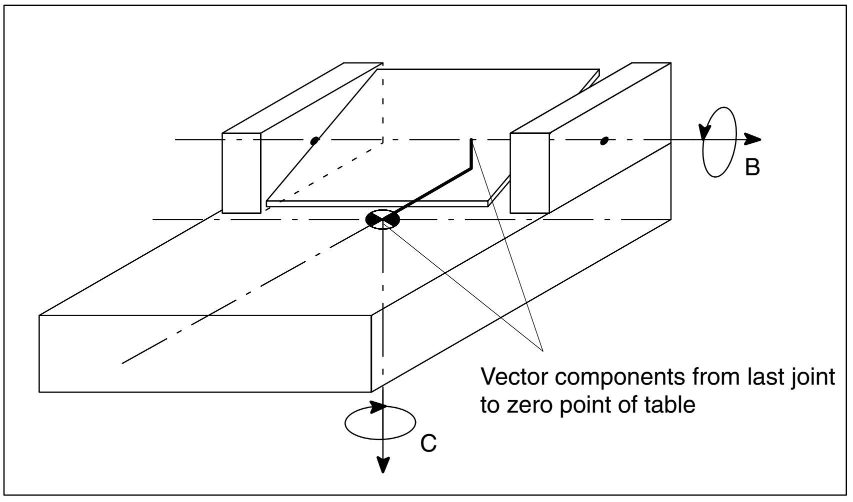


Fig. 2-2 Machine data $MC_TRAFO5_PART_OFFSET_1 for machine type 2

- 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 first to second joint (machine type 1 and 2).
Vector from machine zero to table joint
(machine type 3).

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

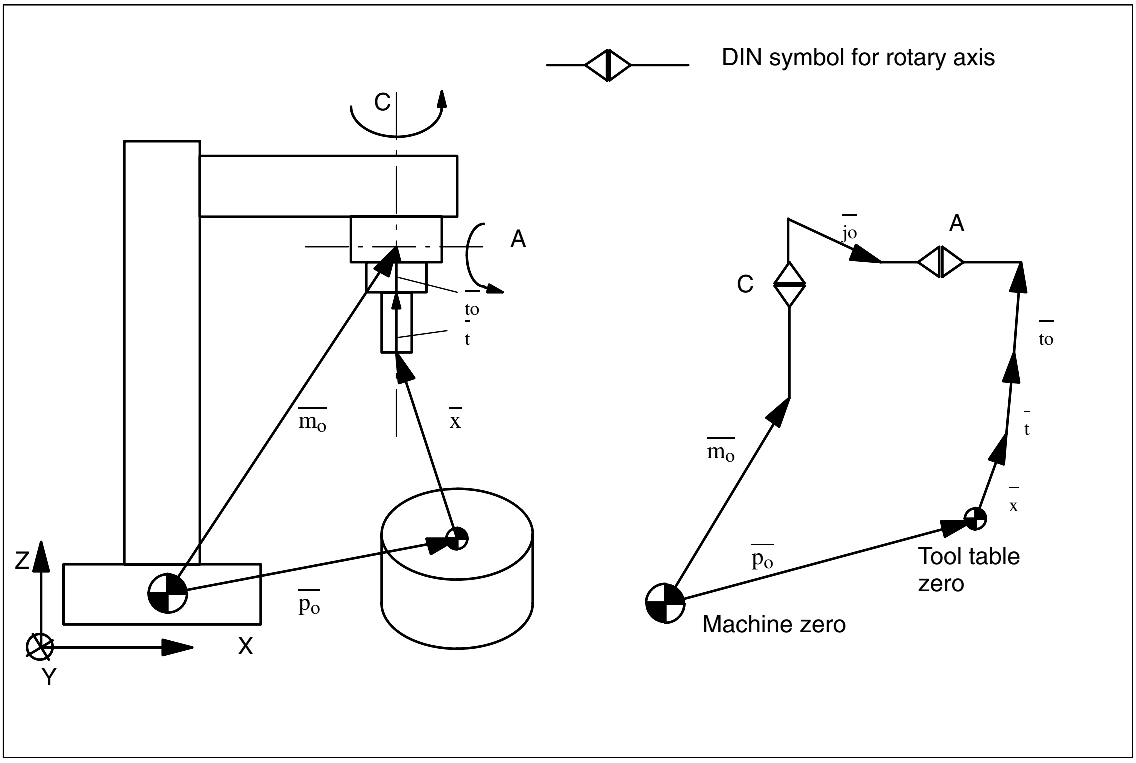


Fig. 2-3 Schematic diagram of CA kinematics, moved tool

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

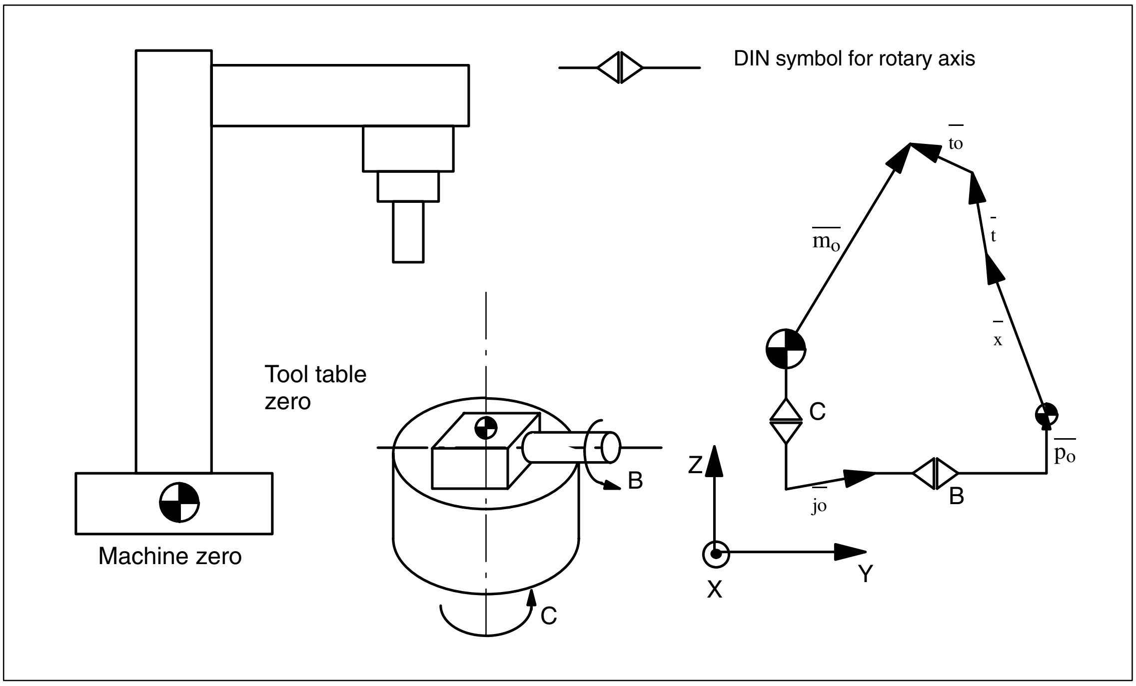


Fig. 2-4 Schematic diagram of CB kinematics, moved workpiece

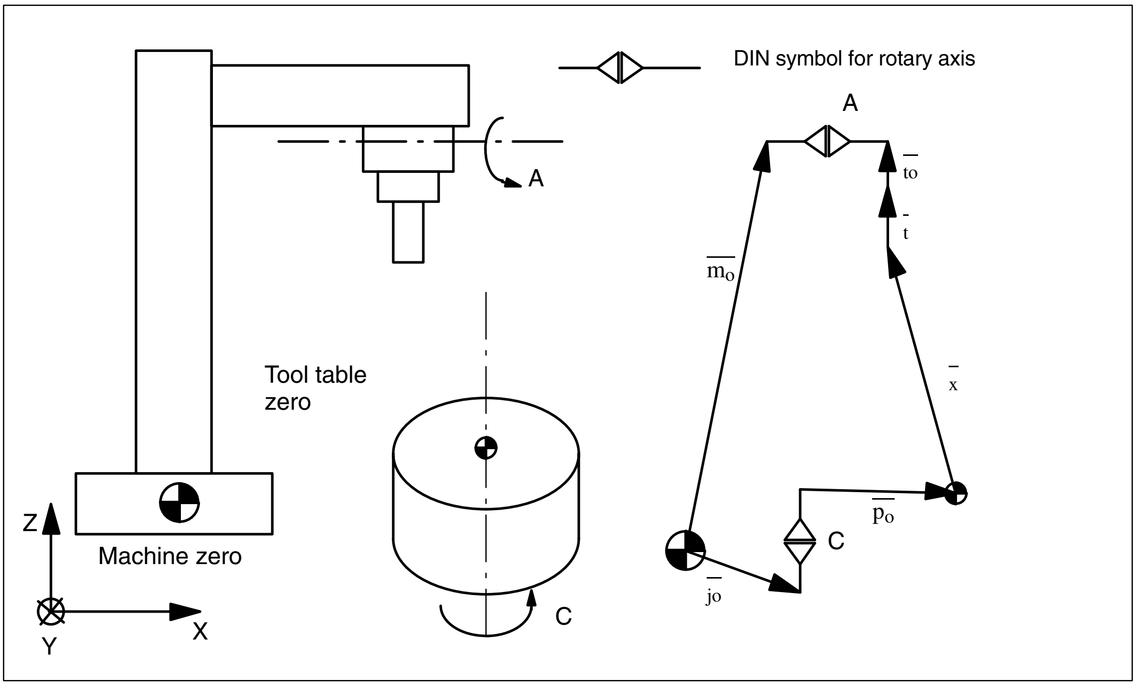


Fig. 2-5 Schematic diagram of AC kinematics, moved tool, moved workpiece

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

Ten transformation types per channel can be configured in machine data TRAFO_TYPE_1 to 10. Of these eight types, a maximum of two may be 5-axis transformations.

Activation

A description of how to activate 5-axis transformations appears in Section 2.5.

2.1.4 Tool orientation

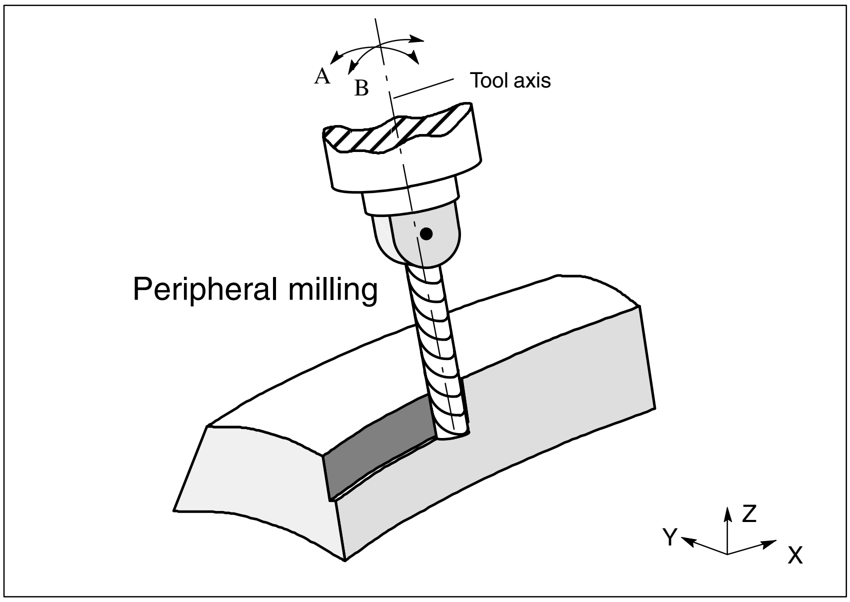


Fig. 2-6 Machining of workpieces with 5-axis transformation

Programming

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

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

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

Euler angle in	MD 10620: EULER_ANGLE_NAME_TAB
Direction vector in	MD 10640: DIR_VECTOR_NAME_TAB

The tool orientation can be located in any block. In particular, it can even be programmed alone in a block, resulting in a change of orientation in relation to the tool tip fixed in its relationship to the workpiece.

Euler or RPY

The MD 21100: ORIENTATION_IS_EULER can be used 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

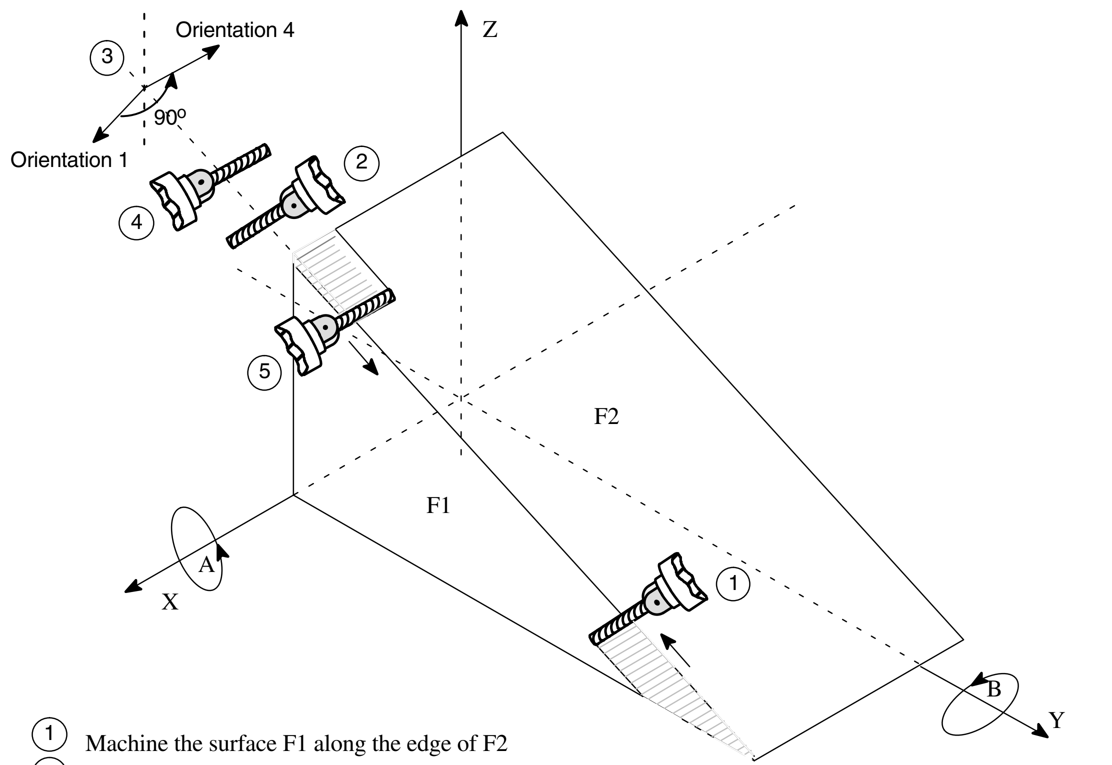
ORIWKS command

The tool orientation is programmed in the workpiece coordinate system (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 (MCS) and is therefore 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.

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



1. Machine the surface F1 along the edge of F2
2. Retract
3. Change in orientation B: +90° A: –30 °
4. Approach the surface F2
5. Machine the surface F2 along the edge of F1

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

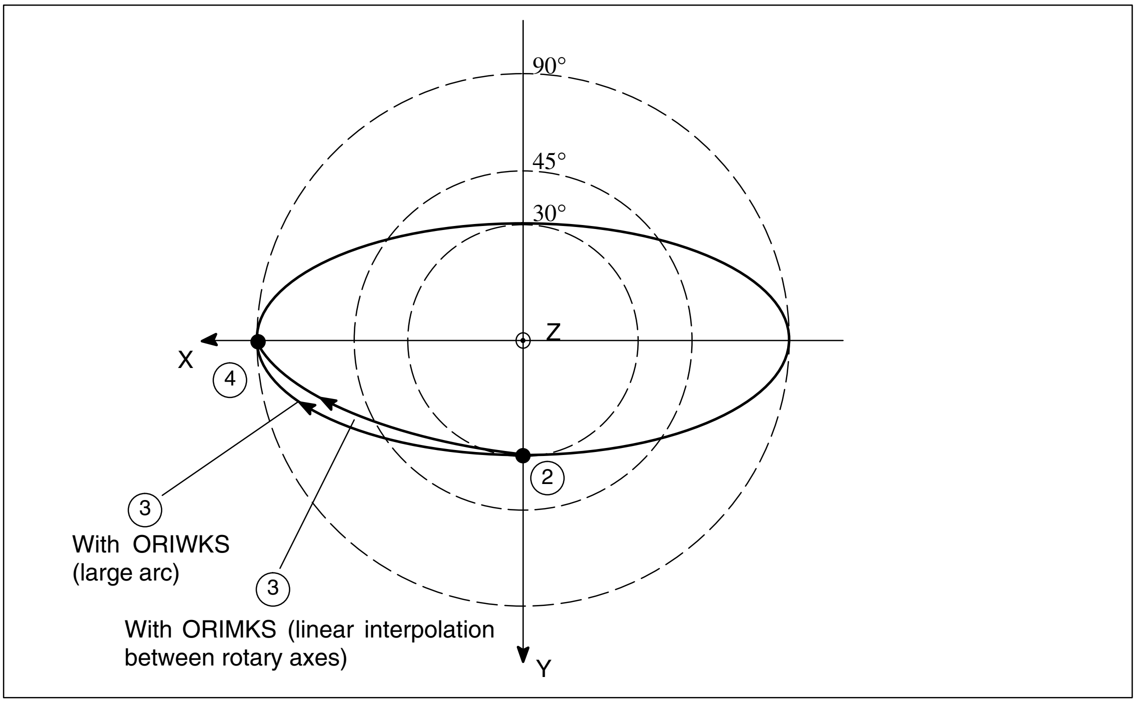


Fig. 2-8 Change in orientation while machining inclined edges

Initial setting is ORIMKS (SW 2 and higher).
MD 20150: GCODE_RESET_VALUES can be used to change the initial setting:
MD 20150: GCODE_RESET_VALUES [24] = 1 ⇒ ORIWKS is the initial setting
MD 20150: GCODE_RESET_VALUES [24] = 2 ⇒ ORIMKS is the initial setting

Illegal 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

Alarm 12130 “Illegal tool orientation” is output when Euler angles and direction vectors are selected. The NC program stops (this alarm can also occur in connection with G331, G332 and G63). 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 to be traversed are involved in the transformation. This applies irrespective of orientation programming.

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

If a transformation operation (activation, deactivation or transformation change) is carried out when tool radius compensation is active, alarm 14400 is output. However, conversely, no alarm is output if tool radius compensation is selected or deselected during active transformation.

Multiple input of tool orientation

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

N50 A3=1 B3=1 C3=1

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

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

error message 12240 “Channel X block Y tool orientation xx defined more than once” is displayed and the NC part program stops.

Tool orientation using orientation vectors

In SW 5.3 and higher, 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 2.10.1.

Note

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

References: /FB/, W1 “Tool Compensation, Orientable Toolholder”

2.1.5 Singular positions and handling

Extreme velocity overshoot

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 “Extreme velocity overshoot on one axis” is output. The programmed velocity is then reduced to a value, which does not exceed the maximum axis velocity.

Behavior atpole

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 MD 24630: 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 2.7.1 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 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 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 so 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.

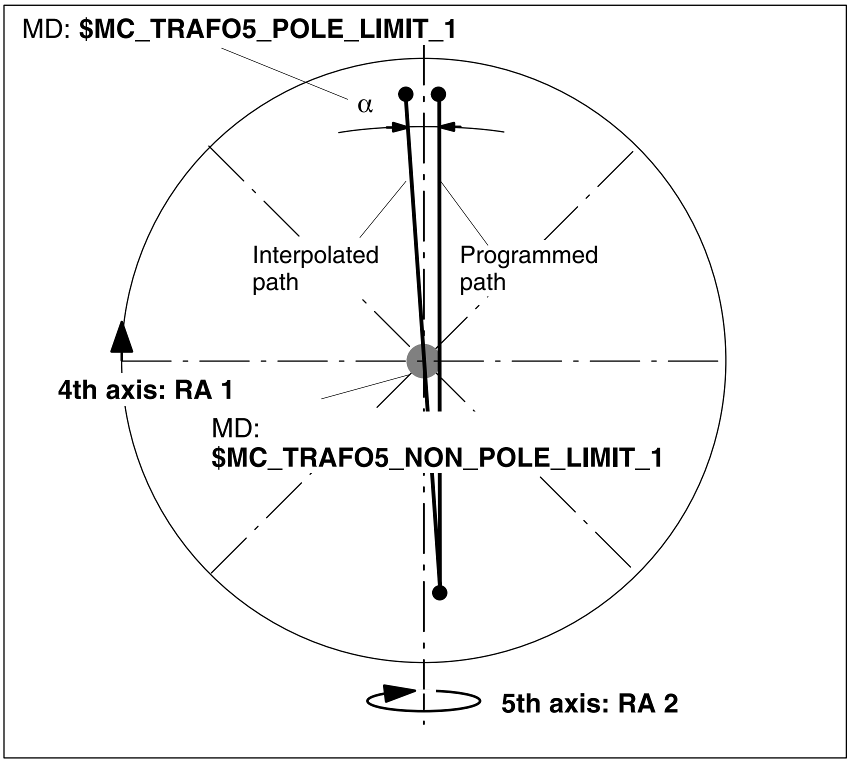


Fig. 2-9 5-axis transformation; orientation path in pole vicinity. Example for machine type 1: 2-axis swivel head with rotary axis RA 1 (4th axis of transformation) and rotary axis RA 2 (5th axis of transformation)

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 changes in orientation during large circle interpolation unless the starting orientation is equal to the pole orientation or approximates to it and the end orientation of the block is outside the tolerance circle defined in 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 decimally-coded.
The **unit digits** define the behavior if the start orientation precisely matches the pole orientation and the
ten digits define the behavior if the start orientation is within the tolerance circle defined in MD TRAFO5_NON_POLE_LIMIT1/2.
All settings are described in “Channel-specific machine data”.

2.2 3-axis and 4-axis transformations

Introduction

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 in 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 so that the change in orientation 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 swiveling linear axis forms a transformation group of its own. It is specified in machine data $MC_TRAFO_TYPE_n (n = 1, 2, 3, 4) using the following values:

Transf. 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 (SW 3.2 and higher)	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.

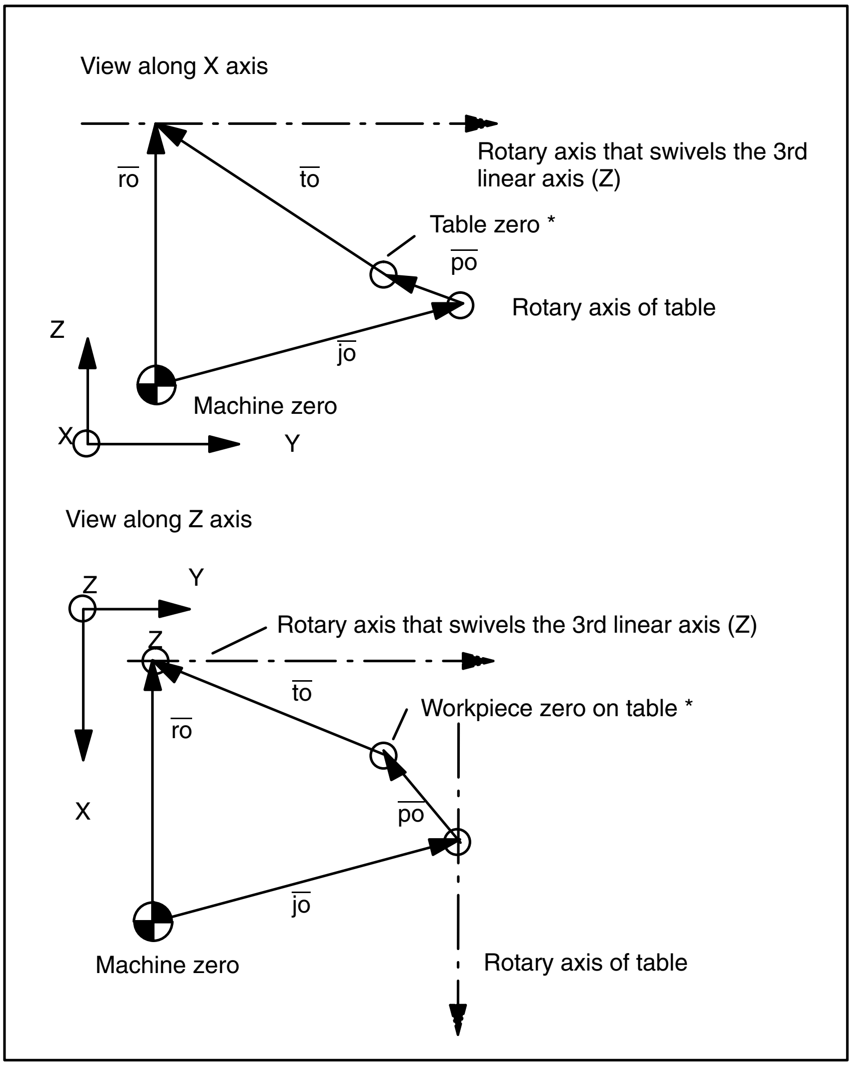


Fig. 2-10 Projections of the vectors to be set in MD

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.

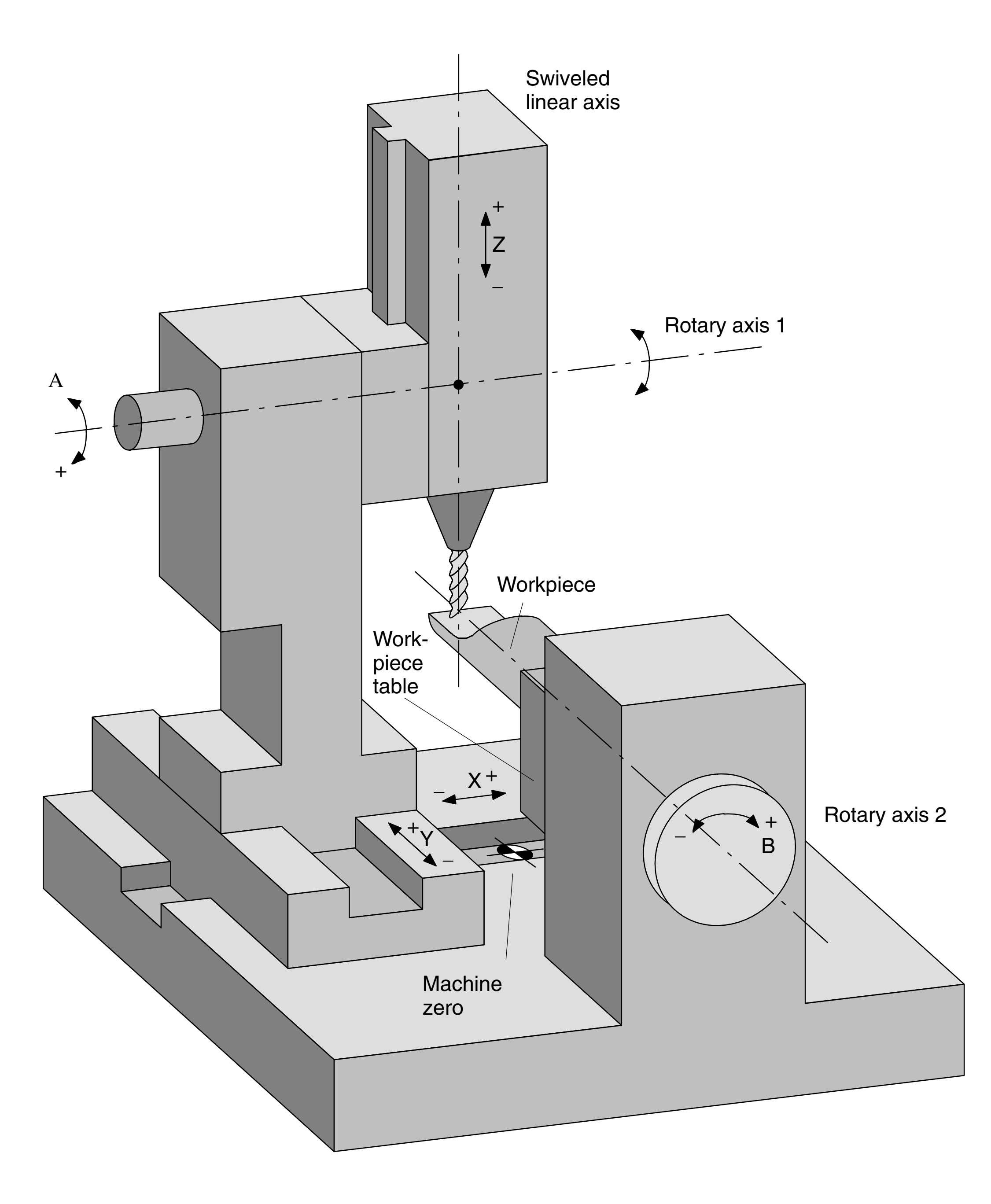


Fig. 2-11 Machine with swiveling linear axis in position zero

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

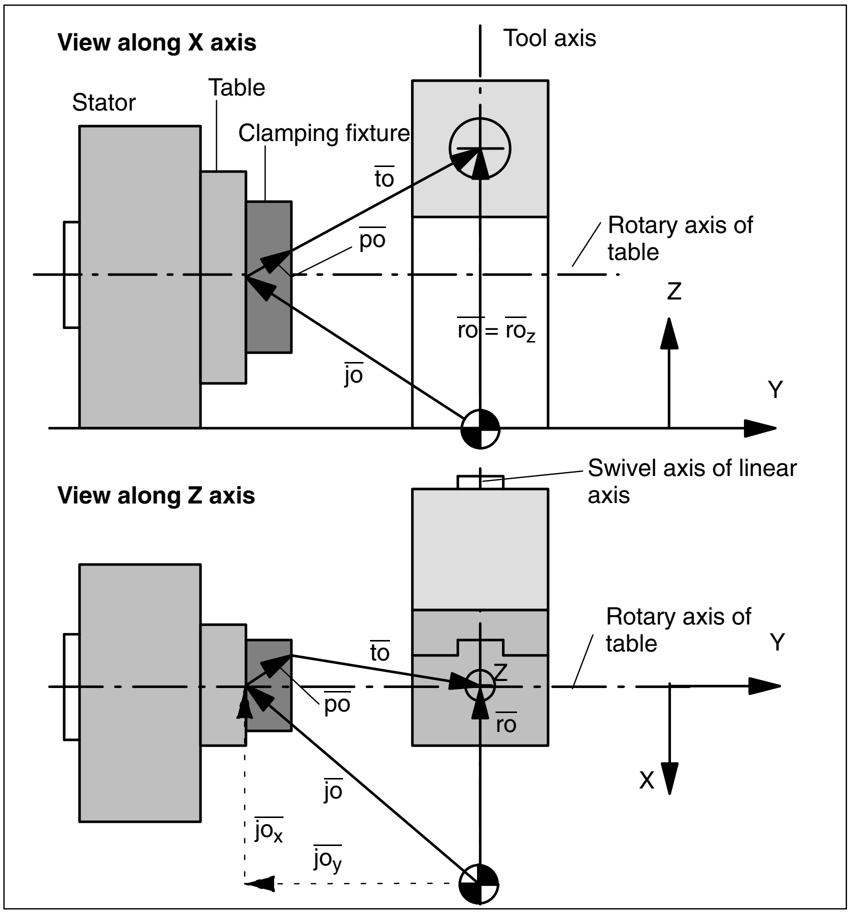


Fig. 2-12 Example of vector designations for MD settings for Fig. 2-11

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 accordingly:
  $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.

Machine type

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 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 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.

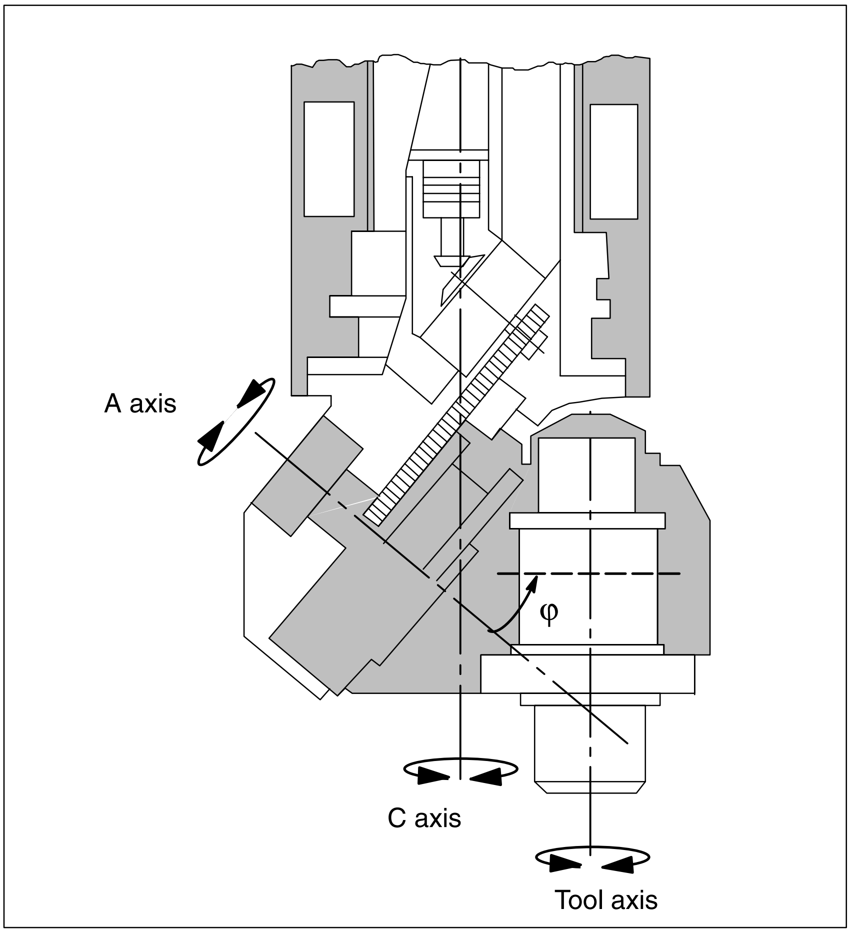


Fig. 2-13 Universal milling head

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 sequences of the rotary axes and the orientation direction of the tool at zero position are set for the different types of kinematics using machine data $MC_TRAFO_TYPE_1 ... $MC_TRAFO_TYPE_10.

Axis designation scheme

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

A is parallel to X	A' is positioned beneath the angle φ to the X axis
B is parallel to Y	B' positioned beneath the angle φ to the Y axis
C is parallel to Z	C' is positioned beneath the angle φ to the Z axis

Angle definition

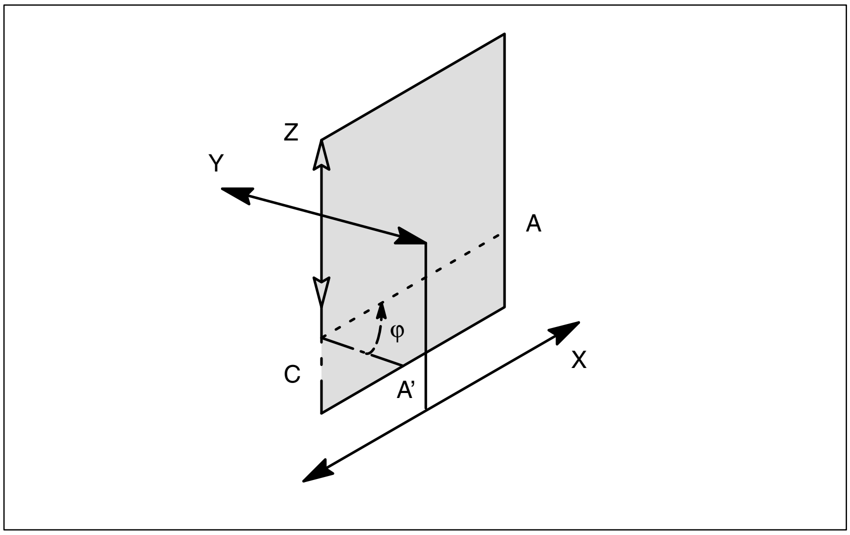


Fig. 2-14 Position of axis A'

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 indicating the type of transformation: 1: Transformation for universal milling head
7 6	0 32 64	00: Moving tool 01: Moving workpiece 10: Moving tool and workpiece
5 4	 0 8 16	Orientation of tool in zero position 00: X direction 01: Y direction 10: Z direction
3 2 1	 0 1 2 3 4 5	Axis sequence 000: AB' 001: AC' 010: BA' 011: BC' 100: CA' 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 SW 3.1, the others in SW 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 se- quence	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 in the zero position, tool orientation points in the Z direction.
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 zero position 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 so 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 3-axis to 5-axis transformation

Power on

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

Once the TRAORI(n) command has been executed and the transformation thus activated, 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”.

Power OFF

TRAFOOF or TRAFOOF() can be used to deactivate the currently active 3-axis to 5-axis transformation.
This sets IS “Transformation active” (DB21-30, DBX33.6) to “0”.

Switchover

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

RESET/ EOP

The behavior of the control with regard to 3-axis/5-axis transformations after run-up, end of program or RESET is determined by
MD 20110: RESET_MODE_MASK
Bit 7: Reset behavior of “active kinematic transformation”

Bit 7=0: Initial setting for active transformation after end of part program or RESET in acc. with 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 part program.

Option

The “5-axis transformation” function 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 TRAORI command, 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 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 and variants

2.6.1 Functionality

Scope of functions

The generic 5-axis transformation covers the functions implemented in SW up to and including 5.1 (see Section 2.1) for perpendicular rotary axes as well as 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 may not be possible to compensate the conventional transformation accuracy, e.g. if:

– The rotary axes are not exactly mutually perpendicular

– One of the two rotary axes is not positioned exactly parallel to the linear axes

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

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

Activation

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

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

2.6.2 Description of machine kinematics

Machine types

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

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

Configurations

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 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
Transform. types	24	40	56

Rotary axis direction

The direction of the rotary axes is defined in machine data $MC_TRAFO5_AXIS1_n (1st rotary axis) and $MC_TRAFO5_AXIS2_n (2nd rotary axis). n is 1 or 2 for the first/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 orientable toolholder. The absolute value of the vectors is insignificant; only the defined direction is relevant.

Example:

1. Rotary axis is 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 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 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 also missing. A zero is entered 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-axis 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

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

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

Priority:
High Programmed orientation,
Medium Tool orientation and
Low Basic orientation

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

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

Note

For further information about programmable tool orientation and basic tool orientation, please see

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

Comparison

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

Description	In software version 6.1 and higher
Position of the rotary axis	1. Can be any. 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 Parameterizing data for orientable toolholders (SW 7.2 and higher)

Application

Machine types on which the table or tool can be rotated can either be operated as true 5-axis machines or as conventional machines with orientable toolholders. In both cases, the machine kinematics are determined by the same data, which, due to the different parameters, previously had to be entered twice for

orientable toolholders via system variable and for transformations by means of machine data.

The new transformation type 72 can be used to specify that these two machine types use identical data.

Transformation type 72

MD 24100: TRAFO_TYPE_1 or MD 24200: TRAFO_TYPE_2 can be used to define a generic 5-axis transformation for transformation type 72 with kinematic data read from the data for an orientable toolholder. Its number provides the data via machine data
MD 24582: TRAFO5_TCARR_NO_1 for the first and
MD 24682: TRAFO5_TCARR_NO_2 for the second orientation transformation respectively. The corresponding transformation type can then be derived from the content of kinematic type with parameter $TC_CARR23. See Table 2-6

Table 2-6 Machine types for generic 5-axis transformation in SW 7.2 and higher

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

Note

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

The transformation machine data for the first orientation transformation listed in the tables below are equally valid for the second orientation transformation. All other machine data that may affect the transformation characteristics and **do not** appear in the tables below (e.g. MD 24110/24210: TRAFO_AXES_IN_1/2 or MD 24574/24674: TRAFO5_BASE_ORIENT_1/2) remain valid and effective. If in the tables below a second additive parameter appears in brackets for the parameters of the orientable toolholder (e.g. $TC_CARR24 (+ $TC_TCARR64)), the sum of both values will only be effective if the fine offset specified in setting data SD 42974:TOCARR_FINE_CORRECTION = TRUE is active when the data are transferred from the orientable toolholder.

Activation

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

Assignment for all types of transformation

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

Assignment for all types of transformation together identical	
MD 24100: TRAFO_TYPE_1 24 40 56	$TC_CARR23 = T P M
MD 24570: TRAFO5_AXIS1_1[0]	$TC_CARR7
MD 24570: TRAFO5_AXIS1_1[1]	$TC_CARR8
MD 24570: TRAFO5_AXIS1_1[2]	$TC_CARR9
MD 24572: TRAFO5_AXIS2_1[0]	$TC_CARR10
MD 24572: TRAFO5_AXIS2_1[1]	$TC_CARR11
MD 24572: TRAFO5_AXIS2_1[2]	$TC_CARR12
MD 24510: TRAFO5_ROT_AX_OFFSET_1[0]	$TC_CARR24 (+$TC_TCARR64)
MD 24510: TRAFO5_ROT_AX_OFFSET_1[1]	$TC_CARR25 (+$TC_TCARR65)
MD 24520: TRAFO5_ROT_SIGN_IS_PLUS_1[0]	TRUE*
MD 24520: TRAFO5_ROT_SIGN_IS_PLUS_1[1]	TRUE*

*) Machine data MD 24520/24620: TRAFO5_ROT_SIGN_IS_PLUS_1/2 are redundant. They are used to invert the direction of rotation of the assigned rotary axis. However, this can also be achieved by inverting the direction of axis vector $MC_TRAFO5_AXIS1/2_1/2. It is for this reason that there is no corresponding parameter for the orientable toolholder. For the purpose of absolute clarity,
MD 24520/24620: TRAFO5_ROT_SIGN_IS_PLUS_1/2 must be ignored.

Assignments for transformation type 24

Toolholder data assignments dependent on transformation type 24

Transformation type "T" (corresponds to MD24100: TRAFO_TYPE_1 = 24)	
MD 24500: TRAFO5_PART_OFFSET_1[0]	$TC_CARR1 (+$TC_TCARR41)
MD 24500: TRAFO5_PART_OFFSET_1[1]	$TC_CARR2 (+$TC_TCARR42)
MD 24500: TRAFO5_PART_OFFSET_1[2]	$TC_CARR3 (+$TC_TCARR43)
MD 24560: TRAFO5_JOINT_OFFSET_1[0]	$TC_CARR4 (+$TC_TCARR44)
MD 24560: TRAFO5_JOINT_OFFSET_1[1]	$TC_CARR5 (+$TC_TCARR45)
MD 24560: TRAFO5_JOINT_OFFSET_1[2]	$TC_CARR6 (+$TC_TCARR46)
MD 24550: TRAFO5_BASE_TOOL_1[0]	$TC_CARR15 (+$TC_TCARR55)
MD 24550: TRAFO5_BASE_TOOL_1[1]	$TC_CARR16 (+$TC_TCARR56)
MD 24550: TRAFO5_BASE_TOOL_1[2]	$TC_CARR17 (+$TC_TCARR57)

Assignments for transformation type 40

Toolholder data assignments dependent on transformation type 40

Transformation type "P" (corresponds to MD24100: TRAFO_TYPE_1 = 40)	
MD 24550: TRAFO5_BASE_TOOL_1[0]	$TC_CARR4 (+$TC_TCARR44)
MD 24550: TRAFO5_BASE_TOOL_1[1]	$TC_CARR5 (+$TC_TCARR45)
MD 24550: TRAFO5_BASE_TOOL_1[2]	$TC_CARR6 (+$TC_TCARR46)
MD 24560: TRAFO5_JOINT_OFFSET_1[0]	$TC_CARR15 (+$TC_TCARR55)
MD 24560: TRAFO5_JOINT_OFFSET_1[1]	$TC_CARR16 (+$TC_TCARR56)
MD 24560: TRAFO5_JOINT_OFFSET_1[2]	$TC_CARR17 (+$TC_TCARR57)
MD 24500: TRAFO5_PART_OFFSET_1[0]	$TC_CARR18 (+$TC_TCARR58)
MD 24500: TRAFO5_PART_OFFSET_1[1]	$TC_CARR19 (+$TC_TCARR59)
MD 24500: TRAFO5_PART_OFFSET_1[2]	$TC_CARR20 (+$TC_TCARR60)

Assignments for transformation type 56

Toolholder data assignments dependent on transformation type 56

Transformation type "M" (corresponds to MD24100: TRAFO_TYPE_1 = 56)	
MD 24560: TRAFO5_JOINT_OFFSET_1[0]	$TC_CARR1 (+$TC_TCARR41)
MD 24560: TRAFO5_JOINT_OFFSET_1[1]	$TC_CARR2 (+$TC_TCARR42)
MD 24560: TRAFO5_JOINT_OFFSET_1[2]	$TC_CARR3 (+$TC_TCARR43)
MD 24550: TRAFO5_BASE_TOOL_1[0]	$TC_CARR4 (+$TC_TCARR44)
MD 24550: TRAFO5_BASE_TOOL_1[1]	$TC_CARR5 (+$TC_TCARR45)
MD 24550: TRAFO5_BASE_TOOL_1[2]	$TC_CARR6 (+$TC_TCARR46)
MD 24558: TRAFO5_JOINT_OFFSET_PART_1[0]	$TC_CARR15 (+$TC_TCARR55)
MD 24558: TRAFO5_JOINT_OFFSET_PART_1[1]	$TC_CARR16 (+$TC_TCARR56)
MD 24558: TRAFO5_JOINT_OFFSET_PART_1[2]	$TC_CARR17 (+$TC_TCARR57)
MD 24500: TRAFO5_PART_OFFSET_1[0]	$TC_CARR18 (+$TC_TCARR58)
MD 24500: TRAFO5_PART_OFFSET_1[1]	$TC_CARR19 (+$TC_TCARR59)
MD 24500: TRAFO5_PART_OFFSET_1[2]	$TC_CARR20 (+$TC_TCARR60)

Example parameterization

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

```
MD 24100: TRAFO_TYPE_1 = 24  ; first 5-axis transformation
MD 24200: TRAFO_TYPE_2 = 72  ; second 5-axis transformation

                             ; parameterize data of 3rd
MD 24682: TRAFO5_TCARR_NO_2 = 3; orientable toolholder
```

2.6.5 Extension of the generic transformation to 6 axes

Application

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

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

Kinematics for the 6-axis transformation

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

machine data **$MC_TRAFO_TYPE_1 = transformation type**.

Table 2-7 Overview of machine types for the generic 6-axis transformation

Machine type	1	2	3	4
Swivel/rotatable	Tool	Workpiece	Tool/ workpiece	Tool/ workpiece
Transform. types	24	40	56	57
Orientation in space, rotation of the axes	Unchanged. All three axes rotate the tool	Unchanged. All three axes rotate the workpiece	Tool by two axes, work-piece by a rotary axis	Tool by one axis, work-piece by two rotary axes

In all four cases, the first rotary axis is the one which closest to the workpiece and the third rotary axis the one which closest to the tool in the kinematic chain.

Note

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

Dedicated machine data exist for each general transformation or for each orientation transformation that are differentiated by the suffixes _1, _2 etc. (e.g. $MC_TRAFO_TYPE_1, $MC_TRAFO_TYPE_2 etc.). In the following, only the names for the first transformation are specified, i.e. those with the suffix _1. If a transformation other than the first is parameterized, the correspondingly modified names must be used.

Configuration

The extensions of the following machine data are required for the configuration of **a 6-axis transformation**:

- The channel axis index of the 3rd rotary axis must be entered in $MC_TRAFO_AXES_IN_1[5].
- The direction of the 3rd rotary axis must be specified in $MC_TRAFO5_AXIS3_1[0..2].
- An orientation normal vector with a length not equal to zero and which is not parallel or anti-parallel to the $MC_TRAFO5_BASE_ORIENT_1[0..2] defined orientation vector, must be specified in machine data $MC_TRAFO6_BASE_ORIENT_NORMAL_1[0..2].

To the **previous offsets:**	**Vector**
• $MC_TRAFO5_BASE_TOOL_1[0..2]	Base tool
• $MC_TRAFO5_JOINT_OFFSET_1[0..2]	Kinematic offset
• $MC_TRAFO5_JOINT_OFFSET_PART_1[0..2]	Kinematic offset on the table
• $MC_TRAFO5_PART_OFFSET_1[0..2]	Offset vector for 5-axis transformation 1

as **new offset**, the machine data	**Vector**
• $MC_TRAFO6_JOINT_OFFSET_2_3_1[0..2]	Kinematic offset

is added, which describes the offset between the second and third rotary axis.

Note

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

Programming of the orientation

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

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

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

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

Default setting of the orientation normal vector

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

Specification for the **activation of the transformation**

1. The vector components are transferred as parameters 8 to 10:
   Parameter 1: Transformation number,
   Parameters 2 – 4: Orientation vector,
   Parameters 5 – 7: Rotary axis offsets
2. If **no** orientation normal vector has been specified and **a** tool is active, the vector is taken from the tool data.
3. If **no** orientation normal vector has been specified and also **no** tool is active, the vector defined in machine data $MC_TRAFO6_BASE_ORIENT_NORMAL_1[0..2] is used.

The position of the orientation coordinate system of a standard tool depends on the **active plane** G17, G18, G19 according to the following table:

Table 2-8 Position of the orientation coordinate system

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

Note

The orientation vector of a tool can be defined differently from the default setting with the system variables
$TC_DPV or $TC_DPV3 – $TC_DPV5 in the tool data,
see functional description FB1, Tool Offsets, W1
“Sum and setup offsets”.
This option is extended with the specification of the orientation normal vector with the system variables
$TC_DPVN3 – $TC_DPVN5. The meaning of the vector components is analogous to the meaning of the components of the tool orientation:
$TC_DPVN3 is the component in the direction of tool length L1, $TC_DPVN4 the component in the direction of tool length L2 and $TC_DPVN5 the component in the direction of tool length L3.

So that the new tool parameters can be used, machine data MD18114: MM_ENABLE_TOOL_ORIENT must have the value 3.

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

A programming example
for generic 6-axis transformation is described in Section 6.6.

Supplementary conditions

The generic 6-axis transformation requires 6 axes and is therefore only available on systems with at least 6 axes.

2.6.6 Cartesian manual travel with generic transformation

Functionality

As reference system for the JOG mode, the “Cartesian manual travel” function allows you, in the Cartesian coordinate systems:

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

to set axes independently. Machine data
MD21106: CART_JOG_SYSTEM not only activates the function, but also sets the permitted coordinate systems.

For the JOG motion, one of the three reference systems can be set not only for

the translation but also for the	movement of the geometry axes
tool orientation	movement of the orientation axes

via the setting data SD42650: CART_JOG_MODE independently from one another.

Activation

MD21106: CART_JOG_SYSTEM not only activates the function, but also sets the permitted coordinate systems.

SD42660: ORI_JOG_MODE sets the virtual kinematics used for the traversing of the orientation. As opposed to the generic 5-/6-axis transformation, only kinematics can be set in which the rotary axes are perpendicular to one another.

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

Translations

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

Note

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

References: /FB2/, Description of Functions, Extended Functions, M1, “Cartesian manual travel (810D powerline)”

Tool orientation

The tool can be aligned to the workpiece surface via an orientation movement. The motion of the orientation axes is triggered by the PLC via the VDI interface signals of the orientation axes. The virtual orientation axes execute rotations around the fixed directions of the relevant reference system. The virtual kinematics are specified with

SD42660 = 0: by the active transformation.

Rotations of the orientations

Further settings of the setting data SD42660: ORI_JOG_MODE specify the rotations of the orientation axes.
The options are as follows:

Rotations with JOG

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

SD42660 = 1: During jogging, **Euler angles** are traversed, i.e. the
1st axis rotates around the z direction, the
2nd axis rotates around the x direction and, if available, the
3rd axis rotates around the new z direction.

SD42660 = 2: During jogging, **RPY angles** are traversed with the rotation sequence XYZ, i.e. the
1st axis rotates around the x direction, the
2nd axis rotates around the y direction and, if available, the
3rd axis rotates around the new z direction.

SD42660 = 3: During jogging, **RPY angles** are traversed with the rotation sequence ZYX, i.e. the
1st axis rotates around the z direction, the
2nd axis rotates around the y direction and, if available, the
3rd axis rotates around the new x direction.

Rotation sequence of the rotary axes

The rotation sequence of the rotary axes is set with

SD42660 = 4:
via machine data MD21120: ORIAX_TURN_TAB_1.

SD42660 = 5:
via machine data MD21130: ORIAX_TURN_TAB_2.

For further explanations of the orientation movements, see:
Section 2.8 “Orientation”
Section 2.9 “Orientation axes”

Note

For further information about the programming of the rotations, see:

References: /PGA/, Programming Guide Advanced,
“Programming of the tool orientation” transformation

2.7 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 so 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 causes 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.

Please consider the features of polynomial interpolation of orientation vectors described in Subsection 2.10.1.

Further explanations about the tool orientation by means of orientation vectors for machine tools can be found in Subsection 2.10.1 and in the following sections.

2.7.1 Singularities of orientation

Description of problem

As described in Subsection 2.1.5 for SW up to 5.2, 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.

Proceed as follows in a situation of this type:
Only one machine data remains relevant
MD 24540: TRAFO5_POLE_LIMIT_1 or
MD 24640: TRAFO5_POLE_LIMIT_2,

describing a circle around the pole (as previously).
For further information about the handling of singular positions, see:
References: /PGA/, Programming Guide Advanced, “Cartesian PTP travel” transformations

Example for machine type 1

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

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

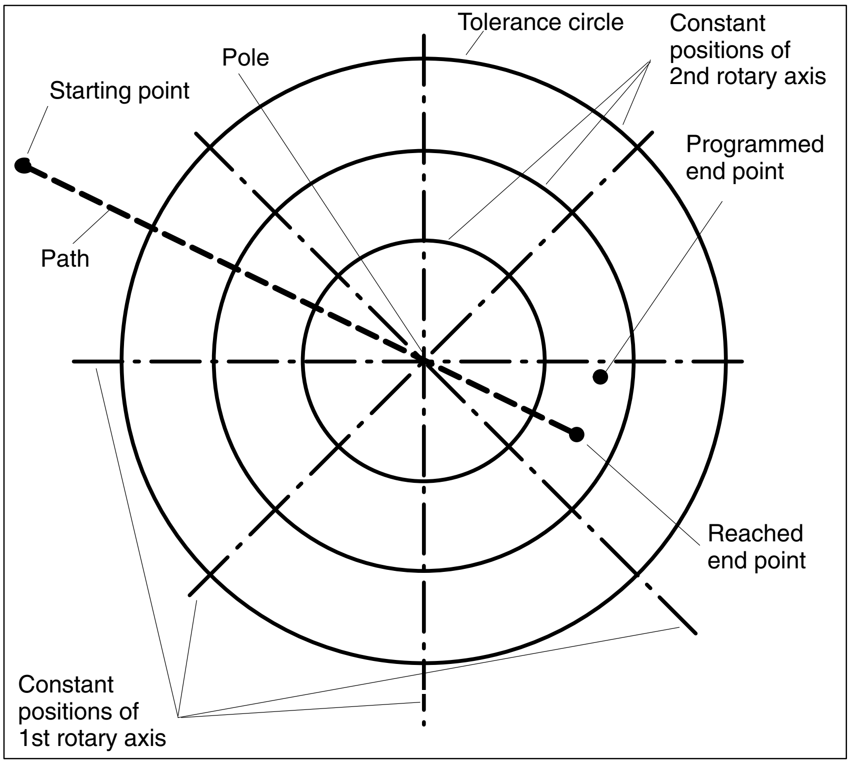


Fig. 2-15 Generic 5-axis transformation; end point of orientation inside tolerance circle

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 outside of a taper.

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.

2.8 Orientation

2.8.1 Basic orientation

Differences to the previous 5-axis transformations

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

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

If an orientation is programmed by means of Euler 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 so that a tool positioned in basic orientation is traversed to the programmed orientation.

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

Definition

There are three different ways to define the basic orientation:

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

Via the transformation call

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

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

Note

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

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

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

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

Via the orientation of the active tool

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

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

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

It can be arbitrarily modified toolholders with orientation capability, see:
References: /FB1/, Description of Functions, Basic Machine, W1 Tool Offset, “Orientable Toolholders”

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

Re 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 run-up 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 met, “dead” ranges will occur.

Examples:

1. Extreme example: A machine with rotatable tool has a C axis as its first rotary axis and an A axis as its second. If the basic orientation is defined in parallel to the A axis, the orientation can only be changed in the X-Y plane (when the C axis is rotating), i.e. 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.

2.8.2 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 the conditions under which the rotary axis positions may be modified:

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

Switchover 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.

2.8.3 Compression of orientation (SW 6.3 and higher)

Introduction

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

Extended function

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

Preconditions

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
– Feed
– Comment

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

Function in SW 6.3

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

Compression with 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 carried out under the following conditions:

1. Please note that 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. Please note that 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.

General format of an NC block which can be compressed:
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 change in orientation 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 every axis. This curve may deviate from the programmed end point of each axis by a maximum of the value set with 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: Max. 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 on 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 specified with SD 42475: COMPRESS_CONTUR_TOL, tolerance for the orientation is specified via axial tolerances MD 33100: COMPRESS_POS_TOL.

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

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

It is only possible to specify a maximum angular displacement for 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.

2.8.4 Orientation relative to the path (SW 7.3 and higher)

Functionality

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

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

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

Note

The orientation relative to the path interpolation with ORIPATH or ORIPATHS and ORIROTC, **cannot be used together** with the “Orientation smoothing” function. For this, OSOF must be active in the part program. Otherwise alarm 10980 “Orientation smoothing not possible” is output.

Activate orientation relative to the path

The extended function “Orientation relative to the path” is activated with

MD21094: ORIPATH_MODE > 0.

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

Orientation at block transition

In the case of MD21094: ORIPATH_MODE = 0
the tool orientation is always continuous at a block transition.

With MD21094: ORIPATH_MODE > 0, a jump in the tool orientation can occur at a block transition. A jump in the orientation always occurs when either the path tangent or the surface normal vector does not change smoothly at a block transition.

Deviation from the desired orientation

During the interpolation of the block, the orientation may deviate greatly or not so greatly from the desired relative orientation. The orientation achieved in the previous block is transferred to the programmed end orientation using large-radius circular interpolation. The resulting deviation from the desired relative orientation has two main causes:

1. The **end orientation of the previous block** refers to the tangent and the normal vector at the end of the previous block. Both can differ from this at the start of the current block. Therefore, the start orientation in the current block does not have the same alignment with respect to the tangent and the normal vector as at the end of the previous block.
2. No only the tangent, but also the normal vector can **change throughout the entire block**. This is the case, when circles, splines or polynomials are programmed for the geometry axes, or when not only a start, but also an end value is programmed for the normal vector. In this case, the tool orientation must change accordingly during the interpolation of the block, in order to have the same reference to the path tangent and to the surface normal vector in each path point.

Set orientation relative to the path

Machine data MD21094: ORIPATH_MODE is used to set in which way the orientation relative to the path is to be interpolated. With ORIPATH, the behavior of the tool orientation interpolation relative to the path can be activated for various functions:

Meaning of the **unit digit** Activate real orientation relative to the path

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

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

Meaning of the **ten digit** Interpretation of the angle of rotation TILT

0: LEAD = Rotation around the direction perpendicular to the tangent and normal vector (forward angle)
TILT = Rotation of the orientation around the **normal vector**

1: LEAD = Rotation around the direction perpendicular to the tangent and normal vector (forward angle)
TILT = Rotation of the orientation around the **direction of the path tangent** (side angle)

Meaning of the **hundred digit** Retracting movement for re-orientation

0: There is no retracting movement

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

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

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

Smoothing of the orientation jump ORIPATHS

The smoothing of the orientation jump is performed within a distance specified via setting data SD42670: ORIPATH_SMOOTH_DIST. The programmed reference of the orientation to the path tangent and normal vector is then no longer maintained within this distance. If this distance is set to small, the path velocity may have to be significantly reduced.

A velocity jump of the orientation axes can also be smoothed. In the case where the orientation path does not perform a jump, but whose first derivation is not smooth, the resulting velocity jump can be smoothed. Setting data SD42672: ORIPATH_SMOOTH_TOL > 0 is used to specify how much the orientation may deviate from the "tangential" alignment. This orientation smoothing is only performed if G code ORIPATHS is active and SD42672: ORIPATH_SMOOTH_TOL > 0.

Insertion of intermediate blocks for the smoothing of the orientation path
If SD42670: ORIPATH_SMOOTH_DIST = 0.0 is set, a separate intermediate block is inserted for the smoothing of the orientation path. This means that the path motion then stops at the corner of the contour and only then is the jump in the tool orientation executed. The orientation change is then only performed with with continuous acceleration when ORIPATHS is active. The orientation is otherwise transferred from the start orientation to the end orientation by means of linear large circle interpolation.

Execute tool retracting movement

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

In which coordinate system the tool retracting vector is interpreted, depends on the value of MD21094: ORIPATH_MODE. This specifies in which coordinate system the retracting vector is interpreted.

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

Normally the retracting movement is performed simultaneously to the orientation change. A factor can be programmed with the identifier ORIPLF = r , which defines a "safety clearance". In this way, the tool orientation only changes when the tool has retracted by

Formula

r * retractingdistance

The programmed retraction factor must be in the interval 0 <= r < 1, in order to avoid alarm14126.

Interpolation of the rotation relative to the path ORIROTC

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

2.8.5 Programming orientation polynomials (SW 7.1 and higher)

Functionality

Orientation polynomials and even axis polynomials can be programmed with different types of polynomials regardless of the type of polynomial interpolation currently active. In SW 7.1 and higher, this can be applied to:

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

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

Note

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

References: /PGA/, Programming Guide Advanced

Two different types of orientation polynomial are defined:

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

Type 1 polynomials

Type 1 orientation polynomials are polynomials **for angles**

PO[PHI]: in the plane between start and end orientation

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

Type 2 polynomials

Type 2 orientation polynomials are polynomials **for coordinates**

PO[XH]: x coordinate of reference point on tool

PO[YH]: y coordinate of reference point on tool

PO[ZH]: z coordinate of reference point on tool

Polynomials for angle of rotation and rotation vectors

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

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

Angle of rotation with ORIPATH and ORIPATHS

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

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

LEAD	Angle relative to the surface normal vector, in the plane defined by the path tangent and surface normal vector.
TILT	Rotation of orientation around the z direction or rotation around the path tangent
THETA	Rotation around the tool direction. Is only possible if the tool orientation has a total of 3 degrees of freedom, see “Extension of the generic transformation to 6 axes”.

Different settings of MD21094:ORIPATH_MODE can be used to specify how the LEAD und TILT angle is to be interpreted.

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

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

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

The higher polynomial coefficients, which are zero, can be omitted when programming. For example, with

PO[PHI] = (a2)

a parabola programmed for the LEAD angle.

Rotations of the rotation vectors with ORIROTC

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

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

Note

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

For further information about programming, please see:
References: /PGA/, Programming Guide Advanced, “Transformations” interpolation method (ORIPATH, ORIPATHS)

Supplementary conditions

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

Orientation polynomials cannot be programmed:

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

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

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

- If the orientation is interpolated using ORIAXES axis interpolation. In this case, polynomials can be programmed directly with PO[A] and PO[B] for orientation axes A and B.

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

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

Alarms

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

Alarm 14136:	Orientation polynomial is generally not permitted.
Alarm 14137:	PO[PHI] and PO[PSI] polynomials are not permitted.
Alarm 14138:	PO[XH], PO[YH], PO[ZH] polynomials are not permitted.
Alarm 14139:	Polynomial for angle of rotation PO[THT] is not permitted.

2.8.6 Tool orientation with 3-/4-/5-axis transformations

The tool direction can be read with the following system variables:

$P_TOOLO[n]	Tool orientation active in the **interpreter** cannot be used in synchronized actions
$AC_TOOLO_ACT[n]	Set orientation active in the **interpolator**
$AC_TOOLO_END[n]	End orientation of the active block
$AC_TOOLO_DIFF	Residual angle of the tool orientation in the active block
$VC_TOOLO[n]	Actual orientation direction
$VC_TOOLO_DIFF	Angle between actual and set orientation
$VC_TOOLO_STAT	Status of the calculation of the actual orientation direction.

2.8.7 Orientation vectors for 6-axis transformations

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

- The **first** vector points in the direction of the tool (see above), while
- the **second** is in the plane perpendicular to this and describes rotations of the tool around itself.

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

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

$P_TOOLROT[n]	Direction of rotation vector active in the **interpreter** cannot be used in synchronized actions
$AC_TOOLR_ACT[n]	Direction of rotation vector active in the **interpolator**
$AC_TOOLR_END[n]	End direction of rotation vector of the active block
$AC_TOOLR_DIFF	Residual angle of the direction of rotation vector in the active block in degrees
$VC_TOOLR[n]	Actual value of the direction of rotation vector
$VC_TOOLR_DIFF	Angle between actual value and setpoint of the direction of rotation vector in degrees
$VC_TOOLR_STAT	Status of the calculation of the actual value of the direction of rotation vector.

References: /PGA1/, LHB System Variables

For further information about the programming of polynomials for axis movements with orientation vectors, see Section “Orientation vectors”.

2.9 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 ORIMKS and ORIWKS commands:

– ORIMKS: Reference system = Basic coordinate system

– ORIWKS: Reference system = Workpiece coordinate system

Order of rotation

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

1. First rotation around the axis of the reference system specified in MD 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 initial machine setting is defined in
MD 24580: TRAFO5_TOOL_VECTOR_1 or
MD 24680: TRAFO5_TOOL_VECTOR_2.

Assignment to channel axes

Using 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”.

2.9.1 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: ACCEL_ORI[n].

2.9.2 Programming for orientation transformation

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

Programming of the 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)

With MD 21102: ORI_DEF_WITH_G_CODE can be used to define 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

References: /PG/, Programming Guide Fundamentals

Note

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

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

Example

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

Interpolation type

The 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 the case of 6 transformed axes, rotation around the orientation vector takes place in addition to the swiveling motion.
    If ORIVECT is selected, the path traversed by the orientation axes is always the shortest possible.

Value range

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

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

Feedrate

Feedrate when programming ORIAXES:

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

2.9.3 Programmable offset for orientation axes (SW 6.4 and higher)

How the programmable offset works

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

The programmable offset can be specified in two ways:

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

Programming the offset directly

When transformation is activated, the offset can be programmed directly as TRAORI(n, x, y, z, a, b).
In SW 6.4 and higher, the following parameters are available as options:

n: Number of transformation n = 1 or 2

x, y, z: Components of the vector for the basic orientation of the tool (generic 5-axis transformation only).

a, b: Offset for rotary axes

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

TRAORI(,,,, a, b) is programmed, for example.

For further information about programming, please see:
References: /PGA/, Chapter 7 "Transformations"

Programming the offset automatically

As the offset is transferred automatically from the currently active zero offset on the orientation axes, the effects of zero offset on rotary axes are always the same both with and without active transformation. The automatic transfer of the offset from the zero offset is made possible via machine data
MD 24590: TRAFO5_ROT_OFFSET_FROM_FR_1 = TRUE for the first and
MD 24690: TRAFO5_ROT_OFFSET_FROM_FR_2 = TRUE for the second transformation in the channel.

Note

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

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

Orientable toolholder with additive offset

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

The automatic transfer of the offset from the zero offset is made possible via machine data MD 21186: TOCARR_ROT_OFFSET_FROM_FR = TRUE.

Note

For more information about orientable toolholders, please see:
References: /FB1/, Basic Machine W1, "Tool Compensation"

2.9.4 Orientation transformation and orientable toolholders

Note

Orientation transformation and orientable toolholders can be combined.

The resulting orientation of the tool is produced by linking the orientation transformation and the orientable toolholder.

2.10 Orientation vectors

2.10.1 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.

MD 10674: By default, PO_WITHOUT_POLY = FALSE is set and MD 10674: PO_WITHOUT_POLY = **TRUE** can always be used for programming:

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

In SW 7.1 and higher, orientation polynomials can be programmed together with various types of interpolation and are described in Subsection 2.8.5.

POLYPATH

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

(“AXES”):	For all path axes and special axes
(“VECT”):	For orientation axes
(“AXES”,“VECT”):	For path axes, special axes and orientation axes
(No argument):	**deactivates** polynomial interpolation for all axes groups

Polynomial interpolation is activated for all axis groups by 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 defined by 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
$PSI(u) = b_0 + b_1*u + b_2*u^2 + b_3*u^3+ b_4*u^4 + b_5*u^5$.

PL — Length of the parameter interval where polynomials are defined. The interval always starts at 0.
Theoretical value range for PL: 0,0001 ... 99999,9999.
The PL value is valid for the block in which it is located.
If no PL has been programmed, PL = 1 will be applied.

Rotation of the orientation vector

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

Example 1 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.

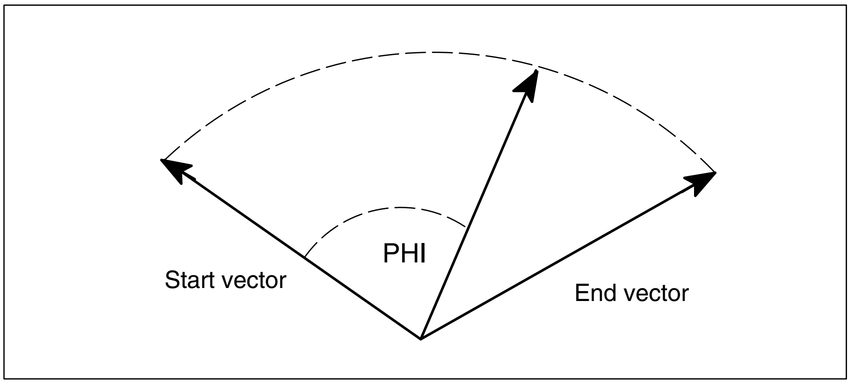


Fig. 2-16 Rotation of the orientation vector in the plane between start and end vector

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.

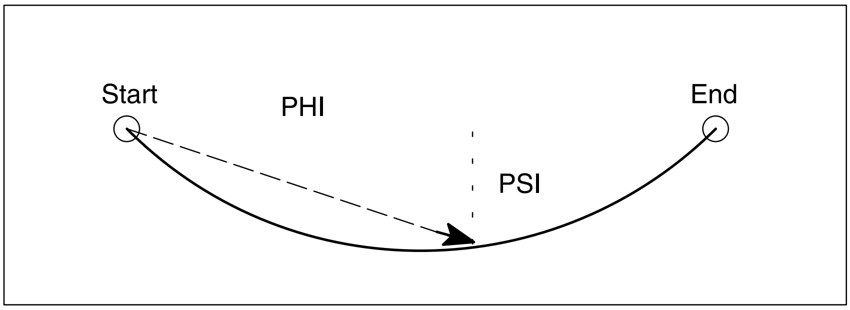


Fig. 2-17 Movement of the orientation vector in the top view

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 angle, the orientation vector is always interpolated in the plane defined by the start and end vector.

The PHI angle in this plane is interpolated according to the programmed polynomial for PHI. This mainly achieves 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 depending on the programmed polynomial.

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

Note

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

References: /PGA/, Programming Guide Advanced, Chapter 5

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.

2.10.2 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.

of orientation direction and rotation

While the direction of rotation is already defined when you program the orientation with RPY angles, additional parameters are needed 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 due to the additional programming of a polynomial for the angle of rotation θ, 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.

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

The linear coefficient d_n
is defined by means of the end angle θ_e and entered in degrees.

The end angle θ_e
is derived from the programming of the rotation vector.

The starting angle θ_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

$$\theta(u) = \theta_s + d_1 u + d_2 u^2 + d_3 u^3 + d_4 u^4 + d_5 u^5 \qquad (14)$$

For parameter interval 0 ... 1, this produces the following values for linear coefficients:

Formula

$$d_1 = \theta_e - \theta_s - d_2 - d_3 - d_4 - d_5 \qquad (15)$$

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 as an absolutely fixed 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 change in orientation for THETA=0.

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

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 reached at the end of the block.
THETA = θ_e	programmed angle θ_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 of the tangential rotation vector of the orientation vector to the orientation change
ORIROTC	Angle of rotation relative to the change of the tangential rotation vector of the orientation vector to the path tangent
PL	Length of the parameter interval where polynomials are defined. The interval always starts at 0. If no PL has been programmed, PL = 1 will be applied.

These G codes define the reference direction of the angle of rotation. The meaning of the programmed angle of rotation 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 rotation of the tool orientation (e.g. 6-axis machines).

2.10.3 Extended interpolation of orientation axes (SW 6.1 and higher)

Functionality

To execute a change in orientation along the peripheral surface of a taper 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.

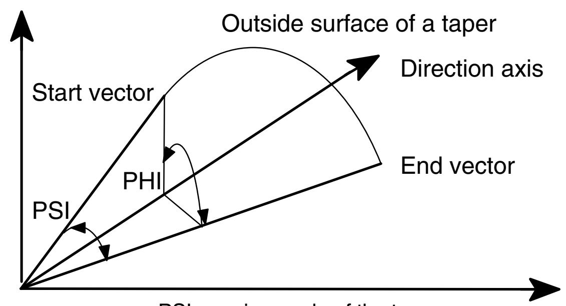


Fig. 2-18 Change in orientation of the peripheral surface of a taper located in space

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 taper** is programmed as a (normalized) vector with A6, B6, C6.
- The **opening angle of the taper** 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 opening angle specifies whether the traversing angle is to be greater of less than 180 degrees.

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

Programming

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

ORICONCW **ori**entation interpolation on a **con**e **c**lock**w**ise

ORICONCCW **ori**entation interpolation on a **con**e **c**ounter **c**lock**w**ise

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 end orientation is specified, a full outside taper with 360 degrees is interpolated.

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

Note

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

The identifiers have the following meanings:

NUT = +... 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.

Settings for intermediate orientation

ORICONIO **ori**entation interpolation on a **con**e with **i**ntermediate

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.

Angle of rotation and opening angle

The following may be programmed for the angle of the taper

PHI	**Angle of rotation** of the orientation around the direction axis
PSI	**Opening angle** of the taper

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

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

Further interpolation options

It is possible to interpolate the orientation on a taper connected tangentially to the previous change in orientation.
This orientation interpolation is achieved by programming the ORICONTO G code.

ORICONTO **ori**entation interpolation on a **con**e with **t**angential **o**rientation

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**

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

XH, YH, ZH Except for the relevant end values, you can also program additional

polynomials in the format

PO[XH] = (xe, x2, x3, x4, x5)	(xe, ye, ze) is 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)	(maximum 5th degree).

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.

Supplementary 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 taper in space is activated with the G code of Group 51 through extended interpolation of the orientation vector using the following commands:

ORIPLANE	Interpolation in a plane with specification of the end orientation (same as ORIVECT)
ORICONCW	Interpolation on a peripheral surface of a taper in clockwise direction with specification of the end orientation and taper direction or opening angle of the taper.
ORICONCCW	Interpolation on a peripheral surface of a taper in counter clockwise direction with specification of the end orientation and taper direction or opening angle of the taper.
ORICONIO	Interpolation on a peripheral surface of a taper with specification of the end orientation and an intermediate orientation.
ORICONTO	Interpolation on a peripheral surface of a taper with tangential transition and specification of the end orientation.
ORICURVE	Interpolation of the orientation specifying a movement between two contact points of the tool.
ORIPATH	Tool orientation in relation to the path.
ORIPATHS	Tool orientation in relation to the path, when, for example, a kink in the orientation path, e.g. at a corner in the contour, is to be smoothed, see Section "Orientation relative to the path".

Examples

The various changes in orientation 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 in Y/Z plane rotated at 45
                                ; degrees, orientation (0, 1/√2, 1/√2)
                                ; reached at end of block.
N70 ORICONCW                    : The orientation vector is interpolated
                                ; clockwise on a taper with the direction
                                ;
N80 A6=0 B6=0 C6=1 A3=1 B3=0 C3=1; (0, 0, 1) until orientation
                                ; (1/√2, 0, 1/√2) is reached.
                                ; The angle of rotation is 270 degrees.
                                ; The tool orientation passes through a
N90 A6=0 B6=0 C6=1              ; complete revolution on the same taper.
...
```

2.11 Online tool length offset

Functionality

The effective tool lengths can be changed in real time so that these changes in length are also considered for changes in orientation in the tool. System variable $AA_TOFF[ ] applies the tool length compensations 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 compensation.

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

Application

The online tool length compensation function can be used for:

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

Note

The online tool length 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 correct axis velocities are applied quickly.

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

Note

Changing the effective tool length using online tool length compensation produces changes in the compensatory movements of the axes involved in the transformation in the event of changes in orientation. The resulting velocities can be higher or lower depending on 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 content of the synchronization 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 online tool length compensation:

Machine data / setting data	Meaning for online tool length compensation
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 compensation
MD 21194: TOFF_ACCEL	Acceleration of online tool length compensation
SD 42970: TOFF_LIMIT	Upper limit of tool length compensation value

With the acceleration margin, 20% is reserved for the overlaid movement of the online tool length compensation, which can be changed via machine data MD 20610: ADD_MOVE_ACCEL_RESERVE.

Activation

The TOFFON instruction can be used to activate the online tool length compensation from the part 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 instruction TOFFON( ) with an offset causes the new offset to be applied. The offset value is added to variables $AA_TOFF[ ] as an absolute value.

Note

For further information about programming with examples, please see:
References: /PGA/, Chapter 7 “Transformations”

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

During a compensatory movement, the VDI →signal
IS ”TOFF motion active” (DB21, ... DBX318.3) is set to 1.

Reset

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

The tool length compensations 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 compensation.

Once the ”online tool length compensation” has been deselected for a tool direction, the value of system variable $AA_TOFF[ ] or $AA_TOFF_VAL[ ] is zero for this tool direction.
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, illegal change of tool direction active due to $AA_TOFF active” is suppressed:

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

Mode change

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

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

Behavior with REF and block search

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

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

System variable

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

System variable	Meaning for online tool length compensation
$AA_TOFF[ ]	Position offset in the tool coordinate system
$AA_TOFF_VAL[ ]	Integrated position offset in the WCS
$AA_TOFF_LIMIT[ ]	Query whether tool length compensation 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: /PGA1/, LHB System Variables

Supplementary conditions

The online tool length offset function is an option and is available during ”generic 5-axis transformation” by default and for ”orientable 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 compensation dimension, deviations compared to the actual offset surface are produced. It is not possible to produce exact offset surfaces with one tool length compensation alone.

■

Supplementary Conditions 3

There are no other supplementary conditions to note

■

Notes

Data Descriptions (MD, SD) 4

4.1 General machine data

10620 MD number	**EULER_ANGLE_NAME_TAB** Name of Euler angle [GEOaxisNo.]: 0...2				
Default setting: A2, B2, C2		Minimum input limit: –		Maximum input limit: –	
Changes effective after POWER ON			Protection level: 2/7		Unit: –
Data type: STRING				Applies as of SW 2.1	
Special cases, errors, ...	• The name entered must not clash with the designation and allocation of machine and geometry axes. • The name entered must not overlap with the channel axis names in channel (MD 20080: AXCONF_CHANAX_NAME_TAB), names for direction 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). • The name entered must not feature the following reserved address letters: – D Tool offset (D function) – E Not used – F Feedrate (F function) – G Distance condition – H Auxiliary function (H function) – L Subroutine call – M Special function (M function) – N Subblock – P Number of subroutine repetitions – R R parameters – S Spindle speed (S function) – T Tool (T function) • Keywords (e.g. DEF, SPOS, etc.) and predefined identifiers (e.g. ASPLINE, SOFT) are also not permitted. • An angle identifier comprises a valid letter (A, B, C, I, J, K, Q, U, V, W, X, Y, Z), followed by an optional numerical extension (1–99).				

10630 MD number	**NORMAL_VECTOR_NAME_TAB** Name of normal vectors				
Default setting: A4, B4, C4 Default setting: A5, B5, C5		Minimum input limit: –		Maximum input limit: –	
Changes effective after POWER ON			Protection level: 2/7		Unit: –
Data type: STRING				Applies as of SW 4.1	
Meaning:	List of identifiers for normal vector components at block start and block end. The rules for axis identifiers described in MD 20080: AXCONF_CHANAX_NAME_TAB apply when choosing identifiers. The identifiers must be selected so that there are no conflicts with other identifiers, e.g. 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 MD number	**DIR_VECTOR_NAME_TAB** Name of direction vectors				
Default setting: A3, B3, C3		Minimum input limit: –		Maximum input limit: –	
Changes effective after POWER ON			Protection level: 2/7		Unit: –
Data type: STRING				Applies as of SW 4.1	
Meaning:	List of identifiers for direction vector components. The rules for axis identifiers described in MD 20080: AXCONF_CHANAX_NAME_TAB apply when choosing identifiers. The identifiers must be selected so that there are no conflicts with other identifiers, e.g. 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 MD number	**ROT_VECTOR_NAME_TAB** Name of rotation vectors				
Default setting: A6, B6, C6		Minimum input limit: –		Maximum input limit: –	
Changes effective after POWER ON			Protection level: 2/7		Unit: –
Data type: STRING				Applies as of SW 6.1	
Meaning:	List of identifiers for rotation vector components. The rules for axis identifiers described in MD 20080: AXCONF_CHANAX_NAME_TAB apply when choosing identifiers. The identifiers must be selected so that there are no conflicts with other identifiers, e.g. 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 MD number	**INTER_VECTOR_NAME_TAB** Name of intermediate vector components				
Default setting: A7, B7, C7		Minimum input limit: –		Maximum input limit: –	
Changes effective after POWER ON			Protection level: 2/7		Unit: –
Data type: STRING				Applies as of SW 6.1	
Meaning:	List of identifiers for intermediate vector components. The rules for axis identifiers described in MD 20080: AXCONF_CHANAX_NAME_TAB apply when choosing identifiers. The identifiers must be selected so that there are no conflicts with other identifiers, e.g. 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 MD number	**ORIENTATION_NAME_TAB** Identifier for programming a 2nd orientation path				
Default setting: XH, YH, ZH		Minimum input limit: –		Maximum input limit: –	
Changes effective after POWER ON			Protection level: 2/7		Unit: –
Data type: STRING				Applies as of SW 6.1	
Meaning:	List of identifiers for programming the second curve in space for tool orientation. The rules described in MD 20080: AXCONF_CHANAX_NAME_TAB apply when choosing identifiers. The identifiers must be selected so that there are no conflicts with other identifiers, e.g. 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 MD number	**NUTATION_ANGLE_NAME** Name of the aperture angle	
Default setting: NUT	Minimum input limit: –	Maximum input limit: –
Changes effective after POWER ON	Protection level: 2/7	Unit: –
Data type: STRING	Applies as of SW 6.1	
Meaning:	Identifier for the aperture angle for orientation interpolation. The identifiers must be selected so that there are no conflicts with other identifiers, e.g. 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 MD number	**STAT_NAME** Name of position information	
Default setting: STAT	Minimum input limit: –	Maximum input limit: –
Changes effective after POWER ON	Protection level: 2/7	Unit: –
Data type: STRING	Applies as of SW 5.2	
Meaning:	Identifier for position information to resolve ambiguities for Cartesian PTP travel. The identifiers must be selected so that there are no conflicts with other identifiers, e.g. axes, Euler angle, normal vector, direction vector, intermediate coordinate.	

10672 MD number	**TU_NAME** Name of position information of the axes	
Default setting: TU	Minimum input limit: –	Maximum input limit: –
Changes effective after POWER ON	Protection level: 2/7	Unit: –
Data type: STRING	Applies as of SW 5.2	
Meaning:	Identifier for position information of axes to resolve ambiguities for Cartesian PTP travel. The identifiers must be selected so that there are no conflicts with other identifiers, e.g. axes, Euler angle, normal vector, direction vector, intermediate coordinate.	

10674 MD number	**PO_WITHOUT_POLY** Permits programming of PO[ ] without POLY having to be active.	
Default setting: FALSE	Minimum input limit: 0	Maximum input limit: 1
Changes effective after POWER ON	Protection level: 2/7	Unit: –
Data type: BOOLEAN	Applies as of SW 5.3	
Special cases, errors, ...	The machine data can be used to specify how the control behaves when programming polynomials with PO[...]. **MD 10674 = 0 (FALSE):** Previous behavior, active when programming PO[...] without POLY. An error message is displayed. **MD10674 = 1 (TRUE):** Programming of PO[...] is permitted without the G code POLY being active. POLY and POLYPATH( ) produce only the actual execution of the polynomial interpolation in this case.	

4.2 Channelspecific 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.

21100 MD number	**ORIENTATION_IS_EULER** Definition of angle for programming of orientation		
Default setting: 1	Minimum input limit: 0	Maximum input limit: 1	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)	Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)	Unit: –	
Data type: BOOLEAN	Applies as of SW 2.1		
Meaning:	**MD 21100 = 0 (FALSE):** The values programmed with A2, B2, C2 during orientation programming are interpreted as RPY angles (in degrees). The orientation vector results from the fact that one vector is first rotated by C2 in the Z direction around the Z axis, then by B2 around the new Y axis and, finally, by A2 around the new X axis. Unlike Euler programming, in this case all three values affect the orientation vector. **MD 21100 = 1 (TRUE):** The values programmed with A2, B2, C2 during orientation programming are interpreted as Euler angles (in degrees). The orientation vector results from the fact that one vector is first rotated by A2 in the Z direction around the Z axis, then by B2 around the new X axis and, finally, by C2 around the new Z axis. The value of C2 is therefore meaningless.		

21108 MD number	**POLE_ORI_MODE** Behavior during large circle interpolation at pole position				
Default setting: 0, 0, 0, 0, ...		Minimum input limit: 0		Maximum input limit: 22	
Changes effective after NEWCONF (SW 5.2 and higher)			Protection level: 7 / 7 (SW 5.2 and higher)		Unit: –
Data type: DWORD			Applies as of SW 5.2		
Meaning:	Defines the treatment of changes in orientation 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. The position of the polar axis is arbitrary in the polar position. For large circle interpolation, however, a specific orientation is required for this axis. The MD is decimally-coded. The **unit digits** define the behavior if the start orientation precisely matches the pole orientation, the **ten digits** the behavior if the start orientation of the block is inside the tolerance circle defined in MD TRAFO5_NON_POLE_LIMIT_1/2. The **unit digits** can have the following values (effective if the starting orientation is equal to the pole orientation): **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. **1**: A block is inserted in front of the block in which the situation described above occurs. This block positions the polar axis so that large circle interpolation can be performed in the following block. **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. 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). 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): **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. **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. **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). The values of the unit digits are added to those of the ten digits.					

24100, 24440 **24200, 24450** **24300, 24460** **24400, 24470** **24430, 24480** MD number	**TRAFO_TYPE_1 TRAFO_TYPE_6** **TRAFO_TYPE_2 TRAFO_TYPE_7** **TRAFO_TYPE_3 TRAFO_TYPE_8** **TRAFO_TYPE_4 TRAFO_TYPE_9** **TRAFO_TYPE_5 TRAFO_TYPE_10** Definition of transformation 1/2/3/4/5/6/7/8/9/10 in channel	
Default setting: 0	Minimum input limit: 0	Maximum input limit: –
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)	Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)	Unit: –
Data type: DWORD	Applies as of 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 tenth (MD 24480) transformation.	
MD irrelevant for	If no transformation is installed.	
Figure see Chapter 1 / 2	Figures: 1-1, 1-2, 1-3, 2-1.	
Application example(s)	See Chapter 6.	
Special cases, errors, ...	See Section 2.5.	
Additional references	/FB/, M1, “Kinematic transformations”	

The tables below show the decimal numbers to be set in MD TRAFO_TYPE_n (n = 1 ... 10) for appropriate kinematics and kinematics which have been implemented.

5-axis transformations					
1. Rotary axis	2. Rotary axis	3. Rotary axis	Movable tool **TRAFO_TYPE**	Movable workpiece **TRAFO_TYPE**	Movable tool and workpiece **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		
Generic 5-axis transformation (SW 5.2 and higher)			24	40	56
Generic 3/4/5-axis transformation defined using an orientable toolholder (SW 6.4 and higher)					72
Generic 6-axis transformations (SW 7.2 and higher)					
A	B	C	24	40	56, 57

3-axis and 4-axis transformations				
Rotary axis	Orientation plane	Movable tool **TRAFO_TYPE**	Movable workpiece **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**
Polar transformation (TRANS)	256
with improved tool compensation	257
Cylinder surface transformation (TRACYL)	512
with X-Y-Z-C kinematics	513
with improved tool compensation	514
Inclined axis (TRAANG)	1024
Centerless grinding	2048
OEM transformation	from 4096 to 4098
Linked transformation	8192

24110, ,24432 **24210, 24442** **24310, 24452** **24410, 24462** **24472, 24482** MD number	**TRAFO_AXES_IN_1[n] TRAFO_AXES_IN_5[n]** **TRAFO_AXES_IN_2[n] TRAFO_AXES_IN_6[n]** **TRAFO_AXES_IN_3[n] TRAFO_AXES_IN_7[n]** **TRAFO_AXES_IN_4[n] TRAFO_AXES_IN_8[n]** **TRAFO_AXES_IN_9[n] TRAFO_AXES_IN_10[n]** Axis assignment for transformation 1/2/3/4/5/6/7/8/9/10 [axis index]: 0 ... 7				
Default setting: 1,2,3,4,5,0,0,...		Minimum input limit: 0		Maximum input limit: 10 [max. number of channels]	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)			Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)		Unit: –
Data type: Byte				Applies as of SW 1.1	
Meaning:	This MD contains the axis assignment at the input of the first (TRAFO_AXES_IN_1), second (TRAFO_AXES_IN_2), third (TRAFO_AXES_IN_3) to tenth transformation (TRAFO_AXES_IN_10). The index entered in the nth position indicates which axis is imaged internally on axis n by the transformation. The numbers of the channel axes must be entered. This is important for multi-channel systems.				
MD irrelevant for	If no transformation is installed.				
Application example(s)	On machine with 6 axes, the default setting might be e.g.: 1 2 3 4 5 6. The following is then possible: 1 2 3 5 4 6: 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 **24220, 24444** **24320, 24454** **24420, 24464** **24474, 24484** MD number	**TRAFO_GEOAX_ASSIGN_TAB_1[n] TRAFO_GEOAX_ASSIGN_TAB_5[n]** **TRAFO_GEOAX_ASSIGN_TAB_2[n] TRAFO_GEOAX_ASSIGN_TAB_6[n]** **TRAFO_GEOAX_ASSIGN_TAB_3[n] TRAFO_GEOAX_ASSIGN_TAB_7[n]** **TRAFO_GEOAX_ASSIGN_TAB_4[n] TRAFO_GEOAX_ASSIGN_TAB_8[n]** **TRAFO_GEOAX_ASSIGN_TAB_9[n] TRAFO_GEOAX_ASSIGN_TAB_10[n]** Assignment of geometry axes to channel axes with transformation 1/2/3/4/5/6/7/8/9/10 [geometry axis number]:0 ... 2.				
Default setting: 1, 2, 3		Minimum input limit: 0		Maximum input limit: 8	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)			Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)		Unit: –
Data type: Byte				Applies as of 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	Same as MD 20050: AXCONF_GEOAX_ASSIGN_TAB, but only when transformation is active				

24500 **24600** MD number	**TRAFO5_PART_OFFSET_1[n]** **TRAFO5_PART_OFFSET_2[n]** Offset vector of 5-axis transformation 1/2 [axis number]: 0 ... 2			
Default setting: 0,0	Minimum input limit:		Maximum input limit:	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)		Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)		Unit: mm
Data type: DOUBLE			Applies as of 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: Machine type 1 (two-axis swivel head for tool): Vector from machine reference point to zero point of table. This will generally be a zero vector if both coincide. Machine type 2 (two-axis rotary table for workpiece): Vector from second joint of workpiece rotary table to zero point of table. Machine type 3 (single-axis rotary table for workpiece and single-axis swivel head for tool): 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 **24610** MD number	**TRAFO5_ROT_AX_OFFSET_1[n]** **TRAFO5_ROT_AX_OFFSET_2[n]** Position offset of rotary axis 1/2 for 5-axis transformation 1/2 [axis no.] 0 ... 1			
Default setting: 0,0	Minimum input limit:		Maximum input limit:	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)		Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)		Unit: degrees
Data type: DOUBLE			Applies as of 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 **24620** MD number	**TRAFO5_ROT_SIGN_IS_PLUS_1[n]** **TRAFO5_ROT_SIGN_IS_PLUS_2[n]** Sign of rotary axis 1/2 for 5-axis transformation 1/2 [axis no.]: 0 ... 1				
Default setting: 1		Minimum input limit: 0		Maximum input limit: 1	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)			Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)		Unit: –
Data type: BOOLEAN				Applies as of 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 the second (MD: TRAFO5-ROT_SIGN_IS_PLUS_2) 5-axis transformation of a channel. MD = 0 (FALSE): Sign is inverted. MD = 1 (TRUE): Sign is not inverted and the traversing direction is defined according to AX_MOTION_DIR. The machine data does not mean that the direction of rotation of the rotary axis concerned will be inverted but indicates whether movement in a positive direction will be in a positive or negative direction in mathematical terms. Therefore, rather than changing the direction of rotation, changes to this machine data change the compensating movement of the linear axes. However, if a direction vector is entered and consequently a compensating movement set implicitly, this will change the direction of rotation of the rotary axis concerned. The machine data should therefore only be set to FALSE (or zero) on an actual machine if during movement in a positive direction the rotary axis rotates counter-clockwise.				
MD irrelevant for	If no transformation is installed.				

24530 **24630** MD number	**TRAFO5_NON_POLE_LIMIT_1** **TRAFO5_NON_POLE_LIMIT_2** Definition of pole range for 5-axis transformation 1/2				
Default setting: 2		Minimum input limit:		Maximum input limit:	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)			Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)		Unit: degrees
Data type: DOUBLE				Applies as of 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. 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 so that the interpolation passes through the pole. 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 option is installed. Also irrelevant for programming in the ORIMKS machine coordinate system.				
See Chapter 2 for Figure	Fig. 2-9 shows how this MD is used.				
Related to	MD: TRAFO5_POLE_LIMIT_1 or _2				

24540 **24640** MD number	**TRAFO5_POLE_LIMIT_1** **TRAFO5_POLE_LIMIT_2** End angle tolerance with interpolation through pole for 5-axis transformation 1/2		
Default setting: 2	Minimum input limit:	Maximum input limit:	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)	Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)	Unit: degrees	
Data type: Double	Applies as of 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: 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. 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 option is installed. Also irrelevant for programming in the ORIMKS machine coordinate system.		
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 **24650** MD number	**TRAFO5_BASE_TOOL_1[n]** **TRAFO5_BASE_TOOL_2[n]** Vector of base tool for activation of 5-axis transformation 1/2 [axis number]: 0 ... 2		
Default setting: 0,0	Minimum input limit:	Maximum input limit:	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)	Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)	Unit: mm	
Data type: DOUBLE	Applies as of 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 TRAFO5_BASE_TOOL_2) without tool length offset having been selected. Programmed tool length offsets take effect in addition to the base tool.		
MD irrelevant for	If no transformation is installed.		

24560 **24660** MD number	**TRAFO5_JOINT_OFFSET_1[n]** **TRAFO5_JOINT_OFFSET_2[n]** Vector of kinematic offset of 5-axis transformation 1/2 [axis number]: 0 ... 2				
Default setting: 0,0		Minimum input limit:		Maximum input limit:	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)			Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)		Unit: mm
Data type: DOUBLE				Applies as of SW 1.1	
Meaning:	This machine data defines the vector from the first to the second joint for the first (MD: TRAFO5_JOINT_OFFSET_1) or second TRAFO5_JOINT_OFFSET_2) transformation of a channel and has a specific meaning for the various machine types: Machine type 1 (two-axis swivel head for tool) and Machine type 2 (two-axis rotary table for workpiece): Vector from first to second joint of tool swivel head or workpiece rotary table. Machine type 3 (single-axis rotary table for workpiece and single-axis swivel head for tool): Vector from machine reference point to joint of workpiece table.				
MD irrelevant for	If no transformation option is installed. The same applies for 3-axis and 4-axis transformation.				

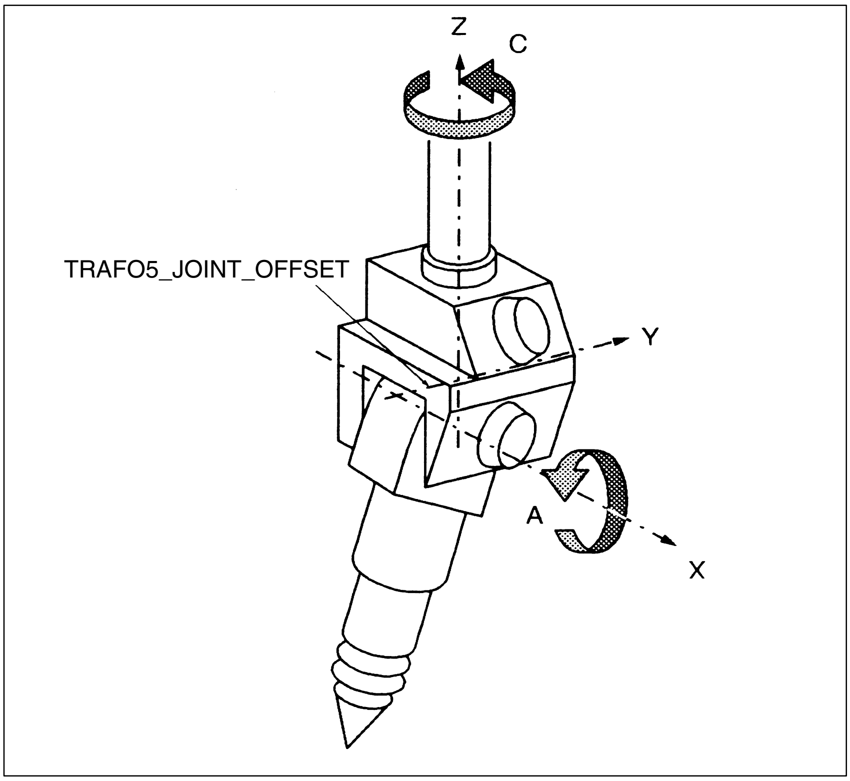


Fig. 4-1 Example for MD: TRAFO5_JOINT_OFFSET (joint offset for a 5-axis machine with two-axis swivel head for tools) (e.g. laser machining with machine type 1)

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 MD number	**TRAFO5_TOOL_ROT_AX_OFFSET_1** Offset of focus of 1st 5-axis transformation with swiveled linear axis.		
Default setting: 0.0, 0.0, 0.0	Minimum input limit: –	Maximum input limit: –	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)	Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)	Unit: degrees	
Data type: DOUBLE	Applies as of 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		
Figure	2-10		
Related to	24662		

24662 MD number	**TRAFO5_TOOL_ROT_AX_OFFSET_2** Offset of focus of 2nd 5-axis transformation with swiveled linear axis.		
Default setting: 0.0, 0.0, 0.0	Minimum input limit: –	Maximum input limit: –	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)	Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)	Unit: degrees	
Data type: DOUBLE	Applies as of 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		
Figure	2-10		
Related to	24562		

4.2.2 Channel-specific MD for universal milling head

24564 MD number	**TRAFO5_NUTATOR_AX_ANGLE_1** Angle of 2nd rotary axis to relevant axis in rectangular coordinate system, 1st transform.			
Default setting: 45	Minimum input limit: –89		Maximum input limit: 89	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)		Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)		Unit: degrees
Data type: DOUBLE			Applies as of SW 3.1	
Meaning:	Angle between the second rotary axis and the corresponding axis in the rectangular coordinate system. 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 MD number	**TRAFO5_NUTATOR_AX_ANGLE_2** Angle of 2nd rotary axis to relevant axis in rectangular coordinate system, 2nd transform.			
Default setting: 45	Minimum input limit: –89		Maximum input limit: 89	
Changes effective after NEWCONF (SW 5.2 and higher) POWER ON (up to SW 5.1)		Protection level: 7 / 7 (SW 5.2 and higher) 2 / 7 (up to SW 5.1)		Unit: degrees
Data type: DOUBLE			Applies as of SW 3.1	
Meaning:	Angle between the second rotary axis and the corresponding axis in the rectangular coordinate system. 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 MD and SD compression of orientation (SW 6.3)

20482 MD number	**COMPRESSOR_MODE** Mode of the compressor	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 3
Change valid after NEWCONF	Protection level:	Unit: –
Data type: BYTE	Applies as of SW 6.3	
Meaning:	This MD can be used to set the mode of operation of the compressor. The following options are available: 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. 1: For the compressor, the geometry axes specified by setting data SD 42475: COMPRESS_CONTUR_TOL is active for the contour tolerance. For the orientation axes, the axis-specific tolerances from MD 33100: COMPRESS_POS_TOL are active. 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. 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, SD 42476: COMPRESS_ORI_TOL, SD 42477: COMPRESS_ORI_ROT_TOL	

42475 MD number	**COMPRESS_CONTOUR_TOL** Max. contour deviation for compressor	
Default setting: 0.05	Minimum input limit: 0.000001	Maximum input limit: 999999
Changes effective IMMEDIATELY	Protection level:	Unit: mm
Data type: DOUBLE	Applies as of 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 MD number	**COMPRESS_ORI_TOL** Max. angular displacement of tool orientation for the compressor	
Default setting: 0.05	Minimum input limit: 0.000001	Maximum input limit: 90
Changes effective IMMEDIATELY	Protection level:	Unit: degrees
Data type: DOUBLE	Applies as of SW 6.3	
Meaning:	This setting data is used to define the maximum tolerance for tool orientation for the compressor. The data defines the maximum permissible angular displacement for tool orientation. This data is only effective if orientation transformation is active.	
Related to	MD 20482: COMPRESSOR_MODE	

42477 MD number	COMPRESS_ORI_ROT_TOL Max. angular displacement of tool rotation for the compressor	
Default setting: 0.05	Minimum input limit: 0.000001	Maximum input limit: 90
Changes effective IMMEDIATELY	Protection level:	Unit: degrees
Data type: DOUBLE	Applies as of SW 6.3	
Meaning:	This setting data is used to define the maximum tolerance for rotation of the tool orientation for the compressor. This data defines the maximum permissible angular deviation for tool rotation. This data is only effective if orientation transformation is active. The tool orientation may only be rotated on 6-axis machines.	
Related to	MD 20482: COMPRESSOR_MODE	

4.2.4 Channel-specific MD for orientation axes

20621 MD number	HANDWH_ORIAX_MAX_INCR_SIZE Limitation of handwheel increment for orientation axes	
Default setting: 0.0	Minimum input limit: 0.0	Maximum input limit: –
Changes effective after Reset	Protection level: 2 / 7	Unit: degrees
Data type: DOUBLE	Applies as of SW 5.3	
Meaning:	Limitation of handwheel increment for orientation axes	
MD irrelevant for	If no transformation is installed.	

20623 MD number	HANDWH_ORIAX_MAX_INCR_VSIZE Orientation velocity overlay	
Default setting: 0.1	Minimum input limit: 0.0	Maximum input limit: –
Changes effective after Reset	Protection level: 2 / 7	Unit: rev/min
Data type: DOUBLE	Applies as of SW 5.3	
Meaning:	Orientation velocity overlay	
MD irrelevant for	If no transformation is installed.	

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

21104 MD number	ORI_IPO_WITH_G_CODE Definition of interpolation type for orientation	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 1
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: BOOLEAN	Applies as of SW 5.3	
Meaning:	Definition of interpolation type for orientation MD = 0 (FALSE): The G codes ORIWKS and ORIMKS are the references MD = 1 (TRUE): The G codes ORIVECT and ORIAXES are the references	

21120 **21130** MD number	**ORIAX_TURN_TAB_1[n]** **ORIAX_TURN_TAB_2[n]** Assignment of rotations of orientation axes about the reference axes, definition 1 or definition 2				
Default setting: 1, 2, 3		Minimum input limit: 1		Maximum input limit: 3	
Changes effective after POWER ON			Protection level: 2 / 7		Unit: –
Data type: Byte				Applies as of SW 5.3	
Meaning:	Assignment of rotations of orientation axes about the reference axes, definition 1 or definition 2: 1 = Rotation about 1st reference axis (X) 2 = Rotation about 2nd reference axis (Y) 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.
This results in 12 options for specifying an orientation.
If a different axis assignment has been programmed, the alarm “Configuration axes configured incorrectly” appears.

Definition analogous to RPY angles			**Definition analogous to Euler angles**		
1. Rotation	2. Rotation	3. Rotation	1. Rotation	2. Rotation	3. 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 (row Z, Y, X) — (dark shading) corresponds to Euler angle NC (row Z, X, Z)

21150 MD number	**JOG_VELO_RAPID_ORI[n]** Rapid traverse in jog mode for orientation axes in the channel				
Default setting: 10.0, 10.0, 10.0		Minimum input limit: 0.0		Maximum input limit: –	
Changes effective after Reset			Protection level: 2 / 7		Unit: rev/min
Data type: DOUBLE				Applies as of SW 5.3	
Meaning:	Rapid traverse in jog mode for orientation axes in the channel				
MD irrelevant for	If no transformation is installed.				

21155 MD number	**JOG_VELO_ORI[n]** Orientation axis velocity in jog mode				
Default setting: 2.0, 2.0, 2.0		Minimum input limit: 0.0		Maximum input limit: –	
Changes effective after Reset			Protection level: 2 / 7		Unit: rev/min
Data type: DOUBLE				Applies as of SW 5.3	
Meaning:	Orientation axis velocity in jog mode				
MD irrelevant for	If no transformation is installed.				

21160 MD number	**JOG_VELO_RAPID_GEO[n]** Rapid traverse in jog mode for geometry axes in the channel				
Default setting: 10000./60.,10000./60.,10000./60.,		Minimum input limit: 0.0		Maximum input limit: –	
Changes effective after Reset		Protection level: 2 / 7		Unit: mm/min	
Data type: DOUBLE			Applies as of SW 5		
Meaning:	Rapid traverse in jog mode for geometry axes in the channel				

21165 MD number	**JOG_VELO_GEO[n]** Geometry axis velocity in jog mode				
Default setting: 2000./60.,2000./60.,2000./60.,		Minimum input limit: 0.0		Maximum input limit: –	
Changes effective after Reset		Protection level: 2 / 7		Unit: mm/min	
Data type: DOUBLE			Applies as of SW 5		
Meaning:	Geometry axis velocity in jog mode				

21170 MD number	**ACCEL_ORI[n]** Acceleration for orientation axes				
Default setting: {2.0, 2.0, 2.0}, ...		Minimum input limit: 0.0		Maximum input limit: –	
Changes effective after Reset		Protection level: 2 / 7		Unit: rev/s^2	
Data type: DOUBLE			Applies as of SW 5.3		
Meaning:	The acceleration for the orientation axes can be set by means of this MD.				

21186 MD number	**TOCARR_ROT_OFFSET_FROM_FR** Offset of TOCARR rotary axes from WO				
Default setting: 0		Minimum input limit: 0		Maximum input limit: 1	
Changes effective after Reset		Protection level: 7 / 7		Unit: –	
Data type: BOOLEAN			Applies as of SW 6.4		
Meaning:	The offset of the rotary axes for orientable toolholders is automatically transferred from the work offset active when the orientable toolholders for the rotary axes are activated.				
MD irrelevant for	If no orientable toolholders are available.				
Related to	MD 24590: TRAFO5_ROT_OFFSET_FROM_FR_1 MD 24690: TRAFO5_ROT_OFFSET_FROM_FR_2				

24580 **24680** MD number	**TRAFO5_TOOL_VECTOR_1** **TRAFO5_TOOL_VECTOR_2** Direction of tool vector for orientation transformation 1 or 2				
Default setting: 2,2,2, ...		Minimum input limit: 0		Maximum input limit: 2	
Changes effective after POWER ON		Protection level: 2 / 7		Unit: –	
Data type: Byte			Applies as of SW 5.1		
Meaning:	The machine data indicates the direction of the tool vector for orientation transformation 1 or 2: 0 = Tool vector in X direction 1 = Tool vector in Y direction 2 = Tool vector in Z direction				
MD irrelevant for	If no transformation is installed.				
Application example(s)	See Section 6.4				

24585 **24685** MD number	**TRAFO5_ORIAX_ASSIGN_TAB_1[n]** **TRAFO5_ORIAX_ASSIGN_TAB_2[n]** Assignment of orientation axes to channel axes			
Default setting: 0, 0, 0		Minimum input limit: 0		Maximum input limit: 18
Changes effective after POWER ON		Protection level: 2 / 7		Unit: –
Data type: BOOLEAN			Applies as of 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			

24590 **24690** MD number	**TRAFO5_ROT_OFFSET_FROM_FR_1[n]** **TRAFO5_ROT_OFFSET_FROM_FR_2[n]** Offset of transf. rotary axes from WO			
Default setting: 0		Minimum input limit: 0		Maximum input limit: 1
Changes effective after Reset		Protection level: 7 / 7		Unit: –
Data type: BOOLEAN			Applies as of SW 6.4	
Meaning:	The programmable offset of the rotary axes for orientation axes is automatically transferred from the work offset active when orientation transformation 1 or 2 is activated for the orientation axes.			
MD irrelevant for	If no transformation is installed.			
Related to	MD 21186: TOCARR_ROT_OFFSET_FROM_FR			

4.2.5 Machine data for generic 5-axis transformation

<table>
<tr><td colspan="2">21180
MD number</td><td colspan="4">ROT_AX_SWL_CHECK_MODE
Check software limits for orientation axes</td></tr>
<tr><td colspan="2">Default setting: 0</td><td colspan="2">Minimum input limit: 0</td><td colspan="2">Maximum input limit: 2</td></tr>
<tr><td colspan="3">Changes effective IMMEDIATELY</td><td colspan="2">Protection level: 2 / 7</td><td>Unit: –</td></tr>
<tr><td colspan="4">Data type: DWORD</td><td colspan="2">Applies as of SW 6.3</td></tr>
<tr><td>Meaning:</td><td colspan="5">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.

The machine data can have the following three values:

0: No path modification takes place. If a movement across the shortest path is not possible, alarm 10620 or 10720 (SW-LIMITSWITSCH) is output.

1: If the initial orientation path violates the axis 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 a permissible solution by modifying both rotary axes by multiples of 360 degrees for both solutions. The changes to the end positions described are only made if axis interpolation has been activated for the rotary axes.

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.</td></tr>
</table>

<table>
<tr><td colspan="2">24558
24658
MD number</td><td colspan="4">TRAFO5_JOINT_OFFSET_PART_1
TRAFO5_JOINT_OFFSET_PART_2
Vector of kinematic offset in table</td></tr>
<tr><td colspan="2">Default setting: 0, 0, 0</td><td colspan="2">Minimum input limit: -</td><td colspan="2">Maximum input limit: –</td></tr>
<tr><td colspan="3">Change valid after NEWCONF</td><td colspan="2">Protection level: 7 / 7</td><td>Unit: mm</td></tr>
<tr><td colspan="4">Data type: DOUBLE</td><td colspan="2">Applies as of SW 6.3</td></tr>
<tr><td>Meaning:</td><td colspan="5">MD 24558 for the 1st orientation transformation, MD 24658 for the second.
This machine data is only evaluated for generic 5-axis transformation with movable workpiece and movable tool (TRAFO_TYPE = 56).
It indicates the part of the vector between the table and swivel head assigned to the table. Only the sum of this machine data and MD 24560: TRAFO5_JOINT_OFFSET_1 are included in the transformation equations. The only exception concerns reading out the total tool lengths with the GETTCOR function. In this case, only MD 24560: TRAFO5_JOINT_OFFSET_1 is taken into account. The second transformation is handled in the same way.
This machine data can be used to uniquely assign the machine data for 5-axis transformation and the parameters of the orientable toolholder on a machine with mixed kinematics as follows:
Orientable toolholder: 5-axis transformation (1st or 2nd transformation):
11 MD 24560/24660: TRAFO5_JOINT_OFFSET_1/2
12 MD 24550/24650: TRAFO5_BASE_TOOL_1/2
13 MD 24558/24658: TRAFO5_JOINT_OFFSET_PART_1/2
14 MD 24500/24600: TRAFO5_PART_OFFSET_1/2</td></tr>
<tr><td>MD irrelevant for ... ...</td><td colspan="5">If no 5-axis transformation option is installed.</td></tr>
</table>

24570 **24572** MD number	**TRAFO5_AXIS1_1** **TRAFO5_AXIS2_1** Vector for the first/second rotary axis and the first orientation transformation			
Default setting: 0, 0, 0		Minimum input limit: –		Maximum input limit: –
Change valid after NEWCONF			Protection level: 7 / 7	Unit: –
Data type: DOUBLE			Applies as of SW 5.2	
Meaning:	Specifies the vector, which describes the direction of the first or second rotary axis when a general 5-axis transformation is the first orientation transformation in the system (TRAFO_TYPE_* = 24, 40, 56). Absolute value can be set to any value, although ≠ 0 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).			
MD irrelevant for	If no 5-axis transformation option is installed.			
Related to	24670, 24672			

24574 **24674** MD number	**TRAFO5_BASE_ORIENT_1** **TRAFO5_BASE_ORIENT_2** Basic orientation for the first or second orientation transformation			
Default setting: 0, 0, 0		Minimum input limit: –		Maximum input limit: –
Change valid after NEWCONF			Protection level: 7 / 7	Unit: –
Data type: DOUBLE			Applies as of SW 5.2	
Meaning:	Describes the basic orientation applied as the basis for the transformation when general 5-axis transformation is the first or second orientation in the system (TRAFO_TYPE_* = 24, 40, 56) in cases where the basic orientation has not been defined by activating a transformation or by means of a tool. The absolute value of the vector can by set to any value, although it must not be equal to zero.			
MD irrelevant for	If no 5-axis transformation option is installed.			
Related to				

24582 **24682** MD number	**TRAFO5_TCARR_NO_1** **TRAFO5_TCARR_NO_2** TCARR number for the first or second orientation transformation			
Default setting: 0, 0, 0		Minimum input limit: 0		Maximum input limit: 999999999
Change valid after NEWCONF			Protection level: 2 / 2	Unit: –
Data type: DWORD			Applies as of SW 7.2	
Meaning:	If the value of this machine data **is not equal to zero and** machine data $MC_TRAFO_TYPE_X, which refers to the first or second orientation transformation, **has the value 72**, the kinematic data and the available offset, etc. are not read directly from the parameterized machine data for 5-axis transformation. The data for the first or second orientation transformation is read from the data of the orientable toolholder, which refers to this machine data.			
MD irrelevant for	If no 5-axis transformation option is installed.			
Application example(s)	See Section 2.6 Generic 5-axis transformation			

24670 **24672** MD number	**TRAFO5_AXIS1_2** **TRAFO5_AXIS2_2** Vector for the first/second rotary axis and the second orientation transformation			
Default setting: 0, 0, 0		Minimum input limit: –		Maximum input limit: –
Change valid after NEWCONF			Protection level: 7 / 7	Unit: –
Data type: DOUBLE			Applies as of SW 5.2	
Meaning:	As **MD 24 570: TRAFO5_AXIS1_1 or MD 24 572: TRAFO5_AXIS2_1**, but for the first or second orientation transformation in the channel.			
Related to	24570, 24572			

4.2.6 MD and SD online tool length offset (SW 6.4)

21190 MD number	**TOFF_MODE** Effect of online offset in tool direction		
Default setting: 0	Minimum input limit: 0	Maximum input limit: –	
Changes effective after Reset	Protection level: 2 / 7	Unit: 0F HEX	
Data type: BYTE	Applies as of SW 6.4		
Meaning:	This machine data is used to set the effect of online offset in the tool direction via $AA_TOFF[ ]. Bit 0 : Behavior of $AA_TOFF in the event of a RESET Bit 0 = 0 $AA_TOFF deselected in the event of a RESET Bit 0 = 1 $AA_TOFF maintained beyond RESET Bit 1 : Effect of value assignment on the 1st component of $AA_TOFF[ ] Bit 1 = 0 Absolute value Bit 1 = 1 Incremental value (integrator) Bit 2 : Effect of value assignment on the 2nd component of $AA_TOFF[ ] Bit 2 = 0 Absolute value Bit 2 = 1 Incremental value (integrator) Bit 3 : Effect of value assignment on the 3rd component of $AA_TOFF[ ] Bit 1 = 0 Absolute value Bit 1 = 1 Incremental value (integrator)		
Related to	MD 21194: TOFF_VELO velocity of online offset in tool direction MD 21196: TOFF_ACCEL acceleration of online offset in tool direction		

21194 MD number	**TOFF_VELO** Velocity of online offset in tool direction		
Default setting: 0	Minimum input limit: ≥ 0	Maximum input limit: –	
Change valid after NEWCONF	Protection level: 2 / 7	Unit: mm/min	
Data type: DOUBLE	Applies as of 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 MD 21196: TOFF_ACCEL acceleration of online offset in tool direction		

21196 MD number	**TOFF_ACCEL** Acceleration of online offset in tool direction		
Default setting: 100	Minimum input limit: 0,001	Maximum input limit: –	
Change valid after NEWCONF	Protection level: 2 / 7	Unit: m/s^2	
Data type: DOUBLE	Applies as of 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 MD 21194: TOFF_ACCEL Velocity for online offset in tool direction		

42970 MD number	**TOFF_LIMIT** Upper limit for offset value $AA_TOFF		
Default setting: 100000000.0	Minimum input limit: ≥ 0	Maximum input limit: 100000000.0	
Changes effective IMMEDIATELY	Protection level: 2 / 7	Unit: mm/inches	
Data type: DOUBLE	Applies as of SW 6.4		
Meaning:	Upper limit of the offset value, which can be preset by means of synchronization via system variable $AA_TOFF. This limit value is applied to the absolute effective offset value via $AA_TOFF. System variable $AA_TOFF_LIMIT can be used to query whether the offset value is close to the limit.		

4.2.7 MD and SD Cartesian manual travel (SW 6.3 and higher)

21106 MD number	**CART_JOG_SYSTEM** Coordinate system for Cartesian JOG	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 7
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: DWORD	Applies as of SW 6.3	
Meaning:	This machine data has two different meanings. First, it is used to activate the function Cartesian manual travel. Second, it can be used to specify the reference systems between which a switchover can be performed. The meaning of the individual bits is specified as follows: The meaning of the individual bits is specified as follows: Bit 0 : Basic coordinate system (BCS) Bit 1 : Workpiece coordinate system (WCS) Bit 2 : Tool coordinate system (TCS) If no bit is set, setting data SD 42650: CART_JOG_MODE is not interpreted. Traversing is carried out as before in JOG mode. 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. 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 “Handling transformation package”	

42650 MD number	**CART_JOG_MODE** Coordinate system for Cartesian manual travel	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 0x0404
Changes effective IMMEDIATELY	Protection level: 7 / 7	Unit: –
Data type: DWORD	Applies as of SW 6.3	
Meaning:	This allows the reference coordinate system to be set for Cartesian manual travel. Bits 0 to 7 are provided for selecting the coordinate system for the translation, Bits 8 to 15 for selecting the coordinate system for the orientation. 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 translation or orientation, Cartesian manual travel is not active either. The meaning of the individual bits is specified as follows: Bit 0 : Translation in the basic coordinate system (BCS) Bit 1 : Translation in the workpiece coordinate system (WCS) Bit 2 : Translation in the tool coordinate system (TCS) Bit 3 : Not used Bit 4 : Not used Bit 5 : Not used Bit 6 : Not used Bit 7 : Not used Bit 8 : Orientation in the basic coordinate system (BCS) Bit 9 : Orientation in the workpiece coordinate system (WCS) Bit 10 : Orientation in the tool coordinate system (TCS) Bit 11 : Not used Bit 12 : Not used Bit 13 : Not used Bit 14 : Not used Bit 15 : Not used	
MD irrelevant for	Transformation type not equal to “Handling transformation package”	

4.2.8 Channel-spec. MD for Cartesian point-to-point travel

20150 MD number	**GCODE_RESET_VALUES[n] 0 up to max. no. of G codes–1** Initial setting of G group 49		
Default setting: See MD in K1	Minimum input limit: –		Maximum input limit: –
Changes effective after Reset		Protection level: 2 / 7	Unit: –
Data type: BYTE		Applies as of SW 5	
Meaning:	This machine data defines whether CP or PTP or PTPG0 is active after a Reset/end of part program. MD=1: CP is active after Reset MD=2: PTP is active after Reset MD=3: PTPG0 is active after Reset		
Related to	MD 20152: GCODE_RESET_MODE[48]		
Additional references	/FB/, K1, "Axes, Coordinate Systems, Frames", Channel-Specific Machine Data, MD 20150 with assignment of all standard G codes currently in existence		

20152 MD number	**GCODE_RESET_MODE[n] Max. no. of G codes–1** Setting and response after Reset and end of part program in G group 49		
Default setting: 0	Minimum input limit: 0		Maximum input limit: 1
Changes effective after Reset		Protection level: 2 / 7	Unit: –
Data type: BYTE		Applies as of SW 5	
Meaning:	This machine data can be used to modify the setting after Reset and end of part program. MD=0: Specifies for every entry in MD GCODE_RESET_VALUES (i.e. for every G group) whether the setting in MD GCODE_RESET_VALUES is applied in the event of a Reset/end of part program. MD=1: The current setting is maintained following a Reset or end of part program.		
Related to	MD 20150: GCODE_RESET_VALUES[48]		

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_ACT[i], i = 1, 2, 3	ith component of the vector of the end orientation of the current block
$AC_TOOLO_DIFF	Residual angle in degrees, i.e. this is the angle between 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 the orientation are normalized so that the orientation vector has the value 1.

These system variables can be read by part programs and in synchronized 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 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 is close to the limit.
$AA_TOFF_LIMIT [ ] = 0:	Offset not in limit range
$AA_TOFF_LIMIT [ ] = 1:	Offset reached in positive direction
$AA_TOFF_LIMIT [ ] = –1:	Offset reached in negative direction
$AA_TOFF_PREP_DIFF [ ]	Size of the difference between the currently active value of $AA_TOFF[ ] and the value when the current motion block was prepared.

References: /PGA/, Programming Guide Advanced
List of System Variables

■

Notes

Signal Descriptions 5

5.1 Channel-specific signals

DB21–30 DBB232 Data Block	**Number of active G function of G function group 25 (tool orientation reference)** 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 in SW2.1 and higher.	

DB 21–30 **DBX 33.6** Data Block	**Transformation active** Signal(s) from channel (NCK → PLC)	
Edge evaluation: No	Signal(s) updated: Cyclic	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	
Additional references	/PA/, “Programming Guide Fundamentals”	

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

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

DB21 – DB30 DBX318.2 Data Block	**TOFF active** Signal(s) to channel (NCK–> PLC)
Edge evaluation: Yes	Signal(s) updated:
Signal state 1 (or signal transition 0 –> 1)	Activate online tool length offset
Signal state 0 (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 DBX318.3 Data Block	**TOFF motion active** Signal(s) to channel (NCK–> PLC)
Edge evaluation: Yes	Signal(s) updated:
Signal state 1 (or signal transition 0 –> 1)	Activate offset motion
Signal state 0 (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

6 Examples

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:     1st rotary axis is parallel to Z
;                 2nd 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
```

```
;——————————————————————————————————————————
; Example program 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 offset 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 output 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: 3-axis transformation for the machine illustrated in Fig. 1-1 can be configured as follows:

$MC_TRAFO_TYPE_n = 18

```
$MC_TRAFO_GEOAX_ASSIGN_TAB_n[0] = 1   ; Assignment of channel axes to geo 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            ; No second rotary axis
```

6.2.2 Example of a 4-axis transformation

Example: 4-axis transformation for the machine illustrated in Fig. 1-2 but with an additional axis (Y) can be configured as follows:

$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            ; No second rotary axis
```

6.2.3 Set of machine data and part program (extract)

Machine data for a 3-axis and a 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              ; Only for 4-axis transformation
$MA_IS_ROT_AX[AX4] = 1

; 3-axis transformation for moved tool and orientation in 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 linear axis available
$MC_TRAFO_AXES_IN_1[3] = 4                 ; Rotary axis
$MC_TRAFO_AXES_IN_1[4] = 0                 ; No 2nd rot. axis, i.e. 3-axis transf.

; 4-axis transf. for moved workpiece and orientation in 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 linear axis available
$MC_TRAFO_AXES_IN_2[3] = 4                 ; Rotary axis
$MC_TRAFO_AXES_IN_2[4] = 0                 ; No 2nd rot. axis, i.e. 4-axis transformation

CHANDATA(1)
M17                                        ; End of machine data
```

Part 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)                              ; Activate 3-axis transformation
N80 a30                                    ; Axis programming, rotation through 30 degrees
N90  a3=-0.5 b3=0.866025 c3=0.0            ; Prog. direction vector
N100 TRAFOOF()                             ; End of 3-axis transformation
N110 TRAORI(2)                             ; 2nd transformation defined in MD on (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 the tool in zero position in Z
direction
$MC_TRAFO_TYPE_1 = 148

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

;Angle of 2nd rotary axis
$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 offset 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 direction vector
G1 G90
a3 = 0 b3 = 1 c3 = 0

;Programming in Euler angles
G1 G90
a2 = 0 b2 = 0 X0

;Programming of movement of rotary axes
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
```

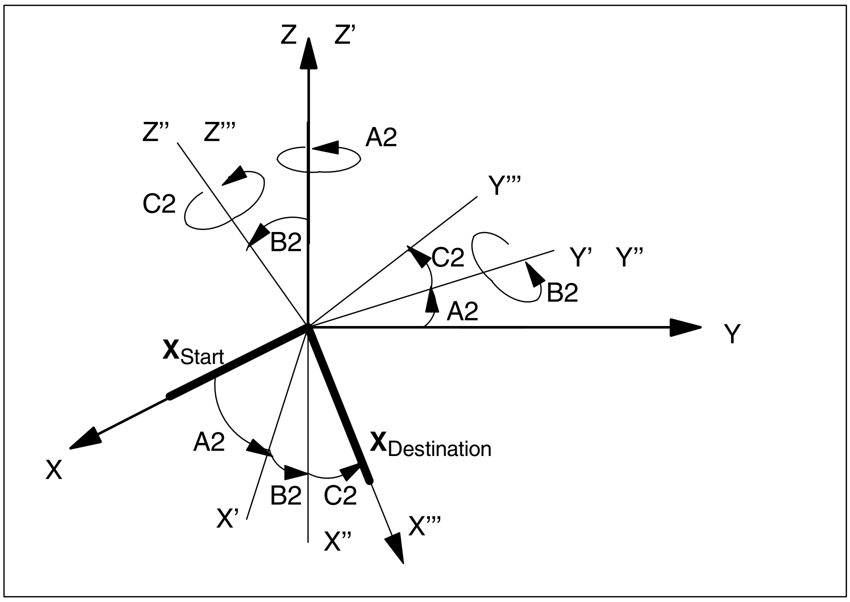


Fig. 6-1 3 orientation axes for the 1st orientation transformation for kinematics with 6 transformed axes

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
```

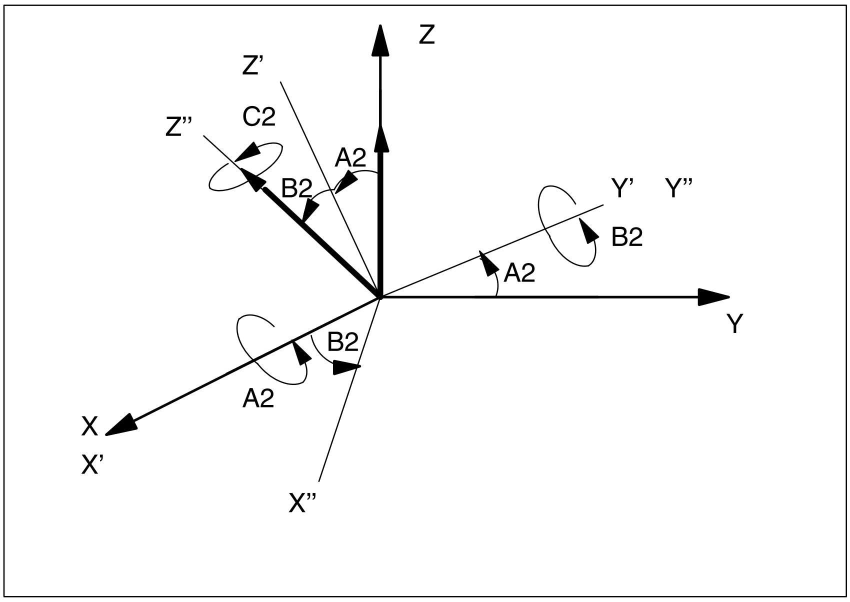


Fig. 6-2 3 orientation axes for the 2nd orientation transformation for kinematics with 5 transformed axes

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.

Reference: /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 possible.
N30 A3=0 B3=0 C3=1                ; Orientation in +Z direction (start vector)
N40 A3=1 B3=0 C3=0                ; Orientation in +X direction (end vector)
```

In N40, the orientation vector is rotated in the Z-X plane 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 possible.
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 the 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 a change in orientation. The tool orientation is rotated from the Y direction to the X direction.

```
N10 TRAORI                             ; Activation of 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 about 90 degrees
N50 A3=0 B3=1 C3=0 PO[THT]=(180.90)    ; in Y direction and rotation about 180 degrees
N60 A3=0 B3=1 C3=0 THETA=IC(–90)       ; remains constant and rotation about 90 degrees.
N70 ORIROTT                            ; Angle of rotation relative to change in orientation.
N80 A3=1 B3=0 C3=0 THETA=30            ; Rotation vector in angle 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 a change in orientation taking place.

N80 The tool orientation is rotated from the Y direction to the X direction. The change in orientation takes place in the X-Y plane and the rotation vector describes an angle of 30 degrees to this plane.

6.6 Example for 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.). 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 offset 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 orientation parallel to Z ⇒ B–45 C0
N130 T1 D1                   ; Basic orientation is now parallel to Z
N140 C3 = 1                  ; Set orientation parallel to Z ⇒ B0 C0
N150 G19                     ; Basic orientation is now parallel to X
N160 C3 = 1                  ; Set orientation parallel to Z ⇒ B–90 C0
N170 G17 TCARR=1 TCOABS      ; Basic orientation is now bisecting
                             ; line Y-Z
N180 A3=1                    ; Set orientation parallel to X ⇒ B–90 C–135
N190 B3 =1 C3 = 1            ; Set ori. parallel to basic orientation ⇒ B0 C0
N200 TRAORI(,2.0, 3.0, 6.0)  ; Transfer basic orientation in call
N210 A3 = 2 B3 = 3 C3 = 6    ; Orient. parallel to basic orientation ⇒ B0 C0
N220 TOFRAME                 ; Z axis pointing in direction of orientation
N230 G91 Z7                  ; Travel 7mm in new Z dir. ⇒ X2 Y3 Z6
N240 C3 = 1                  ; Orientation parallel to new Z axis ⇒ B0 C0
N250 M30
```

6.6.1 Example for a generic 6-axis transformation (SW 7.2 and higher)

Activation of a 6-axis transformation with subsequent orientation changes and traversing:

```
N10   A0 B0 X0 Y0 Z0            ;
N20   TRAORI(1, ,,, 0,0,0, 0,1,0); Transfer of the orientation vector and the
                                ; orientation normal vector, transformation
                                ; selection
N30   T1 D1 X10 Y20 Z30 A3=0.5 C3=1 BN3=1 ORIPLANE ORIWKS
                                ; Orientation change, rotation and
                                ; traversing
N40   B3=0.5 C3=1 AN3=–1        ; Rotation programmed, orientation constant
N50   M30
```

A tool, whose orientation differs from the default, is defined in the following example. With G17, the orientation vector is in the X-Z plane and is inclination to the Z axis of 26.565 degrees because of tan(26.565) = 0.5 = $TC_DPV3[2,2] / $TC_DPV5[2,2].

The orientation normal vector is also specified. As only $TC_DPVN4[2,2] is not equal to zero, it points in the Y direction. Orientation vector and orientation normal vector are perpendicular to one another.
An orthogonalization is therefore **not necessary** and the programmed orientation normal vector is therefore not modified.

```
N100  $TC_DP1[2,2]=120          ; End mill
N110  $TC_DP3[2,2]=  20         ; Length offset vector

N120  $TC_DPV[2,2]= 0           ; Tool edge orientation
N130  $TC_DPV3[2,2]= 1          ; X component of tool edge orientation.
N140  $TC_DPV4[2,2]= 0          ; Y component of tool edge orientation.
N150  $TC_DPV5[2,2]= 0.5        ; Z component of tool edge orientation.

N160  $TC_DPVN3[2,2]= 0         : X component of orientation normal vector
N170  $TC_DPVN4[2,2]= 1         ; Y component of orientation normal vector
N180  $TC_DPVN5[2,2]= 0         ; Z component of orientation normal vector

N200  TRAORI( )                 ; Transfer basic orientation in call
N210  A3=5 C3=10 BN3=1          ; Bring rotary axes into initial position
N220  C3=1                      ; Orientation in Z direction ⇒ tool
                                ; rotates through 26.565 degrees
N230  THETA=IC(90)              ; Rotate orientation normal vector through
                                ; 90 degree increments. Vector points in
                                ; negative X direction.
N240  M30
```

6.6.2 Example for the modification of the rotary axis motion (SW 6.1 and higher)

The machine is a 5-axis machine of machine type 1 (two-axis swivel head with CA kinematics) on which both rotary axes rotate the tool (transformation type 24). The first rotary axis is a modulo axis parallel to Z (C axis); the second rotary axis is parallel to Y (B axis) and has a traversing range from –5 degrees to +185 degrees.

To allow modification at any time, 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 on the outside of the taper at the same time. Although the programmed orientation changes are executed one after the other, but in an unsteady way, the compressor generates a smooth motion of the orientation.

```
DEF INT NUMBER = 60
DEF REAL RADIUS = 20
DEF INT COUNTER
DEF REAL ANGLE
N10 G1 X0 Y0 F5000 G64

$SC_COMPRESS_CONTUR_TOL = 0.05          ; Maximum deviation
                                        ; of contour 0.05 mm
$SC_COMPRESS_ORI_TOL = 5                ; Maximum deviation
                                        ; of orientation 5 degrees

TRAORI
COMPCURV

              ; A circle generated from polygons is traversed.
              ; The orientation moves on a taper about the
              ; Z axis with an aperture angle of 45 degrees.

N100 X0 Y0 A3=0 B3=–1 C3=1
N110 FOR COUNTER = 0 TO NUMBER
N120 ANGLE = 360 * COUNTER/NUMBER
N130 X=RADIUS*cos(ANGLE) Y=RADIUS*sin(ANGLE)
     A3=sin(ANGLE) B3=–cos(ANGLE) C3=1
N140 ENDFOR
...
```

■

Notes

7 Data Fields, Lists

7.1 Interface signals

DB number	Bit, byte	Name	Reference
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 offset motion	

7.2 Setting data

Number	Identifier	Name	Reference
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)	
42660	ORI_JOG_MODE	Definition of virtual kinematics for JOG (SW 7.2 and higher)	
42670	ORIPATH_SMOOTH_DIST	Distance for the smoothing of the orientation (SW 7.3 and higher)	
42672	ORIPATH_SMOOTH_TOL	Tolerance for the smoothing of the orientation (SW 7.3 and higher)	
42970	TOFF_LIMIT	Upper limit for offset value $AA_TOFF (SW 6.4 and higher)	

7.3 Machine data

Number	Identifier	Name	Refe-rence
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 components	
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 the axes	
10674	PO_WITHOUT_POLY	Allows Programming of PO[ ] without having to activate POLY (SW 5.3 and higher)	

Number	Identifier	Name	Refe-rence
Channelspecific ($MC_ ...)			
20150	GCODE_RESET_VALUES[n]	Reset G groups	
20152	GCODE_RESET_MODE[n]	Setting after RESET/end of part 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	
21094	ORIPATH_MODE	Setting for orientation relative to the path (SW 7.3 and higher)	
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]	

Number	Identifier	Name	Refe-rence
21160	JOG_VELO_RAPID_GEO[n]	Rapid traverse in jog mode for geometry axes in the channel [n = 0..2]	
21165	JOG_VELO_GEO[n]	Geometry axis velocity in jog mode [n = 0..2]	
21170	ACCEL_ORI[n]	Acceleration for orientation axes [n = 0..2]	
21180	ROT_AX_SWL_CHECK_MODE	Check software limits for orientation axes (SW 6.1 and higher)	
21186	TOCARR_ROT_OFFSET_FROM_FR	TOCARR rotary axes offset (SW 6.4 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.]	

Number	Identifier	Name	Refe-rence
24460	TRAFO_TYPE_8	Definition of transformation 8 in channel	
24462	TRAFO_AXES_IN_8[n]	Axis assignment for transformation 8 [axis index]	
24464	TRAFO_GEOAX_ASSIGN_TAB_8[n]	Assignment geometry axis to channel axis for transformation 8 [geometry no.]	
24470	TRAFO_TYPE_9	Definition of transformation 9 in channel	
24472	TRAFO_AXES_IN_9[n]	Axis assignment for transformation 9 [axis index]	
24474	TRAFO_GEOAX_ASSIGN_TAB_9[n]	Assignment geometry axis to channel axis for transformation 9 [geometry no.]	
24480	TRAFO_TYPE_10	Definition of transformation 10 in channel	
24482	TRAFO_AXES_IN_10[n]	Axis assignment for transformation 10 [axis index]	
24484	TRAFO_GEOAX_ASSIGN_TAB_10[n]	Assignment geometry axis to channel axis for transformation 10 [geometry no.]	
24500	TRAFO5_PART_OFFSET_1[n]	Offset vector for 5-axis transfor. 1 [n = 0.. 2]	
24510	TRAFO5_ROT_AX_OFFSET_1[n]	Position offset of rotary axis 1/2 for 5-axis transformation 1 [axis no.]	
24520	TRAFO5_ROT_SIGN_IS_PLUS_1[n]	Sign of rotary axis 1/2 for 5-axis transformation 1 [axis no.]	
24530	TRAFO5_NON_POLE_LIMIT_1	Definition of pole range for 5-axis transformation 1	
24540	TRAFO5_POLE_LIMIT_1	End angle tolerance with interpolation through pole for 5-axis transformation 1	
24550	TRAFO5_BASE_TOOL_1[n]	Vector of base tool for activation of 5-axis transformation 1 [n = 0.. 2]	
24558	TRAFO5_JOINT_OFFSET_PART_1[n]	Vector of kinematic offset in table for 5-axis transformation 1 [n = 0.. 2]	
24560	TRAFO5_JOINT_OFFSET_1[n]	Vector of kinematic offset for 5-axis transformation 1 [n = 0.. 2]	
24561	TRAFO6_JOINT_OFFSET_2_3_1[n]	Vector of kinematic offset for 6-axis transformation 2_3_1 (SW 7.2 and higher)	
24562	TRAFO5_TOOL_ROT_AX_OFFSET_1[n]	Offset of focus of 1st 5-axis transformation with swiveled linear axis.	
24564	TRAFO5_NUTATOR_AX_ANGLE_1	Angle of 2nd rotary axis for the universal milling head	
24570	TRAFO5_AXIS1_1[n]	Vector for the first rotary axis and the first orientation transformation [n = 0.. 2] (SW 5.2 and higher)	
24572	TRAFO5_AXIS2_1[n]	Vector for the second rotary axis and the first transformation [n = 0.. 2] (SW 5.2 and higher)	
24673	TRAFO5_AXIS3_1[n]	Direction of third rotary axis for generic 6-axis transformation (transformation type 24, 40, 56, 57 for SW 7.2 and higher)	
24574	TRAFO5_BASE_ORIENT_1[n]	Basic orientation for the first transformation [n = 0.. 2] (SW 5.2 and higher)	
24576	TRAFO6_BASE_ORIENT_NORMAL_1[n]	Tool normal vector for the first transformation [n = 0.. 2] (SW 7.2 and higher)	
24580	TRAFO5_TOOL_VECTOR_1	Tool vector direction for the first 5-axis transformation 1	

Number	Identifier	Name	Refe-rence
24582	TRAFO5_TCARR_NO_1	TCARR number for the first 5-axis transformation 1 (SW 7.2 and higher)	
24585	TRAFO5_ORIAX_ASSIGN_TAB_1[n]	Assignment of orientation axes to channel axes for orientation transformation 1 [n = 0.. 2]	
24590	TRAF5_ROT_OFFSET_FROM_FR_1	Offset of transf. rotary axes from WO (SW 6.4 and higher)	
24600	TRAFO5_PART_OFFSET_2[n]	Offset vector for 5-axis transformation 2 [n = 0.. 2]	
24610	TRAFO5_ROT_AX_OFFSET_2[n]	Position offset of rotary axis 1/2 for 5-axis transformation 2 [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]	
24658	TRAFO5_JOINT_OFFSET_PART_2[n]	Vector of kinematic offset in table for 5-axis transformation 2 [n = 0.. 2]	
24660	TRAFO5_JOINT_OFFSET_2[n]	Vector of kinematic offset for 5-axis transformation 2 [n = 0.. 2]	
24661	TRAFO6_JOINT_OFFSET_2_3_2[n]	Vector of kinematic offset for 6-axis transformation 2_3_2 (SW 7.2 and higher)	
24662	TRAFO5_TOOL_ROT_AX_OFFSET_2[n]	Offset of focus of 2nd 5-axis transformation with swiveled linear axis.	
24664	TRAFO5_NUTATOR_AX_ANGLE_2	Angle of 2nd rotary axis for the universal milling head	
24670	TRAFO5_AXIS1_2[n]	Vector for the first rotary axis and the second orientation transformation [n = 0.. 2] (SW 5.2 and higher)	
24673	TRAFO5_AXIS3_2[n]	Direction of third rotary axis for generic 6-axis transformation (type 24, 40, 56, 57)	
24672	TRAFO5_AXIS2_2[n]	Vector for the second rotary axis and the first transformation [n = 0.. 2] (SW 5.2 and higher)	
24674	TRAFO5_BASE_ORIENT_2[n]	Basic orientation for the second transformation [n = 0.. 2] (SW 5.2 and higher)	
24676	TRAFO6_BASE_ORIENT_NORMAL_2[n]	Tool normal vector for the second transformation [n = 0.. 2] (SW 7.2 and higher)	
24680	TRAFO5_TOOL_VECTOR_2	Tool vector direction for the second 5-axis transformation 2	
24682	TRAFO5_TCARR_NO_2	TCARR number for the second 5-axis transformation 2 (SW 7.2 and higher)	
24685	TRAFO5_ORIAX_ASSIGN_TAB_2[n]	Assignment of orientation axes to channel axes for orientation transformation 2 [n = 0.. 2]	
24590	TRAF5_ROT_OFFSET_FROM_FR_2	Offset of transf. rotary axes from WO (SW 6.4 and higher)	
28580	MM_ORIPATH_CONFIG	Configuration for orientation relative to the path ORIPATH (SW 7.3 and higher)	

7.4 Alarms

Detailed explanations of the alarms, which may occur, appear in

References: /DA/, “Diagnostics Guide”

or in the Online help.

■

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

Gantry Axes (G1)

■

Notes

1 Brief Description

Gantry axes

The “gantry axes” function (see Chapter 3) can be used to traverse two or more mechanically-coupled machine axes 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 (SW 7.1 and higher). 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 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 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 leading axis and the synchronized axis. As these axes 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 leading axis based on machine data settings. The leading axis and synchronized axes cannot be traversed separately.

Leading axis: The leading axis is the gantry axis, which actually exists from the point of the view of the operator and programmer and can therefore be controlled accordingly in the same way as a normal NC axis. The axis name of the leading axis identifies all axes in the gantry axis grouping.

Synchronized axis: The synchronized axis is the gantry axis whose setpoint position is always derived from the traversing motion of the leading axis. It therefore moves in exact synchronism with the leading 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, which one
- Whether the axis is defined as a **leading axis** or a **synchronized axis** within this grouping.

A total of up to 3 (8 in SW 7.1 and higher) gantry groupings can be defined.

Each gantry grouping consists of **one** leading 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 leading axis in another axis grouping.

Note

Each axis in the gantry grouping must be set so that it can take over the function of the leading axis at any time, i.e. matching velocity, acceleration and dynamic response settings.

The control performs a plausibility check on the axis definition.

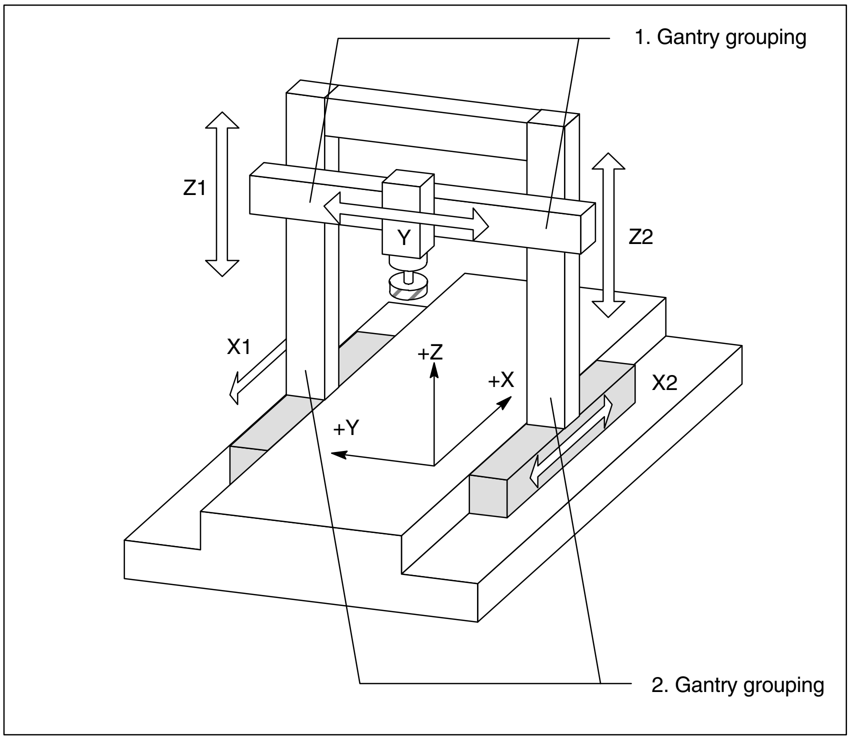


Fig. 2-1 Example: Gantry-type milling machine with 2 gantry groupings

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 leading 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 leading axis, were programmed in the NC. Analogously, only the leading axis is programmed in the part program. The commands and traverse requests from the operator, the PLC interface or via the part 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 leading axis in all operating modes.

Note

The dynamic control response of the leading and synchronized axes must be set identically.

Monitoring of actual value difference

The position actual values of the leading 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.

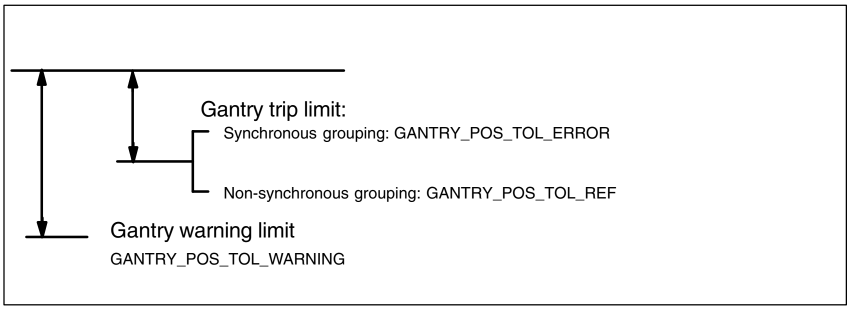
Machine data can be set to specify the following limit values for alarm output and termination of the traversing motion for specific axes:

- **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. In addition, IS “Gantry warning limit exceeded” (DB31, ... ; DBX101.3) is output to the PLC. The warning message and interface signal are canceled automatically when the value falls back below the warning limit.

  If 0 is entered in MD 37110: GANTRY_POS_TOL_WARNING no warning message is output.

- **Gantry trip 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 has not yet been synchronized, the limit value 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.

Extended monitoring

MD 37150: GANTRY_FUNCTION_MASK can be used to activate an extended monitoring function. Effect:

– An offset between the master and slave axis resulting from follow-up or Break_Up 1 is included in the synchronous operation difference and triggers an alarm when the corresponding warning or error limits are reached

After power-up, extended monitoring is activated when the first reference point or synchronization (absolute encoder) is reached, irrespective of the axis status: e.g. RFG, NF, reference point approach, etc.

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 intact in all operating modes as well as immediately after power ON. In cases where an incremental measuring system is being used for the leading or the synchronized axis, the reference point must be approached while maintaining the axis coupling immediately after machine power ON.

After every axis in the gantry grouping has approached its reference point, any misalignment that may exist between the axes must be eliminated (this is known as the gantry synchronization process). As soon as this takes place, IS “Gantry grouping is synchronized” (DB31, ... ; DBX101.5) is signaled 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 in normal operation, the leading and synchronized axes move in positional synchronism even during acceleration and braking.

Load effects are compensated by the appropriate 611D drive of the leading 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 leading 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 leading 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 leading 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

All gantry axes must first be referenced and then synchronized after the control system is switched on. During gantry synchronization, all gantry axes approach **the reference position of the gantry grouping in the decoupled state**. The reference position of the gantry grouping for referencing the gantry axes corresponds to the **reference position of the leading axis** (MD 34100: REFP_SET_POS) or the **current actual position of the leading axis**.

These operations for referencing and synchronizing the gantry axes are executed automatically in accordance with a special flowchart.

Referencing operation

The flowchart for referencing gantry axes using an incremental measuring system is as follows:

Chapter 1: **Referencing of leading axis**
The user program starts axis-specific referencing of the gantry axes with IS "Travel key plus/minus" (DB31, ... ; DBX4.7/4.6) on the leading axis when the REF machine function is active.

The leading 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 leading axis.

IS "Referenced/synchronized" of the leading axis is output to indicate that the reference point has been reached.

Chapter 2: **Referencing of synchronized axes**
As soon as the leading axis has approached its reference point, the synchronized axis is referenced **automatically** (corresponding to reference point approach (see **References:** /FB/, R1 "Reference Point Approach"). The dependency between the leading axis and synchronized axis is inverted in the control for this phase so that the leading axis 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.

Chapter 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 leading axis.

The next step in the operating sequence depends on the difference calculated between the actual values of the leading 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 leading axis in MD 34100: REFP_SET_POS.

   If the leading 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 leading 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 leading axis. The signal is set on the leading axis. The operational sequence is then the same as that described above.

The following flowchart illustrates the referencing and synchronization processes.

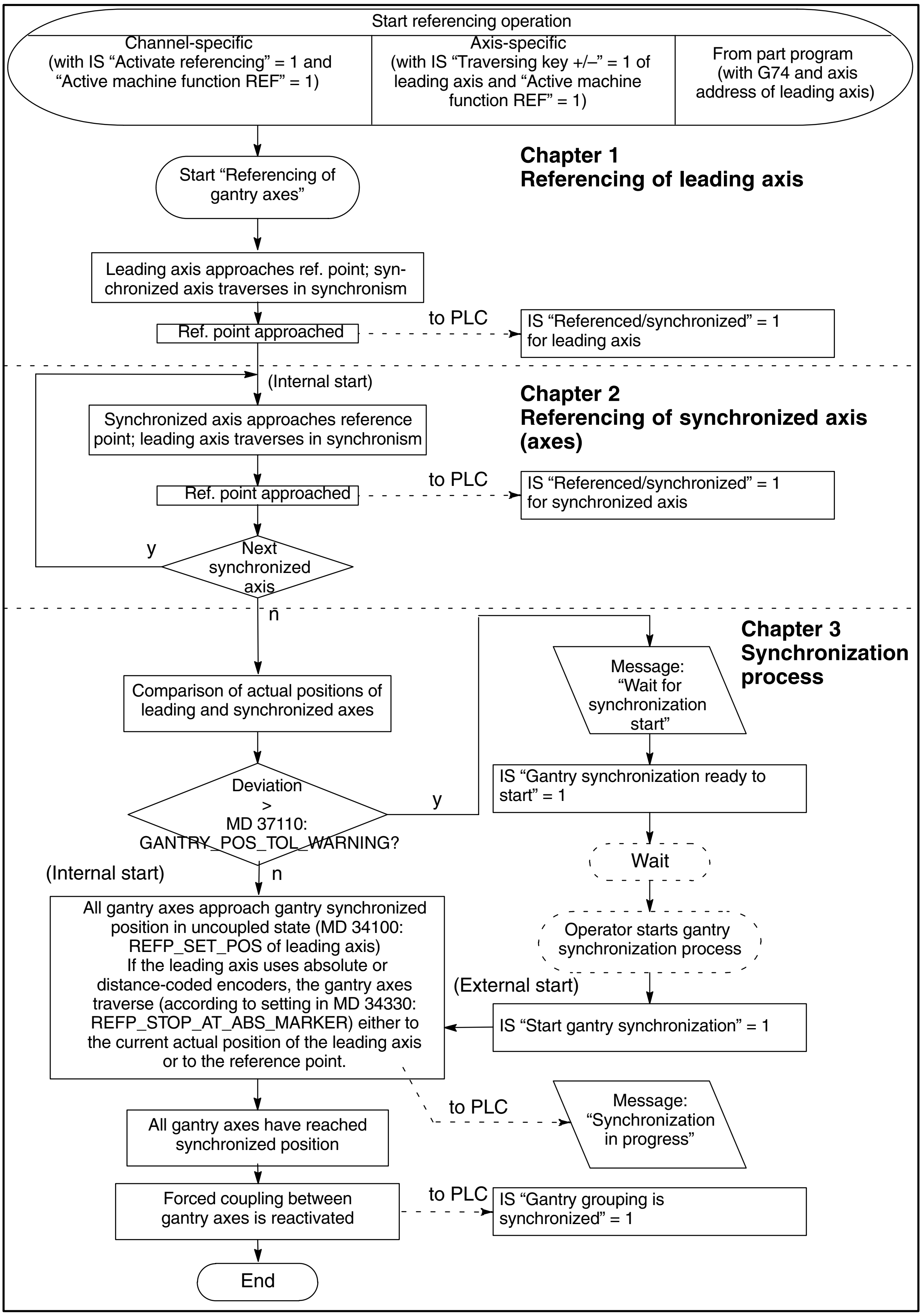


Fig. 2-2 Flowchart for referencing and synchronization of gantry axes

Synchronization process

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 sequence

If the referencing process described above is interrupted as a result of disturbances or a RESET, proceed as follows:

1. **Abort during Chapter 1 or 2:**
   Approach the reference point again with the leading axis (see Chapter 1)
2. **Abort during Chapter 3:**
   In cases where the gantry axes have not yet been referenced (IS “Referenced/Synchronized” = 1), the gantry synchronization process can be started again with IS “Synchronize gantry grouping”.

Restart gantry synchronization

Synchronization of the gantry axes can be started with IS “Start gantry synchronization” under the following conditions only:

- 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 of the grouping are within the tolerance window or
  IS (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 instead of the reference position (MD 34100: REFP_SET_POS) being specified as the target position for the synchronized axes, **the current actual position of the leading axis** is specified and is approached in the uncoupled state.

Note

Automatic synchronization can be interlocked for the leading axis using IS DB31, ... DBX29.5. 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 gantry grouping). In this case, the synchronized axes traverse to the current actual position of the leading axis in the uncoupled state.
(SW 5.3 AND HIGHER) If an Emergency Stop command is issued and then canceled again while a gantry grouping is in motion and the two axes have drifted apart by less than the standstill tolerance of the slave axes, then the grouping is automatically synchronized. It is no longer necessary to switch to REFP mode. Automatic synchronization can be disabled using IS DBxx.DB29.5 on the slave axis.

Reference point selection

To ensure that the shortest possible paths are traversed when the gantry axes are referenced, the reference point values of the leading 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.

During referencing, the reference point value of the leading axis is specified as the target position for all axes in the grouping for the synchronization compensatory motion. This position is then approached without axis coupling. If the leading 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 leading axis or to the reference point.

If only one reference cam is used for the leading 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 Subsection 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 leading and synchronized axes is disabled. The grouping is no longer synchronized as a result. IS “Gantry grouping is synchronized” (from leading axis, DB31, ... DBX101.5) is set to 0, irrespective of the axis positions.

If the gantry grouping is switched from follow-up mode to position control mode, axis synchronism is automatically restored provided the actual-value monitor does not detect a difference between the positions of the leading and synchronized axes greater than the setting in 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

IS “No automatic synchronization” DB31, ... DBX29.5 blocks automatic synchronization in all modes except referencing mode. If automatic synchronization is to be activated here, set IS “No automatic synchronization” DB31, ... DBX29.5 = 0 must be defined. Then switch one of the axes in the gantry grouping from follow-up mode to position-controlled mode. This is achieved with:

IS “Follow-up mode” DB31, ... DBX1.4 = 1 and
IS “Delete distance to go/spindle reset” DB31, ... DBX2.1 = 1 signal change from 1 to 0 to 1”.

2.2.3 Special features

2nd position measuring system for each 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 leading axis is used as the axis sequence for channel-specific referencing. After the reference point of the leading axis has been reached, the synchronized axes are referenced first as described above.

Referencing from part program with G74

The referencing and synchronization process for gantry axes can also be initiated from the part program by means of command G74. In this case, only the axis name of the leading axis may be programmed. The operational sequence is analogous to that described for axis-specific referencing.

Position measuring system with distance-coded reference marks

In order that return traverses do not have to be made over large distances, it is possible to use a position measuring system with distance-coded reference marks as a sole or second measuring system for gantry axes. In this way the measuring system is referenced after traversal of a short path (e.g. 20 mm). The procedure for referencing the gantry axes is the same as that described for 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 leading axis defined in MD 34100: REFP_SET_POS in the decoupled state. If the leading 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 leading axis or to the reference point.

Activation of axis compensations

Compensation functions can be activated for both the leading axis and the synchronized axes. Compensation values are applied separately for each individual gantry axis. These values must therefore be defined and entered for the leading axis and the synchronized axes during start-up.

The compensations do not become operative internally in the control until the axis is referenced or the gantry grouping synchronized. The following applies:

Compensation type	Takes effect when	PLC interface signal
Backlash compensation	Axis is referenced	“Referenced/Synchronized”
LEC	Axis is referenced	“Referenced/Synchronized”
Sag compensation	Gantry grouping is synchronized	“Gantry grouping is synchronized”
Temperature compensation	Gantry grouping is synchronized	“Gantry grouping is synchronized”

If a movement by the synchronized axis (axes) is caused by an active compensation, a travel command is displayed for the synchronized axis (axes) independently of the leading axis.

Monitoring functions effective

Analogous to normal NC axes, the following monitoring functions do not take effect for gantry axes until the reference point is reached (IS “Referenced/Synchronized”):

- Working area limits
- Software limit switch
- Protection zones

The axial machine data values are used as monitoring limit values for the synchronized axes as well.

Multi-channel block search in SW 6.1 and higher

The cross-channel block search in Program Test mode (SERUPRO “**se**arch **r**un by **pro**gram test”) can be used to simulate the traversal of gantry axis groupings in SW 6.2 and higher.

Note

For more information about multi-channel SERUPRO block search, please see:

References: /FB/, K1, “Mode Group, Channels, Program Operation” 2.4 Program test

---

2.3 Start-up of “gantry axes”

General

Owing to the forced coupling which is normally present between leading 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 leading and synchronized axes must always be defined and entered jointly.

If the synchronized axis is being overloaded by the leading axis due to reduced dynamics, this is acknowledged with alarm 10656.

References: /IAD/, “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

In MD 37100: GANTRY_AXIS_TYPE, the following must be specified for the gantry axis:

- To which gantry grouping (1, 2 or 3) the axis must be assigned
- Whether it is to act as the leading axis (single-decade MD value only) or as a synchronized axis.

Note

Please make sure that a gantry grouping specified as cross-channel or cross-NCU does not clash with gantry grouping numbers already assigned. In such cases, unique numbers must be assigned for cross-channel and cross-NCU gantry groupings. If clashes are detected, alarm 10651 is output with reason 40XX. XX is the gantry grouping causing the clash.

For start-up purposes, all axes in a gantry grouping must be declared either as linear axes or as rotary axes (MD 30300: IS_ROT_AX).

Table 2-1 Examples for defining the gantry axis grouping:

MD: GANTRY_AXIS_TYPE	Gantry axis	Gantry grouping
0	None	–
1	Leading axis	1
11	Synchronized axis	1
2	Leading axis	2
12	Synchronized axis	2
3	Leading axis	3
13	Synchronized axis	3
...		
8	Leading axis	8
18	Synchronized axis	8

Entering gantry trip limits

For the purposes of monitoring the position actual value deviation between the synchronized axis and the actual position of the leading axis, the trip limit values (MD: 37120 GANTRY_POS_TOL_ERROR or MD 37130: GANTRY_POS_TOL_REF) must be entered for the leading 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 setpoint 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 leading and synchronized axes are set to the **same dynamic response value**.

To ensure the best possible synchronism, the leading 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 leading 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 leading 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 adaptation

The leading 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.

Leading axis MD 32810: EQUIV_SPEEDCTRL_TIME [n] = 5 ms
Synchronized axis MD 32810: EQUIV_SPEEDCTRL_TIME [n] = 3 ms

→ Time constant of dynamic response adaptation for synchronized axis:
MD 32910: DYN_MATCH_TIME [n] = 5 ms – 3 ms = 2 ms

The dynamic response adaptation function must be activated axially with MD 32900: DYN_MATCH_ENABLE.

Check of dynamic response adaptation:
The following errors of the leading 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 gantry axes

The positions of the reference points of the leading and synchronized axes must first be set to almost identical values.

To ensure that the synchronization compensatory motion of the gantry axes is not started automatically, the gantry warning limit (MD 37100: GANTRY_POS_TOL_WARNING) must be set to 0 prior to referencing on first start-up. This will prevent a warning message being output during traversing motion.

In cases where an excessively high additional torque is acting on the drives due to misalignment between the leading 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 leading 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 leading and synchronized axes are identical after execution of the compensatory motion.

Synchronizing 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 leading and synchronized axes must be checked to ensure that it equals 0. Corrections may need to be made in the machine data mentioned above.

Input of gantry warning limit

Once the reference point values for the leading and synchronized axes have been optimized so that the gantry axes are perfectly aligned with one another after synchronization, the warning limit values for all axes must be entered in 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.

Calculating and activating compensations

In cases where the gantry axes require compensation (backlash, sag, temperature or leadscrew error), the compensation values for the leading 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
In all software versions up to and including SW 3.1, the function generator and measuring function may only be activated on the **leading axis**.
The **synchronized axis** joins the traversing motion automatically (by means of coupling 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 leading 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 may damage the machine.

SW 3.2 and higher
In SW 3.2 and higher, the activation of the function generator and measuring function on the synchronized axis is aborted with an error message.
If the synchronized axis absolutely has to be activated (e.g. in order to measure the machine), the leading axis and synchronized axis must be inverted 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 leading or synchronized axes. Table 2-2 below shows all special gantry-PLC interface signals along with their codes and indicates whether the IS is evaluated on the leading axis or the synchronized axis.

Table 2-2 Assignment of gantry-PLC interface signals to leading and synchronized axes

PLC interface signal	PLC↔NCK	DB31, ... ; DBX ...	Leading axis	Synchronized axis
Start gantry synchronization	→	29.4	X	
No automatic synchronization	→	29.5	X	
Gantry axis	←	101.7	1	1
Gantry leading 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 axial interface signals on gantry axes

a) **Axial interface signals from PLC to axis (PLC –> NCK)**

The axial interface signals from the PLC to the axis are always referred to all gantry axes in the grouping. In this case, all gantry axes (leading and synchronized axis) have equal priority.

For example, if the leading axis sets IS “Servo enable” (DB31, ... ; DBX2.1) to “0”, all axes of the gantry grouping are brought to a standstill at the same time.

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 leading and synchronized axes

PLC interface signal	DB31, ... ; DBX ...	Effect on	
		Leading 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 leading and synchronized axes

PLC interface signal	DB31, ... ; DBX ...	Effect on	
		Leading 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 switchover between position measuring systems 1 and 2 applies individually for each gantry axis. However, deactivation of both position measuring systems (known as the parking position) applies as a common signal for all gantry axes.
2. IS “Servo enable” (DB31, ... ; DBX2.1)
   If the servo enable signal on one gantry axis is canceled, all axes in the gantry grouping are shut down simultaneously. The method by which shutdown is implemented (e.g. with fast stop) is identical for all gantry axes.

Depending on 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 leading 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:
IS “Travel command plus or minus” (DB31, ... ; DBX64.6 and 64.7) is also set for the synchronized axis when the leading axis traverses.

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 leading axis in coupled axis mode. In this case, the synchronized axes traverse in synchronism with the leading axis.

DRF offset

A DRF offset can only be applied to the leading axis. In this case, the synchronized axes traverse in synchronism with the leading axis.

Programming in part program

Only the leading axis of a gantry axis grouping may be programmed in the part program. An alarm is generated when a synchronized axis is programmed.

PLC or command axes

Only the leading axis of the gantry grouping can be traversed by the PLC using FC18 or as a command axis by means of synchronized actions.
References: /FB/, P3, “PLC Basic Program”
/FBSY/, Synchronized Actions

PRESET

The PRESET function can only be applied to the leading axis. All axes in the gantry grouping are reevaluated internally in the control when PRESET is activated. The gantry axes then lose their reference and synchronization (IS “Gantry grouping synchronized” = “0”).

Exchange

All axes in the gantry grouping are released automatically in response to a RELEASE command (leading axis).

A replacement of the master axis of a closed gantry grouping is only possible, if all axes of the grouping are known in the channel in which they are to be transferred, otherwise alarm 10658 is signaled.

Axes of a gantry grouping must not be known in all channels. The check is not performed when powering up, but only when an attempt is made to replaced the master axis in the channel.

If an attempt is made to close a broken up gantry grouping with MD 37140: GANTRY_BREAK_UP, no automatic axis replacement and no automatic adjustment of the axis states of the gantry axes are performed. The user is responsible for this. However, a check of the axes states is performed after break-up and, if necessary, alarm 10658 is issued. If a gantry grouping is to be closed again, the user must ensure that all axes of the grouping are in one channel with the appropriate axis state.

MD 30450: IS_CONCURRENT_POS_AX=1 for slave axes (closed gantry grouping):

With active gantry grouping, the MD is ignored for the slave axes. The state of the master axis is assumed. The user is informed about the inappropriate configuration with display alarm 4300.

Display data

The position actual value display shows the actual values of both the leading axis and the synchronized axes. The same applies to the service display values in the “Diagnosis” operating area.

SW limit switch

The SW limit switch monitor is processed for the leading axis only. If the leading axis crosses the limit switch, all axes in the gantry grouping are braked to a standstill.

Differences in comparison with the "Coupled motion" function

The main differences between the "gantry axes" and "coupled motion" functions are listed below:

- The axis coupling between the gantry axes must always be active. It is therefore not possible to separate the axis coupling between "gantry axes" by means of the part program. In contrast, the coupled axis grouping can be separated by means of the part program and the axes then traversed individually.
- With "gantry axes", the difference between the position actual values of the leading and synchronized axes is continuously monitored and the traversing motion terminated in response to illegal deviations. No such monitoring takes place with the "coupled motion" function.
- Gantry axes must remain coupled even during referencing. For this reason, special procedures are applied for the reference point approach of gantry axes. In contrast, coupled-motion axes are referenced as individual axes.
- To allow "gantry axes" to traverse without a mechanical offset, the dynamic control response settings of the synchronized axes and the leading axis must be identical. In contrast, the "coupled motion" function permits axes with different dynamic control response characteristics to be coupled.

References: /FB/, M3 "Coupled Motion"

Multiple channels

Please make sure, if a gantry grouping's master axis is declared in several channels, that its slave axes are also declared in those channels.

If this is not the case, alarm 10651 is output with reason 60XX. XX is the gantry grouping causing the clash.

■

Supplementary Conditions 3

There are no other supplementary conditions to note.

■

Notes

Data Descriptions (MD, SD) 4

4.1 Axisspecific machine data

37100 MD number	**GANTRY_AXIS_TYPE** Gantry axis definition	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 13
Changes effective after POWER ON	Protection level: 2/4	Unit: –
Data type: BYTE	Applies as of SW: 2.1	
Meaning:	Use axial MD: GANTRY_AXIS_TYPE to set in the form of a two-decade value specification: • Whether the axis belongs to a **gantry grouping** and if so, to which (**1st decade**) • and whether the axis is declared within the grouping as a **leading axis** (MD value has only 1 decade) or as a **synchronized axis** (2nd decade is set to value 1) [2. decade | 1. decade] 2. decade → **Gantry axis type**: (no entry): Leading axis; 1: Synchronized axis 1. decade → **Gantry axis grouping**: 0: No gantry axis; 1: Gantry axis grouping 1; 2: Gantry axis grouping 2; i: Gantry axis grouping i A maximum total of 3 (8 in SW 7.1 and higher) gantry axis groupings can be defined. A gantry axis grouping has a leading axis and at least one synchronized axis (a maximum of two synchronized axes are possible). **Declaration conditions:** • All axes in a gantry axis grouping must be declared either as linear axes or as rotary axes (MD 30300: IS_ROT_AX). • A spindle may not be declared within a gantry axis grouping. • A synchronized axis may not be declared either as a geometry axis or as a "concurrent positioning axis". • A synchronized axis may not be declared as the leading axis of another gantry grouping. The gantry axis definition is subject 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.	
MD irrelevant for	SINUMERIK 840D with NCU 571	
Application example(s)	0: No gantry axis 1: Axis is leading axis in gantry grouping 1 11: Axis is synchronized axis in gantry grouping 1 2: Axis is leading axis in gantry grouping 2 12: Axis is synchronized axis in gantry grouping 2 3: Axis is leading axis in gantry grouping 3 13: Axis is synchronized axis in gantry grouping 3	
Special cases, errors, ...	Alarm 10650 "Incorrect gantry machine data" and 10651 "Gantry unit undefined" in response to incorrect gantry axis definition.	
Related to	MD 37110: GANTRY_POS_TOL_WARNING Gantry warning limit MD 37120: GANTRY_POS_TOL_ERROR Gantry trip limit MD 37130: GANTRY_POS_TOL_REF Gantry trip limit for referencing	

37110 MD number	**GANTRY_POS_TOL_WARNING** Gantry warning limit				
Default setting: 0		Minimum input limit: 0		Maximum input limit: plus	
Changes effective after Reset			Protection level: 2/4		Unit: Linear axis: mm Rotary axis: degrees
Data type: DOUBLE				Applies as of SW: 2.1	
Meaning:	Value > 0 With gantry axes, the difference between the position actual values of the leading and synchronized axes are continuously monitored. 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 so that the machine can withstand the position actual value deviation between the gantry axes without sustaining mechanical damage. In addition, 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. 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. Effect of gantry warning limit on gantry synchronization process: The position actual value difference between the leading 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 started automatically internally in the control. The synchronizing motion must otherwise be initiated via the PLC interface (IS “Start gantry synchronization process”). Value = 0 Setting MD: GANTRY_POS_TOL_WARNING to 0 deactivates the monitoring for violation of the warning limit. Gantry synchronization is not initiated internally in the control.				
MD irrelevant for	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 MD 37120: GANTRY_POS_TOL_ERROR MD 37130: GANTRY_POS_TOL_REF IS “Gantry warning limit exceeded” IS “Start gantry synchronization”			Gantry axis definition Gantry trip limit Gantry trip limit for referencing (DB31, ... ; DBX101.3) (DB31, ... ; DBX29.4)	

37120 MD number	**GANTRY_POS_TOL_ERROR** Gantry trip limit				
Default setting: 0.0		Minimum input limit: 0		Maximum input limit: plus	
Changes effective after POWER ON			Protection level: 2/4		Unit: Linear axis: mm Rotary axis: degrees
Data type: DOUBLE				Applies as of SW: 2.1	

37120 MD number	**GANTRY_POS_TOL_ERROR** Gantry trip limit	
Meaning:	With gantry axes, the difference between the position actual values of the leading and synchronized axes are continuously monitored. MD: GANTRY_POS_TOL_ERROR is the maximum permissible position actual value deviation between the synchronized axis and the leading axis for the gantry axis grouping. 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. If 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. In addition, IS “Gantry trip limit exceeded” to the PLC is set to “1”.	
MD irrelevant for	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 MD 37110: GANTRY_POS_TOL_WARNING MD 37130: GANTRY_POS_TOL_REF IS “Gantry grouping is synchronized” IS “Gantry trip limit exceeded”	Gantry axis definition Gantry warning limit Gantry trip limit for referencing (DB31, ... ; DBX101.5) (DB31, ... ; DBX101.2)

37130 MD number	**GANTRY_POS_TOL_REF** Gantry trip limit for referencing	
Default setting: 0.0	Minimum input limit: 0	Maximum input limit: plus
Changes effective after POWER ON	Protection level: 2/4	Unit: Linear axis: mm Rotary axis: degrees
Data type: DOUBLE	Applies from SW 2.1	
Meaning:	On gantry axes, the difference between the position actual values of the leading and synchronized axes are continuously monitored. MD: GANTRY_POS_TOL_REF is the maximum permissible position actual value deviation between the synchronized axis and the leading axis that is monitored if the gantry axis grouping is **not yet synchronized** (IS “Gantry grouping is synchronized” = “0”). If 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. In addition, IS “Gantry trip limit exceeded” to the PLC is set to “1”.	
MD irrelevant for	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 MD 37110: GANTRY_POS_TOL_WARNING MD 37120: GANTRY_POS_TOL_ERROR IS “Gantry grouping is synchronized” IS “Gantry trip limit exceeded”	Gantry axis definition Gantry warning limit Gantry trip limit (DB31, ... ; DBX101.5) (DB31, ... ; DBX101.2)

37140 MD number	**GANTRY_BREAK_UP** Invalidate gantry axis grouping	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 1
Changes effective after Reset	Protection level: 2/4	Unit: –
Data type: BOOLEAN	Applies as of SW: 2.1	

37140 MD number	**GANTRY_BREAK_UP** Invalidate gantry axis grouping
Meaning:	**GANTRY_BREAK_UP = “0”** The forced coupling of the gantry axis grouping remains valid. Monitoring for violation of the gantry warning/trip limit is active! **GANTRY_BREAK_UP = “1”** This cancels the **forced coupling of the gantry grouping**! All gantry axes in this grouping can therefore 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”. **Caution:** **If the gantry axes are mechanically coupled, the machine may be damaged if the leading or synchronized axis is traversed in this mode!** **The gantry axes cannot be referenced individually.**
MD irrelevant for	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)

37150 MD number	**GANTRY_FUNCTION_MASK** Gantry functions	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 1
Changes effective after RESET	Protection level: 2 / 7	Unit: –
Data type: DWORD	Applies as of SW: 7.2	
Meaning:	This MD is used to set special gantry functions. The MD is bitcoded. The following bits are assigned: Bit 0 = 0: Extended monitoring of actual value difference is not active. An offset between the master and slave axis resulting from follow-up or BREAK_UP is not considered when monitoring the actual value difference. No output of alarm 10657 if alarm 10563 before Power Off. Bit 0 = 1: Extended monitoring of actual value difference is active. An offset between the master and slave axis resulting from follow-up or BREAK_UP is not considered when monitoring the actual value difference. Prerequisite: The gantry grouping must be referenced or synchronizedl once after the control is powered up. Output of alarm 10657 if alarm 10563 before Power Off.	

■

Signal Descriptions 5

5.1 Signals to axis/spindle

DB31, ... ; DBX29.4 Data Block	Start gantry synchronization Signal(s) to NC (PLC → NC)	
Edge evaluation: No	Signal(s) updated: Cyclic	Signal(s) valid from SW: 2.1
Signal state 1 or signal transition 0 ——> 1	Request from PLC user program to synchronize the leading 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). Synchronization of the gantry axes can be started only under the following conditions: • Machine function REF must be active (IS "Active machine function REF" = "1") • IS "Gantry grouping is synchronized" = "0" • IS "Gantry synchronization ready to start" = "1" • 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"). If the IS is set continuously to "1", the gantry synchronization run would be started automatically as soon as the above conditions are fulfilled.	
Signal irrelevant for	Gantry synchronized axis	
Application example(s)	If the deviation between the position actual values and the reference position is greater than the gantry warning threshold after referencing of the gantry axes, automatic gantry synchronization is not started and IS "Gantry synchronization ready to start" is set to "1". Synchronization of the gantry axes can be started by the user or the PLC user program with IS "Start gantry synchronization".	
Related to	IS "Gantry grouping is synchronized" IS "Gantry synchronization ready to start" IS "Active machine function REF" IS "Referencing active"	(DB31, ... ; DBX101.5) (DB31, ... ; DBX101.4) (DB11, DBX5.2) (DB21-30, DBX33.0)

DB31, ... ; **DBX29.5** Data Block	**Start automatic synchronization** Signal(s) to NC (PLC → NC)	
Edge evaluation: No		Signal(s) valid from SW: 5
Signal state 1 or signal transition 0 ––> 1	No automatic synchronization	
Signal state 0 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 disabled by sending a VDI signal to the axial PLC –> NC interface of the master axis. This always makes sense when the axes are not activated by default. In this case, the synchronization process should also be started explicitly.	

5.2 Signals from axis/spindle

DB31, ... ; **DBX101.2** Data Block	**Gantry trip limit exceeded** Signal(s) to PLC (NC → PLC)
Edge evaluation: No	Signal(s) updated: Cyclic; Signal(s) valid from SW: 2.1
Signal state 1 or signal transition 0 ——> 1	The difference between the position actual values of the leading and synchronized axes has exceeded the maximum permissible limit value. The axes in the gantry grouping are shut down internally in the control. Alarm 10653 “Error limit exceeded” is also output. The monitored limit value is derived from the following machine data: • MD 37120: GANTRY_POS_TOL_ERROR if gantry grouping is synchronized· • MD 37120: GANTRY_POS_TOL_REF, if gantry grouping is not yet synchronized. **Note:** IS “Gantry trip limit exceeded” is output to the PLC via the PLC interface of the synchronized axis.
Signal state 0 or signal transition 1 ——> 0	The difference between the position actual values of the leading and synchronized axes is still within the permissible tolerance range.
Signal irrelevant for	Gantry leading axis
Related to	MD 37120: GANTRY_POS_TOL_ERROR Gantry trip limit MD 37130: GANTRY_POS_TOL_REF Gantry trip limit for referencing IS “Gantry grouping is synchronized” (DB31, **... ;** DBX101.5)

DB31, ... ; **DBX101.3** Data Block	**Gantry warning limit exceeded** Signal(s) to PLC (NC → PLC)
Edge evaluation: No	Signal(s) updated: Cyclic; Signal(s) valid from SW: 2.1
Signal state 1 or signal transition 0 ——> 1	The difference in the position actual values of the leading and synchronized axes has exceeded the limit value defined with MD 37110: GANTRY_POS_TOL_WARNING. The message “Warning limit exceeded” is also output. **Note:** IS “Gantry warning limit exceeded” is output to the PLC via the PLC interface of the synchronized axis.
Signal state 0 or signal transition 1 ——> 0	The difference between the position actual values of the leading and synchronized axes is less than the limit value defined with MD 37110: GANTRY_POS_TOL_WARNING.
Signal irrelevant for	Gantry leading axis
Application example(s)	When the gantry warning limit is exceeded, the necessary measures (e.g. program interruption at block end) can be initiated by the PLC user program.
Special cases, errors, ...	Setting MD 37110: GANTRY_POS_TOL_WARNING to zero deactivates monitoring of the warning limit.
Related to	MD 37110: GANTRY_POS_TOL_WARNING Gantry warning limit

DB31, ... ; **DBX101.4** Data Block	**Gantry synchronization ready to start** Signal(s) to PLC (NC → PLC)
Edge evaluation: No	Signal(s) updated: Cyclic; 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 leading 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. The compensatory motion must be started by the user or the PLC user program (IS “Start gantry synchronization”). The signal is processed for the gantry leading axis only.

DB31, ... ; **DBX101.4** Data Block	**Gantry synchronization ready to start** 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 IS “Start gantry synchronization” IS “Referenced/synchronized 1/2”	Gantry warning limit (DB31, **...** ; DBX29.4) (DB31, **...** ; DBX60.4 and 60.5)

DB31, ... ; **DBX101.5** Data Block	**Gantry grouping is synchronized** Signal(s) to PLC (NC → PLC)	
Edge evaluation: No	Signal(s) updated: Cyclic	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 leading and synchronized axes (e.g. after start-up of the machine) is eliminated by gantry axis synchronization (see Section 2.3). The synchronization process is initiated either automatically once the gantry axes have been referenced or via the PLC user program (IS “Start gantry synchronization”). The compensation values for temperature and sag do not become effective internally in the control until the gantry grouping is synchronized. **Note:** IS “Gantry grouping is synchronized” is output to the PLC via the PLC interface of the leading axis.	
Signal state 0 or 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 leading and synchronized axes may not be ideally aligned (e.g. gantry misalignment). Workpiece machining with a non-synchronized gantry axis grouping will result in impaired machining accuracy or mechanical damage to the machine. The gantry grouping becomes desynchronized if · • The gantry axes were in “Follow-up” mode • The reference position of a gantry axis is no longer valid or the axis is referenced again (IS “Referenced/Synchronized”). • The gantry grouping has been invalidated (via 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” IS “Referenced/synchronized 1 / 2” MD 37140: GANTRY_BREAK_UP	(DB31, **...** ; DBX29.4) (DB31, **...** ; DBX60.4 and 60.5) Invalidate gantry axis grouping

DB31, ... ; **DBX101.6** Data Block	**Gantry leading axis** Signal(s) to PLC (NC → PLC)	
Edge evaluation: No	Signal(s) updated: Cyclic	Signal(s) valid from SW: 2.1
Signal state 1 or signal transition 0 ––> 1	The axis is defined as the **leading axis** within a gantry axis grouping (see MD 37100: GANTRY_AXIS_TYPE). **Note:** The following interface signals are evaluated or output to the PLC via the PLC interface of the gantry leading axis: • IS “Start gantry synchronization” (DB31, **...** ; DBX29.4) • IS “Gantry grouping is synchronized” (DB31, **...** ; DBX101.5)	

DB31, ... ; **DBX101.6** Data Block	**Gantry leading axis** 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). It is not possible to traverse a synchronized axis directly by hand (in JOG mode) or to program it in a part program. **Note:** The following interface signals are output to the PLC via the PLC interface of the gantry synchronized axis: • IS "Gantry warning limit exceeded" (DB31, **...** ; DBX101.3)· • IS "Gantry trip limit exceeded" (DB31, **...** ; DBX101.2) 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 IS "Gantry axis" (DB31, **...** ; DBX101.7)

DB31, ... ; **DBX101.7** Data Block	**Gantry axis** Signal(s) to PLC (NC → PLC)	
Edge evaluation: No	Signal(s) updated: Cyclic	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). The PLC user program can read IS "Gantry leading axis" to detect whether the axis has been declared as a leading or synchronized axis.	
Signal state 0 or 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 IS "Gantry leading axis" (DB31, **...** ; DBX101.6)	

■

Notes

6 Example

6.1 Creating a gantry grouping

Introduction

The gantry grouping, the referencing of its axes, the orientation of possible offsets and, finally, the synchronization of the axes involved are complicated procedures. The individual steps involved in the process are explained below by an example constellation.

Constellation

Machine axis 1 = Gantry leading axis	Incremental measuring system
Machine axis 3 = Gantry synchronized axis	Incremental measuring system

Machine data

The following machine data 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. 1 mm
MD 37130: GANTRY_POS_TOL_REF	= e.g. 100 mm (max. misalignment)
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. 1 mm
MD 37130: GANTRY_POS_TOL_REF	= e.g. 100 mm (max. misalignment)
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	= corresponds to max. traversing distance
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.

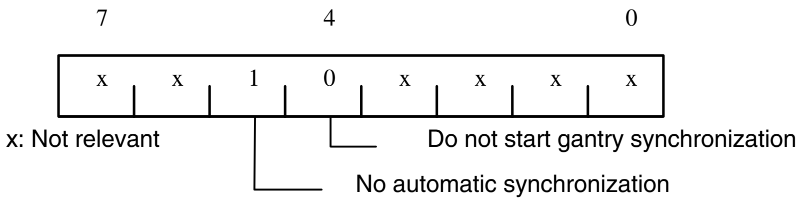
6.2 Setting of NCK PLC interface

Introduction

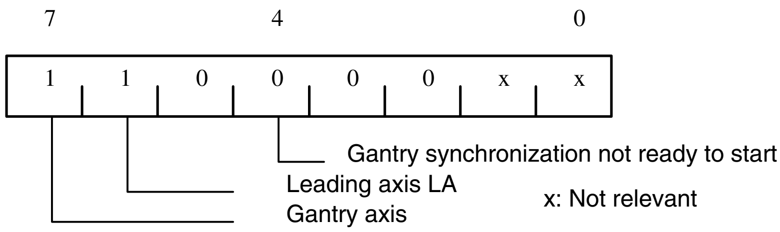
An automatic synchronization process during axis referencing must be disabled initially so as to prevent any damage to grouping axes that are misaligned.

Disabling of automatic synchronization

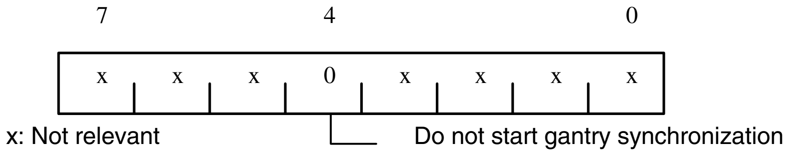
The PLC user program sets:
DB31, ... ; DBX 29.4 = 0
DB31, ... ; DBX 29.5 = 1
for the axis data block of axis 1



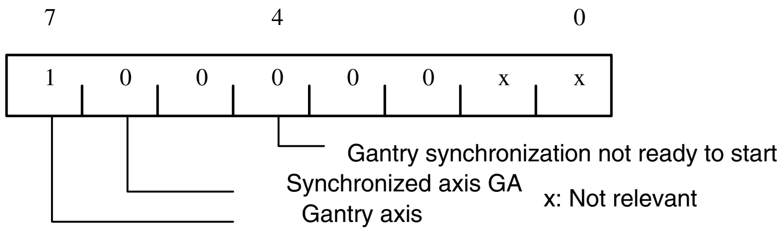
The NCK sets the following as a confirmation in the axis block of axis 1:
DB31, ... ; DBB101:



The PLC user program 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:



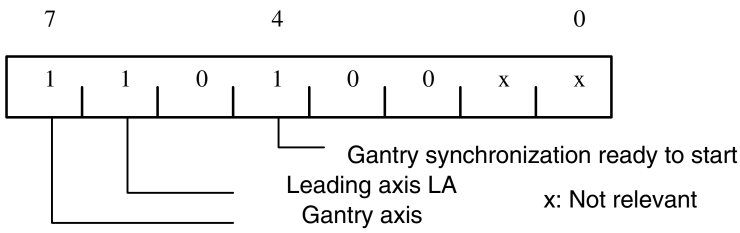
6.3 Commencing start-up

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 at the IS (DB31, ... ; DBB101 with:



4. RESET
5. Value read-off 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

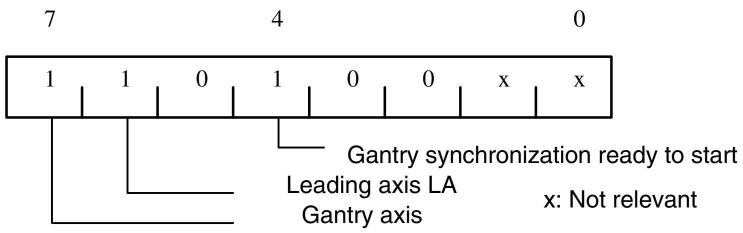
Note

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 at the IS (DB31, ... ; DBB101 with:



9. Examine actual positions of machine. Case A or B might apply:

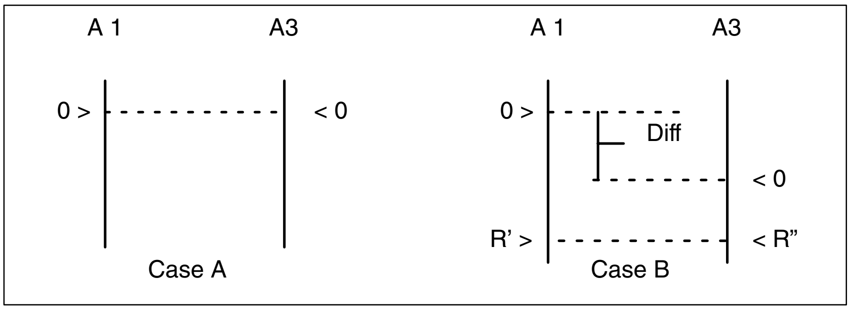


Fig. 6-1 Possible results after referencing of axis 1 (master axis)

If Case A applies, the synchronization process can be started immediately. See Step 10.
In case B, the offset “diff.” must be calculated and taken into account:
a) Measuring of diff
b) The position difference can be tracked back in JOG using two reference points R’ und R”set at right angles in the machine base.
The offset “diff” can then be read as a difference for the position display.
The 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. PLC sets:
DB31, ... ; DBX 29.4= 1

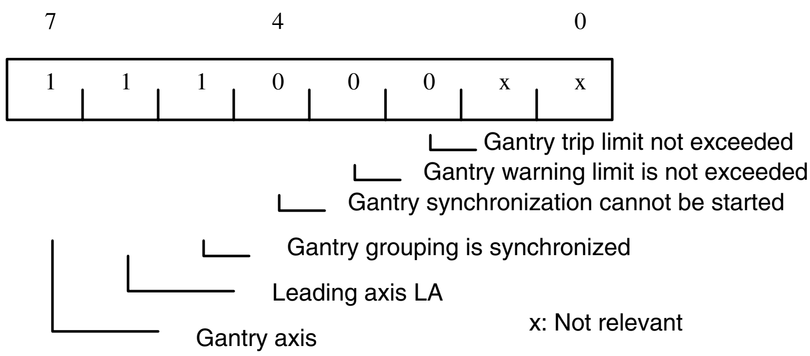
6.4 Setting warning and trip limits

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 initially for all axes.
– Assign a very small value to MD 37110: GANTRY_POS_TOL_WARNING.
Now, if you expose the axes to high-dynamic loads, the self-clearing alarm:
“10652 Channel %1 axis %2 Gantry warning limit exceeded”
should be output repeatedly.

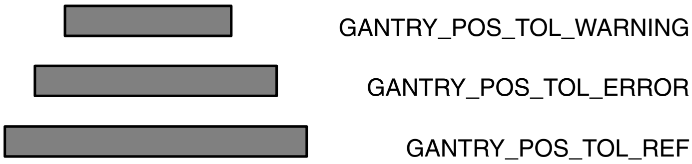
– Now increase the setting in MD 37110: GANTRY_POS_TOL_WARNING until the alarm no longer appears. The interface indicates the status specified below. (This must happen in a window suitable for production.)
If monitoring is still only very sporadic, an edge flag can be programmed in the user PLC program.



– 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 the value of GANTRY_POS_TOL_ERROR or GANTRY_POS_TOL_REF is smaller than GANTRY_POS_TOL_WARNING, this monitoring is not active!
Set GANTRY_POS_TOL_WARNING to 0 to deactivate warning limit monitoring.
The maximum possible LEC value must be set in GANTRY_POS_TOL_REF.

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).

---

7 Data Fields, Lists

7.1 Interface signals

DB number	Bit, byte	Name	Refer-ence
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	
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 leading axis	
31, ... ;	101.7	Gantry axis	

7.2 Machine data

Number	Identifier	Name	Refer-ence
Axis/channelspecific($MA_ ...)			
30300	IS_ROT_AX	Rotary axis	R2
32200	POSCTRL_GAIN	Servo gain factor (Kv)	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	Initial setting for axial jerk limitation	B2
32430	JOG_AND_POS_MAX_JERK	Axial jerk	B2
32610	VELO_FFW_WEIGHT	Feedforward control factor for speed feedfor. 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 feed-forward 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 positioning 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	Dist.-coded linear meas. 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	Max. 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

Detailed explanations of the alarms, which may occur, appear in

References: /DA/, Diagnostics Guide

or in the Online help.

■

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

Cycle Times (G3)

■

Notes

Brief Description 1

This description explains the relationships and machine data of the various system cycles of the NC:

- System basic cycle
- Interpolator cycle
- Position control cycle

810D and 840D

For SINUMERIK 840D and SINUMERIK 810D, the position control cycle and the interpolator cycle (IPO cycle) are derived from the system basic cycle, which is set in the machine data of the NC.

840Di

For SINUMERIK 840Di, the position control cycle and the interpolator cycle are derived from the system basic cycle. The system basic cycle is not set via the machine data of the NC, but set the same as the isochronous DP cycle T_{DP} set in the S7 project during the creation of the configuration.

■

Notes

Detailed Description 2

2.1 General information about cycle times

Required

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.

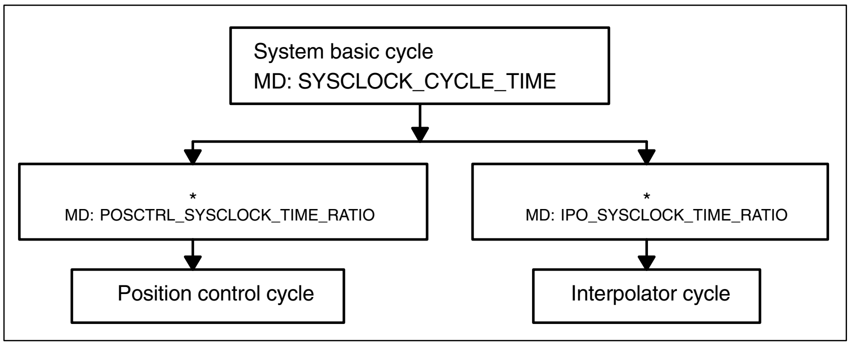


Fig. 2-1 Derivation of cycle times

Note

MD 10050: SYSCLOCK_CYCLE_TIME
The system clock cycle is entered in seconds.

MD 10060: POSCTRL_SYSCLOCK_TIME_RATIO
The position control cycle is entered as a factor of the system clock cycle.

MD 10070: IPO_SYSCLOCK_TIME_RATIO
The interpolator cycle is entered as a factor of the system clock cycle.

Default values for cycle times

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 default cycle times are as follows:

Clock	810D CCU	840D NCU 571	840D NCU 572	840D NCU 573	Setting via MD
System basic cycle	2.5 ms	6 ms	4 ms	4* / 8# ms	SYSCLOCK_CYCLE_TIME
Position control cycle	2.5 ms	6 ms	4 ms	4* / 8# ms	POSCTRL_SYSCLOCK_TIME_RATIO
Interpolator cycle	10 ms	18 ms	12 ms	12* / 40# ms	IPO_SYSCLOCK_TIME_RATIO

* with 2 channels and 12 axes
\# with > 2 channels

General example for cycle settings

The machine data assignments for cycle times are as follows:

If MD ... = ...	then ... = ...
SYSCLOCK_CYCLE_TIME = 0.002	System clock cycle = 2 ms
POSCTRL_SYSCLOCK_TIME_RATIO = 1	Position-control cycle = 2 ms (1 2 ms)
IPO_SYSCLOCK_TIME_RATIO = 3	Interpolator cycle = 6 ms (3 2 ms)

2.2 SINUMERIK 810D and 840D

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 to a value greater than 1, the position control cycle corresponds to the basic system clock cycle.

The ratio of interpolator to position-control cycle must be an integer value and greater than or equal to 1. If it is not, it will be corrected automatically and an alarm will appear.

Alarm: “4102 IPO cycle increased to [ ] ms”

$$\frac{\text{Interpolator cycle}}{\text{Position control cycle}} \geq 1$$

2.3 SINUMERIK 840Di with PROFIBUS DP

For more information about SINUMERIK 840Di, see:

References:
/HBI/ SINUMERIK 840Di Manual, PROFIBUS-DP Communication

2.3.1 Description of a DP cycle

Actual values

At time T_I, the current actual values are read from all isochronous drives (DP slaves). In the next DP cycle, the actual values are transferred to the DP master in the time T_{DX}.

Position 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.

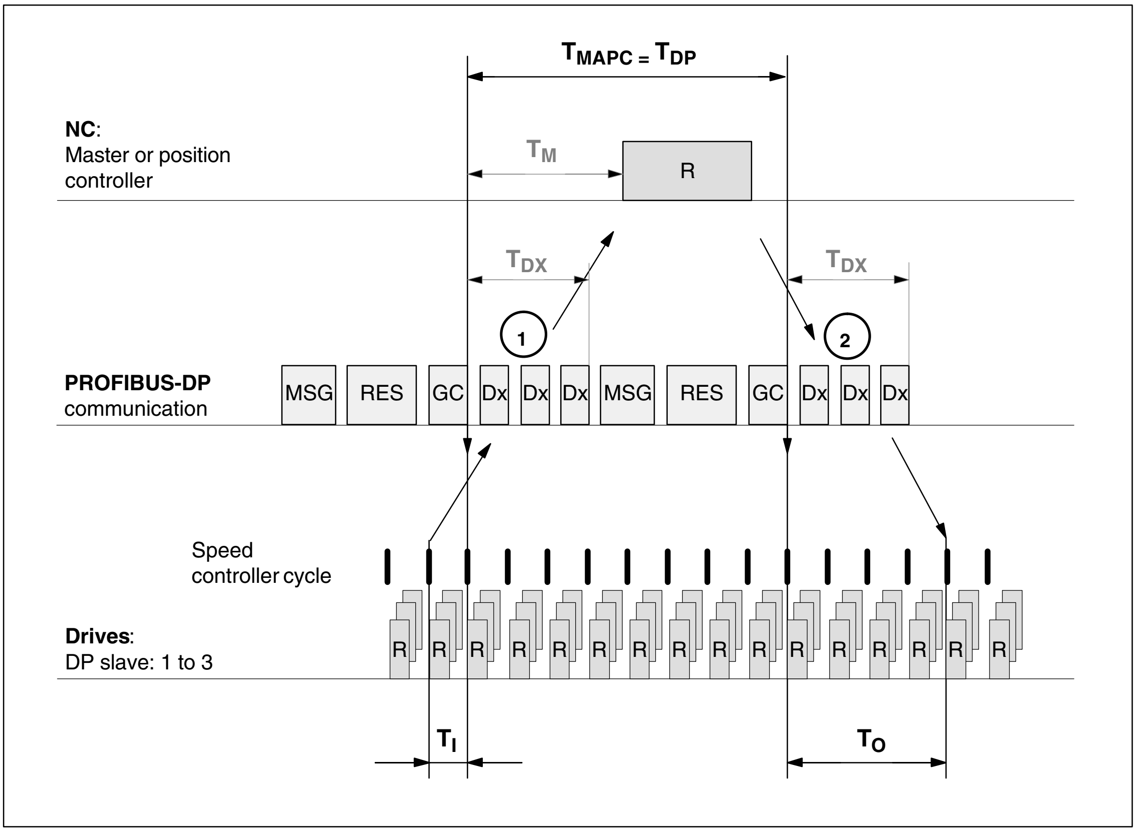


Fig. 2-2 Optimized DP cycle with 3 DP slave with a SIMODRIVE 611 universal

Key to Fig. 2-2:

T_{MAPC}	Master application cycle: NC position control cycle 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 time for all DP slaves
T_M	Master time: Offset of start time for NC position control
T_I	Input time: Time of actual-value acquisition
T_O	Output time: Time of setpoint transfer
GC	Global control message frame (broadcast message frame) for cyclic synchronization of the equidistance between the DP master and DP slaves
R	Speed or position controller computing time
Dx	Useful data exchange between the DP master and DP slaves
MSG	Acyclic services (e.g. DP/V1, pass token)
RES	Reserve: “active break” until the equidistant cycle has elapsed
①	The actual values for the current DP cycle / 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 basic 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 cycle)

Position control cycle

The position control cycle is set to the fixed ratio 1:1 with respect to the system basic cycle. It cannot be changed.

- MD10061: POSCTRL_CYCLE_TIME (position-control cycle)

Interpolation cycle

The interpolator cycle can be freely selected as an integer multiple of the position control cycle.

- MD10070: IPO_SYSCLOCK_TIME_RATIO (factor for interpolator cycle)

Position control cycle offset

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)

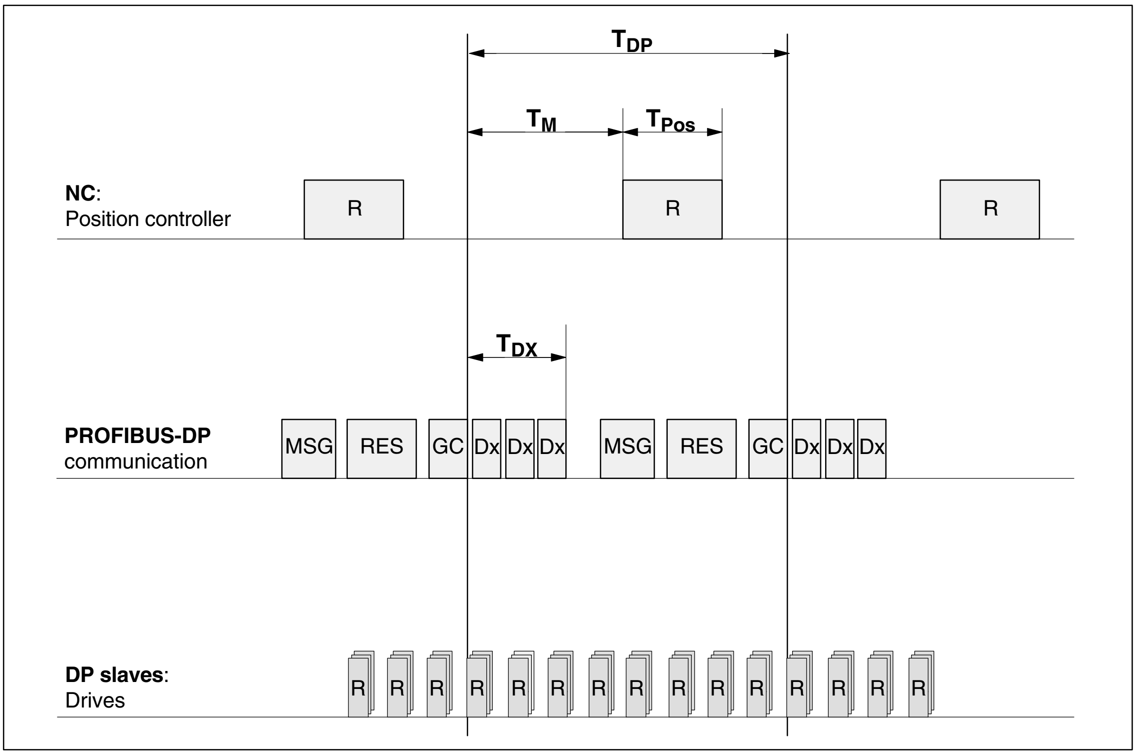


Fig. 2-3 Position control cycle offset compared to PROFIBUS DP cycle

Key to Fig. 2-3:

T_{Pos}	CPU time required by position controller
T_{DP}	DP cycle time: DP cycle time
T_{DX}	Data exchange time: Total transfer time for all DP slaves
T_M	Master time: Offset of the start time for NC position control
GC	Global control: Broadcast message for cyclic synchronization of the equidistance between DP master and DP slaves
R	CPU time
Dx	Useful data exchange between the DP master and DP slaves
MSG	Acyclic services (e.g. DP/V1, pass token)
RES	Reserve: “active break” until the equidistant cycle has elapsed

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 so that the following conditions are satisfied within a PROFIBUS-DP/system cycle:

– 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 position controller maximum time is displayed in the **NC Load** dialog box 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 selected is too small. 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 startup

- In this machine data, alarm requests on the PROFIBUS level are stored even after reboot.

  If a conflict occurs during startup 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 startup. These alarm requests are stored here.

Special features

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 the position control cycle is set permanently to 1 and cannot be changed.

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 control quality for the drive and the better the surface quality on the workpiece.

■

Notes

3 Supplementary Conditions

None

4 Data Descriptions (MD, SD)

4.1 General machine data

10050 MD number	**SYSCLOCK_CYCLE_TIME** System clock cycle	
Default setting: 0.004 for the 840D Default setting: 0.002 for the 840Di	Minimum input limit: 0.000125	Max. input limit: 0.032 for the 840D Max. input limit: 0.008 for the 840Di
Changes effective after POWER ON	Protection level: 2 / 7	Unit: s
Data type: DOUBLE		Applies as of SW: 1.1 SW 6.4 and higher PROFI-BUS
Meaning:	Clock cycle of the system software, i.e. the unit of time after which time interrupts are triggered in the operating system. The interpolator and position control cycle/IPO settings are multiples of this system clock cycle. The system clock cycle corresponds to the position control cycle if MD 10060: POSCTRL_SYSCLOCK_TIME_RATIO is not set higher than 1. If a digital drive is used, the system basic clock cycle and MD 10060: POSCTRL_SYSCLOCK_TIME_RATIO must be set so that the position control cycle is no longer than 16 ms (otherwise a drive alarm is issued). Automatic offsets during start-up can cause a shift in the value set for MD 10050: SYSCLOCK_CYCLE_TIME. On systems with PROFIBUS-DP, MD 10050: SYSCLOCK_CYCLE_TIME: corresponds to the PROFIBUS-DP cycle time. The PROFIBUS-SDB reads this time from the configuration file during start-up and writes it to this machine data. This machine data can only be modified via the configuration file.	
Special cases, errors, ...	**Note:** 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 startup, the entered value is automatically rounded to a multiple of this incrementing. Details: After a POWER OFF/ON, discrete timer division ratios may produce a value that is not an integer of the input value. e.g.: Entry = 0.005s after POWER OFF/ON = 0.00499840 or Entry = 0.006s after POWER OFF/ON = 0.0060032	
Related to	MD 10060: POSCTRL_SYSCLOCK_TIME_RATIO, MD 10080: SYSCLOCK_SAMPL_TIME_RATIO	

10059 MD number	**PROFIBUS_ALARM_MARKER** Profibus alarm marker (internal only)				
Profibus adpt.: 0.0		Minimum input limit: –		Maximum input limit: –	
Changes effective after POWER ON			Protection level: 0 / 0		Unit: –
Data type: BYTE				Applies as of SW: 5.2	
Meaning:	PROFIBUS alarm flag: Alarm requests at PROFIBUS level are stored in this machine data even after a reboot. If a conflict is found between machine data 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 a corresponding alarm is set during the next start-up. These alarm requests are stored here.				
Related to	MD 10050: SYSCLOCK_CYCLE_TIME, MD 10060 POSCTRL_SYSCLOCK_TIME_RATIO				

10060 MD number	**POSCTRL_SYSCLOCK_TIME_RATIO** Factor for position-control cycle				
Default setting: 1		Minimum input limit: 1		Maximum input limit: 31 for the 840D Max. input limit: 1 for the 840Di	
Changes effective after POWER ON			Protection level: 2 / 7		Unit: Factor × MD 10050
Data type: DWORD				Applies as of 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. The **normal setting is 1**. The position-control cycle then corresponds to the system clock cycle. 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. If a digital drive is being used, the value set for the position-control cycle can change because of an automatic correction on startup. 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. 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 MD number	**POSCTRL_CYCLE_TIME** Position control cycle				
Default setting: 0.0		Minimum input limit: –		Maximum input limit: –	
Changes effective after POWER ON			Protection level: 0 / 7		Unit: –
Data type: DOUBLE				Applies as of SW: 5	
Meaning:	Position-control cycle Display of the position-control cycle time (cannot be modified !). 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 MD number	**POSCTRL_CYCLE_DELAY** Position-control cycle offset				
Default setting: 0.003 for the 840D Default setting: 0.0015 for the 840Di		Minimum input limit: 0.000 Profibus adpt.: 0.000		Max. input limit: 0.008 for the 840D Max. input limit: 0.008 for the 840Di	
Changes effective after POWER ON			Protection level: 2 / 7		Unit: s
Data type: DOUBLE				Applies as of 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 MD number	**IPO_SYSCLOCK_TIME_RATIO** Factor for interpolator cycle	
Default setting: 4 for 840Di 3 for 840D 2-channel 5 for 840D 10-channel	Minimum input limit: 1	Maximum input limit: 100
Changes effective after POWER ON	Protection level: 2 / 7	Unit: Factor × MD 10050
Data type: DWORD	Applies as of 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. Only integer 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 rounded up to the next integer multiple of the position-control cycle before they become active (next POWER ON). This is accompanied by the alarm 4102 “IPO cycle increased to [ ] ms”. 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 MD number	**IPO_CYCLE_TIME** Interpolator cycle	
Default setting: 0.0	Minimum input limit: ***	Maximum input limit: ***
Changes effective after POWER ON	Protection level: 0 / 7	Unit: s
Data type: DOUBLE	Applies as of SW: 5	
Meaning:	Display of the interpolator cycle time (cannot be modified!). Generated 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	

<table>
<tr><td>10080
MD number</td><td colspan="4">SYSCLOCK_SAMPL_TIME_RATIO
Division factor of position-control cycle for actual-value acquisition</td></tr>
<tr><td colspan="2">Default setting: –
No meaning for 840Di
4 for 840D 2-channel
5 for 840D 10-channel</td><td colspan="2">Minimum input limit: 1</td><td>Maximum input limit: 31</td></tr>
<tr><td colspan="2">Changes effective after POWER ON</td><td colspan="2">Protection level: 0 / 0</td><td>Unit: –</td></tr>
<tr><td colspan="3">Data type: DWORD</td><td colspan="2">Applies as of SW: SW1, SW 6.4 and higher PROFIBUS</td></tr>
<tr><td>Meaning:</td><td colspan="4">This machine data sets the division factor of a cycle divider, which is hardware located between the measured value sampler and the interrupt controller.

MD 10080: SYSCLOCK_SAMPL_TIME_RATIO
may only be set to values > 1
subject to observance of the conditions described in the special cases.

Produce a larger increment, in which the time for the system clock cycle can also be set (see MD 10050: SYSCLOCK_CYCLE_TIME).

If a digital drive controller and systems with PROFIBUS-DP connection are used, the division factor is set automatically (see special cases).</td></tr>
<tr><td>Special cases, errors, ...</td><td colspan="4">1. Using an analog speed interface
When using this conventional drive interface, the division can be set according to the following criteria:
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.

If MD 10080: SYSCLOCK_SAMPL_TIME_RATIO is set to values > 1:
The delay time of the position control output can be set to fractions of the position control clock cycle.

The input/output hardware can be triggered more than once during a position controller cycle.
However, this is not recommended due to the effect on the interrupt load of the operating system.

2. Using a digital drive controller
If a digital drive controller is used, the division factor is set automatically.

The scan cycle time is set as
a multiple (100%, 200% or 300%) of 0.000625 s SINUMERIK 810D
Standard case:
100% of 0.000625 s SINUMERIK 810D new design
100%, 200%, 300% ... 800% of 0.000125s SINUMERIK 840D

The SIMODRIVE 611 digital drive can synchronize its own clock generation to these values.

3. Systems with PROFIBUS DP connection
If systems with PROFIBUS-DP are used, the division factor is set automatically.</td></tr>
<tr><td>Related to ....</td><td colspan="4">MD 10050: SYSCLOCK_CYCLE_TIME, MD 10060 POSCTRL_SYSCLOCK_TIME_RATIO</td></tr>
</table>

11250 MD number	PROFIBUS_SHUTDOWN_TYPE PROFIBUS shutdown handling	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 2
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: BYTE	Applies as of SW: SW 6.3	
Meaning:	Handling the PROFIBUS DP on shutdown of the NCK (NCK reset) Value 0:The bus is deactivated directly from within cyclic mode, without pre-warning Value 1: In the event of an NCK shutdown, the PROFIBUS DP is first set to the CLEAR status for at least 20 clock cycles before it is deactivated. If this is not possible on account of the hardware, the procedure is as for value 2. Value 2: In the event of an NCK shutdown, the PROFIBUS-DP is first set to a status in which all drives are sent a zero as ControlWord1 and ControlWord2 for at least 20 clock cycles (pseudo clear). The bus itself remains operational.	
Related to	–	

4.2 Axis-specific machine data

37600 MD number	PROFIBUS_ACTVAL_LEAD_TIME Actual-value acquisition time (PROFIBUS Ti)	
Default setting: 0.000125	Minimum input limit: 0.0	Maximum input limit: 0.032
Changes effective after POWER ON	Protection level: 0 / 0	Unit: s
Data type: DOUBLE	Applies as of SW: 6.1	
Meaning:	Date and setting of the setpoint transfer time (Ti) of the PROFIBUS encoder Unit: Seconds, default is thus 125µs (this is also the default set by Step 7 on a SIMODRIVE 611 universal)	
Special cases, errors, ...	**Caution:** The actual Ti value is read directly from the PROFIBUS configuration or the drive if possible. 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 MD number	PROFIBUS_OUTVAL_DELAY_TIME Setpoint delay time (PROFIBUS To)	
Default setting: 0.003	Minimum input limit: 0.0	Maximum input limit: 0.032
Changes effective after POWER ON	Protection level: 0 / 0	Unit: s
Data type: DOUBLE	Applies as of SW: 6.3	
Meaning:	Date and setting of the setpoint transfer time (To) of the PROFIBUS encoder Unit: Seconds, default is thus 3ms	
Special cases, errors, ...	**Caution:** The actual To value is read directly from the PROFIBUS configuration or the drive if possible. 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	–	

■

Notes

5 Signal Descriptions

None

6 Example

None

Notes

Data Fields, Lists 7

7.1 Machine data

Number	Identifier	Name	Refer-ence
General ($MN_ ...)			
10050	SYSCLOCK_CYCLE_TIME	System basic cycle	
10059	PPOFIBUS_ALARM_MARKER	PTOFIBUS alarm marker (internal only)	
10060	POSCTRL_SYSCLOCK_TIME_RATIO	Factor for position-control cycle	
10061	POSCTRL_CYCLE_TIME	Position-control cycle (SW 5 and higher)	
10062	POSCTRL_CYCLE_DELAY	Position-control cycle offset (SW 5 and higher)	
10070	IPO_SYSCLOCK_TIME_RATIO	Factor for interpolator cycle	
10071	IPO_CYCLE_TIME	Interpolation cycle (SW 5 and higher)	
10080	SYSCLOCK_SAMPL_TIME_RATIO	Division factor of the position control cycle for actual value acquisition	
11250	PROFIBUS_SHUTDOWN_TYPE	PROFIBUS DP shutdown handling (SW 6.3 and higher)	
Axis/channelspecific ($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

Detailed explanations of the alarms, which may occur, appear in

References: /DA/, Diagnostics Guide

or in the Online help.

■

Notes

SINUMERIK 840D sl/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

- Along a braking ramp or
- With speed setpoint zero and follow-up mode.

Application

The function can be used for 2D and 3D paths. On 2D paths, the monitoring area is defined using lines in parallel with the programmed path.
The monitoring area is defined using 2 or 3 **geometry axes**.

Other axes

Monitoring of synchronized 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.

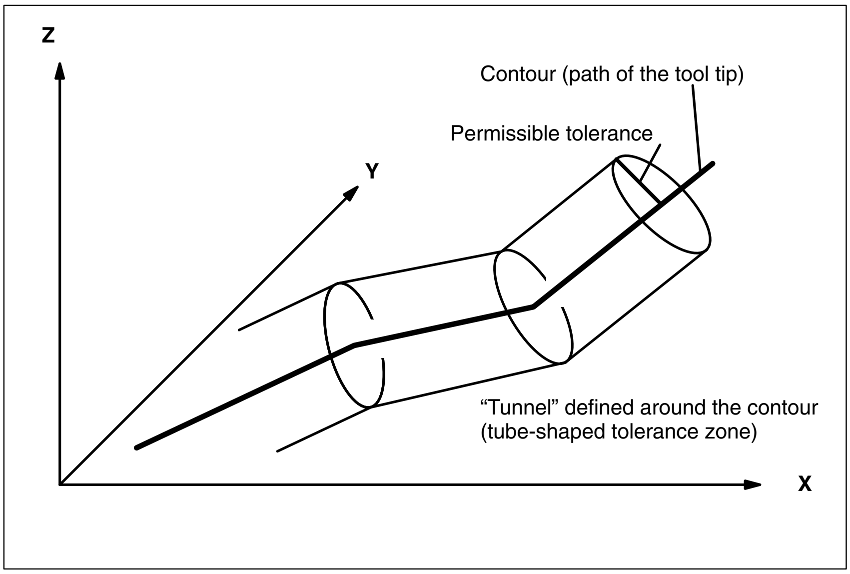


Fig. 1-1 Position of the contour tunnel around the programmed path

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, motion continues in the normal way. If the calculated actual position violates the tunnel, an alarm is triggered (in the default setting) and the axes are stopped with a ramp. This response to the violation of the tunnel can be disabled (alarm triggered but movement continued) or intensified (rapid stop) by means of a machine data setting.

Analysis

The calculated distance between the programmed path and the actual values can be routed to an analog output to analyze the progression of the contour errors during normal operation (quality control).

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 SW 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. It is specified in MD21050: $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.

Parameterizable deceleration behavior

The deceleration behavior for the monitoring response can be set via the following MD: MD21060: $MC_CONTOUR_TUNNEL_REACTION.

0: Display alarm and continue machining

1: Deceleration according to the deceleration ramps (default setting)

2: Rapid stop (speed setpoint = 0)

Encoder switchover

Switching between two encoder systems usually causes a sudden change in the actual position of the tool tip. This change resulting from encoder switchover must not be so large as to cause the tool tip to violate the monitoring tunnel. The radius 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

The value of MD: $MC_CONTOUR_ASSIGN_FASTOUT can be set to define whether the deviation values of the actual value of the tool tip from the programmed path are to be routed to a fast analog output and, if so, to which one (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

Scale:
The tunnel radius stored in MD: $MC_CONTOUR_TUNNEL_TOL corresponds to a voltage of 10 V 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 “Programmable contour accuracy” function permits the user to specify a maximum error for the contour in the NC program, which may not be exceeded. 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 ensures that the velocity necessary for maintaining the required contour accuracy is not exceeded at any point along the path.

Application

The function ensures 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 machine data MC_CPREC_WITH_FFW is set to 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 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 G code group 39 in 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

Parameterizable deceleration behavior

The deceleration behavior for the monitoring response is parameterized via the following MD:

MD21060: $MC_CONTOUR_TUNNEL_REACTION = <deceleration behavior>

With deceleration behavior = 1 and deceleration behavior = 2, deceleration is always performed with rapid stop (speed setpoint = 0).

Coupled motion

If coupled motion between two geometry axes is programmed with contour tunnel monitoring, this always results in activation of the contour tunnel monitoring.

In this case, the contour tunnel monitoring must be switched off before programming the coupled motion:
MD21050: $MC_CONTOUR_TUNNEL_TOL = 0.0

Data Descriptions (MD, SD) 4

4.1 Channelspecific machine data

21050 MD number	**CONTOUR_TUNNEL_TOL** Response threshold for contour tunnel monitoring				
Default setting: 0.0		Minimum input limit: 0.0		Maximum input limit: ***	
Changes effective after NEW_CONF			Protection level: 2/4		Unit: mm
Data type: DOUBLE				Applies as of SW: 2.0	
Meaning:	Response threshold for contour tunnel monitoring Indicates the radius of the tunnel set around the tool tip path. If three geometry axes have been defined, the tunnel may be set up like a tube through the center of which the tool tip path runs. If only two geometry axes have been defined, this tube appears in planographic format at the level 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 MD number	**CONTOUR_TUNNEL_REACTION** Reaction to response of contour tunnel monitoring				
Default setting: 1		Minimum input limit: 0		Maximum input limit: 2	
Changes effective after POWER ON			Protection level: 2/4		Unit: –
Data type: BYTE				Applies as of SW: 2.0	
Meaning:	Reaction to activation of alarm 0: Display alarm only, continue machining 1: Ramp stop 2: Rapid stop				
MD irrelevant for	Contour tunnel monitoring option not available				
Related to	CONTOUR_TUNNEL_TOL, CONTOUR_ASSIGN_FASTOUT				

21070 MD number	**CONTOUR_ASSIGN_FASTOUT** Assignment of an analog output for contour error output				
Default setting: 0		Minimum input limit: 1		Maximum input limit: 8	
Changes effective after POWER ON			Protection level: 2/4		Unit: –
Data type: BYTE				Applies as of SW: 2.0	
Meaning:	Assignment of an analog input to which the contour error calculated can be output. 0: No output 1: Output to output 1 2: Output to output 2, etc. 8: Output to output 8 A fault in the setting for the response threshold $MC_CONTOUR_TUNNEL_TOL appears at the output as a 10 V voltage. Multiple assignment of the same output by other signals is checked automatically.				
MD irrelevant for	Contour tunnel monitoring option not available				
Related to	CONTOUR_TUNNEL_TOL, CONTOUR_TUNNEL_REACTION				

20470 MD number	**CPREC_WITH_FFW** Progr. Contour accuracy				
Default setting: 0		Minimum input limit: 0		Maximum input limit: 1	
Changes effective after POWER ON			Protection level: 2 / 7		Unit: –
Data type: BOOLEAN				Applies as of SW: 3.2	
Meaning:	This machine data is set to define the response of the programmable CPRECON function in conjunction with feedforward control. FALSE: The CPRECON function is inoperative if feedforward control is active simultaneously. TRUE: CPRECON is also operative with feedforward control.				
Related to	$SC_CONTPREC, $SC_MINFEED				

4.2 Channel-specific setting data

42450 MD number	**CONTPREC** Contour accuracy			
Default setting: 0.1		Minimum input limit: 0.000001		Maximum input limit: 999999
Changes effective after POWER ON		Protection level: 7 / 7		Unit: mm
Data type: DOUBLE			Applies as of SW: 3.2	
Meaning:	Contour accuracy. You can use this setting data to specify the accuracy 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 MD number	**MINFEED** Minimum path feed with CPRECON			
Default setting: 1.0		Minimum input limit: 0.000001		Maximum input limit: 999999
Changes effective after POWER ON		Protection level: 7 / 7		Unit: mm/min
Data type: DOUBLE			Applies as of 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

Cutout

```
N10 X0 Y0 G0
N20 CPRECON              ; Activation of the contour accuracy defined using MD
N30 F10000 G1 G64 X100   ; Machining at 10 m/min in continuous-path mode
N40 G3 Y20 J10           ; Automatic limiting of feed in circle block
N50  G1 X0               ; Feed again 10 m/min
...
N100 CPRECOF             ; Deactivation of programmed contour accuracy
N110 G0 ...
```

■

Notes

7 Data Fields, Lists

7.1 Machine Data

Number	Identifier	Name	Reference
Channelspecific ($MC_ ...)			
20470	CPREC_WITH_FFW	Programmed Contour accuracy	
21050	CONTOUR_TUNNEL_TOL	Response threshold for contour tunnel monitoring	
21060	CONTOUR_TUNNEL_REACTION	Reaction to response of contour tunnel monitoring	
21070	CONTOUR_ASSIGN_FASTOUT	Assignment of an analog output for output of the contour error	
Axis/channelspecific ($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 feed with CPRECON	

7.2 Alarms

Detailed explanations of the alarms, which may occur, appear in

References: /DA/, Diagnostics Guide

or in the Online help.

■

Notes

SINUMERIK 840D sl/840D/840Di 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

Due to function extensions for “coupled motion and master value coupling”, the title of the Description of Functions has been changed to:

“Coupled axes and ESR”

ESR = Extended Stop and Retract

1.1 Coupled motion

The function: Coupled Motion (see Section 2.1, Page 3/M3/2-9) enables the definition of simple axis links between a master axis and a slave axis, allowing for a coupling factor.

The function: Coupled Motion has the following features:

– Any axis of the NC can be defined as a master axis.
– Any axis of the NC can be defined as a coupled axis with a specific coupling factor.
– The master axis and coupled-motion axis or axes together form a coupled-axis grouping.
– Any number of coupled-motion axes can be assigned to a master axis.
– Up to 2 master axes can be assigned to a coupled-motion axis.
– A coupled-motion axis can be the master axis of a further coupled-axis grouping.
– Traversing movements of the master axis are executed in synchronism on all slave axes based on the coupling factor.
– Coupled-motion axes can be moved independently of the master axis while the coupling is active (overlaid movements).
– The master and coupled-motion axes of a coupled-axis grouping are defined, and the coupling activated/deactivated, by programming instructions in the part program or by synchronized action.
– Coupled motion is also possible in the manual modes: JOG, JOG REF, JOG INC etc.

1.2 Curve tables

The “curve tables” function (see Section 2.2) can be used to define the complex sequence of motions of an axis in a curve table.

Any axis can be defined as a leading axis and a following axis 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 slave axis are defined and activated/deactivated with the curve table in the NC part program or using a synchronized action.

The curve tables in SRAM remain valid after the end of a part program or power down.

Processing speed can be increased by loading the curve table from SRAM to DRAM when the NC is powered up.

Axis groupings with curve tables must be reactivated independently of the storage location of the curve table after POWER ON.

Linear curve table segments are stored in separate areas to save memory space.

1.3 Master value coupling

The "master value coupling" function (see Section 2.3, Page 3/M3/2-35) can be used to process short programs cyclically with close coupling of the axes to one another and a master value that is either generated internally or input from an external source.

The master value can for example be derived from a conveyor belt or a line shaft.

Either an axis or path master value coupling can be used, and activated and deactivated in the NC part program or using a synchronized action.

The coupling with the master value is defined using a curve table.

1.4 Electronic gear EG

With the "Electronic gear" function (see Section 2.4, Page 3/M3/2-43) it is possible to control the motion of one **leading axis** as the function of up to five **following axes**. The relationship between each leading axis and the following axis is defined by the coupling factor. The following axis motion components derived from the individual leading axis motion components have an additive effect. The coupling can be based on:

– Actual value of the leading axis

– Setpoint of the leading axis

The following functions of a gear grouping can be programmed using part program instructions:

– Defining
– Switching on
– Switching off
– Delete

Curve tables

Non-linear associations can also be implemented using curve tables (Section 2.2, Page 3/M3/2-16).

Electronic gears can be cascaded, i.e. the following axis of an electronic gear can be the leading axis for a further electronic gear.

Synchronous position

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

Application examples:

– Machine tools for gear cutting
– Gear trains for production machines

1.5 Extended stop and retract (ESR)

The “Extended stop and retract” function (ESR) provides a means to react flexibly to selective error sources while preventing damage to the workpiece.

- **Extended stop**
  Where possible, all axes involved in the electronic coupling are brought to a normal standstill.
- **Retract**
  The tool currently in use is retracted from the workpiece as quickly as possible.
- **Generator operation**
  In the event of a power outage, the electrical energy needed for retraction is provided by additional backup capacitors in the DC link or by regenerative braking using a specially provided drive operating in generator mode.

NC-controlled functions

With NC-controlled stop and retract, the coupling remains active for a configurable time, unlike stop and retract under independent drive control.

Straight line as retraction path

A straight line can be programmed as a retraction path, as an alternative to pure axial retraction.

■

Notes

Detailed Description 2

Note

- **External master value axes**

  When using the REPOS or REPOSA part program instructions in conjunction with external master value axes, please make sure that these are released by the channel or switched to a "neutral state" using the RELEASE instruction.

  When attempting to reposition without release of the axis, the message: "Wait: Feed stop active" is displayed and the processing of the part program is not continued.

2.1 Coupled motion

2.1.1 Function

Coupled-axis grouping

The function: "Coupled motion" allows the definition of simple axis couplings. The coupling is performed from one leading axis to one or more following axes, the so-called coupled-motion axes. A separate coupling factor can be specified for each coupled-motion axis.

The leading axis and all the coupled-motion axes assigned to it together form a coupled-axis grouping. If the leading axis is traversed, all coupled-motion axes traverse in accordance with their coupling factors.

A coupled-axis grouping can consist of any combination of linear and rotary axes.

Leading axes

Any axis of the NC including simulated axes can be used as leading axis.

Coupled-motion axes

Any axis of the NC can be used as coupled-motion axis.

Coupling factor

The ratio in which the coupled-motion axis moves in relation to the leading axis is specified via the coupling factor.

Coupling factor K = motion of the coupled-motion axis / motion of the leading axis

Negative coupling factors (motion of the coupled-motion axis in the opposite direction) are also permitted.

	Master axis	Coupled-motion axis	Coupling factor
Coupled-axis grouping 1:	Axis Y	Axis V	1
Coupled-axis grouping 2:	Axis Z	Axis W	–1

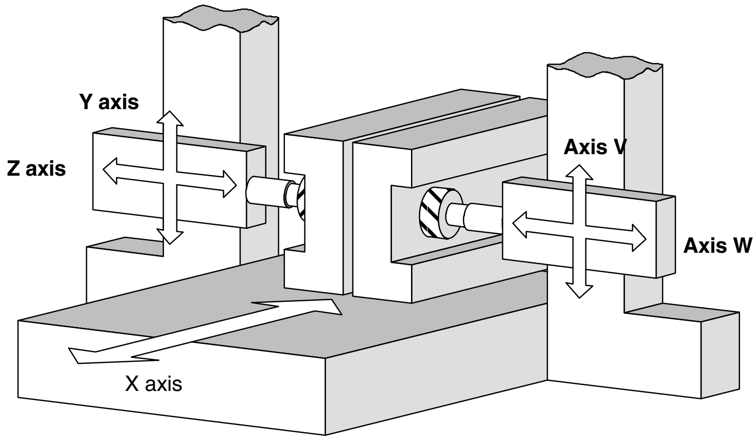


Fig. 2-1 Sample application: Two-side machining

Multiple couplings

Up to two leading axes can be assigned to one coupled-motion axis. The traversing movement of the coupled-motion axis then results from the sum of the traversing movements of the leading axes.

"Dependent coupled-motion axis"

A coupled-motion axis is a "dependent coupled-motion axis" when it traverses as a result of a leading axis movement.

"Independent coupled-motion axis"

A coupled-motion axis is an "independent coupled-motion axis" when it traverses as a result of a direct traverse instruction. The traversing movement resulting on the coupled-motion axis is then the sum total of the traversing movements as a "dependent" and an "independent" coupled-motion axis.

Coupled-motion axis as leading axis

A coupled-motion axis can, at the same time, be the leading axis of a further coupled-axis grouping.

Coordinate system

Coupled motion is always executed in the base coordinate system (BCS).

Activation/deactivation

Coupled motion can be activated/deactivated via part programs and synchronized actions. It must be ensured that the coupled motion is deactivated via the same programming with which it was activated:

- Activation: Part program –> Deactivation: Part program
- Activation: Synchronized action –> Deactivation: Synchronized action

Synchronization on-the-fly

If the activation is performed while the leading axis is in motion, the coupled-motion axis is first accelerated to the velocity corresponding to the coupling. The position of the leading axis at the time the velocities of the leading and coupled-motion axes are synchronized then serves as the start position for further coupled motion.

Operating modes

Coupled motion is effective in the modes: AUTOMATIC, MDA and JOG.

Referencing

The following applies for referencing of axes of a coupled-axis grouping:

- Leading axes

  When referencing the leading axis of a coupled-axis grouping, the coupling to all coupled-motion axes is retained. The coupled-motion axes move in synchronism with the leading axis, as a function of their coupling factors.

- Coupled-motion axis: JOG/REF mode

  When referencing a coupled-motion axis of a coupled-axis grouping, the coupling to the leading axis is canceled. If the coupling is canceled, the following alarm is displayed:

  - Alarm: "16772 Channel *Channel no.* Block *Block no.* Axis *Axis no.* is following axis, coupling is opened."

  The coupling is not activated again until JOG/REF mode is canceled.

  The display of this alarm can be suppressed using the following machine data:

  - MD11410: SUPPRESS_ALARM_MASK, Bit 29 = 1

Caution

When the coupled-motion axis is referenced, the coupling to the leading axis is canceled. If referencing is now executed immediately with the leading axis, i.e. without changing the JOG/REF mode, the coupled-motion axis does <u>not</u> traverse with the leading axis.

- Coupled-motion axis: Part program instruction G74

  It is not possible to reference a coupled-motion axis of a coupled-axis grouping using the G74 programming instruction.

Delete distance-to-go: Coupled-motion axis

Delete distance-to-go for a coupled-motion axis only results in the abort of the independent traversing movement of the leading axis.

Initial settings

Depending on the setting in MD 20112: START_MODE_MASK, the behavior will be as follows:

- Coupled-axis groupings are retained (bit 8=0)
- Coupled-axis groupings are invalidated (bit 8=1)

Depending on the setting in MD 20110: RESET_MODE_MASK, the behavior will be as follows:

- Coupled-axis groupings are invalidated on RESET/part program end (bit 8=0)
- Coupled-axis groupings remain active after RESET/part program end and part program start (bit 8=1)

2.1.2 Programming a coupled-axis grouping

Definition and activation

Definition and activation of a coupled-axis grouping are performed simultaneously with the part program command:

TRAILON(<coupled-motion axis>, <leading axis>, [<coupling factor>])

Effective: Modal

Parameters: Coupled-motion axis, leading axis

Data type: AXIS
Value range: Axis identifier

Parameter: Coupling factor

Data type: REAL
Default value: 1

The ratio of the traversing movement of the coupled-motion axis to the leading axis is specified via the optional coupling factor:

$$\text{Coupling factor} = \frac{\text{Paths of the coupled axis}}{\text{Paths of the leading axis}}$$

A negative coupling factor results in motion in opposite directions for the leading and coupled-motion axis.

Deactivation

The deactivation of the coupling of a coupled-motion axis to **one** leading axis is performed via the part program command:

TRAILOF (coupled-motion axis, leading axis)

Deactivation of the coupling results in a preprocessing stop.

Note

An active coupling can also be deactivated from the PLC by means of an ASUB.

References: /FB/, P3, "Basic PLC Program"

Example 1

Example of an NC part program for the axis constellation shown in Fig. 2-1:

```
TRAILON(Y,V,1)      ;Activate coupled-axis grouping 1
TRAILON(Z,W,–1)     :Activate coupled-axis grouping 2
G0 Z10              ;Infeed of Z and W axis in opposite
                    ;axis direction
G0 Y20              ;Infeed of Y and V axes in same axis direction
      :
G1 Y22 V25          ;Overlay a dependent and independent movement
      :             ;of coupled-motion axis "V"
      :
TRAILOF(Y,V)        ;Deactivate coupled-axis grouping 1
TRAILOF(Z,W)        ;Deactivate coupled-axis grouping 2
      :
```

Example 2

The dependent and independent movement components of a coupled-motion axis are added together for the coupled motion. The dependent component can be regarded as a coordinate offset with reference to the coupled-motion axis.

```
N01 G90 G0 X100 U100
N02 TRAILON(X,U,1)        ;Activate coupled-axis grouping
N03 G1 F2000 X200         :Dependent movement of U, Upos=200, UTrail=100
N04 U201                  ;Independent movement, Upos=U201+UTrail=301
N05 X250                  ;Dependent movement of U,
                          ;UTrail=UTrail(100)+50=150, Upos=351
N06 G91 U100              ;Independent movement, Upos(351)+U100=451
N07 G90 X0                :Dependent movement of U,
                          ;Upos=Upos(451)-UTrail(250)=201
N10 TRAILOF(X,U)
```

2.1.3 Effectiveness of PLC interface signals

Independent coupled-motion axis

All relevant channel-specific and axis-specific interface signals of the coupled-motion axis are effective for the independent motion of a coupled-motion axis, e.g.:

- Activate DRF (DB21, ... DBX0.3),
- Feed override (DB31, ... DBX0.0 – 0.7),
- Axis disable (DB31, ... DBX1.3),
- Servo enable (DB31, ... DBX2.1),
- Activate handwheel (DB31, ... DBX4.0 – 4.2),
- Feed stop (DB31, ... DBX4.3), etc.

This allows the velocity to be changed for the independent motion of a coupled-motion axis using a feed override or allows a DRF offset to be defined using the handwheel in AUTOMATIC and MDA modes.

Dependent coupled-motion axis

With respect to the motion of a coupled-motion axis, which is dependent on the leading axis, only the coupled-motion axis interface signals that effect termination of the motion (e.g. axis-specific feed stop, axis inhibit, servo enable, etc.) are effective.

Leading axis

When a coupled-axis grouping is active, the interface signals (IS) of the leading axis are applied to the appropriate coupled-motion axis via the axis coupling, i.e.

- A position offset or feed control action of the leading axis is applied via the coupling factor to effect an appropriate position offset or feed control action in the coupled-motion axis.
- Shutdown of the leading axis as the result of an IS (e.g. axis-specific feed stop, axis inhibit, servo enable, etc.) causes the corresponding coupled-motion axis to shut down.

Position measuring system 1/2 (DB31, ... DBX1.5 / 1.6)

Switchover of the position measuring system for the leading and coupled-motion axes is not inhibited for an active coupled-axis grouping. The coupling is not invalidated.

Recommendation: Switch the measuring system over when the coupling is deactivated.

Follow-up DB31, ... DBB1.4

The follow-up mode for an axis is activated in the PLC user program by setting the NC/PLC interface signals:
DB31, ... DBB (servo enable) = 0
DBB31, ... DBB4.1 (follow-up mode) = 1

When activating follow-up mode for a coupled-axis grouping, the specified NC/PLC interface signals must be set simultaneously for all axes (master and slave axes) of the coupled-axis grouping.

If follow-up mode is activated for the master axis only, a permanent offset results within the coupling.

The following NC/PLC interface signals and system variables can be used to determine which axis, if any, is a master or slave axis:
DB31, ... DBB99.0 (master axis/spindle active)
DB31, ... DBB99.1 (slave axis/spindle active)
$AA_COUP_ACT[axis identifier] (see Subsection 2.1.4: Status of coupling)

2.1.4 Supplementary conditions

Control dynamics

It is recommended to align the position control parameters of the leading axis and the coupled-motion axis within a coupled-axis grouping.

Note

The alignment of the position control parameters of the leading axis and the coupled-motion axis can be performed via a parameter set changeover.

Status of coupling

The coupling status of an axis can be determined using the following system variable:
$AA_COUP_ACT[axis identifier]

$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 coupling mode may be active at any given time.

Distance-to-go: Coupled-motion axis

The distance-to-go of a coupled-motion axis refers to the total residual distance to be traversed from dependent and independent traversing.

RESET response

If with NC RESET or end of part program in a channel, the leading axis is not stopped as well (cross-channel coupling, command axis, PLC axis, etc.), the requested Reset cannot be completed.

Because of the traversing of the leading axis, the coupled-motion axis is still active for the channel in which the Reset is requested. With suitable actions (NC Reset in the channel of the leading axis, stopping of the command or PLC axis), the leading axis must also be stopped in parallel to the coupled-motion axis.

Dynamic limit

The dynamic limit is dependent upon the activation of the coupled-axis grouping:

- Part program
  If the activation is performed in the part program, the dynamics of all coupled-motion axes are taken into account so that no coupled-motion axis is overloaded during the traversing of the leading axis.
- Synchronized action
  If the activation is performed in a synchronized action, the dynamics of the coupled-motion axes are not taken into account during the traversing of the leading axis. This can result in an overload for coupled-motion axes with a dynamic response which is less than that required for the coupling.

Caution

If a coupled-axis grouping is activated via synchronized actions, it is the special responsibility of the user / machine manufacturer to provide suitable measures to ensure that an overload of the coupled-motion axes does not occur through the traversing of the leading axis.

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.

Within a curve segment, the relation between the master value and following value is generally described by a polynomial up to the third order.

In **SW 6** and higher, polynomials up to the **5th degree** are permissible. See also

References: /PGA/, Programming Guide Advanced.

Polynomial segments are required for curve tables when the following applies:

- Polynomials or circles are programmed
- Spline is active
- Compressor is active
- Polynomials or circles are generated internally (chamfer/rounding, approximate positioning with G643, TRC etc.)

Tool radius compensation

Curve tables are available in which it is possible to specify the tool radius compensation in the table definition even if polynomial blocks or blocks with no motion for an axis, or jumps for the following axis, occur in the curve table (G41/G42/G40 in the table definition).

Tool radius compensation

The equidistant curve (tool center point path of the tool radius compensation) of a curve consisting of polynomials can no longer be displayed exactly using polynomials. The associated curve tables must be approximated stepwise using polynomials in this case. This means that the number of segments in the curve table no longer matches the number of programmed segments. The number of segments required for the curve table is defined by the bend of the curve. The larger the bend for the programmed curve, the more segments are required for the curve table.

On account of the tool radius compensation 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 compensation can produce segments for which the following axis or leading axis have no movement. Although 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 created in this case. The following options are available:

0:
No curve tables that contain a jump in the following axis are produced. Alarm 10949 is output and 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 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 programmed curve.

Defining the memory type for curve tables

It is possible to specify the memory type to be used for creating the curve table when defining the curve table. The following options are available:

– SRAM

– DRAM

Table definitions for SRAM type memory remain available following control run-up. Curve tables for DRAM must be redefined following control run-up. Additional machine data have been provided for reserving space in the DRAM:

MD 20905: CTAB_DEFAULT_MEMORY_TYPE defines the memory type if no memory type has been specified in the table definition (see below).

Storing curve tables in a variety of memory types entails an optional specification of the memory type in delete calls for curve tables (see below).

2.2.2 Programming a curve table

Definition

The following modal language commands work with curve tables
(you will find explanations of the parameters 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()

- Deleting curve table(s):

  CTABDEL(n) ; Curve table n
  CTABDEL(n, m) ; [n < m], several in number range
  ; deleted in SRAM and DRAM.

  CTABDEL(n, m, memType) ; Deletion with indication of memory type:
  The curve tables with the numbers in the range stored in the memory type specified are deleted from that memory type. All other curve tables 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, [slave axis, master axis])

- Read the master value for a following value

  CTABINV(following value, approx. master value, n, degrees, [following axis, leading axis])

As of SW 6.3

Access to curve table segments:

- 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(master value, n, degrees, [following axis, master axis])

Note

If the curve table functions such as CTAB(), CTABINV(), CTABSSV() etc., are used in **synchronized actions**, only **real-time variables**, e.g. $AC_PARAM[...] or $R[...] are allowed as the return value and the “degrees” argument of the function.

Example:
ID=1 WHEN TRUE DO $R1 = CTABSSV(10, 1, $R2)
or
ID=1 WHEN TRUE DO $AC_PARAM[1] = CTABSSV(10, 1, $AC_PARAM[2])

Enable/cancel lock

The following functions can be used to enable or cancel deletion and overwrite locks for part programs:

- **Enable** deletion and overwrite **lock**.

  General form: **CTABLOCK**(n, m, memType)

- **Cancel** deletion and overwrite **lock**.

  CTABUNLOCK releases the tables locked with CTABLOCK. Tables involved in an active coupling remain locked, i.e. they cannot be deleted. However, the CTABLOCK command is 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 forms:

Curve table number n
CTABLOCK(n)
Curve tables in the number range n to m
CTABLOCK(n, m)
All curve tables regardless of memory type
CTABLOCK()
All curve tables of memory type specified
CTABLOCK(, , memType)

Curve table number n
CTABUNLOCK(n)
Curve tables in the number range n to m
CTABUNLOCK(n, m)
All curve tables regardless of memory type
CTABUNLOCK()
All curve tables of memory type specified
CTABUNLOCK(, , memType)

As of SW 6.4

Other commands for calculating and differentiating between curve tables for applications for diagnosing and optimizing the use of resources:

- Number of **defined** tables **regardless** of memory type

  CTABNO()

- Number of **defined** tables in SRAM or DRAM

  CTABNOMEM(memType)

- Number of **possible** curve tables in memory memType.

  CTABFNO(memType)

- Table number of **nth** curve table.

  General form: **CTABID**(n, memType)
  Supplies the table number of the nth curve table, in memory type memType. **CTABID**(1, memType) is used to read out the highest curve number (105) of the memory type specified.

  CTABID(n)
  Supplies the table number of the nth curve table in the memory specified using MD 20905: CTAB_DEFAULT_MEMORY_TYPE. **CTABID**(p)
  Supplies the ID (table number) of the curve table entered in the memory as the pth curve table.

Note

If for example the sequence changes between consecutive calls of CTABID(), CTABID(n, ...) can be used to supply a **different** curve table to that before the change was made in the memory.

- Indicates **the lock state** of curve table number n.

  CTABISLOCK(n)
- **Checks** curve table number n.

  CTABEXIST(n)
- Indicates the **memory** in which curve table number n is stored.

  CTABMEMTYP(n)
- Indicates the **table periodicity**.

  CTABPERIOD(n)
- Number of **curve segments** already used in memory memType.

  CTABSEG(memType, segType)
- Number of curve segments used in **curve table** number n

  CTABSEGID(n, segType)
- Number of **still** possible **curve segments** in memory memType.

  CTABFSEG(memType, segType)
- Number of **maximum** possible **curve segments** in memory memType.

  CTABMSEG(memType, segType)
- Number of **polynomials already used** in memory memType.

  CTABPOL(memType)
- Number of **curve polynomials** used in by curve table number n.

  CTABPOLID(n)
- Number of **still** possible **polynomials** in memory memType.

  CTABFPOL(n)
- Number of **maximum** possible **polynomials** in memory memType.

  CTABMPOL(n)

Edge values of curve tables

Behavior of the leading axis/following axes on the edges of the curve table:

- The value at the **beginning** of the curve table is read by a following axis.

  CTABTSV(n, degrees, FAxis) Following value at the beginning of the curve table

- The value at the **end** of the curve table is read by a following axis.

  CTABTEV(n, degrees, FAxis) Following value at the end of the curve table

- The value at the **beginning** of the curve table is read by the leading axis.

  CTABTSP(n, degrees, FAxis) Master value at the beginning of the curve table

- The value at the **end** of the curve table is read by the leading axis.

  CTABTEP(n, degrees, FAxis) Master value at the end of the curve table

- Determine the value range of the following value.

  CTABTMIN(n, FAxis) Minimum following value of curve table

  CTABTMAX(n, FAxis) Maximum following value of curve table

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.

  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 memType memory area)

- applim:

  Behavior at the curve table edges.

  – 0 non-periodic (table is processed only once, even for rotary axes).
  – 1 periodic, modulo (the modulo value corresponds to the LA table values).
  – 2 periodic, modulo (LA and FA are periodic).

- Master value

  Position value for which a following value is to be determined.

- Slave value

  Position value for which a master value is to be calculated.

- aproxmastervalue

  Position value that can be used to determine a unique master value in the case of an ambiguous reversing function of the curve table.

- degrees
  Parameter in which the pitch of the table function is returned.
- memType
  Optional parameter for entering the memory type in which the curve table is to be created.
  Possible values:
  "SRAM" Curve table created in static memory
  "DRAM" Curve table created in dynamic memory.
  If an invalid memType is entered, value –2 is returned.
  If the parameter is omitted, then the memory type set via MD 20905:CTAB_DEFAULT_MEMORY_TYPE takes effect.
- segType
  Optional parameter for the specification of the segment type
  Possible values
  segType "L" linear segments
  segType "P" polynomial segments

References: PGA, Programming Guide Advanced

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 move vertical to the leading axis.
- Axis names from gantry axis groups cannot be used to define a table (only leading axis are possible).

Restrictions SW 6.3 and higher

Depends on 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 compensation, no memory type

```
N100 CTABDEF(AX2, AX1, 3,0)    ;Start of definition for non
                               ;periodic curve table number 3
N110 AX1=0 AX2=0               ;1st motion instruction specifies the starting
                               ;value master value: 0, following value:0
 N110 AX1=20 AX2=0             ;1st curve segment:
                               ;Master value: 0...20, slave value: Starting
                               ;value ...0
N120 AX1=100 AX2=6             :2nd curve segment:
                               ;Master value: 20...100, slave 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, slave value: 6...0
N200 CTABEND                   ;End of definition, curve table
                               ;generated in internal representation.
                               ;Preprocessing reorganizes to state at
                               ;beginning of N100
```

Example 2

Example of a curve table with active tool radius compensation:

Prior to definition of a curve table with CTABDEF(), the tool radius compensation must not be active; otherwise alarm 10942 is output. This means that tool radius compensation **must** be activated **within** the definition of the curve table. Similarly, it must be deactivated again before the end of the curve table definition using CTABEND.

```
              ...
N10 CTABDEF(Y, X, 1, 0)        ;Start of definition for non
                               ;periodic curve table number 1
N20 X0 Y0
N30 G41 X10 Y0                 ; TRC ON
N40 X20 Y20
N50 X40 Y0
N60 X60 Y20
N70 X80 Y0
N80 G40 X90 Y0                 ; TRC OFF
N90 CTABEND
              ....
```

Tool radius compensation is activated in block N30; this causes the approach movement for radius compensation to be made in this block. Similarly, the approach movement for deactivation of the radius compensation is made in block N80.

Note

The value pairs between CTABDEF and CTABEND must be specified for precisely the axis 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.

2.2.3 Access to table positions and table segments

Reading table positions

With the program commands CTAB and CTABINV you can read off the following value for a master value (CAB) from the part program and from synchronized actions, or the reversal of the curve table, i.e. read off the master value for a following value. You can use the pitch value to calculate the velocity 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 CTABSSV and CTABSEV calls.

The language commands CTABSSV and CTABSEV generally provide the start and end values of the internal segments of the curve tables for the following axis. These values only agree with the programmed values of the curve tables if the programmed segments can be converted 1:1 to the internal segments of the curve table. This is always the case if only G1 blocks or axis polynomials are used to define the curve tables and no other functions are active.

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 compensation is active for polynomial interpolation.

In these cases, the language commands CTABSSV and CTABSEV may not be used to query the start and end points of the programmed segments.

CTABINV

When using the inversion function for the curve tables (CTABINV), 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.

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.

Determining the segment associated with master value X

Example of reading the segment starting and end values for determining the curve segment associated with master value X = 30 using CTABSSV and CTABSEV:

```
N10 DEF REAL STARTPOS          ;Start of definition for start and
N20 DEF REAL ENDPOS            ;start position of curve table
N30 DEF REAL GRADIENT          ;
...                            ;
N100 CTABDEF(Y, X, 1, 0)       ;Start of table definition
N110 X0 Y0                     ;Start position, 1st table segment
N120 X20 Y10                   :End position, 1st table segment = Start
N130 X40 Y40                   ;position 2nd table segment ...
N140 X60 Y10                   ;3rd curve segment ...
N150 X80 Y0                    ;4th curve segment ...
N160 CTABEND                   ;End of table definition
...                            ;
N200 STARTPOS =                ;
CTABSSV(30.0, 1, GRADIENT)     ;Start position Y in segment 2 = 10.
...                            ;
N210 ENDPOS =                  ;End position Y in segment 2 = 40.
CTABSEV(30.0, 1, GRADIENT)     ;Segment 2 is associated with MV X = 30.0
```

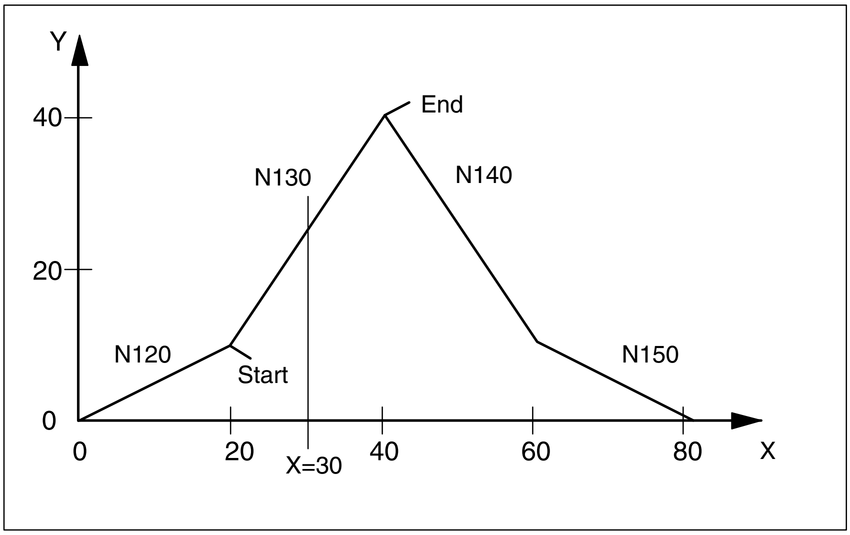


Fig. 2-2 Determining the curve segment associated with master value X = 30

Reading values at start and end

The values of the following axes and also the master axis at the start and end of a curve table can be read with the following calls:

```
R10 =CTABTSV(nth degree, FAxis  Following value at start of table
R10 =CTABTEV(nth degree, FAxis) Following value at end of table
R10 =CTABTSP(nth degree, LAxis  Master value at start of table
R10 =CTABTEP(nth degree, LAxis  Master value at end of table
```

Value range of following value

The following example illustrates how the minimum and maximum values of the table are determined using CTABTMIN and CTABTMAX:

```
N10 DEF REAL STARTVAL         ;Start of definition for start and
N20 DEF REAL ENDVAL           ;start values of curve table
N30 DEF REAL STARTPARA        :
N40 DEF REAL ENDPARA          ;
N50 DEF REAL MINVAL           ;
N60 DEF REAL MAXVAL           ;
N70 DEF REAL GRADIENT         ;
...                           ;
N100 CTABDEF(Y, X, 1, 0)      ;Start of table definition
N110 X0 Y10                   ;Start value, 1st table segment
N120 X30 Y40                  :End position, 1st table segment = Start
N130 X60 Y5                   ;position 2nd table segment ...
N140 X70 Y30                  ;3rd curve segment ...
N150 X80 Y20                  ;4th curve segment ...
N160 CTABEND                  ;End of table definition
...                           ;
N200 STARTPOS =               ;STARTPOS = 0
CTABTSV(1, GRADIENT)          ;Start position of table and
N210 ENDPOS =                 ;ENPOS = 20
CTABTEV(1, GRADIENT)          ;end position of table
N220 STARTPARA =              ;STARTPARA = 10
CTABTSP(1, GRADIENT)          ;Master value at start of curve table
N230 ENDPARA =                ;ENDPARA = 80
CTABTEP(1, GRADIENT)          ;Master value at end of curve table
...                           ;Read from value range of following axis
N240 MINVAL = CTABTMIN(1)     ;Minimum value if Y = 5 and
N250 MAXVAL = CTABTMAX(1)     ;Maximum value if Y = 40
```

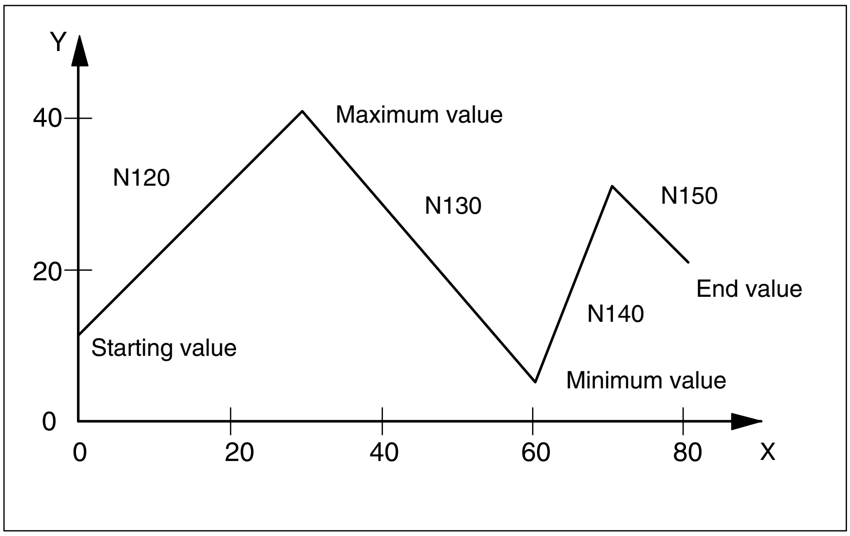


Fig. 2-3 Determining the minimum and maximum values of the table

2.2.4 Activation and coupling axes to a curve table

Activation

Coupling real axes to a curve table:

LEADON (following axis, leading axis, n) n=number of curve table

A concrete channel axis can be coupled with the master value via a curve table:

- in a part program or
- in the definition of a synchronized action.

Multiple use

A curve table can be used several times in a single part program to couple different channel axes.

Deactivation

LEADOF (following axis, leading axis)

You can deactivate the function both in the program and using synchronized actions.

Example

```
N1000   LEADON(A, X, 3)   ;Axis A follows master value X acc. to the
                          ;rules of motion defined in
                          ;curve table number 3.
N1010   LEADOF(A, X)      ;The coupling between axis A and its master
                          ;value is canceled.
...
N10010  CTABDEL(3)        ;Curve table number 3 is deleted, it
                          ;is no longer available for activating a
                          ;coupling.
```

Special cases with modulo master axes

When an axial master value coupling is active, the position of the following axis via a curve table is unique, i.e. an absolute assignment to the master axis exists.

This means that, when a modulo rotary axis is used as the master axis, the position of the master axis is absolute. In other words, the position of the modulo rotary axis entered in the curve table is absolute, and not modulo-reduced.

Example:

Let the position of a modulo rotary axis with LEADON be 210 degrees. The position 210 degrees is used as the starting value in the curve table.

After one rotation of the modulo axis, the axis position is displayed again as 210 degrees. However, the starting value entered in the curve table is 210 degrees plus 1 rotation (360 degrees) = 570 degrees.

2.2.5 Organizing curve tables in the SRAM and DRAM

Memory usage

The memory available in SRAM and DRAM (SW 6.4 and higher) 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.

MD 18402: MM_NUM_CURVE_SEGMENTS specifies the number of curve table segments that can be stored in SRAM

MD 18404: MM_NUM_CURVE_POLYNOMS specifies the number of curve table polynomials that can be stored in SRAM.

MD 18403: MM_NUM_CURVE_SEG_LIN specifies the maximum number of linear segments 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.

MD 18409: MM_NUM_CURVE_SEG_LIN_DRAM specifies the maximum number of linear segments in DRAM.

Memory optimization

For a curve table, **linear segments** can only be stored efficiently in the memory, when the relevant MD 18403: MM_NUM_CURVE_SEG_LIN for tables in SRAM or MD 18409: MM_NUM_CURVE_SEG_LIN_DRAM for tables in DRAM, is greater than zero. If no memory area is created with these machine data, linear segments are stored as polynomial segments requiring unnecessary space.

If memory has been configured for tables with linear and polynomial segments via machine data and the memory for linear segments runs out when generating a linear table, polynomial segments are used for this, if some are still available. In this case, memory is “wasted”, as a polynomial segment requires an unnecessary amount of memory to store a linear segment. This condition is indicated by a warning; alarm 10958 is output, with which the number of unnecessarily used polynomial segments is also specified. This alarm only serves as a **warning** and does not result in the interruption of the program or the generation of the curve table.

If a curve table consists of linear segments and polynomials of a high degree, a memory area for linear segments and a memory area for polynomial segments is required for the storage of the curve table. An alarm is output for insufficient memory in the relevant areas. The alarm parameters specify which resources are not sufficient.

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 so that there is always **enough extra memory to accommodate the table to be overwritten**.

Overwriting curve tables

Curve tables that are not active in a master value coupling and are locked with CTABLOCK() may be **overwritten**.

Deleting curve tables

Curve tables that are not active in a master value coupling and are locked with CTABLOCK() may be **deleted**.

Transformation

Transformations may not be programmed in curve tables. RAANG is an exception. If TRAANG is programmed, the rule of motion 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 that stipulates that “the direction of motion of the leading axis must not reverse at any point of the rule of motion” must then be met in the machine coordinate system. Please note 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_KURVENTABELLEN_WPD
;Def.TAB1 0–100mm Curve no. 1/1 non-perio.
N10 CTABDEF(YGEO,XGEO,1,0) ;FA=Y LA=X Curve no. =1 non-perio.
N1000 XGEO=0 YGEO=0            ; Starting values
N1010 XGEO=100 YGEO=100
CTABEND
M30
```

With polynomial blocks:

```
%_N_TAB_1_NOTPERI_MPF
;$PATH=/_N_WKS_DIR/_N_KURVENTABELLEN_WPD
;Def.TAB1 0–100mm Curve no. 1/1 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
```

2.2.6 Behavior in AUTOMATIC, MDA, JOG modes

Activation

An activated curve table is functional in the AUTOMATIC, MDA and JOG modes.

Basic setting after run-up

No curve tables are active after run-up.

2.2.7 Effectiveness of PLC interface signals

Dependent following axis

With respect to the motion of a following axis that is dependent on the leading axis, only the following axis interface signals that effect termination of the motion (e.g. axis-specific feed stop, axis inhibit, servo enable, etc.) are effective.

Leading axis

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, servo enable, etc.) causes the corresponding coupled-motion axis to shut down.

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.

Recommendation: Switch the measuring system over when the coupling is deactivated.

2.2.8 Diagnosing and optimizing the use of resources (SW 6.4 and higher)

The following functions permit **part 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 described in Subsection 2.2.2 without necessarily having to increase the memory requirement. The description of the parameters in Subsection 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 the number of defined tables in the SRAM or DRAM.
  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 Memory type invalid
- Determine the number of curve segments still possible in the memory type
  CTABFNO(memType**)**
  If memType is not specified, the memory type set in
  MD 220905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  >= 0 Number of tables still possible
  –2 Memory type invalid

- Determine the table number of the pth table in the optionally specified memory type
  CTABID(p, memType)
  If memType is not specified, the memory type set in
  MD 22905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  Table number or
  alarm if p or memType invalid

When using the CTABID(p, memType) function, no assumptions should be made regarding the sequence of the curve tables in the memory. The CTABID(p, ...) function supplies the ID (table number) of the curve table entered in the memory as the pth curve table.

If the sequence of the curve tables in the memory changes between consecutive calls of CTABID(), e.g. due to the deletion of curve tables with CTABDEL(), the CTABID(p, ...) function can supply a different curve table with the same number.

To prevent this happening, the curve tables concerned can be locked using the CTABLOCK(...) language command. In this case, please ensure that the curve tables concerned are then unlocked with CTABUNLOCK().

- Query **locking status**
  Table n
  CTABISLOCK(n)
  Result:
  > 0 Table is locked
  Reason for locking:
  1 by CTABLOCK()
  2 by active coupling
  3 by CTABLOCK() and active coupling
  = 0 Table is not locked
  – 1 Table does not exist

- Check that the curve table **exists**
  CTABEXISTS(n)
  Result:
  1 Table exists
  0 Table does not exist

- Determine the **memory type** of a curve table
  CTABMEMTYP(n)
  Result:
  0 Table in SRAM
  1 Table in DRAM
  –1 Table does not exist

- Determine whether the table has been defined as a **periodic** table
  CTABPERIOD(n)
  Result:
  0 Table is not periodic
  1 Table is periodic in leading axis
  2 Table is periodic in leading and following axis
  –1 Table does not exist

b) Curve table segments

- Determine the number of curve segments of type segType **used** in memory area memType.
  CTABSEG(memType, segType)
  If memType is not specified, the memory type set in
  MD 22905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  >= 0 Number of curve segments
  –2 Memory type invalid
  If segType is not specified, the the sum is output via linear and polynomial segments in the memory type.
  –2 segType not equal to "L" or "P"

- Determine the number of used curve segments of type segType of a certain curve table
  CTABSEGID(n, segType)
  Result:
  >= 0 Number of curve segments
  –1 Curve table with the number n does not exist.
  –2 segType not equal to "L" or "P"

- Determine the number of **free** curve segments of type segType of the memory type
  CTABFSEG(memType, segType)
  If memType is not specified, the memory type set in
  MD 22905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  >= 0 Number of free curve segments
  –2 Memory type invalid, segType not equal to "L" or "P"

- Determine the number of **maximum** possible curve segments of type segType of the memory type
  CTABMSEG(memType, segType)
  If memType is not specified, the memory type set in
  MD 22905: CTAB_DEFAULT_MEMORY_TYPE applies.
  Result:
  >= 0 Number of maximum possible curve segments
  –2 Memory type invalid, segType not equal to "L" or "P"

c) Polynomials

- Determine the 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 Memory type invalid
- Determine the number of curve polynomials used in a specific curve table
  CTABPOLID(n)
  Result:
  >= 0 Number of curve polynomials used
  –1 Curve table number n does not exist
- Determine the 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 free curve polynomials
  –2 Memory type invalid
- Determine the **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 Number of maximum possible curve polynomials
  –2 Memory type invalid

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 rules of motion 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. Set IS_VIRTUAL_AX=1 in MD 30132.

Note

Setpoint coupling is the default setting 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 so 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.

With MD 30130: CTRLOUT_TYPE[n] defines the setpoint value output of the axis type.

0: Simulated axis
1: Standard real axis
2: FM-NC stepper motor
3: Free
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.

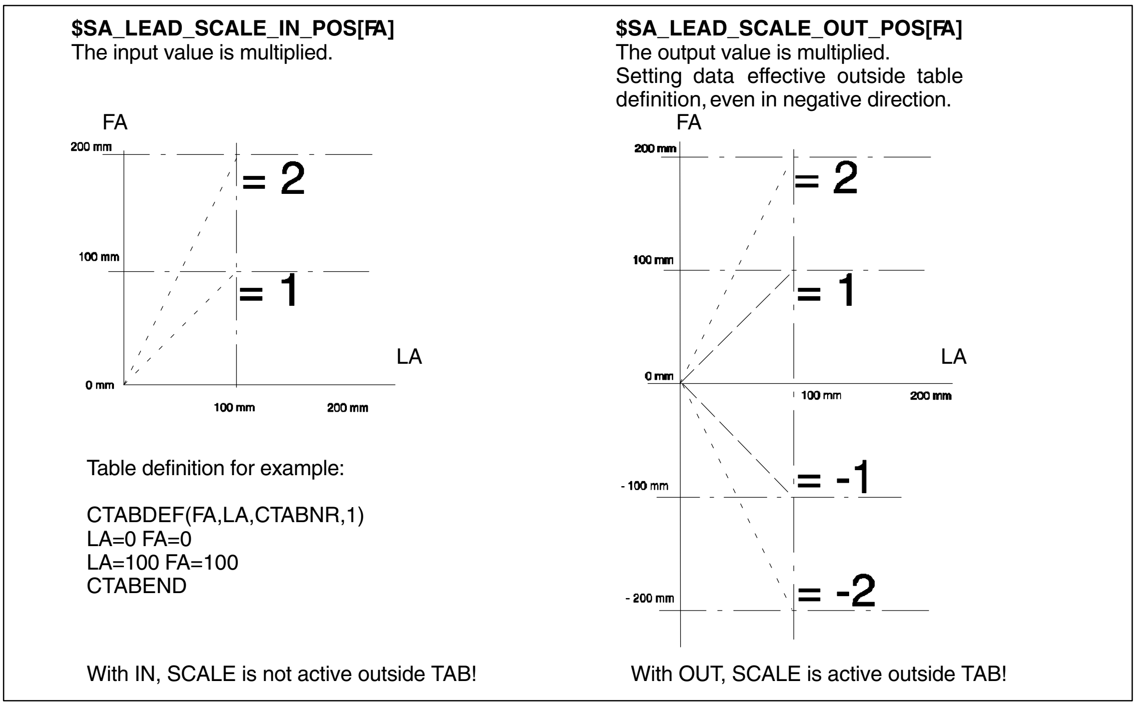


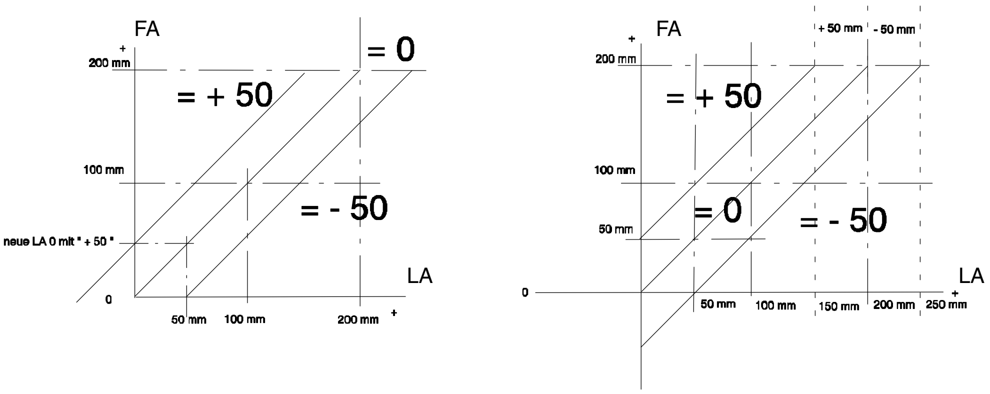
Fig. 2-4 Master value coupling offset and scaling (multiplied)

$SA_LEAD_OFFSET_IN_POS[FA]

The calculated position of the leading axis is offset by the increment specified.

$SA_LEAD_OFFSET_OUT_POS[FA]

The calculated position of the following axis is offset by the increment specified. If the TAB is offset outside the FA LEADON position, the coupling cannot be activated. Setting data effective outside table definition, even in negative direction.



With IN, OFFSET is not active outside TAB!

With OUT, OFFSET active outside TAB!

Fig. 2-5 Master value coupling offset and scaling (with increment offset)

Reaction to stop

All master-value-coupled following axes react to channel stop and MODE GROUP stop.

Master-value-coupled following axes react to a stop due to end of program (M30, M02) if they have not been activated by static synchronized actions (IDS=...). (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).

Axial functions

Actual value coupling causes a position offset between the leading and following axis. This is due to the deadtime in the position controller between the actual value of the leading axis and the following axis necessitated by the IPO cycle.

By default, the position offset and following error are compensated by means of linear extrapolation of the master value by this deadtime. Machine data MD 37160: LEAD_FUNCTION_MASK Bit 0 = 0 deactivates deadtime compensation for master value coupling.

Interface to axis exchange

A master-value-coupled following axis receives its setpoint values from curve tables. **Overlaid programming of this axis is not possible in the part program.** Therefore, the master-value-coupled following axis is removed from the channel in the same way as for axis exchange. This is carried out automatically when the coupling is activated in the part program.

If the coupling is to be activated with synchronized 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 part 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 (activation via synchronized action):

```
SPOS=0
B=IC(0)          ;Switch spindle to axis mode
RELEASE(Y)       ;Release for synchronized 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 of master value coupling

Definition and activation

A master value coupling is defined and activated simultaneously with the modal language 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

  A REF. POINT is not required to activate the coupling. A defined FA cannot be traversed in JOG mode (not even if the synchronism FINE/COARSE interface signals are 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"

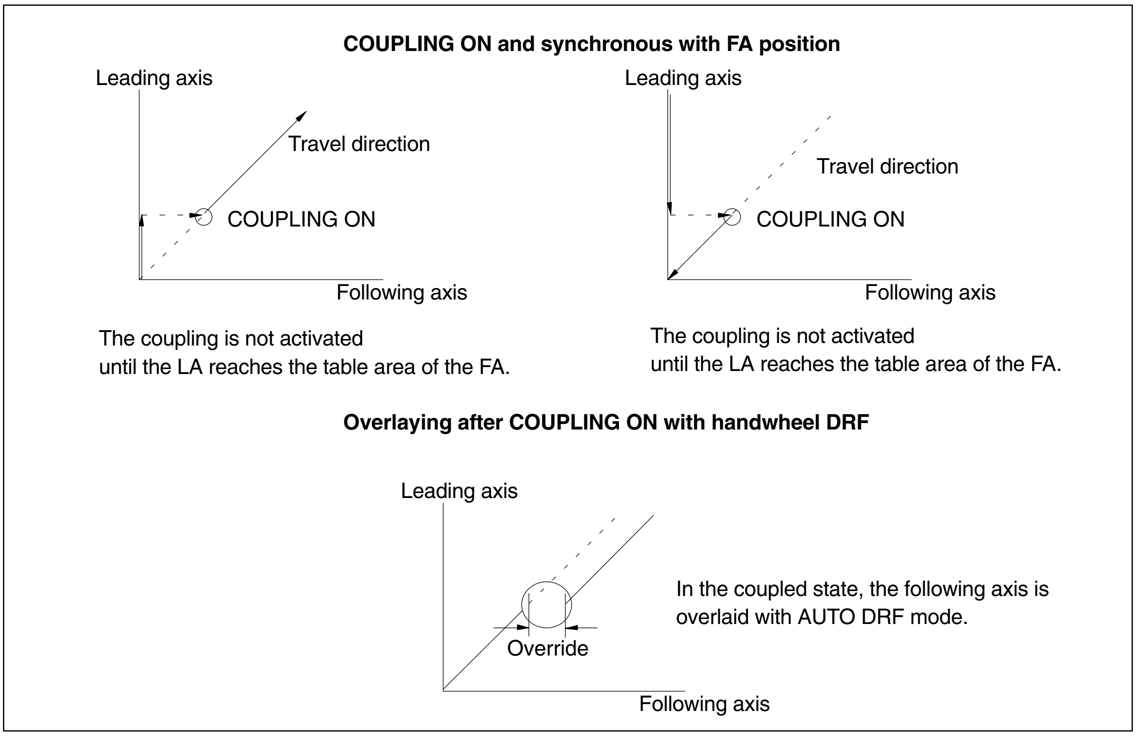


Fig. 2-6 Activating the master value coupling

Deactivation

A master value coupling is deactivated with the model language 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 synchronized action.

Note

Activating and deactivating the axis master value coupling with LEADON, LEADOF is permissible both in the part program and in synchronized actions.

References: /FB/, S5, “Synchronized Actions”

Coupling type

With the 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 axes)

1: Setpoint value coupling (default)

2: Simulated master value (note virtual axis, not evaluated for FA)

Readable system variables of the master value

The system variables of the master value can be read from the part program and from synchronized 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 part program and synchronized 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] can be read from the part program and synchronized action and indicates whether and in what manner following axis FA is synchronized:

0: Not synchronized

1: Coarse synchronism (acc. to MD 37200: COUPLE_POS_TOL_COARSE)

3: Fine synchronism (acc. to MD 37210: COUPLE_POS_TOL_FINE)

The information from system variable $AA_SYNC[ax] corresponds to the assigned VDI signals:

IS “Fine synchronism” DB 31, ... DBX98.0 and

IS “Coarse synchronism” 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 Behavior in AUTOMATIC, MDA, JOG modes

Activation

A master value coupling is active depending on the settings in the part program and in the machine data MD 20110: RESET_MODE_MASK and MD 20112: START_MODE_MASK.

Manual mode

Once a master axis coupling has been activated, traversal of the master axis (e.g. with rapid traverse or incremental dimension INC1 ... INC10000) results in a movement of the slave axis, allowing for the curve table definition.

Referencing

A master-value-coupled following axis is to be referenced prior to activation of the coupling. 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

Depending on the setting in MD 20110: RESET_MODE_MASK (bit 13) and MD 20112: STAR_MODE_MASK (bit 13), the behavior will be as follows:

- 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 invalidated on START.
  However, the master value coupling activated via IDS=... remains valid.
- MD 20110: RESET_MODE_MASK=1H
  ==> Master value coupling is invalidated with RESET, regardless of MD 20112: START_MODE_MASK
  Master value coupling activated via IDS=... can only be deactivated via an operator front panel reset and remains valid after program end/reset (M30, M02).
- MD 20110: RESET_MODE_MASK=0H
  ==> Master value coupling remains valid after RESET and is invalidated on START, regardless of MD 20112: START_MODE_MASK.
  However, master value coupling activated via IDS=... remains valid.

References: /FB/, K2, "Coordinate System, Axis Types, Axis Configurations, ..."

Activating, deactivating

Master value couplings activated via a static synchronized action (IDS=...) are

- not deactivated during program start, regardless of the value of MD 20110: RESET_MODE_MASK and MD 20112: START_MODE_MASK.
- not deactivated during program end reset (M30, M02), regardless of the value of 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, servo enable, etc.) causes the corresponding coupled-motion axis to shut down.

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.

Recommendation: Switch the measuring system over when the coupling is deactivated.

2.3.5 Special characteristics of the axis master value coupling function

Control dynamics

Depending on the application in question, it may be advisable to match the position controller parameter settings (e.g. servo gain factor) of the leading axis and following axis in an axis grouping. It may be necessary to activate other parameter sets for the following axis. The dynamics of the following axis should be the same or better than those of the leading axis.

Status of coupling

See Subsection 2.1.4

Actual value display

The display of the actual value is updated for all axes of in a master-value-coupled axis grouping (only real axes) coupled via a master value.

Interpolation

When the movement defined in the curve table is interpolated, 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.

$FA_{set} = SynPosFA + (LA_1 - SynPosLA_1)*CF_1 + ... + (LA_5 - SynPosLA_5)*CF_5$

Where:
$SynPosFA$, $SynPosLA_i$ from EGONSYN call (see below)
FA_{set} Partial setpoint of following axis
LA_i Setpoint or actual value of ith leading axis (depends on coupling type, see below)
KF_i Coupling factor of ith leading axis (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.

A gear grouping can be:

- defined,
- activated,
- deactivated,
- deleted

from the part program.

Expansions in SW 6 and higher

In SW 6 and higher, the influence of each of the 5 leading axes can be specified using a **curve table** (see Section 2.2) as an alternative to a gear ratio (CF=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.

SW 6.3 and higher:
The function EG with curve tables can be activated with EGON.

SW 6.4 and higher:
The function EGONSYNE is available for approaching the synchronized position of the following axis with a specified approach mode.

For special applications, it may be advisable configure the position controller as a **PI controller**.

Caution

Knowledge of the control technology and measurements with servo trace are an absolute prerequisite for using this function.

References: See /IAD/. Installation and Start-Up Guide
/FB/, G2, Velocities, 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 details see Subsection 2.4.2).

Coupling factor

The coupling factor must be programmed for each leading axis in the grouping. It is defined using the numerator/denominator.
The numerator and denominator coupling factor values are specified in the following activation calls for each leading axis:

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

The following axis of an ELG can be the leading axis of another EG. Chapter 6, page 3/M3/6-103 contains a detailed example of this.

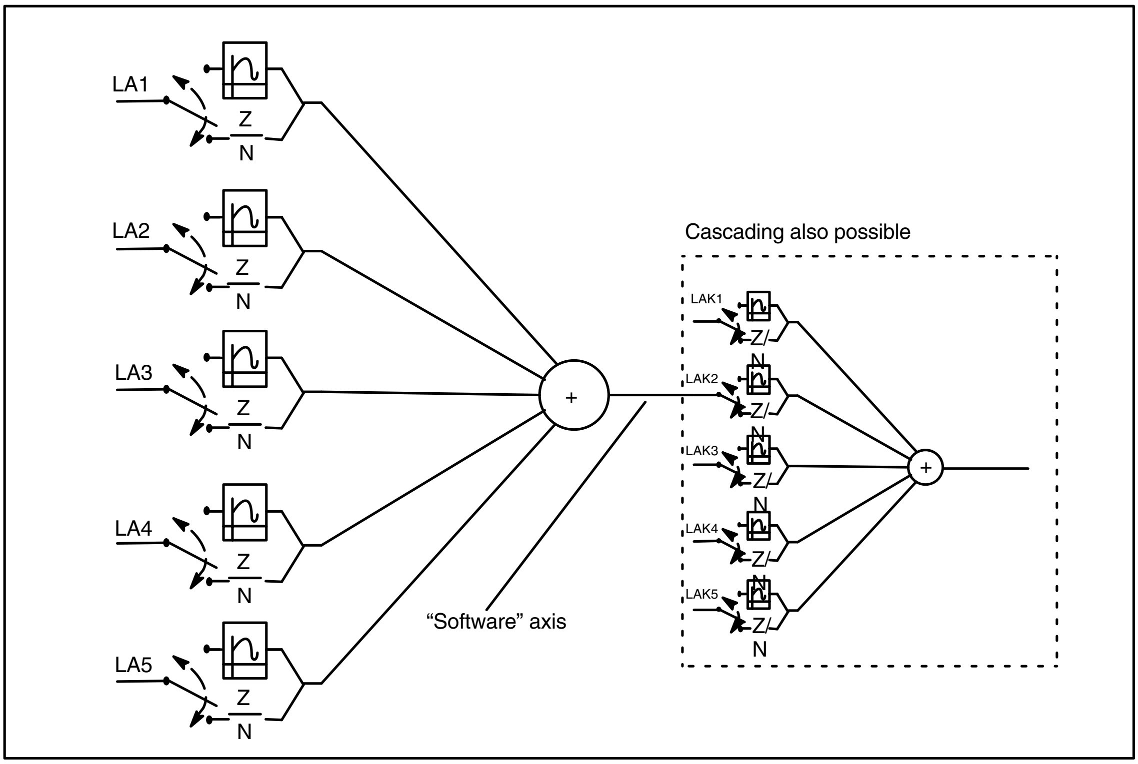


Fig. 2-7 Block diagram of an electronic gear

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 motion of the gear specified by the positions of the leading axes at this time if the part program developer makes sure that it is. The control then uses the motion of the following axis to ensure that the setpoint and actual positions of the following axes correspond as quickly as possible 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 synchronization position for the following axis are specified by the command.

– The control then traverses the following axis with just the right acceleration and velocity to the specified synchronization position so that the following axis is in position with the leading axes at its synchronization position.

– If the following axis **is stationary:** If IS "Feed stop/spindle stop" DB 31, ... DBX 4.3 is set for the following axis, the following axis is **not** set in motion by EGON or EGONSYN. A motion command is sent for the following axis and the block change is disabled until the axis-specific feed is enabled.
EGOSYN is interrupted by RESET and converted to EGON. The programmed synchronized positions are deleted.

– If the following axis is **not stationary**: IS "Feed stop/spindle stop" DB 31, ... DBX 4.3 has no direct effect on the electronic gear. As before, it does have an indirect effect on the leading axes, if these are located in the same channel.

– Nothing has been implemented for the **channel-specific** feed enable and for the override. The override continues to have no direct effect on the electronic gear. The **axis-specific** feed enable is set depending on the current override setting.

Synchronization for EGONSYNE

With EGONSYNE(), the positions of the leading axes and the synchronization position for the following axis are specified by the command.

The control moves the following axis to the synchronized position according to the program approach mode.

Synchronization abort with EGONSYN and EGONSYNE

The EGONSYN/EGONSYNE command is aborted under the following conditions and changed to an EGON command

– RESET
– Axis switches to follow-up

The defined synchronization positions are ignored.
The synchronization monitoring function continues to refer to the synchronization positions.

The position synchronization abort triggers alarm 16774

The alarm can be suppressed with machine data MD 11410 SUPPRESS_ALARM_MASK Bit31 = 1.

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 **difference in synchronism** is the difference between the actual value of the following axis and the value calculated from the leading axis actual values according to the rule of motion. The difference in synchronism can be polled from within the part program. See below.

Changes in the difference in synchronism

The mass inertia of the axis systems during acceleration can cause dynamic fluctuations in the difference in synchronism. The difference in synchronism is checked continuously and the tolerance values in the machine data used to produce interface signals.

The difference 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 "Fine synchronism" DB 31, ... DBX 98.0
IS "Coarse synchronism" DB 31, ... DBX 98.1

Difference > ..TOL_COARSE

As long as the difference in synchronism is greater than MD 37200: COUPLE_POS_TOL_COARSE, the gear is not synchronized and neither IS "Coarse synchronism" DB 31, ... DBX 98.1 nor IS "Fine synchronism" DB 31, ... DBX 98.0 is active. Instead, IS "Synchronization in progress" DB 31, ... DBX 99.4 is displayed.

Difference < ..TOL_COARSE

If the difference in synchronism is less than MD 37200: COUPLE_POS_TOL_COARSE, then IS "Coarse synchronism" " DB 31, ... DBX 98.1 is active at the interface and IS "Synchronization in progress" DB 31, ... DBX 99.4 is canceled.

Difference < ..TOL_FINE

If the difference in synchronism is less than MD 37210: COUPLE_POS_TOL_FINE, the IS "Fine synchronism" DB 31, ... DBX 98.0 is active at the interface.

Difference 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:

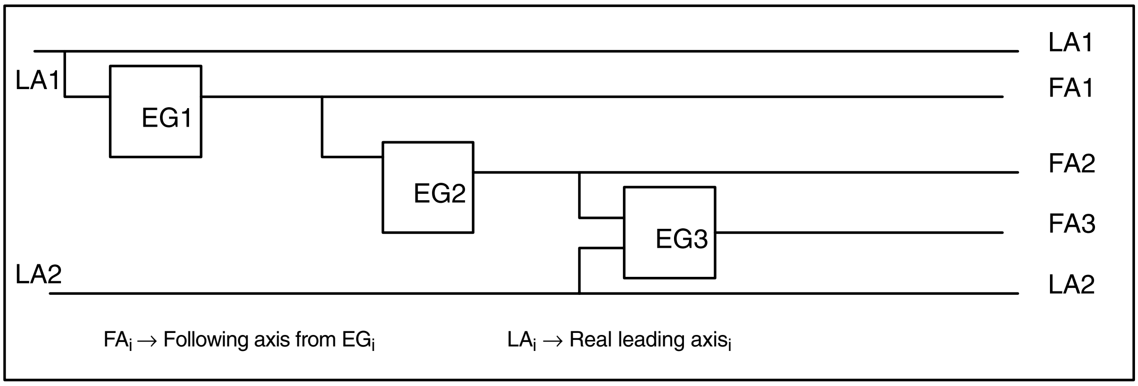


Fig. 2-8 Three-level EG cascade

According to the definition given, the difference in synchronism of following axis FA3 in the example below is determined by the value of following axis $FA3_{Act}$ and the value of leading axis $FA2_{Act}$ and $LA2_{Act}$, but not by $LA1_{Act}$ and $FA1_{Act}$.

If FA2 is not a real axis, the actual value $FA2_{Act}$ is not available. In this case, the **setpoint** of the axis derived solely from the leading axis value $FA1_{Act}$ must be used instead of the actual value of the setpoint of the axis.

Other signals

If an EGON(), EGONSYN() or EGONSYNE() block is encountered in the main run, the signal “Coupling active” is set for the following axis. If the following axis is only overlaid, the signals “Coupling active” and “Axis override” are set. If EGON(), EGONSYN() or EGONSYNE() is active and the following axis is also overlaid, the signals “Coupling active” and “Axis override” are also set.

IS “Following spindle active” DB31, ... DBX 99.1 Coupling active,

IS “Overlaid motion” DB31, ... DBX98.4 Axis is overlaid,

IS “Enable following axis override” DB31, ... DBX26.4

In the case of the commands EGON() and EGONSYNE(), the “Enable following axis override” signal must be present for the gear to synchronize to the specified synchronization position for the following axis. If it is not present, alarm 16771 “Override movement not enabled” is issued. If the signal is present, the following axis travels to the synchronized position with the calculated acceleration and at the velocity set for the approach mode.

Further monitoring signals

Machine data MD 37550: EG_VEL_WARNING
can be used to specify a % share of the velocities and accelerations in
MD 32000: MAX_AX_VELO
MD 32300: MAX_AX_ACCEL

with reference to the following axis, at which interface signals:
IS "Velocity warning threshold" DB 31, ... DBX 98.5
IS "Acceleration warning threshold" DB 31, ... DBX 98.6
are generated. The monitoring signals can be used as trigger criteria for emergency retraction. See 2.6.5.

Machine data MD 37560: EG_ACC_TOL see Chapter 4 can be used to define a % share in relation to
MD 32300: MA_MAX_AX_ACCEL
of the following axis. If this value is exceeded, the signal
IS "Axis accelerating" DB 31, ... DBX 99.3
is generated.

Scanning the synchronism difference value

The result of the synchronism difference calculation can be read as an amount in the part program with system variable $VA_EG_SYNCDIFF. The relevant value with sign is available from SW 6.4 in the system variables $VA_EG_SYNCDIFF_S. The following meanings apply:

– Negative value (leading axis and following axis in positive direction of movement):
  The following axis is trailing its calculated setpoint position.
– Positive value (leading axis and following axis in positive direction of movement):
  The following axis is ahead of its calculated 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 part program block change is to be executed:

Data entries are made using string parameters with the following meanings:

"NOC"	Immediate block change
"FINE"	Block change if "Fine synchronism" is present.
"COARSE"	Block change if "Coarse synchronism" is present
"IPOSTOP"	Block change if "Setpoint synchronism" is present.

Note

When programmed in activation calls EGON, EGONSYN, EGONSYNE, each of the above strings can be abbreviated to the first two characters.

If no block change has been defined for the EG axis group and none is currently specified, "FINE" applies.

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 part 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 synchronized 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 part 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 all 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 for the electronic gear must be **grouped in a single block** in a part program.
All electronic gear commands trigger a **preprocessing stop**, with the exception of

– EGON
– EGONSYN
– EGONSYNE activation commands.

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. (See EGON, EGONSYN, EGONSYNE).

Preconditions for defining an EG axis grouping:
No existing axis coupling may already be defined for the following axis. (If necessary, an existing coupling must be deleted first with EGDEL.)
EGDEF triggers a preprocessing stop with an error message.

For an example of how to use the EG function for gear hobbing, please see Section 6.2.

EGDEF

The gear definition with EGDEF should also be used unaltered when one or more leading axes affect the following axis via a curve table.

The variant to featuring the addition of a non-linear coupling via curve tables is illustrated in an extended example in Subsection 6.2.2.

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.

Where:
FA Following axis
The next block is switched in according to the block change mode:

"NOC"	Immediate block change
"FINE"	Block change on "Fine synchronism"
"COARSE"	Block change on "Coarse synchronism"
"IPOSTOP"	Block change on setpoint synchronism

LA_i	Axis identifier for leading axis i
Z_i	Numerator for coupling factor for leading axis i
N_i	Denominator for coupling factor for 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:

1. EGONSYN

EGONSYN(FA, block change mode, SynPosFA, LA_i, $SynPosLA_i$, Z_LA_i, N_LA_i)
where:
FA Following axis
Block change mode:

"NOC"	Immediate block change
"FINE"	Block change on "Fine synchronism"
"COARSE"	Block change on "Coarse synchronism"
"IPOSTOP"	Block change on setpoint synchronism

SynPosFA	Synchronized position of following axis
LA_i:	Axis identifier for leading axis i
SynPosLAi:	Synchronized position for leading axis i
Zi:	Numerator for coupling factor for leading axis i
Ni:	Denominator for coupling factor for 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. (This is what is known as *relative synchronization*: e.g. the next tooth gap after “centering”).

The synchronized position is only approached if IS “Enable following axis override” DB30 (+ axis number), DBX 26.4 is issued for the following axis. Instead the program is stopped at the EGONSYN block and the self-clearing alarm 16771 is issued until the above mentioned signal is set.

2. EGONSYNE

EGONSYNE(FA, block change mode, SynPosFA, approach mode, LA_i, $SynPosLA_i$, Z_LA_i, N_LA_i)

where:

FA Following axis

Block change mode:

“NOC” Immediate block change
“FINE” Block change on “Fine synchronism”
“COARSE” Block change on “Coarse synchronism”
“IPOSTOP” Block change on setpoint synchronism

SynPosFA Synchronized position of following axis

Approach mode:

“NTGT” NextToothGapTime optimized
Next gap is approached time-optimized
(default setting effective unless specified otherwise)

“NTGP” NextToothGapPath optimized
Next gap approach path-optimized

“ACN” AbsoluteCoordinateNegative, absolute dimension
Rotary axis travels in negative direction of axis motion

“ACP” AbsoluteCoordinatePositive, absolute dimension
Rotary axis travels in positive direction of axis motion

“DCT” DirectCoordinateTime optimized, absolute dimension
Rotary axis travels time-optimized to
programmed synchronized position

“DCP” DirectCoordinatePath optimized, absolute dimension
Rotary axis travels path-optimized to
programmed synchronized position

LA_i: Axis identifier for leading axis i
SynPosLAi: Synchronized position of leading axis i
Zi: Numerator for coupling factor for leading axis i
Ni: Denominator for coupling factor for 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 time-optimized and path-optimized traversing modes are identical.

The table below shows the target positions and traversed paths with direction marker (in brackets) for the particular approach modes:

Programmed synchronized position Fa-SysPos	Position of the following axis before EGONSYNE	Traversing mode NTGT/ NTGP	Traversing mode DCT/ DCP	Traversing mode ACP	Traversing mode ACN
110	150	182 (+32)	110 (–40)	110 (+320)	110 (–40)
110	350	326 (–24)	110 (+120)	110 (+120)	110 (–240)
130	0	346 (–14)	130 (+130)	130 (+130)	130 (–230)
130	30	58 (+28)	130 (+100)	130 (+100)	130 (–260)
130	190	202 (+12)	130 (–60)	130 (+300)	130 (–60)
190	0	334 (–26)	190 (–170)	190 (+190)	190 (–170)
230	0	14 (+14)	230 (–130)	230 (+230)	230 (–130)

Approach response for moving FA

The following axis moves at almost maximum velocity in the positive direction when the coupling is activated by EGONSYNE. The programmed synchronized position of the following axis is 110, the current position 150. This produces the two alternative synchronized positions 110 and 182 (see table above).

In the case of traversing mode NTGP (path-optimized), synchronized position 182 is selected independent of the current velocity. This has the shortest distance from the current position of the following axis. Traversing mode NTGT (time-optimized) considers the current 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).

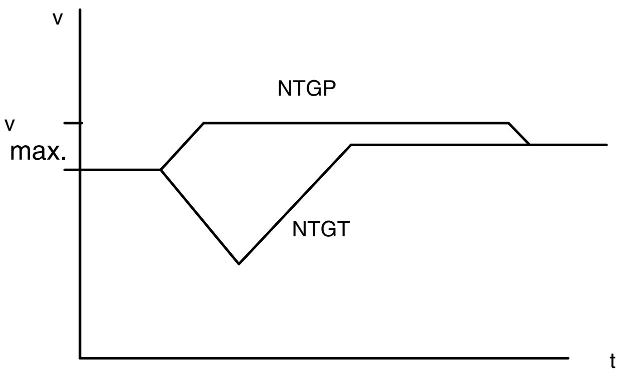


Fig. 2-9 Reaching the next tooth gap, FA path-optimized (top) vs. time-optimized (bottom)

Sample notations

EGONSYNE(A, "FINE", 110, "NTGT", B, 0, 2, 10)
couple A to B, synchronized position A = 110, B = 0, coupling factor 2/10, approach mode = NTGT

EGONSYNE(A, "FINE", 110, "DCT", B, 0, 2, 10)
couple A to B, synchronized position A = 110, B = 0, coupling factor 2/10, approach mode = DCT

EGONSYNE(A, "FINE", 110, "NTGT", B, 0, 2, 10, Y, 15, 1, 3)
couple A to B, and Y, synchronized position A = 110, B = 0, Y = 15,
coupling factor for B = 2/10, coupling factor for Y = 1/3,
approach mode = NTGT

With synchr.

The syntax specified above applies with the following different meanings. If a **curve table** is used for one of the leading axes, then:

N_i The numerator of the coupling factor for linear couplings must be set to 0. (nominator 0 would be illegal for linear couplings) Nominator zero tells the control that

Z_i Should be interpreted as the number of the curve table to be used. The curve table with the specified number must already be defined at POWER ON in accordance with Section 2.2.

LA_i The leading axis specified corresponds to the one specified for coupling via coupling factor (linear coupling).

2.4.4 Deactivating an EG axis grouping

Variant 1

An active EG axis grouping can be deactivated in a number of ways.

EGOFS(following axis)

The electronic gear is deactivated. The following axis is braked to standstill.
The call triggers a preprocessing stop.

Variant 2

The following parameterization of the command makes it possible to **selectively** control the influence of individual leading axes on the motion of the following axis.

EGOFS(following axis, leading axis 1, ... leading axis 5)

Note

At least one leading axis must be specified.

The influence of the specified leading axes on the following axis is selectively inhibited.
This call triggers a preprocessing stop.

If the call still includes active leading axes, then the slave continues to operate under their influence. If the influence of all leading axes is excluded by this method, then the following axis is braked to a standstill.

If the command EGONSYN is deactivated selectively, no axis movement is performed.

Variant 3

EGOFC(following spindle)

The electronic gear is deactivated. The following spindle continues to traverse at the speed/velocity that applied at the instant of deactivation.
The call triggers a preprocessing stop.

Note

Call only available for following **spindles**.
A spindle identifier must be programmed for EGOFC.

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 coupling definition of the axis grouping is deleted.
Additional axis groupings can be defined using EGDEF until the maximum number of simultaneously activated axis groupings is reached.

This call triggers a preprocessing stop.

2.4.6 Interaction between revolutional feedrate (G95) and electronic gear

The FPR( ) part program command can be used 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, RESET, operating mode change, block search

No coupling is active after POWER ON.

The status of active couplings is not affected by RESET or operating mode switchover.

For more detailed information about special states, please see Subsection 2.4.1.

Up to SW 6.4 during block searches, commands for switching, deleting and defining the electronic gear are not executed or collected but are skipped.

Block search with certain simulations

Certain active axis couplings can be simulated in the block search. As this does not apply to all possible coupling types, it is also possible to skip areas in which block searches cannot be made with an “automatic interrupt pointer”. The electronic gear can be simulated for all block search types under the following conditions:

- Simulation always takes place with setpoint coupling.
- No cross-channel leading axes may be disabled.
- Axis movements for which all real positions are known to the NC.

References: /K1/, Mode Group, Channel, Program Operation, Reset Response

2.4.8 System variables for electronic gear

Application

The following system variables can be used in the part 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		Preprocessor stop		Meaning, value	Cond. index
		Part prog.	Sync act.	Part prog.	Sync act.		
$AA_EG_ TYPE[a,b] (SW 5.2 and >)	INT	R		R		Type of coupling: 0: Actual-value coupling 1: Setpoint coupling	Axis identifier a: Following axis b: Leading axis
$AA_EG_ NUMERA[a,b] (SW 5.2 and >) ——— **(SW 6 and >)**	REAL	R		R		Numerator of coupl. factor KF KF = numerator/denominator Default: 0 —— Number of curve table when $AA_EG_DENOM[a,b] is 0.	Axis identifier a: Following axis b: Leading axis
$AA_EG_ DENOM[a,b] (SW 5.2 and higher) ——— **(SW 6 and higher)**	REAL	R		R		Denominator of coupl. fact. KF KF = numerator/denominator Default: 1 Denominator must be positive. —— Denominator is 0 if, instead of the numerator $AA_EG_NUMERA[a,b], the number of a curve table is specified.	Axis identifier a: Following axis b: Leading axis
$AA_EG_ SYN[a,b] (SW 5.2 and >)	REAL	R		R		Synchronized position for specified leading axis Default: 0	Axis identifier a: Following axis b: Leading axis
$AA_EG_ SYNFA[a] (SW 5.2 and higher)	REAL	R		R		Synchronized position for specified following axis Default: 0	Axis identifier a: Following axis
$AA_EG_BC[a]	STRING	R		R		Block change criterion for EG activation calls: EGON, EGONSYN: “NOC” Immediate “FINE” Fine synchronism “COARSE” Coarse synchronism “IPOSTOP” Setpoint synchronism	Axis identifier a: Following axis
$AA_EG_ NUM_LA[a]	INT	R		R		Number of leading axes defined with EGDEF. 0 if no axis has been defined as a following axis with EGDEF.	Axis identifier a: Following axis
$AA_EG_ AX[n,a]	AXIS	R		R		Axis identifier of leading axis whose index n has been specified.	Axis identifier n: Index of leading axis in EG grouping 0 ... 4 a: following axis

Table 2-1 System variables, R means: Read access possible

Name	Type	Access		Preprocessor stop		Meaning, value	Cond. index
		Part prog.	Sync act.	Part prog.	Sync act.		
$AA_EG_ ACTIVE[a,b] (SW 5.2 and higher)	BOOL	R		R		Determine the state of a leading axis at POWER ON: 0: Deactivated 1: Activated	Axis identifier a: Following axis b: Leading axis
$VA_EG_ SYNCDIFF[a]	REAL	R	R	R		Actual value of synchronism difference. The comparison with MD $MA_COUPLE_POS_TOL _COARSE and _FINE supplies interface signals.	Axis identifier a: Following axis

2.5 Dynamic response of following axis

2.5.1 Programmable axis couplings

Couplings

The previous subsections (Descriptions of Functions for S3 and T3) describe axis couplings in which a following axis is moved depending on one or more leading axes/spindles. The response of these coupled axes can be controlled by the user both via machine data and using program commands.

Table 2-2 Programming of axis couplings is possible in

Coupling	In part program	In synchronized actions
Tangential correction	x	
Coupled motion	x	x
Master value coupling	x	x
Electronic gear	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

Reducing or increasing dynamics limits

In SW 6.4 and higher, the dynamics limits specified above may be reduced or increased using commands from the **part program** or from the **synchronized actions**.

The acceleration characteristics set and the dynamics offsets set change the duration for synchronization between following and leading axes during acceleration operations as follows:

- "Dynamics reduction" increases the synchronism difference
  Monitoring from master to slave value may exceed the permissible range by **a longer** time.
- "Dynamics increase" reduces the synchronism difference
  Monitoring from master to slave value may exceed the permissible range by **a shorter** time.

!

Important

The user must restore the technological synchronization between machining and the synchronism difference.

Relevant limits in percent

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 area of validity are rejected with alarm 14811.

The relevant limits are then:

MD	Command	Relevant limit for
32000: MAX_AX_VELO[Ax]	VELOLIMA[Ax]	Axial velocity of FA: MAX_AX_VELO[A] * VELOLIMA[A]
32300: MAX_AX_ACCEL[Ax]	ACCLIMA[Ax]	Axial acceleration of FA: MAX_AX_ACCEL[A]*ACCLIMA[A]

Only BRISKA, i.e. abrupt axis acceleration, is available for the following axis. Acceleration modes SOFTA and DRIVEA are not available for the following axes described.

Furthermore, it is also possible to configure the positions controller as a PI controller.

Caution

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, Velocities, Setpoint/Actual Value Systems, Control

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/end of program

Channel-specific dynamic values for Reset, i.e. the values of VELOLIMA and ACCLIMA are set via MD 22410: F_VALUES_ACTIVE_AFTER_RESET.

MD 22410: F_VALUES_ACTIVE_AFTER_RESET=FALSE, the values for VELOLIMA(FA) and ACCLIMA(FA) are set to 100%.

MD 22410: F_VALUES_ACTIVE_AFTER_RESET=TRUE, the values set last are retained. This response also applies for dynamics offsets set using static synchronized actions.

If this is not the case despite the setting F_VALUES_ACTIVE_AFTER_RESET=FALSE, the dynamics offset must be applied again using an IDS synchronized action or permanent writing.

References: /FBSY/, Synchronized Actions

2.5.2 Examples for limits for ELG and master value coupling

Electronic gear

Axis 4 is coupled to X via an electronic gear coupling. The acceleration capability of the following axis is limited to 70% of maximum acceleration. The maximum permissible velocity is limited to 50% of maximum velocity. After POWER ON, the maximum permissible velocity is set to 100% again.

```
......
N120 ACCLIMA[AX4]=70
N130 VELOLIMA[AX4]=50         ; Reduced velocity
.....
N150 EGON(AX4, “FINE”, X, 1, 2)
.....
N200 VELOLIMA[AX4]=100        ; Full velocity
.....
```

Master value coupling

Axis 4 is coupled to X via a master value coupling. The acceleration capability of the following axis is limited to 80% of maximum acceleration.

```
......
N120 ACCLIMA[AX4]=80          ; 80 %
N130 LEADON(AX4, X, 2)        ; Activate coupling
.....
```

Master value coupling with synchronized action

Axis 4 is coupled to X via a master value coupling. The acceleration response is limited to 80% by static synchronized action 2 from position 100.

```
......
N120 IDS=2 WHENEVER $AA_IM[AX4] > 100 DO ACCLIMA[AX4]=80
N130 LEADON(AX4, X, 2)
.....
```

2.5.3 System variables for programmable following axis dynamics

For geometry axis, channel axis, machine axis and spindle axis types, the following readable system variables are available in part programs and synchronized actions:

Preprocessing

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]	%

Main run

Reading the main run variable implicitly triggers a preprocessing stop.

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

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 feed-forward 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 5 and lower, “Extended stop and retract” provides the following partial reactions:

- “Extended stop”
  (drive-independent or externally driven) is a defined, delayed stop.
- “Retract”
  (drive-independent 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”
  (drive-independent) 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 outage or similar failure.
- “Extended stop” **(NC-controlled)** is a defined, “delayed” and “contour-friendly” stop.

- “Retract” **(NC-controlled)** 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-controlled reactions with ...

NC-controlled 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-controlled *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-controlled *stop*, path interpolation also continues to be processed in the same way as an electronic coupling for a definable period (MD 21380: ESR_DELAY_TIME1), although an alarm, for example, with motion stop is present. It is then decelerated on the path as for NC-STOP.

For NC-controlled 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 via $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 cannot trigger an **EMERGENCY STOP** autonomously. As the EMERGENCY STOP signal is fed from the PLC to the NCK, the PLC is able to actively trigger an ESR if required (e.g. via $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 Drive-independent reactions

Drive-independent reactions are defined axially, that is, if activated each drive processes its stop and retract request independently. There is **no** interpolatory coupling of axes or coupling adhering to the path on stop/retract (only for control management).
The reference to the axes is time-controlled.

During and after execution of drive-independent 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: "Drive-independent stop/retract triggered" indicates this.

Note:
If retraction is not triggered on the 611digital drive, no alarm 26110 is issued.

!

Important

Drive-independent stop and retract are "automatically" triggered (as in the event of a communication failure).

These drive-side reactions are therefore **cross-channel** functions. This means that if drive-independent stop and retract is triggered in one channel, the drives of another channel also produce the drive-independent stop/retract reactions configured (and just enabled) for them.

Drive-independent generator operation ESR_REACTION = 10

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

Drive-independent retraction ESR_REACTION = 11

Drive-independent retraction is:

- Configured, (MD 37500: ESR_REACTION=11; time specification and retract velocity are set in MD; see "Example: Using the drive-independent reaction" in Section 6.3)
- Enabled: System variable $AA_ESR_ENABLE
- Started: System variable $AN_ESR_TRIGGER

Drive-independent stop ESR_REACTION = 12

Drive-independent stopping is:

- Configured (configuration: MD 37500: ESR_REACTION=12)
- Enabled ($AA_ESR_ENABLE) and
- Started: System variable $AN_ESR_TRIGGER

Note

For drive-independent reactions, the behavior can be determined individually for each axis.

Example

For an example of how the drive-independent reaction can be used, please see Section 6.3.

Drive-independent generator operation with NC-controlled stop ESR_REACTION = 13

At the drive end, generator operation is active as for ESR_REAKTION=10.

In SW 6.4 and higher, generator operation is:

- Configured (configuration: MD 37500: ESR_REACTION=13);
- Enabled ($AA_ESR_ENABLE) and
- Activated with system variable $AC_ESR_TRIGGER for “extended stop (NC-controlled)”.

Note

Unlike ESR_REAKTION=10, instead of rapid deceleration of the generator axis, an NC-controlled extended stop takes place (see Subsection 2.6.3 ESR_REAKTION=22).

Generator operation remains active at the drive end. If the DC link voltage on the drive falls below the value entered in MD 1631: LINK_VOLTAGE_GEN_ON, generator operation is activated for DC link backup.

2.6.3 NC-controlled extended stop

Procedure

The schedule for extended stop is defined by the two 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, the axis continues to interpolate unhindered as programmed. Once the period set in MD 21380: ESR_DELAY_TIME1 has elapsed, interpolatory controlled braking (ramp stop).
The maximum time available for interpolatory controlled braking is the time set in MD 21381: ESR_DELAY_TIME2; once this period has elapsed, rapid deceleration with subsequent correction is initiated.

This schedule applies if for at least one of the axes applied by the NCU: MD 37500: ESR_REACTION > 20. If this condition is not met, the alarm reactions are not delayed. If no ESR is active in the cycle, the above-mentioned alarm reactions are delayed by one IPO cycle (it takes one IPO cycle to check whether ESR is active).

For all other axes not specified for which MD 37500: ESR_REACTION = 0, rapid deceleration with subsequent correction is initiated at the start of extended stop ($AC_ESR_TRIGGER = 1).

The processing of all commands, especially those that result in an axis stop (e.g. RESET, Stop, Stopall, StopByAlarm), as well as the standard alarm reactions STOPBYALARM and NOREADY, is delayed by the total of times ESR_DELAY_TIME1 and ESR_DELAY_TIME2.

An NC-controlled stop interacts with the electronic gear (see 2.4). It contains the (selective) switchover of the electronic gear to actual-value coupling if there is a fault on the leading axes, and also upholds interpolation and enables ("continue travel") during a period which can be specified in 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-controlled extended stop can be taken from the figure below.

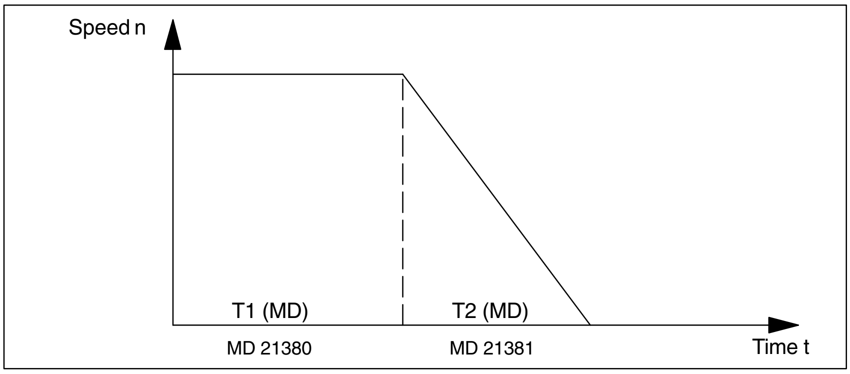


Fig. 2-10 Parameterizable/programmable control-driven shutdown

Note
For safety reasons, the total of times T1 and T2 should not exceed a maximum value, e.g. 1 second.

Effects of ESR_REACTION = 22

Effects for a path axis
If MD 37500: ESR_REACTION = 22 for a continuous path, the "extended stop" is transferred to **all path axes** of the channel.

Effects for a leading axis
If MD 37500: ESR_REACTION = 22 for a leading axis, the "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.

In order for ESR to function correctly, the enable signals must be set and remain set.

2.6.4 NC-controlled retraction

Initial conditions

The following are significant for NC-controlled retraction:

- The axes selected with POLFMASK and POLFMLIN
- The axis-specific positions defined with POLF
- The time slots ESR_DELAY_TIME1 and ESR_DELAY_TIME2
- Triggering by system variable $AN_ESR_TRIGGER
- The agreed ESR reaction MD 37500: ESR_REACTION
- G code LFPOS of modal 46th 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.

Prerequisite:
The retraction position must be programmed in the part program. The enable signals must be set and remain set for the retraction movement.

Fast retraction to the position defined with **POLF** is triggered via G code **LFPOS** of the modal 46th G code group.

The retracting movement configured with **LFPOS**, **POLF** for the axes selected with **POLFMASK** or **POLFMLIN** replaces the path motion defined for these axes in the part program.

During retraction:

- The axes defined in POLFMASK travel to the positions specified with POLF independently.
- The axes defined in POLFMLIN travel to the positions specified with POLF in **linear** relation.

The extended retraction (i.e. LIFTFAST/LFPOS initiated through $AC_ESR_TRIGGER) is **cannot be interrupted** and can only be terminated prematurely via an EMERGENCY STOP.

Velocity and acceleration limits for the axes involved in the retraction are monitored during the retraction motion. The retracting movement takes place with BRISK, i.e. without jerk limitation.

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.

Supplementary conditions

Lift fast is *not* used for axes, which:

- (due to axis replacement, individual axis state or similar) are not controlled by the channel
- are in speed-controlled mode (spindles)
- are interpolated as positioning spindles (SPOS/SPOSA).

Modulo rotary axes respond to lift fast as follows:

- For incremental programming of the target position, the latter is approached without modulo offset.
- For absolute programming, the target position is approached time-optimized with the use of modulo offsets. This is almost identical to positioning via the shortest path.

The retracting movement is interpolated linearly using the maximum acceleration and speed of the axes involved in POLFLIN.

Only **one linear** retraction is permitted in each **channel**. This means that several axis groupings, which approach their retraction positions in linear relation, cannot be created in the channel.

In **parallel** with linear retraction, additional axes can also use **POLFMASK** for independent axial retracting movement to their programmed retraction positions.

If axes are used in both POLFMASK and POLFMIN, make sure that the last state programmed is always active for retracting movement. This means that an axis previously activated with POLFMIN is removed from the linear relation following programming in POLFMASK and the retracting movement would then take place as an independent movement (see the examples in 6.3.6).

The parameters valid at the **triggering time** are decisive for the retracting movement. If one of these parameters (POLF, POLFMASK, POLFMLIN, Frame, etc.) changes during the retracting movement (e.g. due to a block change), this change does not affect the retracting movement that has already started.

Reactions to stop and axis enable signals

Stop characteristics for the retracting movement in response to the “Axial feed stop” and “Feed disable” signals are defined with the channel-specific machine data
MD 21204: LIFTFAST_STOP_COND

Bit0: Axial VDI signal FST DB31 DBB4.3
=0 Stop retracting movement for axial FST
=1 Do not stop retracting movement for axial FST

Bit1: Feed disable in channel DB21 DBB6.0
=0 Stop retracting movement for feed disable in channel
=1 Do not stop retracting movement for feed disable in channel

The “Axial feed stop” stop signal affects the entire retracting movement, i.e. all axis movements defined with POLFMASK and POLFMLIN are stopped.

The retracting movement is restarted by canceling the signals.

The “NC stop” signal, DB21 DBB7.3, has no effect on the retracting movement.

Programming

The destination for the retraction axis is programmed with the language command**:**

POLF

POLF[geo axis name | machine axis name] (**PO**sition **L**ift**F**ast).

POLF is modal.

POLF can also be programmed incrementally.
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 geo axis 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 **independently** to their position defined with **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 geo axes).

POLFMLIN

The language command **POLFMLIN([axisname1], [axisname2], ...)** enables selection of the axes that are to travel **in linear relation** to their position defined with **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 geo axes). Between initiating retraction and the linear relation being reached, a path is traveled, which, although deviating from the programmed path, does not quite reach the linear relation. If the constellation is unsuitable, this transition may last as far as the end points programmed with POLF. The control optimizes to the shortest possible transition.

General

The following sections are equally valid for POLFMASK and POLFMLIN.

The parameters valid at the triggering time are decisive for the retracting movement. If one of these parameters (G code, **POLF**, **POLFMASK** or **POLFMLIN**, Frame, etc.) changes during retraction (block change), this change does not affect the retracting movement that has already started.

Before lift fast to a fixed position can be activated via **POLFMASK** or **POLFMLIN**, a position must be programmed with **POLF** for the selected axes. No machine data is provided for presetting the values of **POLF**.

During interpretation of **POLFMASK** or **POLFMLIN**, alarm 16016 is issued if **POLF** has not been programmed.

If retraction is activated, the position for retraction can still be changed. However, it is no longer possible to change the coordinate system and an attempt is rejected with an alarm 16015.

If **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, lift fast must first be deactivated with **POLFMASK()** or **POLFMLIN()** and then **POLF** used to carry out programming in the new coordinate system.

Deactivate lift fast

POLFMASK() or **POLFMLIN()** without specifying an axis **deactivates** lift fast for all axes activated in that enable call.

POLFMASK / POLFMLIN interactions

The last data entered for a specific axis in one of the two instructions applies. These include:

```
N200 POLFLIN(X, Y, Z)          ; Retraction in linear relation
                               ; activated for axes X, Y and Z
...
N300 POLFMASK(Z)               ; Independent retraction activated for
                               ; axis Z. Axis Z is removed from the linear
                               ; retraction grouping with X and Y.

N500 POLFMLIN(X, Z)            ; Retraction in linear relation
                               ; for X and Z, deleting the independent
                               ; retraction previously activated with
                               ; POLFMASK for axis Z. The retraction
                               ; of axis Y activated previously with
                               ; POLFMLIN is also deleted.
                               ; This means that only X and Z
                               ; continue with retracting movement.
```

Part program start

The positions programmed with **POLF** and the activation by **POLFMASK** and **POLFMLIN** are deleted when the part program starts up. This means that the user *must* program the values for **POLF** and the selected axes (**POLFMASK, POLFMLIN**) in each part program.

Examples

Application examples for parameterizing with several axes and incremental programming appear 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 that can be read in **synchronized actions** are available as error sources. These include:

– 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

Reference: /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 control-internal digital output image that can be read back ($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-controlled 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 synchronized actions** can be used to trigger specific reactions based on sources. Logic operations of all relevant sources by means of static synchronized 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 synchronized actions are effective in all operating modes. For a detailed description of how to use synchronized actions, please see:

References: /FBSY/ Description of Functions Synchronized Actions
/PGA/ Programming Guide Advanced
(Synchronized 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

$AA_ESR_ENABLE
The generator operation, stop and retract functions are **enabled** by setting the associated control signal ($AA_ESR_ENABLE[axis]). This control signal can be modified by synchronized actions, by the part 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 synchronized 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).
- Drive-independent stop and/or retract can also be triggered by part programs/synchronized actions by setting the system variable $AN_ESR_TRIGGER (command to **all** drives). Precondition: Enable.

$AC_ESR_TRIGGER (NC-controlled)

- NC-controlled shutdown is activated with an appropriate parameter setting MD 37500: ESR_REACTION = 22 by setting the control signal “$AC_ESR_TRIGGER”. Precondition: Enable.
- NC-controlled retraction is activated with an appropriate parameter setting MD 37500: ESR_REACTION = 21 and **POLF** and **POLFMASK** in the part program by setting the control signal “$AC_ESR_TRIGGER”. Precondition: Enable.

2.6.8 Power failure detection and bridging

Detection

A power failure can be detected if 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.

In the best-case scenario, 1/2 an IPO cycle should follow activation of the relay and 3 in the worst-case scenario.
This determines the time for mains failure detection:

Worst-case scenario	approx. 120 ms
Best-case scenario	approx. 15 ms

Limits of DC link overvoltage

The DC link is monitored for the following voltage limits:

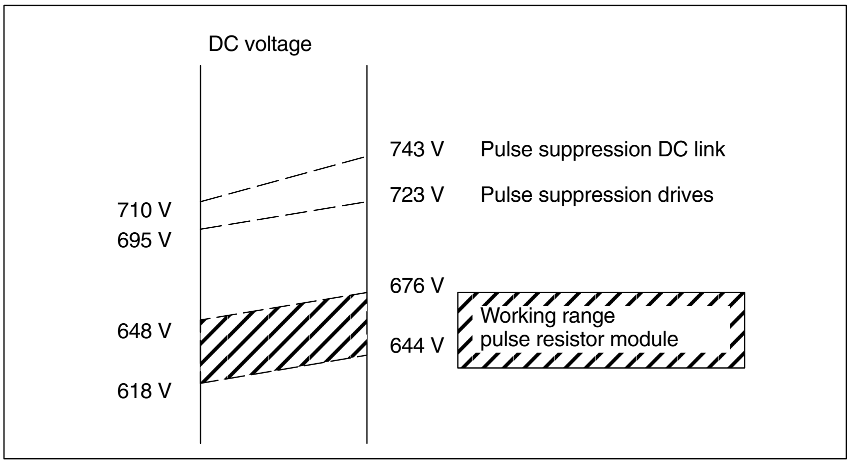


Fig. 2-11 Voltage level of SIMODRIVE 611D DC link

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 an internal error source for retraction. 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 280 V.

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). This is subject to the prerequisite that the synchronized actions are linked dependent on system variable $AA_ESR_STAT. This means that any ESR operations parameterized and programmed will be executed if 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.

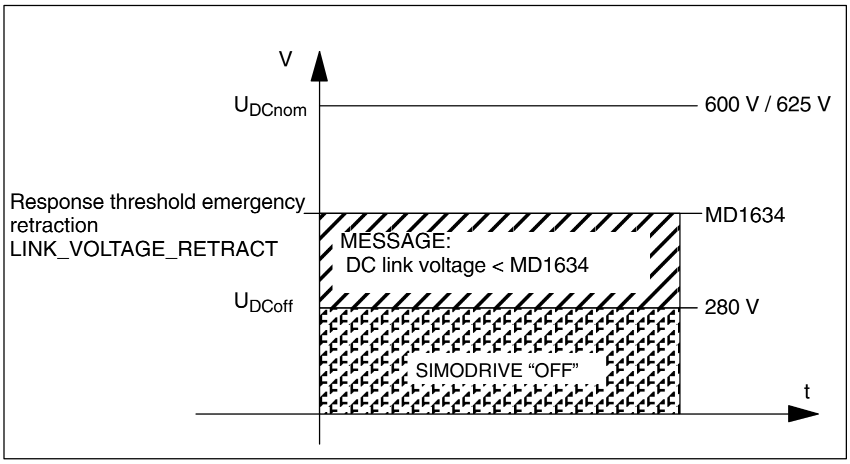


Fig. 2-12 DC link voltage monitoring SIMODRIVE 611D

Communication/ control failure

When the NC sign-of-life monitoring responds, a communication/control failure is detected on the drive bus and a drive-independent ESR is performed if appropriately configured.

Note

In SW 4.2 and higher, changing the default from 600 V to 0 V activates DC voltage measurement by default.
In order to ensure that older HW without the DC link measurement function starts up without errors, on these HW versions, “Calculate controller data MD 1161 (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 synchronized 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).

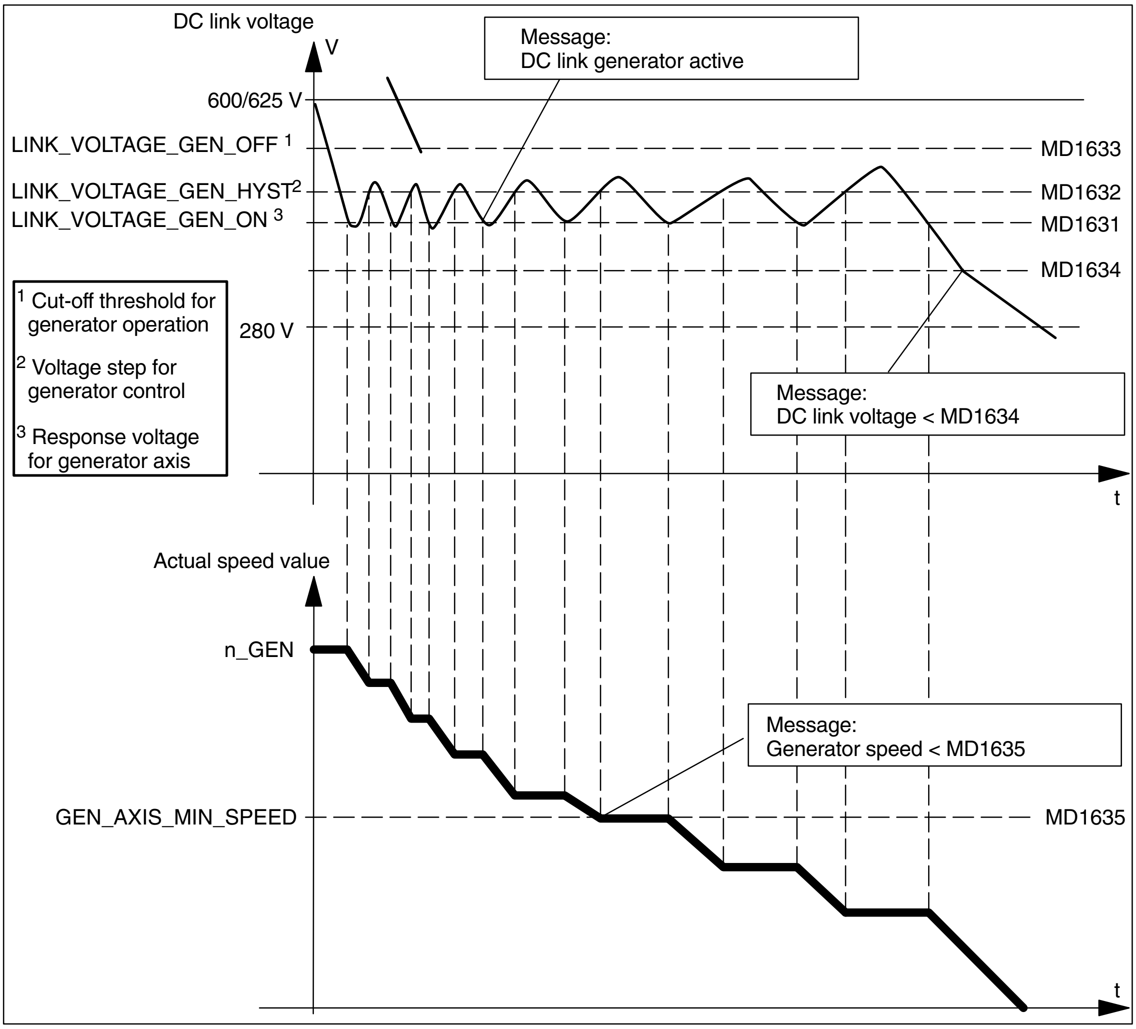


Fig. 2-13 Generator operation

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 feedback 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 error source for ESR.

2.6.10 Drive-independent 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.

Drive-independent 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 specified in MD 1637: GEN_STOP_DELAY.

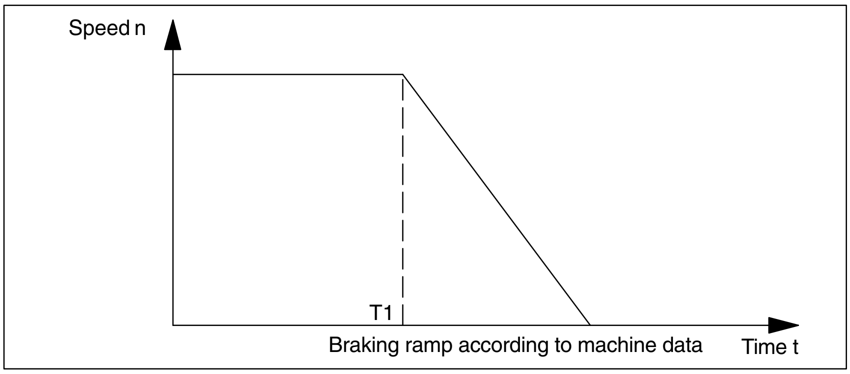


Fig. 2-14 Drive-independent stop SIMODRIVE 611D

Responses

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 Drive-independent retraction

Axes with digital 611D drives can (if configured and enabled) perform a retraction movement independently

- Even if the control fails (sign-of-life failure detection)
- If the DC link voltage drops below a warning threshold
- When triggered by system variable $AN_ESR_TRIGGER

execute a retraction movement independently.

The retract movement is conducted independently by the 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 setting made internally 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. drive-independent stop).

Note

- Drive-independent emergency retraction is only effective if the pulse suppression bit is set to OFF in MD1612: ALARM_REACTION_POWER_ON and MD1613: ALARM_REACTION_RESET.
- When emergency retraction is active, its parameters cannot be modified. 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 HMI Advanced
  – 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 115–230 V AC power supply for the central controller and the operator panel front or alternatively the 24 V DC power supply.
- Software components
  – System software: SW 5
  – 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 24 V 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:

$E = 1/2 * C * (U_{DC}^2 – U_{min}^2)$

E= Energy in watt seconds [Ws]
C= Overall capacitance of the DC link in Farad [F]
U_{DC}= Content 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:
If C= 6000µF (see table 2-4, 1st row) – 20% = 4800 µF
U_{DC}= 550 V (MD 1634)
U_{min}= 350 V (assumption):

$E = 1/2 * 4800\mu F *((550 V)^2 – (350 V)^2) = 432 Ws$

Under load, this energy is available for time

$t_{min} = E / P_{max} * \eta$

in order to initiate emergency retraction.
t_{min} = Backup time in milliseconds [ms]
P_{max} = Power in kilowatts [kW]
η = Efficiency of the drive unit

For the above example where:
E = 432 Ws
P_{max} = 16 kW (see table 2-4, 1st row)
η = 0.90,
the minimum backup time for emergency retraction is:

t_{min} = 432 Ws / 16 kW * 0.9 = 24.3 ms.

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 content (C_{max}) [Ws]	Energy content (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 IPO cycle onwards, the speed setpoints for the configured stop or retraction axes/spindles start to change.
The braking phase starts at this point (unless drive-independent stopping has been 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 16 kW unit under maximum load and minimum DC link capacitance it should be possible to execute an emergency retraction without generator operation. For this purpose, the interpolation cycle time may in theory be a maximum of 4.86 ms, i.e. in this case a maximum of 4 ms can be set.
It may be necessary to switch over to a more powerful NC CPU in order to achieve optimum conditions.

Drive-independent stop/retract

Drive-independent 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.

Note:
A POWER ON must be performed following drive-independent stop and retract.

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.
This does not usually occur in conjunction with a power failure.

Generator operation

Generator operation is possible in the event that the DC link power is insufficient for safe retraction (for a period of at least 3 interpolator 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. (Under normal conditions measurement every 4 ms.)

The energy stored in the drive

$$E = 1/2 * \Theta * \omega^2$$

where

Θ = Total mass moment of inertia
ω = 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 system response

POWER OFF/ POWER ON

If the retraction logic is stored in motion-synchronous actions, they are not yet active on POWER ON.
A logic operation, which is to be active after POWER ON, must be activated in an ASUB started by the PLC.
In the event of a drive-independent stop/retract being triggered, the drive software will subsequently need to be shut down then restarted.

In SW 6 and higher, the “Event-controlled program calls” function may be used instead of ASUB.

Mode change, NC stop

Static synchronized actions can be used for the logic operations (keyword IDS). They are not affected by a mode change or NC STOP/RESET.

Positioning of a command axis/spindle is aborted with channel stop.

ESR is available in AUTO, JOG and MDA modes.

Reset

Static synchronized actions are maintained after a RESET.

On RESET, the values programmed for **POLF** and the activation by **POLFMASK** are not deleted.

Part program start

On start of a part 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 part 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 coupling and traversing movements must remain valid even in the event of an 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 triggered"
  DB31, ... DBX98.7 is available as a checkback signal to the PLC.

If an alarm with cross-channel NOREADY reaction is issued during the active phase of the ESR
(i.e. NOREADY |
NCKREACTIONVIEW |
BAGREACTIONVIEW),
then ESR is triggered in **all** channels.

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 "drives, motors, encoders" axis/spindle components participating in "Extended stop and retract" must be operational. If one of these components fails, the full scope of the described reaction no longer applies. 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-synchronized actions, the runtime of the control for processing the cyclical interpretation of conditions in the synchronized 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 than the corresponding NC-controlled reaction (reason: when broadcast mode is activated for the drive, each drive-independent reaction becomes directly active).

Power ON

If drive-independent 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

3 Supplementary Conditions

There are no other supplementary conditions to note.

Notes

4 Data Descriptions (MD, SD)

4.1 General machine data

11660 MD number	**NUM_EG** Number of possible EG axis groupings	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 31
Changes effective after POWER ON	Protection level: 1 / 1	Unit: –
Data type: Byte	Applies as of 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 MD number	**MM_NUM_CURVE_TABS** Number of curve tables (SRAM)	
Default setting: 0	Minimum input limit: 0	Maximum input limit: plus
Changes effective after POWER ON	Protection level: 1/1	Unit: –
Data type: DWORD	Applies as of 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 MD number	**MM_NUM_CURVE_SEGMENTS** Number of curve segments (SRAM)	
Default setting: 0	Minimum input limit: 0	Maximum input limit: plus
Changes effective after POWER ON	Protection level: 1/1	Unit: –
Data type: DWORD	Applies as of 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	

18403 MD number	MM_NUM_CURVE_SEG_LIN Number of linear curve segments (SRAM)	
Default setting: 0	Minimum input limit: 0	Maximum input limit: –
Changes effective after POWER ON	Protection level: 2 / 2	Unit: –
Data type: DWORD	Applies as of SW: 7.3	
Meaning:	Number of linear curve segments in SRAM available NCK-wide. A curve table can consist of “normal” curve segments and linear segments. The number of “normal” curve segments in the SRAM is specified by MD 18402: MM_NUM_CURVE_SEGMENTS. These curve segments can accommodate polynomials. Linear curve segments can only accommodate straight lines. The linear curve segments are created in the battery-backed memory.	
Related to	MD 18402, MD 18409	
Additional references	PGA	

18404 MD number	MM_NUM_CURVE_POLYNOMS Number of curve table polynomials (SRAM)	
Default setting: 0	Minimum input limit: 0	Maximum input limit: plus
Changes effective after POWER ON	Protection level: 1/1	Unit: –
Data type: DWORD	Applies as of 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 MD 18402: MM_NUM_CURVE_SEGMENTS	

18406 MD number	MM_NUM_CURVE_TABS_DRAM Number of curve tables (DRAM)	
Default setting: 0	Minimum input limit: 0	Maximum input limit: plus
Changes effective after POWER ON	Protection level: 1/1	Unit: –
Data type: DWORD	Applies as of SW: 6.3	
Meaning:	Number of curve tables in DRAM available NCK-wide	
Related to	MD 18408, MD 18410	
Additional references	PGA	

18408 MD number	MM_NUM_CURVE_SEGMENTS_DRAM Number of curve segment (DRAM)	
Default setting: 0	Minimum input limit: 0	Maximum input limit: plus
Changes effective after POWER ON	Protection level: 1/1	Unit: –
Data type: DWORD	Applies as of SW: 6.3	
Meaning:	Number of segments for curve tables in DRAM available NCK-wide	
Related to	MD 18406, MD 18410	
Additional references	PGA	

18409 MD number	**MM_NUM_CURVE_SEG_LIN_DRAM** Number of linear curve segments (DRAM)	
Default setting: 0	Minimum input limit: 0	Maximum input limit: –
Changes effective after POWER ON	Protection level: 2 / 2	Unit: –
Data type: DWORD	Applies as of SW: 7.3	
Meaning:	Number of linear curve segments in DRAM available NCK-wide. A curve table can consist of “normal” curve segments and linear segments. The number of “normal” curve segments in the DRAM is specified by MD 18408: MM_NUM_CURVE_SEGMENTS_DRAM. These curve segments can accommodate polynomials. Linear curve segments can only accommodate straight lines. The linear curve segments are created in the dynamic memory.	
Related to	MD 18408	
Additional references	PGA	

18410 MD number	**MM_NUM_CURVE_POLYNOMS_DRAM** Number of curve table polynomials (DRAM)	
Default setting: 0	Minimum input limit: 0	Maximum input limit: plus
Changes effective after POWER ON	Protection level: 1/1	Unit: –
Data type: DWORD	Applies as of SW: 6.3	
Meaning:	Number or polynomials for curve tables in DRAM available for NCK-wide	
Related to	MD 18408, MD 18406	
Additional references	PGA	

4.2 Channel-specific machine data

20900 MD number	**CTAB_ENABLE_NO_LEADMOTION** Curve tables with jump of following axis			
Default setting: 0		Minimum input limit: 0		Maximum input limit: 2
Changes effective after RESET		Protection level: 2/7		Unit: –
Data type: BYTE			Applies as of SW: 6.3	
Meaning:	This MD configures how jumps of the following axis are processed in the curve tables. The following axis jumps if following axis motion is set in a segment of the curve table but the leading axis does not move. This type of following axis jump can either be programmed directly or simply occur internally in a controller. This type of segment can be generated in particular if a curve table is generated with active tool radius compensation. The following configuration options are possible: 0: No curve tables that contain a jump of the following axis are produced. If a jump of the following axis occurs, alarm 10949 (CTAB_NO_LEADMOTION) is output and program processing is aborted. This setting is compatible with older software versions. 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 output but program processing is not aborted. 2: Curve tables with jumps of the following axis are created without an alarm or message being issued.			

20905 MD number	**CTAB_DEFAULT_MEMORY_TYPE** Default memory type for curve tables			
Default setting: 0		Minimum input limit: 0		Maximum input limit: 1
Changes effective after RESET		Protection level: 2/7		Unit: –
Data type: BYTE			Applies as of SW: 6.3	
Meaning:	This MD sets the default memory type for curve tables: 0: Curve tables are created in buffered memory (SRAM). 1: Curve tables are created in dynamic memory (DRAM).			
Related to	MD 18400, 18402, 18404, 18406, 18408, 18410			

21204 MD number	**LIFTFAST_STOP_COND** Stop characteristics for fast retraction		
Default setting: 0	Minimum input limit: –	Maximum input limit: –	
Changes effective after NEW_CONF	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies as of SW: 6.4		
Meaning:	The MD defines the stop characteristics for the LIFTFAST motion for a variety of stop conditions. Bit0: Axial VDI signal FST DB31 DBB4.3 =0 Stop retracting movement for axial FST =1 Do not stop retracting movement for axial FST Bit1: Feed disable in channel DB21 DBB6.0 =0 Stop retracting movement for feed disable in channel =1 Do not stop retracting movement for feed disable in channel		
Additional references	Programming Guide Advanced		

21380 MD number	**ESR_DELAY_TIME1** Delay time (STOPBYALARM, NOREAD) for ESR axes		
Default setting: 0	Minimum input limit: –	Maximum input limit: plus	
Changes effective after NEW_CONF	Protection level: 2 / 7	Unit: s	
Data type: DOUBLE	Applies as of 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 Subsection 6.3.2		
Related to	ESR_DELAY_TIME2		

21381 MD number	**ESR_DELAY_TIME2** Time for interpolatory braking for ESR axes		
Default setting: 0	Minimum input limit: –	Maximum input limit: plus	
Changes effective after NEW_CONF	Protection level: 2 / 7	Unit: s	
Data type: DOUBLE	Applies as of SW: 6		
Meaning:	When the time $MC_ESR_DELAY_TIME1 expires, the time specified here for interpolatory braking ($MC_ESR_DELAY_TIME2) remains available. When the time $MC_ESR_DELAY_TIME2 expires, rapid deceleration with subsequent follow-up is initiated.		
Application example(s)	see Subsection 6.3.2		
Related to	ESR_DELAY_TIME1		

4.3 Axis-specific machine data

30132 MD number	**IS_VIRTUAL_AX** Axis is virtual axis	
Default setting: 0	Minimum input limit: ***	Maximum input limit: ***
Changes effective after NEW_CONF	Protection level: 2/7	Unit: –
Data type: BOOLEAN	Applies as of SW: 4.1	
Meaning:	Virtual axis. An axis that is interpolated in follow-up mode, too. (Technology electronic transfer, virtual and real master value.) This MD is equivalent to MD 30130: CTRLOUT_TYPE=4. Instead of MD 30130: CTRLOUT_TYPE=4, MD 30130: CTRLOUT_TYPE=0 and IS_VIRTUAL_AX=1 should be set.	
Related to	MD 30130: CTRLOUT_TYPE	

37160 MD number	**LEAD_FUNCTION_MASK** Functions for master value coupling	
Default setting: 0x01	Minimum input limit: 0	Maximum input limit: 0x10
Change valid after NEWCONF	Protection level: 2/2	Unit: –
Data type: DWORD	Applies as of SW: 6.4	
Meaning:	The machine data can be used to set axial master value coupling functions. This machine data is bitcoded: Bit 0: Dead time compensation for actual value coupling = 0 Dead time compensation for actual value coupling is not active = 1 Dead time compensation for actual value coupling is active.	
Related to		

<table>
<tr><td>37500
MD number</td><td colspan="3">ESR_REACTION
Reaction definition with extended stop and retract</td></tr>
<tr><td>Default setting: 0</td><td>Minimum input limit: 0</td><td colspan="2">Maximum input limit: 22</td></tr>
<tr><td>Changes effective after NEW_CONF</td><td>Protection level: 2/7</td><td colspan="2">Unit: –</td></tr>
<tr><td colspan="2">Data type: BYTE</td><td colspan="2">Applies as of SW: 5.1/ 6 and 6.4</td></tr>
<tr><td>Meaning:</td><td colspan="3">Selection of the reaction to be triggered via system variable “$AN_ESR_TRIGGER/$AC_ESR_TRIGGER”:
0 = No reaction (or exclusively external reaction through synchronized action programming of fast digital outputs).
Drive MD 1636: RETRACT_AND_GENERATOR_MODE is set to 0.
10 = Drive-independent generator operation
Drive MD 1636: RETRACT_AND_GENERATOR_MODE is set to 6.
11 = Drive-independent retraction axis
Drive MD 1636: RETRACT_AND_GENERATOR_MODE is set to 4.
12 = Drive-independent stop axis
(as is the case in the event of a communication failure, 11 and 12 are activated together in the drive by broadcast to all drives)
Drive MD 1636: RETRACT_AND_GENERATOR_MODE is set to 2.

Selection of the reaction to be triggered via system variable “$AC_ESR_TRIGGER”:
13 = Drive-independent generator axis with NC-controlled stop (SW 6.4 and higher)
At the drive end, generator operation is active as for ESR_REAKTION=10.
Drive MD 1636: RETRACT_AND_GENERATOR_MODE is set to 6.
21 = NC-controlled retraction axis (SW 6 and higher)
Drive MD 1636: RETRACT_AND_GENERATOR_MODE is set to 5.
22 = NC-controlled standstill axis (SW 6 and higher)
All axes involved in IPO or EG are brought to a smooth stop even without this parameter setting. This parameter is used to configure the appropriate drive-independent response to a communication failure or DC link undervoltage: (21 and 22 contain drive-independent standstill and retraction exclusively for communication failure or DC link undervoltage).
Drive MD 1636: RETRACT_AND_GENERATOR_MODE is set to 3.
If the option “Extended stop and retract” (ESR) is not enabled, the values are reset to 0.</td></tr>
<tr><td>Related to ....</td><td colspan="3"></td></tr>
</table>

37550 MD number	**EG_VEL_WARNING,** Warning threshold for interface signals				
Default setting: 90		Minimum input limit: 0		Maximum input limit: 100	
Changes effective after NEW_CONF			Protection level: 2 / 4		Unit: %
Data type: REAL				Applies as of SW: 5	
Meaning:	Threshold value for VDI signals A velocity warning (signal) is output if an EG axis coupling is active and the current **speed** of the axis reaches the percentage of the maximum velocity set in MD 32000: $MA_MAX_AX_VELO. An acceleration warning (signal) is output if an EG axis coupling is active and the current **acceleration** of the axis reaches the percentage of the maximum acceleration set in MD 32300: $MA_MAX_AX_ACCEL.				
Related to	MD 32000: $MA_MAX_AX_VELO, MD 32300: $MA_MAX_AX_ACCEL				

37560 MD number	**EG_ACC_TOL** Threshold value for VDI signal				
Default setting: 25		Minimum input limit: 0		Maximum input limit: –	
Changes effective after NEW_CONF			Protection level: 2/4		Unit: %
Data type: REAL				Applies as of SW: 5	
Meaning:	Threshold value for VDI signal "**Axis accelerating**" This signal is set if the acceleration rate reaches the specified percentage of maximum acceleration.				
Related to	MD 32300: $MA_MAX_AX_ACCEL				

4.4 Axisspecific setting data

43100 SD number	**LEAD_TYPE** Master value type				
Default setting: 1		Minimum input limit: 0		Maximum input limit: 2	
Changes effective IMMEDIATELY			Protection level: 7/7		Unit: –
Data type: DWORD				Applies as of SW: 4.1	
Meaning:	Defines which value is to be used as the master value: 0: Actual value 1: Setpoint 2: Simulated master value				

43102 SD number	**LEAD_OFFSET_IN_POS** Master value offset				
Default setting: 0		Minimum input limit:		Maximum input limit:	
Changes effective IMMEDIATELY			Protection level: 7/7		Unit: –
Data type: DOUBLE				Applies as of SW: 4.1	
Meaning:	Offset of master value before use for coupling.				
Related to	SD 43104: LEAD_SCALE_IN_POS SD 43106: LEAD_OFFSET_OUT_POS SD 43108: LEAD_SCALE_OUT_POS				

43104 SD number	**LEAD_SCALE_IN_POS** Master value scaling				
Default setting: 1		Minimum input limit:		Maximum input limit:	
Changes effective IMMEDIATELY			Protection level: 7/7		Unit: –
Data type: DOUBLE				Applies as of SW: 4.1	
Meaning:	Scaling of master value before use for coupling.				
Related to	SD 43102: LEAD_OFFSET_IN_POS SD 43106: LEAD_OFFSET_OUT_POS SD 43108: LEAD_SCALE_OUT_POS				

43106 SD number	**LEAD_OFFSET_OUT_POS** Curve table offset				
Default setting: 0		Minimum input limit:		Maximum input limit:	
Changes effective IMMEDIATELY			Protection level: 7/7		Unit: POSN
Data type: DOUBLE				Applies as of SW: 4.1	
Meaning:	Offset of curve table before use for coupling.				
Related to	SD 43102: LEAD_OFFSET_IN_POS SD 43104: LEAD_SCALE_IN_POS SD 43108: LEAD_SCALE_OUT_POS				

43108 SD number	**LEAD_SCALE_OUT_POS** Curve table scaling				
Default setting: 1		Minimum input limit:		Maximum input limit:	
Changes effective IMMEDIATELY			Protection level: 7/7		Unit:
Data type: DOUBLE				Applies as of SW: 4.1	
Meaning:	Scaling of function value of curve table.				
Related to	SD 43102: LEAD_OFFSET_IN_POS SD 43104: LEAD_SCALE_IN_POS SD 43106: LEAD_OFFSET_OUT_POS				

■

Notes

5 Signal Descriptions

5.1 Signals from axis/spindle

DB 31 – DBB 26.4 Data block	Active following axis overlay Signal(s) from NC (PLC → NCK)	
Edge evaluation: No	Signal(s) updated: Cyclic	Signal(s) valid from SW: 5.1
Signal state 1 or signal transition 0 ——> 1	An additional traversing motion can be overlaid on the following axis. This signal is required for on-the-fly synchronization of leading and following axes. As long as the "enable following axis overlay" signal is set to 1, the following axis selected with EGONSYN in the EG coupling group travels to synchronization. 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. If the "enable following axis overlay" signal has not been set for the following axis, the axis will not travel to synchronization. Instead, the program is stopped at the EGONSYN block and the self-clearing alarm 16771 is issued until the "enable following axis overlay" signal is set to 1.	

DB 31 – DBB 99.3 Data block	Axis accelerated Signal(s) from NC (NC → PLC)	
Edge evaluation: No	Signal(s) updated: Cyclic	Signal(s) valid from SW: 5.1
Signal state 1 or signal transition 0 ——> 1	If the acceleration rate of the following axis in the electronic gear axis grouping reaches or exceeds the % share entered in MD 37560: EG_ACC_TOL of the acceleration rate set in MD 32300: MAX_AX_ACCEL, 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 Signal(s) from NC (NC → PLC)	
Edge evaluation: No	Signal(s) updated: Cyclic	Signal(s) valid from SW: 5.1
Signal state 1 or signal transition 0 ——> 1	If the velocity of the following axis in the electronic gear axis grouping reaches or exceeds the % share entered in MD 37550: EG_VEL_WARNING of the velocity set in MD 32000: MAX_AX_VELO, 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 – **DBB 98.6** Data block	**Acceleration warning threshold** Signal(s) from NC (NC → PLC)	
Edge evaluation: No	Signal(s) updated: Cyclic	Signal(s) valid from SW: 5.1
Signal state 1 or signal transition 0 ––> 1	If the acceleration rate of the following axis in the electronic gear axis grouping reaches or exceeds the % share entered in MD 37550: EG_VEL_WARNING of the acceleration rate set in MD 32300: MAX_AX_ACCEL, 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 – **DBX 7** Data block	**ESR reaction is triggered** Signal(s) from NC (NC → PLC)	
Edge evaluation: No	Signal(s) updated: Cyclic	Signal(s) valid from SW: 5.1
Signal state 1 or signal transition 0 ––> 1	Status signal The VDI signal "ESR reaction triggered" is available to the PLC as a check-back signal. The signal is set if $AA_ESR_STAT > 0, i.e. if generator mode, standstill or retraction are active, DC link undervoltage is detected or the generator minimum speed is undershot.	
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. Writing in $A_DBB allows the PLC to extensively influence the execution of the ESR reactions, if appropriate access is also integrated into the synchronized actions. On the 840D, the PLC has a "locking influence" on the ESR response. On the 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 of curve via a coupling from Y to X
N140 LEADON(Y,X,2)
N150 X360
N160 X0
N170 LEADOF(Y,X)

N180 DEPPOS = CTAB(75.0, 2, GRADIENT)  ; Reading of table position at
master value 75.0 from the curve table with table number 2

N190 G0 X75 Y=DEPPOS   ; Positioning of leading and following axis

N200 LEADON(Y,X,2)          ; Once the coupling has been activated, the
                            ; following axis does not need to be synchronized
N210 G1 X110 F1000
N220 LEADOF(Y,X)
N190 M30
```

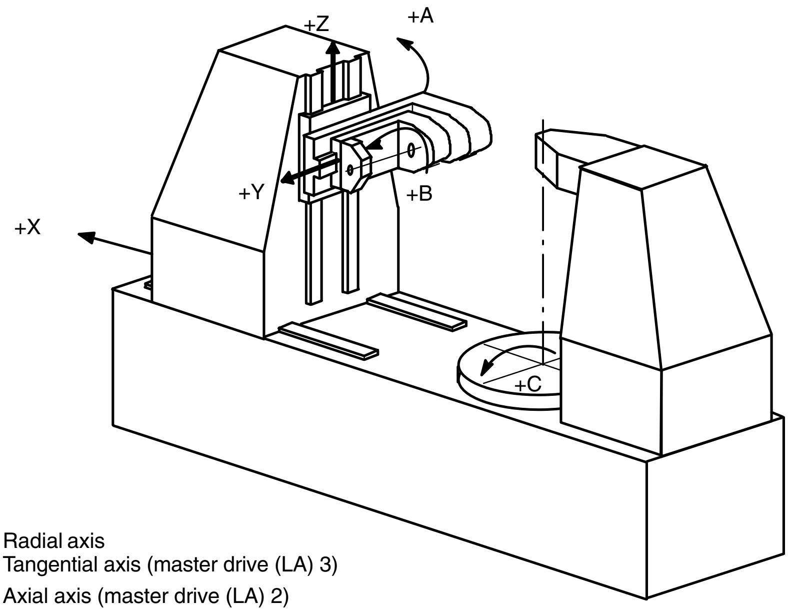
6.2 Electronic gear for gear hobbing

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. These are:

- The rotary motion of the workpiece table (C) and hobbing cutter (B).
- The axial axis (Z) for producing the feed motion over the entire workpiece width.
- The tangential axis (Y) for moving the hobbing cutter along its axis.
- The radial axis (X) for infeeding the cutter to depth of tooth.
- The cutter swivel axis (A) for setting the hobbing cutter in relation to the workpiece as a function of cutter lead angle and angle of inclination of tooth.



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:

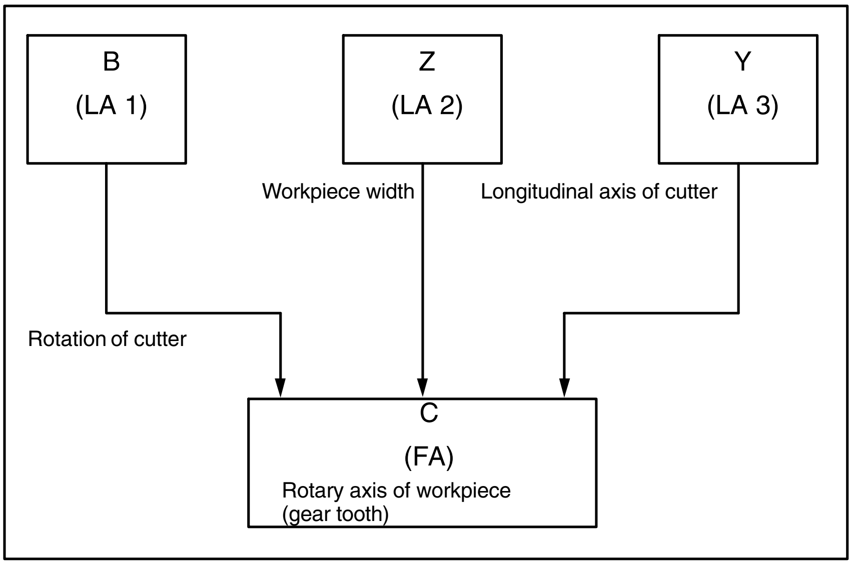


Fig. 6-2

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 part 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.

$$u_{dz} = \frac{\sin\beta^\circ}{m_n \cdot \pi} \cdot 360 \quad \left[\frac{\text{degrees}}{\text{mm}}\right]$$

$$u_{dy} = \frac{\cos\gamma^\circ}{m_n \cdot \pi} \cdot 360 \quad \left[\frac{\text{degrees}}{\text{mm}}\right]$$

with:

m_n	=	Normal module (in mm)
β°	=	Angle of inclined axis on gear wheel
γ°	=	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, z0, z2, Z, udz, z2, Y, udy, z2 )
```

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 teeth geometry component dependent on the Z axis.
This can be used for example 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.

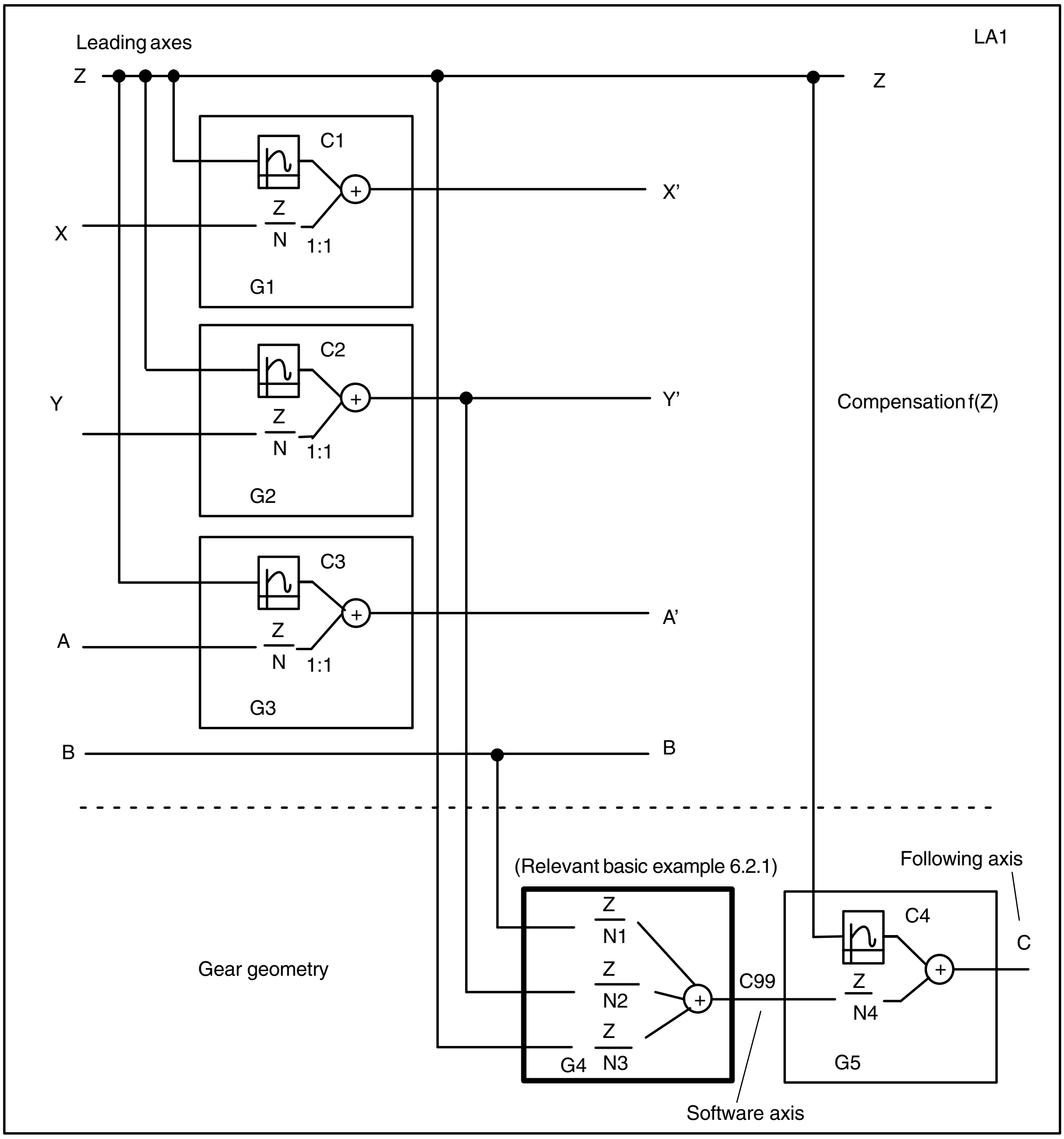


Fig. 6-3 Extended example with non-linear machine fault compensation and non-linear components on the tooth geometry

The following section of a part program is intended to illustrate the general concept; supplementary curve tables and gear wheel/machine parameters are still to be added. Components to be added are marked < ... >. 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 < ... >                    ; Define 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 < ... >                    ; Define 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 < ... >                    ; Define 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 < ... >                    ; Define 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)         ; Declaration of path via C2, setpoint coupling
N610     G1 F1000 Y10           ; Declaration of command component of Y
N620     EGONSYN(Y, "COARSE", <SynPosY>, Z, <SynPosY_Z>, 2, 0)        ; Activation of path via C2

N700     EGDEF(A, Z, 1)         ; Declaration of path via C3, setpoint coupling
N710     G1 F1000 A10           ; Declaration of command component of A
N720     EGONSYN(A, "FINE", <SynPosA>, Z, <SynPosA_Z>, 3, 0)          ; Activation of path via C3

; 1. 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 *    , 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 stage 1 as leading axis for stage 2,
                                ; setpoint coupling
N910                            ; Declaration of path 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, as in the case of an 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: /PGA1/, Lists of System Variables

(SW 7.1 and higher). 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 *      ; 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

Extract from MD:

```
; ************** Channel 1
CHANDATA(1)
; ************** Axis 1, "X"
$MC_AXCONF_GEOAX_NAME_TAB[0] = "X"
$MC_AXCONF_CHANAX_NAME_TAB[0] = "X"
$MC_AXCONF_MACHAX_USED[0] = 1
$MN_AXCONF_MACHAX_NAME_TAB[0] = "X1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX1] = 0
$MA_IS_ROT_AX[AX1] = FALSE
; *************** Axis 2, "Y"
$MC_AXCONF_GEOAX_NAME_TAB[1] = "Y"
$MC_AXCONF_CHANAX_NAME_TAB[1] = "Y"
$MC_AXCONF_MACHAX_USED[1] = 2
$MN_AXCONF_MACHAX_NAME_TAB[1] = "Y1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX2] = 0
$MA_IS_ROT_AX[AX2] = FALSE
; *************** Axis 3, "Z"
$MC_AXCONF_GEOAX_NAME_TAB[2] = "Z"
$MC_AXCONF_CHANAX_NAME_TAB[2] = "Z"
$MC_AXCONF_MACHAX_USED[2] = 3
$MN_AXCONF_MACHAX_NAME_TAB[2] = "Z1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX3] = 0
$MA_IS_ROT_AX[AX3] = FALSE
; *************** Axis 4, "A"
$MC_AXCONF_CHANAX_NAME_TAB[3] = "A"
$MC_AXCONF_MACHAX_USED[3] = 4
$MN_AXCONF_MACHAX_NAME_TAB[3] = "A1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX4] = 0
$MA_IS_ROT_AX[AX4] = TRUE
$MA_ROT_IS_MODULO[AX4] = TRUE
; *************** Axis 5, "B"
$MC_AXCONF_CHANAX_NAME_TAB[4] = "B"
$MC_AXCONF_MACHAX_USED[4] = 5
$MC_SPIND_DEF_MASTER_SPIND = 1
$MN_AXCONF_MACHAX_NAME_TAB[4] = "B1"
$MA_SPIND_ASSIGN_TO_MACHAX[AX5] = 1
$MA_IS_ROT_AX[AX5] = TRUE
```

```
$MA_ROT_IS_MODULO[AX5] = TRUE
; ************** Axis 6, “C”
$MC_AXCONF_CHANAX_NAME_TAB[5] = “C”
$MC_AXCONF_MACHAX_USED[5] = 6
$MN_AXCONF_MACHAX_NAME_TAB[5] = “C1”
$MA_SPIND_ASSIGN_TO_MACHAX[AX6] = 0
$MA_IS_ROT_AX[AX6] = TRUE
$MA_ROT_IS_MODULO[AX6] = TRUE
; ************** Axis 10, “C99”
$MC_AXCONF_CHANAX_NAME_TAB[9] = “C99”
$MC_AXCONF_MACHAX_USED[9] = 10
$MA_SPIND_ASSIGN_TO_MACHAX[AX10] = 0
$MA_IS_ROT_AX[AX10] = TRUE
$MA_ROT_IS_MODULO[AX10] = TRUE
```

6.3 ESR

6.3.1 Use of drive-independent reaction

Example configuration

- Axis A (spindle) must operate as generator drive.
- in the event of an error, axis X must retract by 10 mm at maximum speed, and
- axes Y and Z must stop after a 100 ms delay to give the retraction axis time to cancel the mechanical coupling.

Parameterization

1. Enable the "Ext. stop and retract" and "Static synchronized actions" options.
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. velocity in HEX format
   MD1638: RETRACT_TIME[X]=10 ; mm/max. velocity 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
   ; rpm
4. Function enable (from part program or synchronized 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 rpm)
6. Formulate trigger condition as static synchronized action(s), e.g.:
   - dependent on intervention of the 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 $AN_ESR_TRIGGER=1
   - and dependent on EG synchronization monitoring (if, for example, Y is defined as EG following axis and the maximum permissible synchronization deviation must be 100 μm):
     IDS=03 WHENEVER ABS($VA_EG_SYNCDIFF[Y])>0.1
     DO $AN_ESR_TRIGGER=1
   - or (cumulatively) dependent on all three of the above trigger conditions + PLC + input:
     IDS=01 WHENEVER ($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-controlled reactions

Example using NC-controlled 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 “Extended stop and retract”, “Static synchronized actions” and “ASUB” options must be available.

Parameterization

Parameterization or programming required for the example:

```
$MC_ASUB_START_MASK = 7      ; MD 11602

; Function 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 drive-independent reactions
$MD_RETRACT_SPEED[X]=400000H              ; MD 1639,
                                          ; max. velocity
$MD_RETRACT_TIME[X]=10       ; MD 1638, ms/max. emergency retraction
time
$MD_GEN_STOP_DELAY[Y]=100 ; MD 1637, ms stop delay
$MD_GEN_STOP_DELAY[Z]=100 ; MD 1637, ms stop delay
$MD_GEN_AXIS_MIN_SPEED[A]= ...            ; MD 1635, generator
                                          ; min. speed (rpm)

; Configuration of NC-controlled retraction
LFPOS                        ; Axial retraction to a position
                             ; 46th G group
POLF[X]=IC(10)               ; Retraction target position
POLFMASK(X)                  ; Enable retraction

; Configuration of NC-controlled 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 part program or synchronized actions):
$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 synchronized action**(s), e.g.:

; dependent on intervention of the 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 also dependent on EG synchronized operation (if, for example, Y is defined as the EG following axis and if the max. permissible synchronized operation deviation is to 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                           ; Retraction mode
POLFMASK(X)                     ; Activation of lift fast of X axis
LFON                            ; Interruption of thread cutting ON N25 G33 X100
I10                             :
N30 X130 Z-45 K10               ;
N40 X155 Z-128 K10              ;
N50 X145 Z-168 K10              ;
N55 X120 I10                    ;
N60 G0 Z0 LFOF                  ; Interruption of thread cutting OFF
N70 POLFMASK( )                 ; Disable 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  ; Activation of ASUB via lift fast
                                        ; with fast input 1
N30 LFPOS                               ; Select retraction mode
N40 POLF[X]=19.5 POLF[Y]=33.3           ; Program retraction positions for X and Y
N50 POLF[Z]=100                         ; Program retraction position for Z
N60 X0 Y0 G0
N70 POLFMASK(X, Y)                      ; Select retraction of X axis and Y axis
N80 Z100 G1 F1000                       ; Retraction would set the position of the X axis to
                                        ; 19.5 mm and the Y axis to 33.3 mm
N90 POLFMASK(Z)                         ; Deselect retraction of X axis and Y axis,
                                        ; select retraction of Z axis
N100 Y10                                ; Retraction would set the position of the Z axis to
                                        ; 100 mm
N110 POLFMASK()                         ; Deselect retraction of Z axis, no axis
                                        ; retracting
```

6.3.5 Lift fast with several axes

Parameterization with several axes and incremental programming

```
N10 $AA_ESR_ENABLE[X1]=1                Activation via ESR
N12 $AA_ESR_ENABLE[Z]=1
N14 $AA_ESR_ENABLE[A1]=1
N30 LFPOS                               ; Select retraction mode for lift fast
N40 POLF[X1]=IC(3.0) POLF[A1]=–4.0      ; Program retraction position for axis X1 and
                                        ; A1
N50 POLF[Z]=100                         ; Program retraction position for Z
N60 X0 Y0 A0 G0
N70 POLFMASK(X1, A1)                    ; Select retraction of X axis and axis A1
N80 Z100 G1 F1000                       ; Retraction would set the position of machine axis
                                        ; X1 incrementally by 3.0 mm and axis A1
                                        ; absolutely by –4.0 mm
N82 POLF[X1]=10                         ; Change target position of X1 by 10.0 mm (abso
N80 Y0 G1 F1000                         ; lute) Retraction would set the position of machine
                                        ; axis X1 absolutely to 10.0 mm and axis A1 abso
                                        ; lutely to –4.0 mm
N90 POLFMASK(Z)                         ; Deselect retraction of X1 and axis A1,
                                        ; select retraction of Z axis
N100 Y10                                ; Retraction would set the position of the Z axis to
                                        ; 100 mm
N110 POLFMASK()                         ; Deselect retraction of Z axis, no axis retracting.
```

6.3.6 Lift fast with linear relation of axes

Retraction in linear relation

Example for an activation via a fast input with ASUB:

```
N10    $AA_ESR_ENABLE[X]=1      Activation via ESR
N12    $AA_ESR_ENABLE[Y]=1
N14    $AA_ESR_ENABLE[Z]=1
N30    LFPOS                    ; Select retraction mode
N40    POLF[X]=19.5 POLF[Y]=33.3     ; Retraction positions for X and Y
N50    POLF[Z]=100              ; Program retraction positions for Z
N60    X0 Y0 G0
N70    POLFMLIN(X, Y)           ; Select retraction in linear relation of
                                ; X and Y axis
N80    Z100 G1 F1000            ; Retraction would move the X axis to 19.5
                                ; and the Y axis to 33.3.
N90    POLFMLIN(Y, Z)           ; Select retraction of the X axis,
                                ; Select retraction in linear relation of
                                ; Y and Z axis
N100   Y10
N110   POLFMLIN()               ; Deselect linear retraction
```

Retraction in linear relation and independent

Example for parameterization with several axes and **incremental** programming:

```
N10    $AA_ESR_ENABLE[X1]=1     Activation via ESR
N12    $AA_ESR_ENABLE[Y]=1
N14    $AA_ESR_ENABLE[A1]=1)
N30    LFPOS                    ; Select retraction mode
N40    POLF[X]=IC(3.0) POLF[A1]=–4.0         ; Retraction positions for
;                                            ; axis X and A1
N50    POLF[Y]=100              ; Program retraction position for Z
N60    X0 Y0 A0 G0
N70    POLFMLIN(X, Y)           ; Select retraction in linear relation of
;                               ; X and Y axis
N75    POLFMASK(A1)             ; Select retraction of axis A1
N80    Z100 G1 F1000            ; Retraction would move the X and Y axis in
                                ; linear relation.
                                ; Axis X by 3.0 and
                                ; axis Y absolute to 100.

                                ; Regardless of X and Y, axis A1 would
                                ; travel to –4.0
N90    POLF[X]=10               ; Change target position of X to 10.0
                                ; absolute
N95    Y0 G1 F1000              ; Retraction would move X and Y in linear
                                ; relation. Axis X to
                                ; 10.0 and axis Y to 100.
                                ; Regardless of X and Y, axis A1 would
                                ; travel to –4.0
N100   POLFMLIN()               ; Deselect retraction of X and Y axis,
```

```
                            ; example continued

N110   Y10                  ; Retraction would move the A1 axis to –4.0
N120   POLFMASK()           ; Deselect retraction of A1 axis, no
                            ; axis retracting
```

■

Data Fields, Lists

7

7.1 Interface signals

DB number	Bit, byte	Name	Refer-ence
Channel-specific			
21, ...	0.3	Activate DRF	H1
Axis-specific			
31, ...	0.0–0.7	Feed rate 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	Active 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	Refer-ence
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)	
18403	MM_NUM_CURVE_SEG_LIN	Number of linear curve segments (SRAM)	
18404	MM_NUM_CURVE_POLYNOMS	Number of curve table polynomials (SRAM)	
18406	MM_NUM_CURVE_TABS_DRAM	Number of curve tables in DRAM	

General ($MN_ ...)			
18408	MM_NUM_CURVE_SEG-MENTS_DRAM	Number of curve segments in DRAM	
18409	MM_NUM_CURVE_SEG_LIN_DRAM	Number of linear curve segments (DRAM)	
18410	MM_NUM_CURVE_POLY-NOMS_DRAM	Number of curve polynomials in DRAM	
Channelspecific ($MC_ ...)			
20110	RESET_MODE_MASK	Definition of control basic setting after run-up and RESET/ part program end	K2
20112	START_MODE_MASK	Definition of control basic setting after run-up and RESET	K2
20900	CTAB_ENABLE_NO_LEADMO-TION	Curve tables with jump of following axis	
20905	CTAB_DE-FAULT_MEMORY_TYPE	Default memory type for curve tables	
21204	LIFTFAST_STOP_COND	Stop characteristics for fast retraction	
21380	ESR_DELAY_TIME1	Delay time (STOPBYALARM, NOREAD) for ESR axes	
21381	ESR_DELAY_TIME2	Time for interpolatory braking for ESR axes	
Axisspecific ($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
37160	LEAD_FUNCTION_MASK	Functions of the master value coupling (SW 6.4 and higher)	
37200	COUPLE_POS_TOL_COARSE	Threshold value for “Coarse synchronism”	S3
37210	COUPLE_POS_TOL_FINE	Threshold value for “Fine synchronism”	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	Refer-ence
Axisspecific ($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	Refer-ence
	$AC_STAT	Channel status: invalid, in reset, interrupted and active	PGA1
	$A_IN	Digital input NC	PGA1
	$A_OUT	Digital output NC	PGA1
	$A_DBB	Read/write data byte (8 bits) from/to PLC	PGA1
	$AC_ALARM_STAT	!=0: Alarms are present, the coded associated alarm reactions can be used as a source for “Exten. stop and retract”.	PGA1
	$AN_ESR_TRIGGER	(Global) control signal “Start stop/retract” drive-independent (SW 5 and higher)	PGA1
	$AC_ESR_TRIGGER	Channel-specific control signal “Start stop/retract” NC-controlled (SW 6 and higher)	PGA1
	$AA_ESR_STAT[axis]	(Axial) status feedback signals from “Extended stop and retract” (SW 5 and higher)	PGA1
	$AA_ESR_ENABLE[axis]	1 = (axial) enable of reaction(s) of “Extended stop and retract”. (SW 5 and higher)	PGA1
	$AA_TYP[axis]	Axis type	PGA1
Electronic gear (EG) and master value coupling			
	Identifier	**Name**	**Refer-ence**
	$AA_EG_SYNFA	Synchronized position of following axis a (SW 5 and >)	PGA1
	$P_EG_BC	Block change criterion for EG activation calls: EGON, EGONSYN. WAITC => immediate synchronism fine or coarse and setpoint synchronism. (SW 6.1 and higher)	PGA1
	$AA_EG_NUMLA	Number of leading axes defined with EGDEF (SW 5 and >)	PGA1
	$VA_EG_SYNCDIFF	Difference in synchronism (SW 5 and higher)	PGA1
	$AA_EG_AX	Identifier for nth leading axis (SW 6.1 and higher)	PGA1
	$AA_EG_TYPE	Type of coupling for leading axis b (SW 6.1 and higher)	PGA1
	$AA_EG_NUMERA	Counter of coup. factor for leading axis b (SW 6.1 and >)	PGA1
	$AA_EG_DENOM	Denomin. of coup. factor for leading axis b (SW 6.1 and >)	PGA1
	$AA_EG_SYN	Synchronized position of leading axis b (SW 6.1 and >)	PGA1
	$AA_EG_ACTIVE	Coupling for leading axis b is active, i.e. switched on (SW 6.1 and higher)	PGA1
	$AA_LEAD_SP	Simulated master value - position in MCS (SW 4 and >)	PGA1
	$AA_LEAD_SV	Simulated master value - velocity (SW 4 and higher)	PGA1
	$AA_LEAD_P_TURN	Current master value - position component lost as a result of modulo reduction. (SW 4 and higher)	PGA1
	$AA_LEAD_P	Current master value - pos. (modulo/red.). (SW 4 and >)	PGA1
	$AA_LEAD_V	Current master value - velocity (SW 4 and higher)	PGA1
	$AA_SYNC	Coupling status of following axis for master value coupling. (SW 4 and higher)	PGA1
Dynamics of following axis			
	$PA_ACCLIMA	Acceleration offset set with ACCLIMA during preprocessing (SW 6.4 and higher)	PGA1
	$AA_ACCLIMA	Acceleration offset set with ACCLIMA during main run (SW 6.4 and >)	PGA1

	Identifier	Name	Reference
	$PA_VELOLIMA	Velocity offset set with VELOLIMA during preprocessing (SW 6.4 and higher)	PGA1
	$AA_VELOLIMA	Velocity offset set with VELOLIMA during main run (SW 6.4 and higher)	PGA1

7.5 Alarms

Detailed explanations of the alarms, which may occur, appear in

References: /DA/, Diagnostics Guide

or in the Online help.

■

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

Setpoint Exchange (S9)

■

Notes

Brief Description 1

1.1 Introduction

Function

The “setpoint exchange” function is used in applications in which the same motor is used to traverse different machine axes.

Operating conditions

The function described below replaces the setpoint exchange technology card function (TE5) for systems with NCK SW >= 7.1.

An option is required for the function.

Compatibility

Migration to NCK SW 7.1 requires adaptations to be made to machine data and the PLC user program.

■

Notes

Detailed Description 2

2.1 Function

The “setpoint exchange” function is required in applications in which a single motor needs to drive a number of axes/spindles such as, for example, on milling machines with special millheads. The spindle motor is operated as both a tool drive and a millhead orienting mechanism.

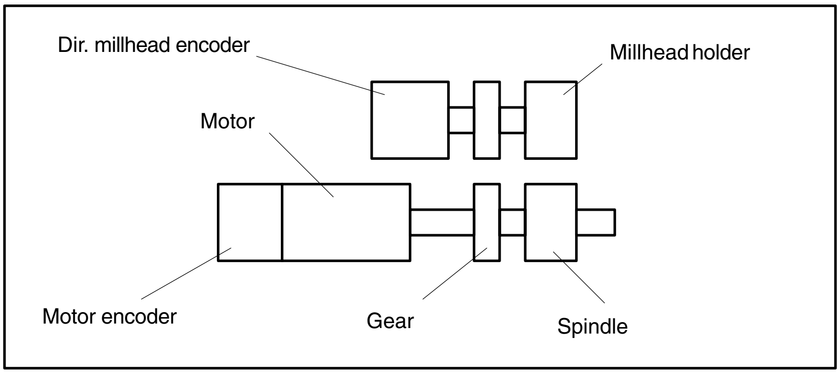


Fig. 2-1 Example 1: 1 motor encoder, extra encoder for millhead

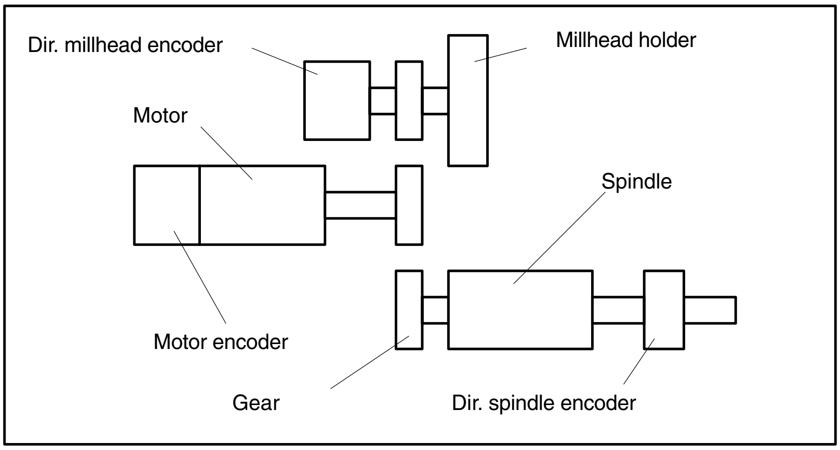


Fig. 2-2 Example 2: 1 motor encoder, separate millhead encoder and spindle encoder

Configuring

Setpoint exchange enables a number of axes to use the same drive.

The same setpoint channel on this drive is assigned a number of times to define the axes participating in setpoint exchange. Machine data MD 30110: CTRLOUT_MODULE_NR must be pre-assigned with the **same** logical drive number for every axis.

Note

Alarm 26018 is output if the option is missing.

Encoder assignment takes place on an axis-specific basis in MD 30230: ENC_INPUT_NR.

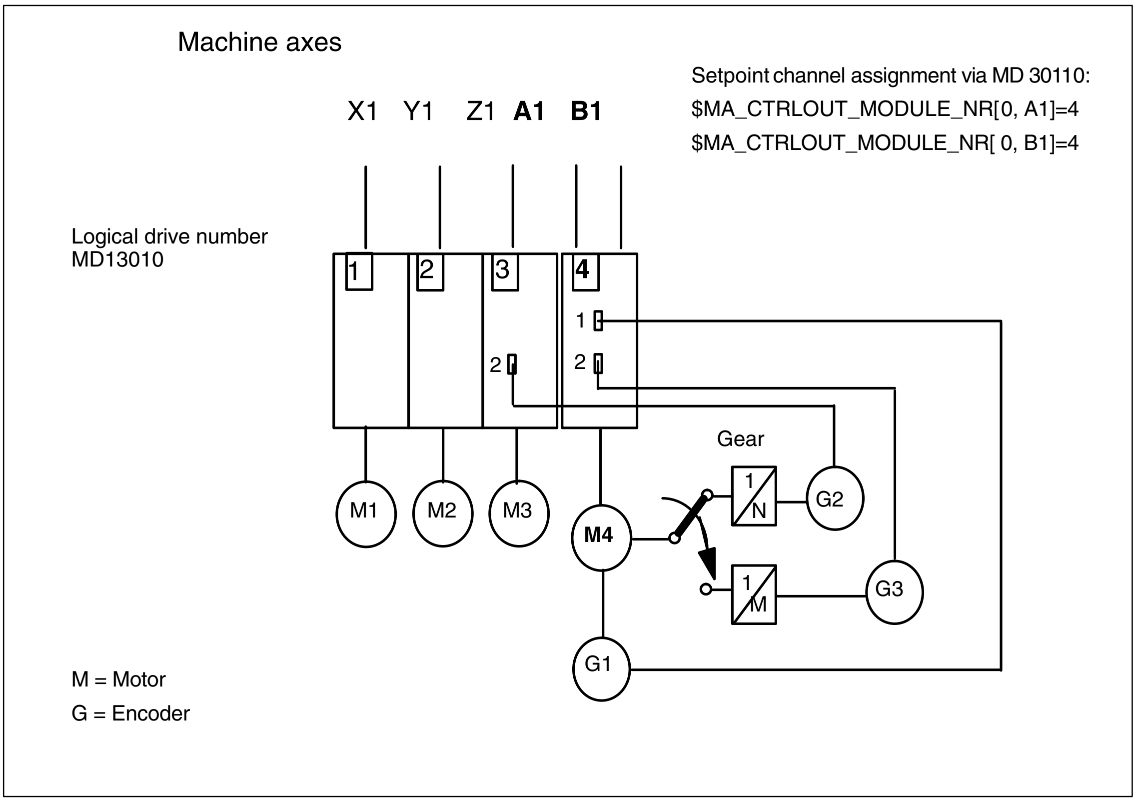


Fig. 2-3 Setpoint exchange with 2 axes

Activation

The setpoint is exchanged and the corresponding interface signals are evaluated in the PLC user program.

Note

An existing PLC user program may need to be modified due to changes in the meaning of interface signals in comparison with the technology card solution.

At any one time, only **one** of the machine axes with the appropriate logical drive number may have control via the setpoint channel of the drive.

Requests to transfer drive control are sent using `DB31–..., DBX24.5`.

The current drive control state is indicated in `DB31–..., DBX96.5.`

Access rights to the shared drive must be managed in the PLC user program.

Transfer conditions

- Axis standstill of all axes involved
- Special functions such as reference point approach, measuring, travel to fixed stop, function generator, star/delta changeover, drive parameter set changeover are not active in the axis with drive control.
- No sign-of-life error and no faults pending on PROFIBUS drive

The PLC interface provides constant information about the current state of the exchange. At any one time, only **one** axis has drive control `DB31–..., DBX96.5=1`.

During exchange, servo enables on all axes involved are automatically withdrawn by the controller.

Axes without drive control are not in closed-loop control. Therefore, a brake control must be set up for vertical axes.

Special cases

If a number of transfer requests are made simultaneously, exchange will not take place. The last axis used remains in control of the drive. This is also the case if there are no transfer requests pending.

If there are no transfer requests during machine power-up, drive control is assigned to the **first** machine axis located with the same logical drive number. Logical drive numbers are scanned in ascending order.

```
$MA_CTRLOUT_MODULE_NR[0,AX1]= 1
$MA_CTRLOUT_MODULE_NR[0,AX2] = 2
$MA_CTRLOUT_MODULE_NR[0,AX3] = 3
$MA_CTRLOUT_MODULE_NR[0,AX4] = 4      ; Drive control
                                        during power-up
$MA_CTRLOUT_MODULE_NR[0,AX5] = 4
```

2.2 Interface signals

Axisspecific signals

Despite assignment to an individual drive, the use of DB31–... remains unchanged.

This requires explicit access coordination in the PLC user program.

As the same drive is being used, **the same status signals** from DB31–..., DBB92–95 are displayed in **all** axes involved in the exchange.

However, control signals DB31–..., DBB20–21 must only be set in the **axis with exclusive control** of the drive. Servo enable DB31–..., DBB2.1 is only effective if DB31–..., DB24.5 have been set.

Axes **without** the corresponding drive control are subject to functional restrictions and are therefore operated in **follow-up mode**. For this purpose, the servo enables are deleted automatically by the controller.

Diagram of the PLC program

The following example illustrates a possible setpoint exchange sequence.

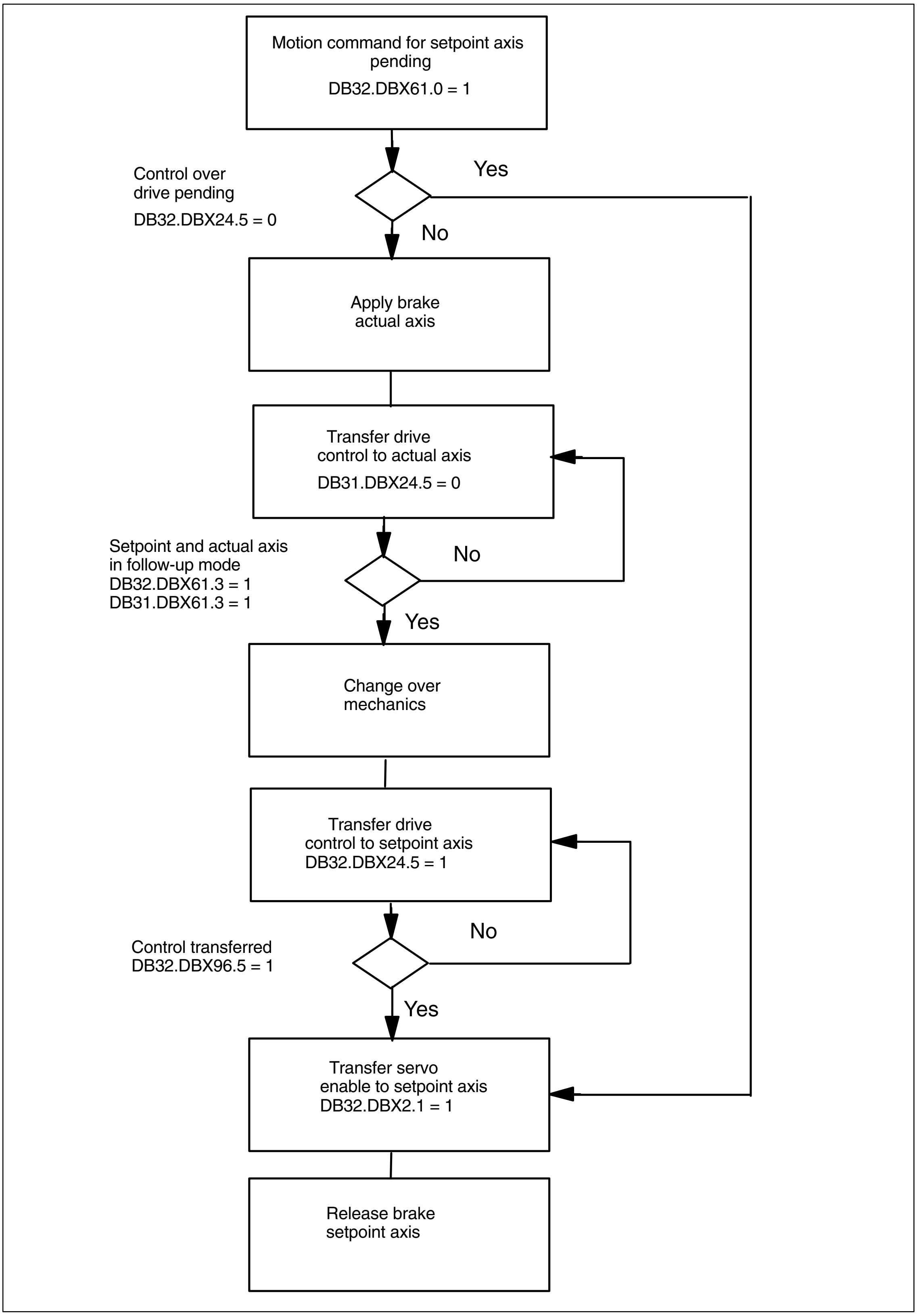


Fig. 2-4 PLC-controlled sequence of a setpoint exchange between AX1 → AX2

2.3 Alarms

Drive alarms are only displayed for axes with drive control.

2.4 Position control loop

During setpoint exchange, the drive train and therefore the position control loop are isolated. In order to avoid instabilities, exchange only takes place at **standstill** and **once all servo enables have been deleted**

The use of a single drive means that only one of the control loops can be closed at any one time. Axes without drive control are operated with open position controller and **following positions**.

2.5 Reference points

The use of load-side encoders does not affect the axial reference points of a setpoint exchange.

However, the mechanical reference to the load can be lost following setpoint exchange for a load-side position derived solely from the motor encoder. These types of axis must be **referenced again** after every setpoint exchange.

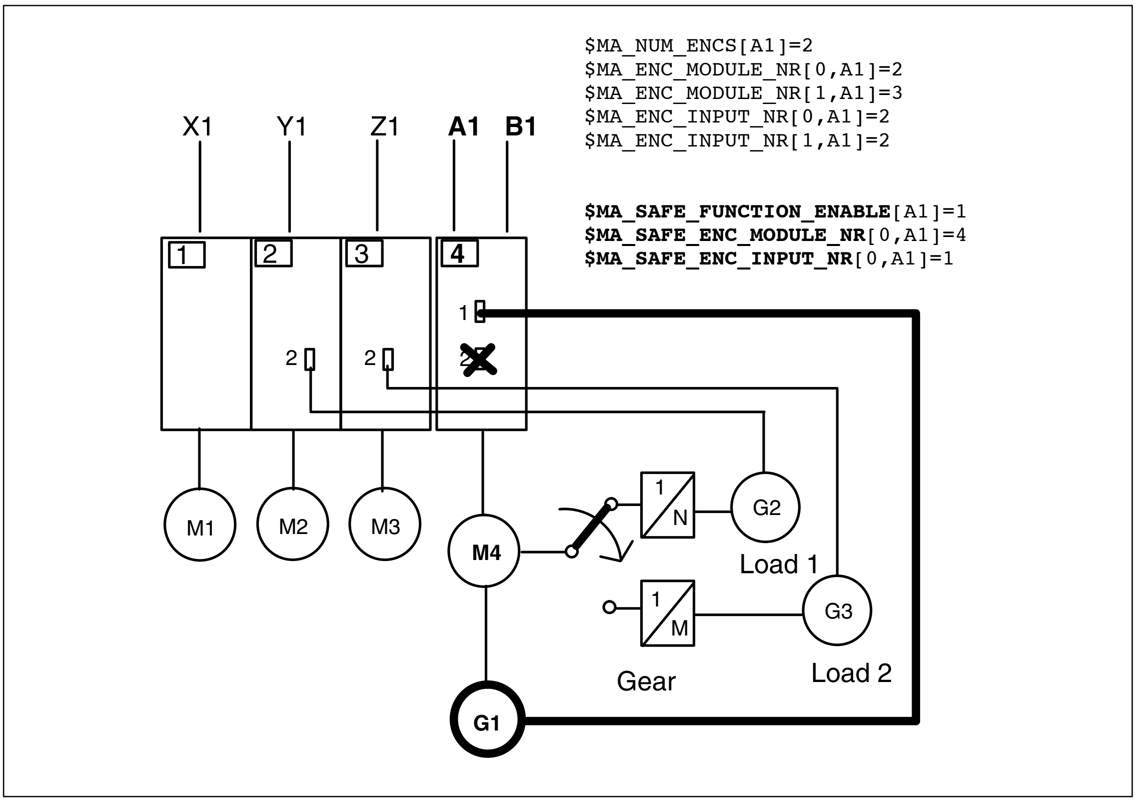


Fig. 2-5 Setpoint exchange in conjunction with single-encoder safety integrated system

2.6 Differences in comparison with the technology card

The setpoint exchange implemented in NCK SW 7.1 and higher differs from the compile cycles solution described in TE5 as follows. These differences must be taken into account during installation and start-up and when creating the PLC user program:

- Machine data MD 63750: CTRLOUT_CHANGE_TAB is no longer used.
- The meanings of associated PLC interface signals have changed. Therefore, PLC user programs must be updated accordingly.
- Alarms 70451 and 70452 are no longer used.
- Setpoint exchange with simulated axes
  MD 30130: CTRLOUT_TYPE=0 is no longer supported.
- Known restrictions of the technology card function no longer apply.

■

Notes

Supplementary Conditions 3

Availability

Setpoint exchange is available in SW 7.1 and higher.

Features

Setpoint exchange is only possible in conjunction with **611D and PROFIBUS drives** MD 30100: CTRLOUT_SEGMENT_NR=**1, 5** or **6**. All other settings generate alarm 26018.

The "Parking" operating state can only be activated via the axis to which drive control has been assigned.

The "Drive Service Display" HMI diagnostics screen currently does not take into account changes in assignments between axis and drive.

Setpoint exchange can only be started up via **SinuComNc** via the Expert List. A dialog is not supported.

See the FBSI description (11.02 and later) for supplementary conditions for **Safety Integrated** in conjunction with setpoint exchange.

Data Descriptions (MD, SD) 4

4.1 Machine data

For descriptions of the machine data, see:

References: LIS, Lists

For descriptions of the machine data for Safety Integrated, see

References: FBSI, Description of Functions, Safety Integrated

■

Notes

Signal Descriptions 5

5.1 Axis/spindle-specific signals

DB31 –... **DBX24.5** Data Block	**Activate setpoint exchange** Signal(s) to axis/spindle (PLC –> NCK)	
Edge evaluation: No	Signal(s) updated: Cyclic	Signal(s) valid from SW: 3.5
Signal state = 1	Request to axis to take over drive control.	
Signal state = 0	Request to axis to relinquish drive control.	

DB31 – ... **DBBX96.5** Data Block	**Status of setpoint exchange** Signal(s) from axis spindle (NCK –> PLC)	
Edge evaluation: No	Signal(s) updated: Cyclic	Signal(s) valid from SW: 3.5
Signal state 1	The axis has taken over control of the drive.	
Signal state 0	The axis has relinquished control of the drive.	
Signal relevant for	All axes involved in setpoint exchange.	

■

Notes

Example 6

6.1 Example

Notes

Data Fields, Lists 7

7.1 Machine data

Number	Identifier	Name	Refer-ence
Axisspecific ($MA_ ...)			
30130	CTRLOUT_TYPE	Output type of setpoint	G2, S6
30200	NUM_ENCS	Number of encoders	G2
30220	ENC_MODULE_NR	Actual-value assignment: Drive number/measuring circuit number	G2
30230	ENC_INPUT_NR	Actual-value assignment: Input on drive module/control loop module	G2

■

Notes

SINUMERIK 840D sl/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 cornerthe 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

- **In 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 for example for the following applications:

- Tangential positioning of a rotatable tool for nibbling operations.
- Follow-up control of tool alignment for a bandsaw.
- Positioning a dressing tool on a grinding wheel.
- Positioning of a gear shaping cutter in glass or paper processing applications.
- Tangential feed of a wire for 5-axis welding.

■

Detailed Description 2

2.1 Characteristics of tangential follow-up control

Task assignment

Follow-up control for the rotary axis must be implemented so that the axis is always positioned at a specified angle on the programmed path of the two leading axes.

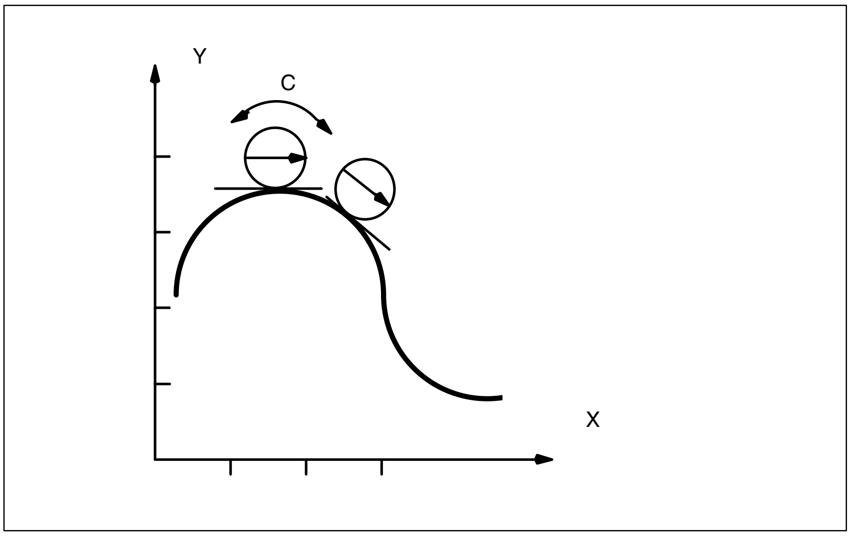


Fig. 2-1 Tangential control, offset angle of zero degrees to path tangent

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 as from SW 3.2

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 to the level defined by the two leading axes is relevant.)
  If there is a hidden corner in space, an intermediate block is inserted before the block (in this case N6) causing the tangential jump. This intermediate block moves the following axis to the new position. This block transition is not smoothed.

```
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 this block is processed.
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)
- 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).

Influence on transformations

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 the follow-up axis

If a following axis, which is being made to follow its leading axes, is positioned explicitly, then the position specification is added to the offset angle programmed in the activation instruction TANGON. See 2.2.2. 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**arch **r**un by **pro**gram test”) can be used to simulate tangential follow-up of axes in SW 6.2 and higher.

For more information about multi-channel SERUPRO block search, please see

References: /FB/, K1, “Mode Group, Channels, Program Operation” 2.4 Program test

2.2.1 Assignment between leading axes and following axis

Programming

The activation is programmed via a predefined subprogram **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
Coordinate system identifier “B” → Basic coordinate system, “W” →Workpiece coordinate system is not available.	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 subprogram **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.

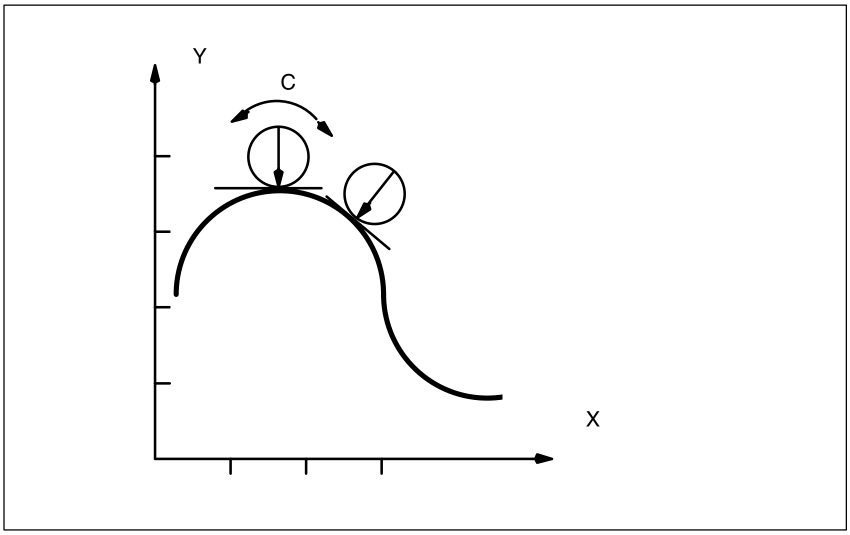


Fig. 2-2 Tangential control, offset angle of 90 degrees to path tangent

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 written if the corner response is to be contained in an intermediate block.

TLIFT (C)

The control reads machine data
MD 37400: EPS_TLIFT_TANG_STEP for the tangential following axis C.
If the tangential angle jump exceeds the value (absolute value) of the angle set in the MD, the control recognizes a "corner" and approaches the new position of the following axis via an intermediate block.

As of SW 6.4

System variable $AC_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 subprogram **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 (preprocessor stop).

RESET/end of part program

An activated tangential control can remain active for further machining operations. See

References: /FB/, K2, "Coordinate Systems, Axis Types, Axis Configurations, Workpiece-Related Actual-Value System, External Zero Offset".

Note

The assignment between 2 master axes and a slave axis programmed with 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
TANG() block 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 subprogram **TANGDEL** is used to cancel the definition of a follow-up axis assignment so that the follow-up axis can be operated dependent on new leading axes when a new follow-up axis assignment is defined.

TANGDEL(C)

The existing definition in the example of TANG(A, X, Y) is canceled.

Example for plane change

```
N10 TANG(A, X, Y, 1)
N20 TANGON(A)
N30 X10 Y20
......
N80 TANGOF(A)
N90 TANGDEL(A)              ; Delete defined coupling from A to X and Y
                            ; as leading axes
......
N120 TANG(A, X, Z)          ; A can be coupled to new leading
                            ; axes
N130 TANGON(A)
......
N200 M30
```

Example for geometry axis switchover

If the definition of the follow-up axis assignment is not canceled, an attempt to execute a geometry axis switchover is suppressed and an alarm is output.

```
N10 GEOAX(2, Y1)
N20 TANG(A, X, Y)
N30 TANGON(A, 90)
N40 G2 F8000 X0 Y0 I0 J50
N50 GEOAX(2, Y2)            ; Alarm 14415, geometry axis to be deleted
                            ; is still leading axis for 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 defined
N20 TANG(A,X, Y)              ; Channel axis Y1 assigned
N30 TANGON(A, 90)             ; Follow-up grouping with Y1 activated
N40 G2 F8000 X0 Y0 I0 J50     ; Traversing block for leading axes
N50 TANGOF(A)                 ; Deactivation of follow-up
N60 TANGDEL(A)                ; Deletion of definition
                              ; of follow-up axis assignment
N70 GEOAX(2, Y2)              ; Geometry axis changeover possible
N80 TANG(A, X, Y)             ; New def. of follow-up axis grouping
N90 TANGON(A, 90)             ; Follow-up grouping with Y2 activated
.......
```

2.3 Limit angle

Description of problem

When the axis moves backwards and forwards along the path, the tangent turns abruptly through 180 degrees at the path reversal point. This response is not generally desirable for this type of machining operation (e.g. grinding of a contour). It is far better for the reverse motion to be executed at the same offset angle (negative) as the forward motion.

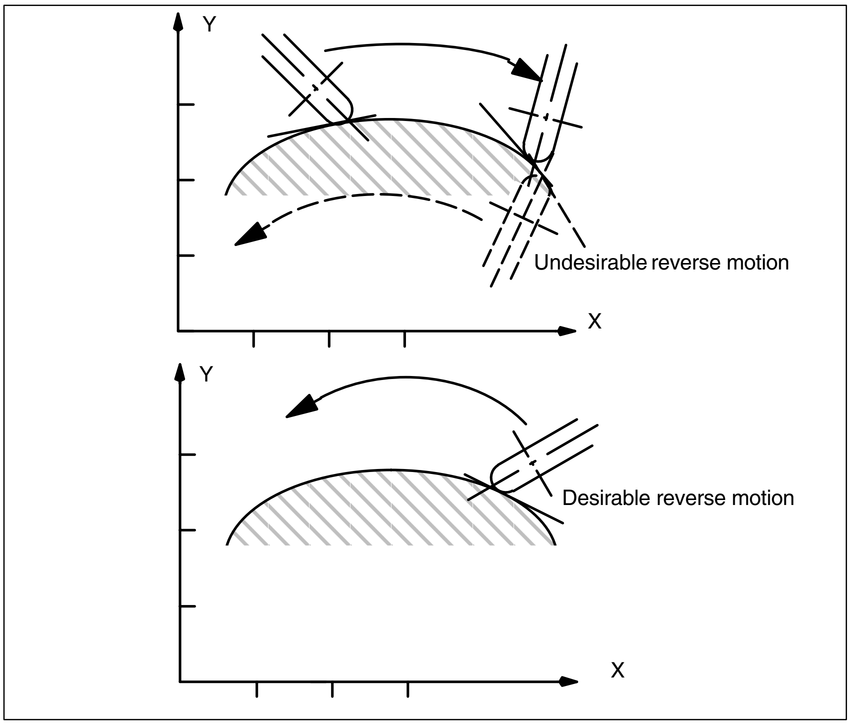


Fig. 2-3 Backward and forward motion on the path

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: /PG/, “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, SW 2 and higher

In SW 3.2 and higher, the special response at path corners, controlled by TLIFT () is available.

Data Descriptions (MD, SD) 4

4.1 Machine data

37400 MD number	**$MA_EPS_TLIFT_TANG_STEP** Tangential angle for corner recognition				
Default setting: 5		Minimum input limit: 0		Maximum input limit: 180	
Changes effective after RESET			Protection level: 2 / 7		Unit: Degrees
Data type: DOUBLE			Applies as of 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 MD number	$MA_TANG_OFFSET Default angle for tangential follow-up control	
Default setting: 0	Minimum input limit: –	Maximum input limit: –
Changes effective after RESET	Protection level: 2 / 7	Unit: Degrees
Data type: DOUBLE	Applies as of 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	

■

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

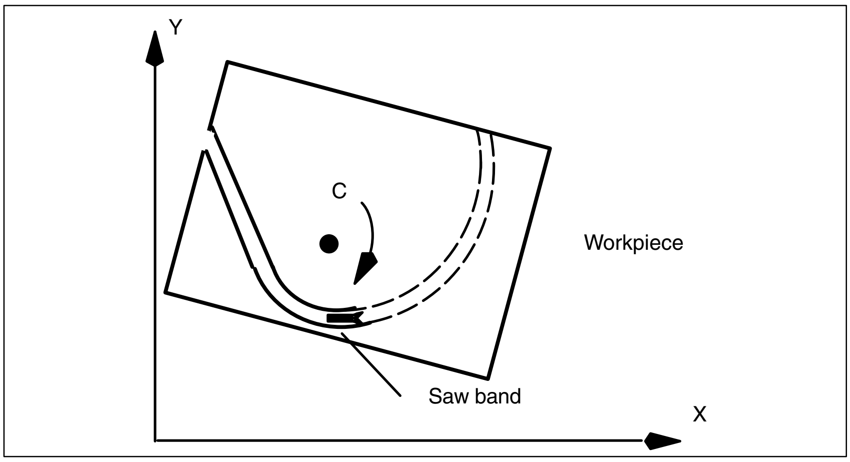


Fig. 6-1 Tangential positioning of a workpiece on a bandsaw

Positioning of tool

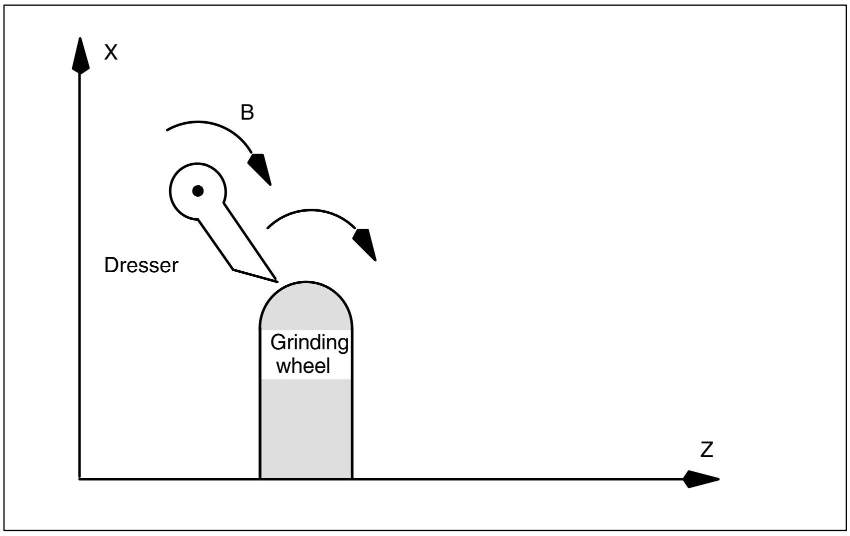


Fig. 6-2 Positioning of a dressing tool on a grinding wheel

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.

■

Data Fields, Lists 7

7.1 Alarms

Detailed explanations of the alarms, which may occur, appear in

References: /DA/, Diagnostics Guide

or in the Online help.

7.2 Machine data

Number	Identifier	Name	Reference
Axisspecific ($MA_ ...)			
37400	EPS_TLIFT_TANG_STEP	Tangential angle for corner recognition	
37402	TANG_OFFSET	Default angle for tangential follow-up control	

7.3 System variable

	Identifier	Name	Reference
	$AC_TLIFT_BLOCK	Current block is an intermediate block generated by TLIFT	PGA1

■

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

Installation of Compile Cycles (TE01)

■

Notes

1 Brief Description

1.1 Brief description (840D)

Aim of the description

This section describes how technology functions are installed and activated in the form of loadable compile cycles. The description applies to all of the following technology functions available from Siemens:

- 1D/3D clearance control in position control cycle
  Order no.: 6FC5 251–0AC05–0AA0
  Compile cycle: CCCLC.ELF
  References: Clearance Control chapter (TE1)
- Handling Transformation Package
  Order no.: 6FC5 251–0AD07–0AA0
  Compile cycle: CCRCTRA.ELF
  References: Handling Transformation Package chapter (TE4)
- Setpoint exchange
  Order no.: 6FC5 251–0AC05–0AA0
  Compile cycle: CCSETP.ELF
  References: Setpoint Exchange chapter (TE5)
- Axial coupling in machine coordinate system (MCS coupling)
  Order no.: 6FC5 251–0AD11–0AA0
  Compile cycle: CCMCSC.ELF
  References: MCS Coupling chapter (TE6)
- Retrace/Continue Machining Support
  Order no.: 6FC5 251–0AE72–0AA0
  Compile cycle: CCRESU
  References: Retrace/Continue Machining Support chapter (TE7)
- High-speed laser switching signal
  Order no.: 6FC5 251–0AE74–0AA0
  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 from 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.

Requirements

One of the following programs is required for the installation:

- SinuCom NC
- SinuCopy FFS
- HMI Advanced, SW 6.3 and higher

required.

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
Chapter: NC Installation and Start-Up With HMI Advanced, Loadable Compile Cycles

■

Detailed Description 2

2.1 Design (840D)

Delivery format

Technology functions that are available as loadable compile cycles must be purchased as options:

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

The further use of existing archives with compile cycles is described in Chapter 3, page 3/TE01/3-11.

2.1.1 Copying compile cycles into the flash file system (FFS)

Target directory

In order that a compile cycle is detected and loaded while powering up the NC, all ELF files (*.ELF) belonging to the compile cycle must be copied into the following FFS directory of the PC card: _N_CCOEM_DIR

Note

It must be ensured that sufficient memory space is available before copying a compile cycle or the corresponding ELF file(s) into the FFS of the PC card.

The ELF files can be copied in different ways into the FFS of the PC card:

- **HMI Advanced (SW 6.3 and higher)**

  With HMI Advanced, the ELF files can be copied directly from various media into the FFS of the active PC card of the NCU.

  References:
  /BAD/ Operator's Guide HMI Advanced

- **External PC and "SINUCOPY FFS"**

  Use the program: "SINUCOPY FFS" to create the "_N_CCOEM_DIR" directory in the FFS of the PC card and copy the ELF files into this directory.

- **PC/PG with MPI connection to the NC and "SinuCom NC"**

  Using the startup tool: "SinuCom NC" (Version 6.2.12 and higher), the ELF files can be copied directly into the FFS of the active PC card of the NCU. SinuCom NC menu command: **File > Load compile cycle**

- **"ddetest.exe" program**

  (The program: "ddetest.exe" is supplied with the SINUMERIK user interface "HMI Advanced".)

  The following three steps must be performed to copy the ELF files into the FFS of the active PC card of the NCU

  1. Create a variable (e.g. COPYELF):

     Execute <Doit> new(COPYELF, –1)

  2. Start a hotlink to this variable:

     Hotlink <Start> COPYELF

  3. Start PI service to load an ELF file stored on diskette in drive: "a:\", e.g. ccmcsc.elf:

     Execute <Doit> copy_to_nc (a:\ccmcsc.elf,
     /NC/_N_NC_CARD_DIR/_N_CCOEM_DIR/_N_CCMCSC_ELF, COPYELF)

     After starting the hotlink, variable "COPYELF" displays the initialization value –1. During the loading of the ELF file, the value increments from 0 to 99. A value of 100 indicates successful loading.

2.1.2 Loading the compile cycles into the NC

Each time the NC is powered up, **all** compile cycles (ELF files) in directory "_N_CCOEM_DIR" available as part of the system software are loaded to the NC. The NC system software accesses the compile cycles via a special interface. The interface version used by the compile cycle and the NC system software must therefore be compatible.

Interface version

Each interface version is displayed under:

- Interface version of the NC system software

  HMI Advanced:
  Diagnosis > Service displays > Version > NCU Version

  Display (excerpt):

  CC Interface Version:
  @NCKOPI . . . .**@Interfaces=*<1st digit>*.*<2nd digit>*** . . . .
  Loaded Compile Cycles:
  . . . .

- Interface version of a compile cycle that has not yet been loaded

  HMI Advanced (excerpt):
  Services > *<Medium>* > Softkey: "Properties"

Display:

Contents: Loadable compile cycle
Interface: . . . . **@Interfaces=<*1st digit*>.<*2nd digit*>** . . . .

- Interface version of a loaded compile cycle

  HMI Advanced:
  Diagnosis > Service displays > Version > NCU Version

  Display (excerpt):

  CC Interface Version:
  @NCKOPI . . . .
  Loaded Compile Cycles:
  <*Identifier*> <*Version*> <*Date generated*>
  CC start address
  _N_<*Identifier*><*Version*>**IF<*1st digit*><*2nd digit*>**_ELF . . .

  Example:
  _N_CLC407**IF003001**_ELF corresponds to interface version: 3.1

SW version of a compile cycle

The SW version of a compile cycle is displayed under:

HMI Advanced:
Diagnosis > Service displays > Version > NCU Version

Display (excerpt):

CC Interface Version:
@NCKOPI . . . .
Loaded Compile Cycles:
<*Identifier*> <*Version*> <*Date generated*>
CC start address
_N_<*Identifier*>**<Version>**IF<*1st digit*>.<*2nd digit*>_ELF . . .
Code=<*Address*> Data=<*Address*> . . .

Example:
_N_CLC**407**IF003001_ELF corresponds to SW version: 4.7

Note

The display of code and data range start addresses of a compile cycle are provided for diagnostics purposes only and have no significance in normal operation.

Dependencies

The following dependencies exist between the interface versionsof a compile cycle and the NC system software:

- 1. Digit of the interface version number
  The 1st digit of the interface version number of a compile cycle and the NC system software must be **equal**.

- 2. Digit of the interface version number
  The 2nd digit of the interface version number of a compile cycle must be **less than or equal** to the 2nd digit of the NC system software.

Caution

If alarm 7200 is displayed after start-up, this means **no** compile cycle has been loaded!

2.1.3 Activating the technology functions in the NCK

Option

The corresponding option (see Section 4.2, page 3/TE01/4-15) must be enabled before activating a technology function as described below.

If the option data has not been set, the following alarm appears every time the NCK boots and the technology function will not be activated:

- Alarm 7202 XXX_ELF_option_bit_missing: <*Bit number*>"

Note

The previous compile cycle option data

– MD 19600: $ON_CC_EVENT_MASK[n]

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_<*name*>[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], Bit0 = 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 (part programs, tool data, etc.) will be deleted on the next run-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 run-up.

Function-spec. startup

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
Chapter: NC Installation and Start-Up With HMI Advanced, Loadable Compile Cycles

■

Notes

Supplementary Conditions 3

3.1 Transition to newer NCK versions (840D)

In order to be able to use technology functions from an existing archive in conjunction with newer NCK versions (NCK 06.03.23 and later), the archive must first be updated before being loaded in the NC.

Requirements

The following requirements must be met in order to update an archive:

- A PC card with a SINUMERIK 840D standard system Version NCK 06.03.23 or later
- The ELF files for the technology functions to be activated
- The conversion program arc4elf.exe (archive conversion)

Update

Proceed as follows to update an archive:

1. Create backup archive
   – Standard procedure or
   – With optimization of SRAM utilization
2. Insert new PC card
3. Incorporate the ELF files
4. NCU RESET
5. Activate technology function
6. Reactivate NCU RESET
7. Convert archive using arc4elf.exe
8. Load converted archive

3.1.1 Create backup archive

Standard

The standard creation of an archive for user data backup is described in:

References:

/IAD/ Installation & Start-up Guide 840D/SIMODRIVE 611D
Chapter: Data Backup

Optimized

Data backup with optimization of SRAM utilization is only necessary if an archive for an NCK Version 6.3.xx is being used and the NCK SRAM needs to be optimized.

The memory configuration machine data:

- MD18238: MM_CC_MD_MEM_SIZE = 1 must be reset to enable the NCK SRAM previously requested explicitly for technology functions.

An explicit request no longer has to be sent to the SRAM memory as the technology functions loaded via ELF files will request the required SRAM memory on a function-specific basis from the NCK memory management.

Resetting the above machine data will reorganize the SRAM memory the next time the NCK starts up and all user data will be lost.

In this state, you must create a new archive BEFORE resetting the NCK.

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 that is most suitable described in 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 Reactivate NCU RESET

The NCK is rebooted; no alarms should appear.

3.1.7 Convert archive

The archive created in 3.1.1 “Standard”/“Option” must be converted. The **arc4elf.exe** program is required for this purpose (available from E-Support).

Use the
arc4elf –h call to access help for using the program.
The general format of the call 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

4 Data Descriptions (MD, Options)

4.1 Machine data

4.1.1 NC-specific machine data

60900 + i where i = 0. 1. 2. 3 ... MD number	**CC_ACTIV_IN_CHAN_XXXX[n]** with: XXXX = function code, n = 0 or 1 n = 0: Activate technology function in NC channels n = 1: Additional functions within the technology function				
Default setting: 0		Minimum input limit: 0		Maximum input limit: FFFF	
Changes effective after RESET			Protection level: 2 / 7		Unit: –
Data type: UINT16			Applies as of SW: 2.2		
Meaning:	Activate technology function in NC channels: The technology function is activated in the NC channels by means of index n = 0. Bit 0 = 1: Technology function activated in NC channel 1 Bit n = 1: Technology function activated in NC channel n+1 For more details about for which NC channels a technology function can be activated, please see the references below. Additional functions within the technology function: The MD with index n = 1 activates additional functions within the relevant technology function. See References below. **References:** /FB3/ Description of Functions Special Functions TE1 – TExx				

4.2 Options

Technology functions that are available as loadable compile cycles must be purchased separately as options.

References: Ordering information in Catalog NC 60

To obtain the required compile cycle in the form of a loadable file (*.ELF), please contact your regional Siemens sales partner.

■

Notes

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

Simulation of Compile Cycles (TE02)

■

Notes

Brief Description 1

1.1 Features

If part programs, which use compile cycles, are simulated on the SINUMERIK user interface, e.g. HMI Advanced, simulation is aborted and corresponding error messages are output. The reason is that compile cycle support has not yet been implemented on the HMI.

The measures described below show how to set up the runtime environment to enable the simulation of part programs, which use compile cycles, without error messages.

1.2 Requirements

1.2.1 OEM transformations

HMI Advanced

At least the following version is required in order to simulate part programs which use the functions of OEM transformations:

- Software version: SW 6.2.12

■

Notes

Detailed Description 2

2.1 OEM transformations

The runtime environment for simulation must be configured as follows when using OEM transformations:

1. Create a new directory: <*Installation path*>/OEM on the computer on which the HMI application, e.g. HMI Advanced, is installed in the directory structure of the HMI application next to the standard existing directory: <*Installation path*>/MMC2.
2. In the OEM directory created under 1., create a file: DPSIM.TEA with the following contents:

```
$MN_NC_USER_CODE_CONF_NAME_TAB[196]="TRAORI"
$MN_NC_USER_CODE_CONF_NAME_TAB[197]="_TRAORI"
$MN_NC_USER_CODE_CONF_NAME_TAB[198]="TRACON"
$MN_NC_USER_CODE_CONF_NAME_TAB[199]="_TRACON"
CHANDATA(1)
$MC_AXCONF_GEOAX_ASSIGN_TAB[0]=1
$MC_AXCONF_GEOAX_ASSIGN_TAB[1]=2
$MC_AXCONF_GEOAX_ASSIGN_TAB[2]=3
$MC_TRAFO_RESET_VALUE=0
; Make sure that transformation types 4096 – 4101 are deleted
$MC_TRAFO_TYPE_1=0
$MC_TRAFO_TYPE_2=0
$MC_TRAFO_TYPE_3=0
; Delete transformation chains with OEM transformations
$MC_TRACON_CHAIN_1[0]=0
$MC_TRACON_CHAIN_1[1]=0
; NOTICE! No spaces after M30
M30
```

3. In the OEM directory created under 1., create a file: DPSIM.INI with the following contents:

```
[PRELOAD]
CYCLES=1
CYCLEINTERFACE=0
```

4. Close and launch the HMI application.
5. In the directory of the manufacturer cycles, create a file: TRAORI.SPF with the following contents:

```
PROC TRAORI(INT II)
RET
```

6. In the directory of the manufacturer cycles, create a file: TRACON.SPF with the following contents:

```
PROC TRACON(INT II)
RET
```

Note

The TRAORI.SPF and TRACON.SPF manufacturer cycles created in 5. and 6. must not be loaded onto the NC.

7. Start the simulation.
8. Run a data comparison for the cycles after the simulation has started up:
   - HMI Advanced: **Data comparison > Compare cycles**

     Note

     At least the password for protection level 3 “End user: Service” is needed for the data comparison.

■

3 Supplementary Conditions

No supplementary conditions exist.

■

Notes

4 Data Descriptions (MD, Options)

No data descriptions are necessary.

■

Notes

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

Clearance Control (TE1)

■

1 Brief Description

Description of functions

The “clearance control” technological function is used to maintain a one-dimensional (1D) or three-dimensional (3D) clearance required for technological reasons during a machining process. The clearance to be maintained may be e.g. the distance of a tool from the workpiece surface to be machined.

Function code

The code for the “clearance control” technological function for function-specific identifiers of program commands, machine data, etc. is:

- CLC = Clearance Control

Availability

The “clearance control” technological function is available for the following systems:

- SINUMERIK 840D
- SINUMERIK 840Di

Restrictions

The “clearance control” technological function is subject to the following restrictions:

- **SINUMERIK 840D**
  The technological function can only be activated in the first channel on the NC in SW up to 6.3.

Compile cycle

The “clearance control” technological function is a compile cycle.

System-specific availability and instructions on how to use compile cycles are described in:

(840D)

References: /FB3/ Description of Functions Special Functions
Installation of Compile Cycles TE0

(840Di)

References: /HBI/ SINUMERIK 840Di Manual
NC Installation and Start-Up With HMI Advanced,
Loadable Compile Cycles

■

Notes

Detailed Description 2

2.1 Description of functions

Laser cutting technology is used as an example for the detailed description of the “clearance control” functionality laser cutting.

Laser cutting

During laser cutting, a divergent parallel laser beam is directed across a fiber-optic cable or via a mirror to a light-collecting lens mounted on the laser machining head. The collecting lens focuses the laser beam at its focal point. Typical focal lengths are from 5 to 20 cm.

The position of the focal point in relation to the workpiece is an extremely critical process parameter in laser cutting operations and must be kept constant with a tolerance of $\leq 100\ \mu m$.

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 output voltage of the clearance sensor is approximately proportional to the distance between the sensor and the workpiece surface.

The output voltage of the clearance sensor is transmitted as a digital input value via an analog I/O module to the control where, in the event of deviations from the setpoint clearance, it generates an additional velocity setpoint for the machining head motion axes.

System overview (840D)

An overview of the system components required for clearance control in conjunction with SINUMERIK 840D is provided in Fig. 2-1.

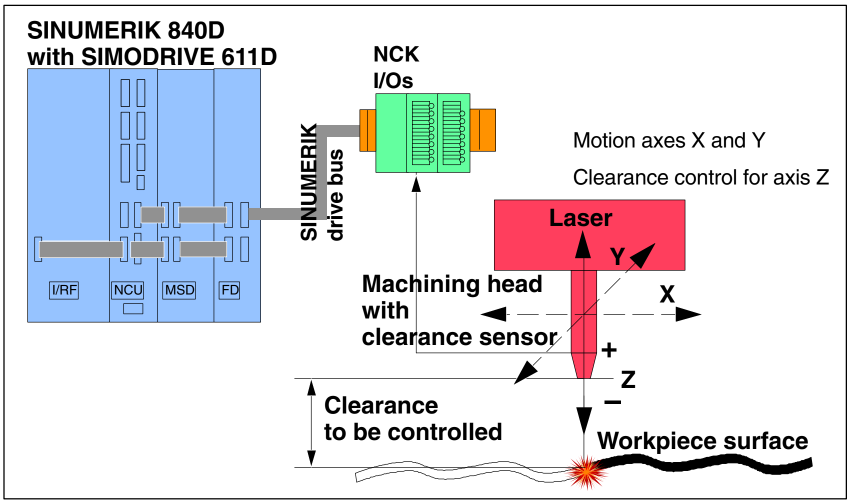


Fig. 2-1 System components for clearance control with SINUMERIK 840D

System overview (840Di)

An overview of the system components required for clearance control in conjunction with SINUMERIK 840Di is provided in Fig. 2-2.

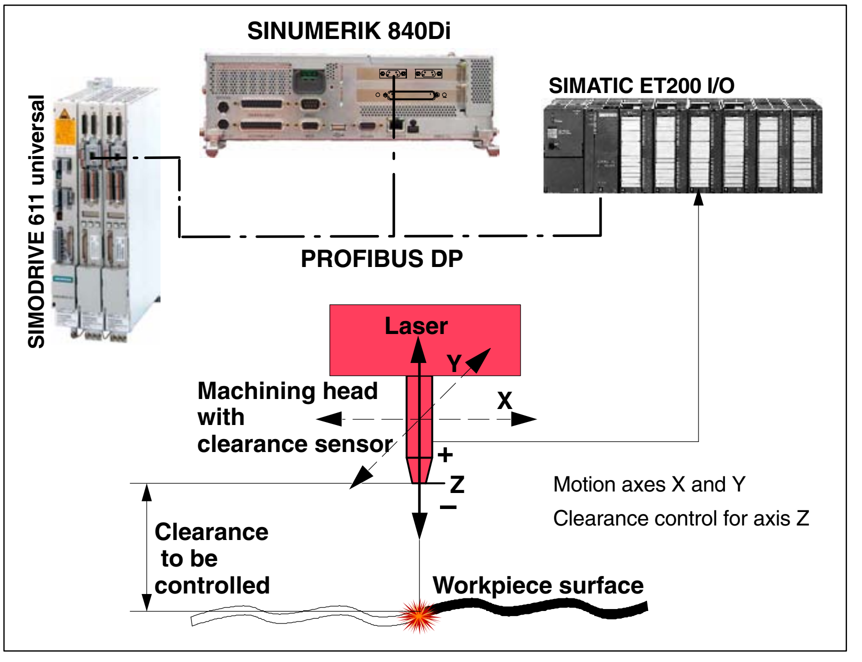


Fig. 2-2 System components for clearance control with SINUMERIK 840Di

1D/3D machining

Clearance control can be used for 1D and 3D machining with up to 5 interpolatory axes.

- 1D machining

  In the case of 1D machining, clearance control is only applied to one axis, e.g. axis Z, as shown in the example machine configuration in the system overview for each SINUMERIK system (see above, Fig. 2-1 and Fig. 2-2). Clearance control acts only in the direction of the Z axis.

- 3D machining

  3 linear axes are used to position the tool. One or two rotary axes are used for the orientation of the tool vector (5-axis machining). Up to 3 linear axes are controlled by the clearance control. The direction of the compensation movement can be defined either in the direction:

  – of the tool orientation vector (normal case)

  – of the programmable compensation vector

2.2 Clearance control

2.2.1 Control dynamics

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:

$$Kv = \frac{\text{Velocity [m/min]}}{\text{following error [mm]}} \quad ; \quad \text{in } \left[\frac{\text{[m/min]}}{\text{[mm]}} \right]$$

Clearance control characteristics

Clearance control is based on the two characteristics shown below in Fig. 2-3:

– Clearance sensor characteristic (sensor property)
– Clearance control characteristic (can be parameterized via machine data)

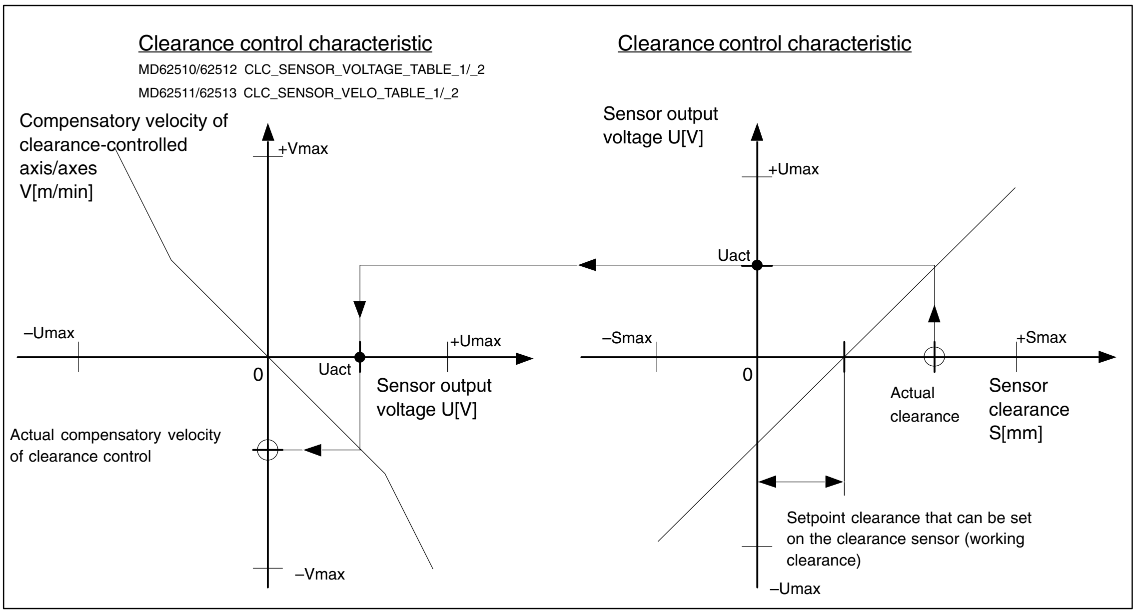


Fig. 2-3 Correlation between characteristics: Clearance sensor and clearance control

- The clearance sensor measures the actual distance from the workpiece surface and returns as its output variable a voltage in [V], which is almost directly proportional to the distance.
- The clearance control function uses the parameterized voltage/velocity characteristic from the voltage provided by the clearance sensor to calculate a compensatory velocity for the clearance-controlled axes that is appropriate for the clearance.

From the point of view of the control, the unit for the closed-loop control gain is [(mm/min)/V]. In the same way as the setpoint clearance in standardized in [mm], values can only be standardized in [(mm/min)/mm] by using the sensor electronics.

Max. closed-loop control gain

The maximum achievable closed-loop control gain is determined by the following delay and reaction times of the overall system:

1. Reaction time of sensor
2. Delay time of A/D converter
3. Signal processing delay times/deadtimes
4. Reaction time of position controller
5. Reaction times of speed and current controllers
6. Time constants of motor and mechanical components

In practice, only items 3 and 4 are relevant.

The influencing variables together produce an effective time constant. A closed-loop control gain set too high based on this time constant will induce natural oscillations in the range of several hertz in the axis/axes to be controlled.

The objective when starting up the clearance control is to minimize important time constants so that the closed-loop control gain required by the process can be set without inducing natural oscillation of this type.

Deadtimes

In order to maximize the dynamics of the control response, clearance control takes place on the highest priority position controller level of the NCK. System-specific distances in the interface between the I/O modules and drives produce the following non-identical deadtimes.

(840D)

SINUMERIK 840D with I/O modules connected to the SINUMERIK drive bus and SIMODRIVE 611D drives produces a deadtime T_{dead} of:

$T_{dead} \approx$ 2 * position controller cycle + 2 * speed controller cycle

(840Di)

SINUMERIK 840Di with I/O modules and drives connected via PROFIBUS-DP produces a deadtime T_{dead} of:

$T_{dead} \approx$ 2 * position controller cycle + 2 * speed controller cycle +
conversion time + channel cycle time +
2 * "PROFIBUS-DP cycle" +
To

- Conversion time, channel cycle time:

  ET 200 S with "2 AI U high-speed" analog electronics module produces the following times:

  – Conversion time 0.1 ms
  – Channel cycle time: 1 ms (both channels)
    => average control deadtime of 0.5 ms

- To

  To is the setpoint transfer time of the drive parameterized in SIMATIC S7: HW-Config.

2.2.2 Velocity feedforward control

Eliminating the delay time

The closed-loop control gain Kv set for the position controller corresponds to a delay time Δt. The delay time Δt is the time that elapses until the actual position of the axis to be controlled correlates with the setpoint position in response to a velocity specification v.

$$\text{With } \Delta t = \frac{1}{Kv}$$

and a closed-loop control gain Kv in seconds:

$$Kv \text{ in } [\frac{m/min}{mm}] = [\frac{1000\, mm/60\, s}{mm}] = 16.667 [\frac{1}{s}]$$

for an assumed closed-loop control gain Kv = 4, the corresponding delay time Δt is:

$$\Delta t = \frac{1}{4 * 16.667 [\frac{1}{s}]} = 14.999\ ms$$

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: Compensations K3,
Following Error Compensation (Feedforward Control)

2.2.3 Control loop structure

The figures below provide an overview of how the clearance control function is embedded in the control loop structure of the NC position controller and the internal structure of the function.

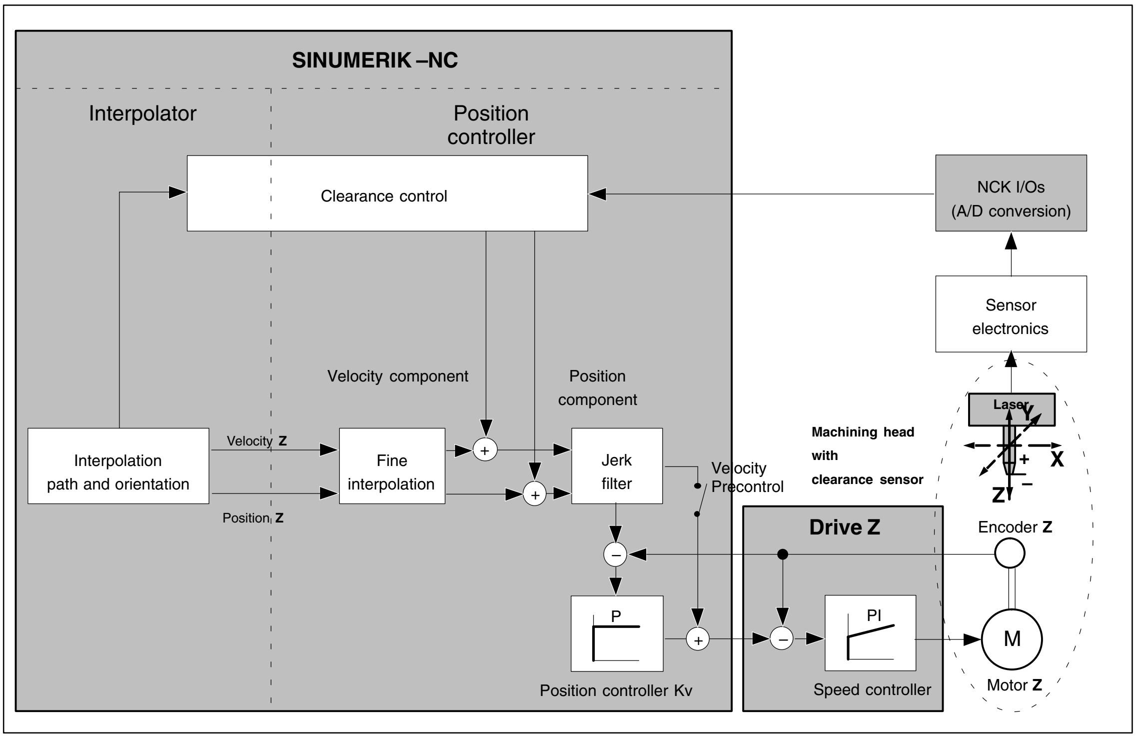


Fig. 2-4 Control structure, position controller with clearance control (principle)

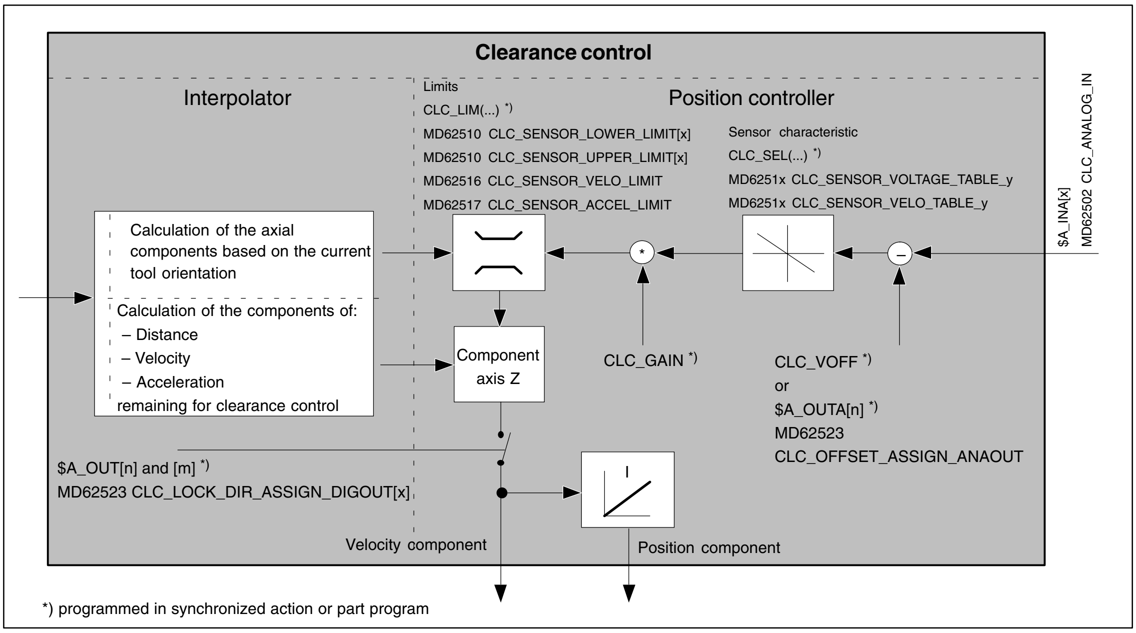


Fig. 2-5 Control structure, clearance control (principle)

2.2.4 Compensation vector

Standard compensation vector

The compensation vector of the clearance control and the tool orientation vector are normally identical. Consequently, the compensation movement of the clearance control is normally always in the direction of the tool orientation.

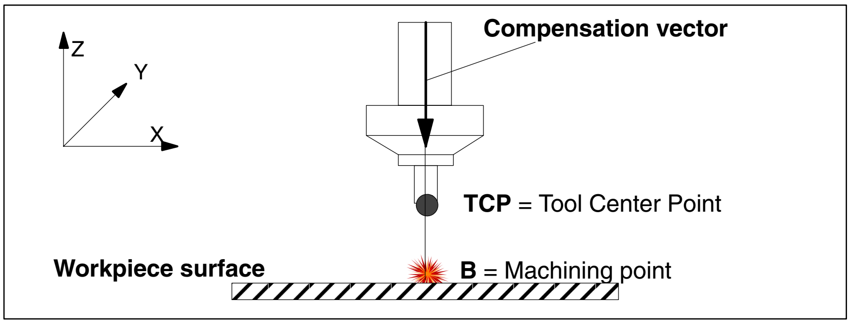


Fig. 2-6 Clearance control with standard compensation vector

Note

In all the figures in this chapter, the traversing movement of the machining head needed in order to machine the workpiece is in the direction of the Y coordinate, i.e. perpendicular to the drawing plane.

As long as the tool orientation, and hence the compensation vector, is perpendicular to the workpiece surface, no disadvantage for the machining process results from the compensation movements of the clearance control.

If a tool setting angle is needed for technological reasons, with the result that the tool orientation is no longer perpendicular to the workpiece surface, the machining point on the workpiece surface is shifted during compensation movements of the clearance control along the standard compensation vector.

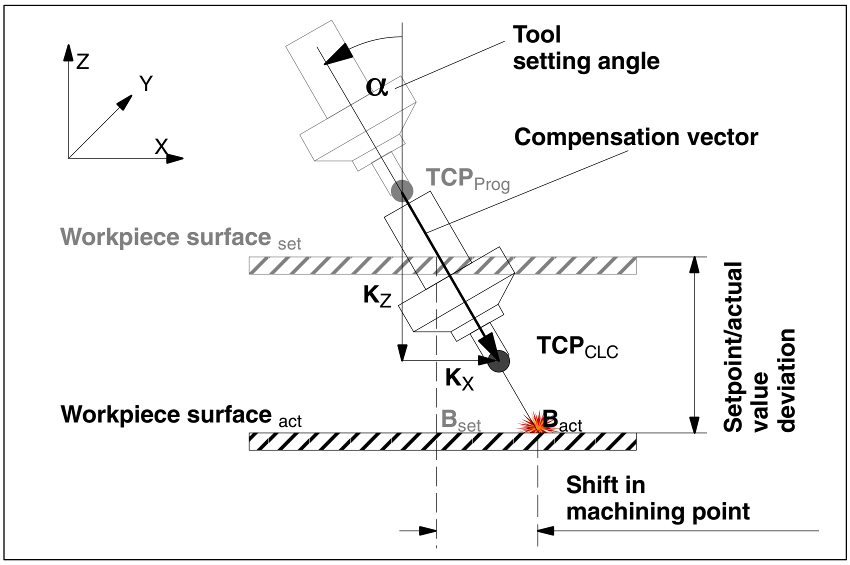


Fig. 2-7 Standard compensation vector

The reason for the shift in the machining point is the X component (K_X) of the compensation vector parallel to the workpiece surface. The TCP of the tool, and thus the machining point B, is shifted by this amount.

Programmable compensation vector

When using the programmable compensation vector, the compensation movements of the clearance control are in the direction of the programmed vector, and not in the direction of the tool orientation.

The X component specified above (K_X) is omitted because the programmable compensation vector is defined perpendicular to the workpiece surface. This does not cause the machining point (B) to be shifted as a result of the compensation movement of the clearance control.

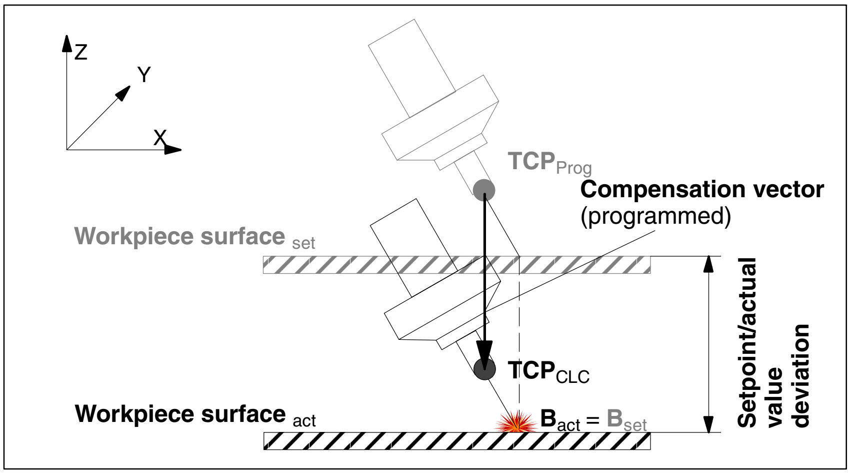


Fig. 2-8 Programmable compensation vector

Changes in orientation

Based on the above observations, a different behavior also results when the orientation of the machining head is changed while the clearance control is active.

In Fig. 2-9: The normal case is shown on the left (compensation vector == tool orientation vector); and the case with the programmed compensation vector is shown on the right.

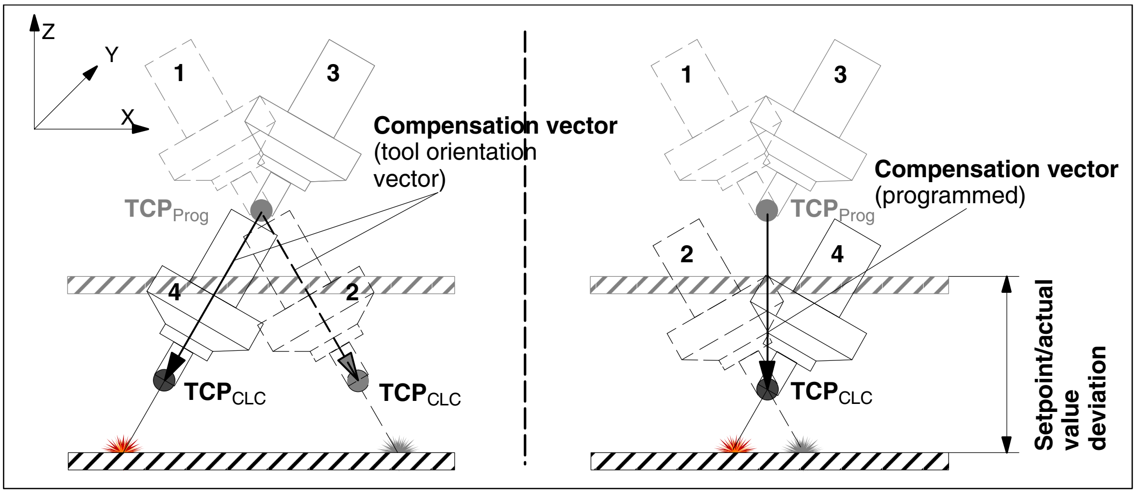


Fig. 2-9 Change in orientation of the machining head

The meaning of the individual positions of the machining head is as follows:

1 Programmed position of the machining head

2 Actual position of the machining head with clearance control active before the orientation change

3 Actual programmed position of the machining head after the change in orientation

4 Position of the machining head with clearance control active after the orientation change

The machining head movement visible on the machine when the change in orientation takes place is direct from position 2 to position 4.

2.3 Technological features of clearance control

Clearance control is characterized by the following technological features:

- **Dynamic response**

  The overlaid sensor motion uses the current residual dynamic response that is still in reserve after the programmed axis motion (velocity and acceleration). The proportion of residual acceleration that must be used can be set as a percentage in a machine data.

- **Sensor characteristic**

  The gain characteristic of a sensor can be defined with up to 10 interpolation points.

- **Sensors**

  Two sensors with different gain characteristics (e.g. a mechanical and a capacitive sensor) can be used simultaneously. The active sensor characteristic can be switched over block-synchronously by means of a language command in the part program.

- **Closed-loop control gain of clearance control**

  The closed-loop control gain configured in the NC machine data for clearance control can be changed block-synchronously by means of a language command in the part program.

- **Motion limitation**

  The lower and upper limits configured in the NC machine data for the axis movements induced by clearance control can be changed block-synchronously by means of a language command in the part program.

  An alarm appears when a limit is reached. The alarm response (stop all traversing movements or display only) can be configured. The current position offset can be frozen by means of a PLC signal.

- **Response on deactivation**

  The deactivation response of the clearance control function can be programmed either for synchronization with the current axis positions (no compensating movement) or for compensating axis movements to the last programmed axis positions (axis positions without clearance control).

- **Programmable clearance setpoint**

  An additional voltage value can be programmed in order to alter the setpoint distance set in the sensor electronics on a block-related basis.

- **Control options via the PLC interface**

  The following signals are available at the PLC interface:

  Status signals:

  – Closed-loop control active
  – Overlaying movement at standstill
  – Lower limit reached
  – Upper limit reached

  Control signals:

  – Path override for sensor movement active

- **Status data of clearance control**

  Both the current values and the min/max values of the sensor signal and of the position offset are available as GUD and/or OPI variables.

- **Sensor signal**

  The sensor signal can be smoothed via a PT1 filter with adjustable time constant.

2.4 Sensor collision monitoring

Sensor signal

If the clearance sensor used has an additional “sensor collision” signal for detecting a collision between the sensor and the workpiece being machined, this signal can be made available to the clearance control function via a digital NCK peripheral input.

In response to this signal, the clearance control function applies a retraction motion in all clearance-controlled axes. The retraction motion is executed independently of the current value of the velocity override with maximum traversing velocity in a positive control direction until the currently valid upper limit of the control range is reached. Path motion is stopped simultaneously.

Once all traversing movements have come to a standstill, part program processing can be resumed with NC START.

Parameterization

The digital peripheral input to which the “sensor collision” signal is wired is assigned to the clearance control function via the following machine data:

- MD62504: CLC_SENSOR_TOUCHED_INPUT (digital peripheral input for “sensor collision” signal)

The digital peripheral input is specified by entering the input number in the same way as $A_IN/$A_OUT digital I/O peripheral system variables are specified ($A_IN[*input number*]).

If a negative input number is entered, the “sensor collision” signal will be processed with internal inversion by the clearance control function (fail safe method).

2.5 Startup

Compile cycle

Before starting up the technological function, make sure that the corresponding compile cycle has been loaded and activated.

(840D)

References: /FB3/ Description of Functions Special Functions
Installation of Compile Cycles (TE01)

(840Di)

References: /HBI/ SINUMERIK 840Di Manual
NC Installation and Start-Up With HMI Advanced,
Loadable Compile Cycles

2.5.1 Activating the technological function

The technological function is activated via the following machine data:

- MD60940: CC_ACTIVE_IN_CHAN_CLC[0], bit n = 1
  where n = channel number – 1; bit0 = 1st channel, bit1 = 2nd channel, etc.

Note

The technological function can be activated for several channels simultaneously.

2.5.2 Configuring the memory

Memory configuration

The technological function requires additional data in the NCK-internal block memory. The following memory-configuring channel-specific machine data must be parameterized:

- MD28090: MM_NUM_CC_BLOCK_ELEMENTS (number of block elements for compile cycles (DRAM)) = x + **4** 1)
- MD28100: MM_NUM_CC_BLOCK_USER_MEM (size of block memory for compile cycles (DRAM) in kBytes) = x + **20** 1)

1) See Note.

Note

The values indicated must be entered in addition to the existing machine data value x.

2.5.3 Parameter settings for input signals (840D)

The following input signals must be parameterized in the machine data:

- Clearance sensor input voltage
  – 1 analog input
- “Sensor collision” input signal (optional)
  – 1 digital input

Analog input

The following machine data must be parameterized for the analog input:

- MD 10300: FASTIO_ANA_NUM_INPUTS
  Number of active analog inputs
- MD 10362: HW_ASSIGN_ANA_FASTIN (for each analog module)
  Specification of the physical address activates the analog input 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. To do this, set the machine data bit with the bit number of the module slot of the analog input module in the terminal block. (Example: Slot 5 –> MD10384 = 10Hex.)
- MD 10380: HW_UPDATE_RATE_FASTIO (for each terminal block)
  Synchronization of the A/D converter with the position controller cycle
  MD 10380 = 2

Digital input

The following machine data must be parameterized for the digital input:

- MD 10350: FASTIO_DIG_NUM_INPUTS
  Number of active digital input bytes
- MD 10366: HW_ASSIGN_DIG_FASTIN (for each digital module)
  Specification of the physical address activates the digital input module

References

A complete description of the analog and digital outputs appears in:

/FB2/ Description of Functions Extended Functions
Digital and Analog NCK I/Os (A4)

2.5.4 Parameter settings for input signals (840Di)

The following input signals must be parameterized in the machine data:

- Clearance sensor input voltage
  – 1 analog input
- “Sensor collision” input signal (optional)
  – 1 digital input

Analog input

The following machine data must be parameterized for the analog input:

- MD 10300: FASTIO_ANA_NUM_INPUTS
  Number of active analog inputs
- MD 10362: HW_ASSIGN_ANA_FASTIN
  I/O address of I/O module

Digital input

Subsequent parameter settings do not have to be made if a digital input on the MCI board extension module (option) is used.

If an I/O module on PROFIBUS-DP is used for the digital input, the following machine data must be parameterized:

- MD 10350: FASTIO_DIG_NUM_INPUTS
  Number of active digital input bytes
- MD 10366: HW_ASSIGN_DIG_FASTIN
  I/O address of I/O module

References

A complete description of the parameter settings for analog and digital I/Os on a SINUMERIK 840Di appears in:

/HBI/ **SINUMERIK 840Di Manual**
NC Installation and Start-Up with HMI Advanced,
Digital and Analog I/Os

A complete description of the analog and digital inputs appears in:

/FB2/ **Description of Functions Extended Functions**
Digital and Analog NCK I/Os (A4)

2.5.5 Parameters of the programmable compensation vector

Reference coordinate system

The programmable compensation vector specifies the direction in which the compensation movement of the clearance control takes place. The compensation vector always refers to the basic coordinate system (machine coordinate system).

The start coordinates [Xa, Ya, Za] of the compensation vector coincide with the origin of the basic coordinate system and are thus always [0, 0, 0].

The end coordinates [Xe, Ye, Ze] of the compensation vector are determined by the actual positions of 3 channel axes, known as the direction axes.

Direction axes

The direction axes must meet the following conditions:

1. The direction axes must be channel axes of the channel in which the clearance control is activated.
2. The direction axes must be linear axes.

   Note
   Since the direction axes are only used to interpolate the direction components, they do not need mechanical axes and can therefore be configured as simulation axes.

3. [mm] or [inch] must be selected as the unit of measurement for the direction axes.
4. The direction axes may not participate in an axis coupling, e.g. transformation, electronic gear, etc.
5. To ensure that the dynamic response of the path is not limited by the dynamic response of the direction axes, the following machine data for the direction axes:
   – MD32000: $MA_MAX_AX_VELO[x]
   – MD32200: $MA_POSCTRL_GAIN[x]
   – MD32230: $MA_MAX_AX_ACCEL[x]
   x = axis number

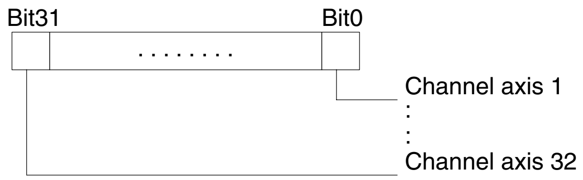
   must be set greater than or equal to the corresponding values of the geometry axes of the channel.

The following machine data is used to specify which channel axis is the direction axis:

- MD62528: $MC_CLC_PROG_ORI_AX_MASK

Each machine data bit corresponds to a channel axis.

Format:



The channel axes are assigned to the end coordinates [Xe, Ye, Ze] of the compensation vector in ascending order of bit numbers.

– Coordinate X = channel axis corresponding to bit a
– Coordinate Y = channel axis corresponding to bit b
– Coordinate Z = channel axis corresponding to bit c
where a < b < c

Current difference angle

The difference angle is the angle between the tool orientation vector and the compensation vector. If the current difference angle of the clearance control is to be output in a system variable $AC_PARAM[n], index n of the system variable should be entered in the following machine data:

- MD65530 $MC_CLC_PROG_ORI_ANGLE_AC_PARAM

Permissible limit angle

The limit angle specifies the maximum difference angle allowed between the tool orientation vector and the compensation vector. The limit angle is configured via the following machine data:

- MD65520 $MC_CLC_PROG_ORI_MAX_ANGLE

2.5.6 Parameter settings for clearance control

Part program identifiers

The following machine data must be parameterized for the declaration of the function-specific part program identifiers CLC_GAIN and CLC_VOFF:

- MD10712: NC_USER_CODE_CONF_NAME_TAB[0] = "OMA1"
- MD10712: NC_USER_CODE_CONF_NAME_TAB[1] = "**CLC_GAIN**"
- MD10712: NC_USER_CODE_CONF_NAME_TAB[2] = "OMA2"
- MD10712: NC_USER_CODE_CONF_NAME_TAB[3] = "**CLC_VOFF**"

1D/3D clearance control

The following machine data is used to select 1D or 3D clearance control:

- MD62500: CLC_AXNO (axis assignment for clearance control) = x
  - x > 0: 1D clearance control where x = axis number of clearance-controlled channel axis
  - x = –1: 1st 5-axis transformation configured in the channel
  - x = –2: 2nd 5-axis transformation configured in the channel

Input signals

The clearance sensor input signals parameterized above:

– 840D: Subsection 2.5.3, page 3/TE1/2-16
– 840Di: Subsection 2.5.4, page 3/TE1/2-16

are declared to the clearance control function via the following machine data:

- MD62502: CLC_ANALOG_IN = x
  x = input number, as when addressing system variables $A_INA[x]
- MD62504: CLC_SENSOR_TOUCHED_INPUT = x
  x = input number, as when addressing system variables $A_IN[x]

Exact stop

In order to be able to meet a programmed "Exact stop coarse/fine reached" block change condition (G601/G602), the traversing velocity induced by the clearance control function in the clearance-controlled axes must be lower than the standstill velocity tolerance at least for the duration of the standstill delay time.

The following machine data must be modified to optimize the block change time:

- MD 36000: STOP_LIMIT_COARSE[x]
- MD 36010: STOP_LIMIT_FINE[x]
- MD 36020: POSITIONING_TIME [x]
- MD 36040: STANDSTILL_DELAY_TIME[x]
- MD 36060: STANDSTILL_VELO_TOL[x]
  x = Axis number of clearance-controlled machine axis

Complete parameterization

Set the remaining clearance control parameters in accordance with the data and signal descriptions in the following sections:

– 4 Data descriptions (MD, SD), page 3/TE1/4-43

– 5 Signal descriptions, page 3/TE1/5-57

2.5.7 Starting up clearance control

Clearance sensor connecting

Connect the clearance sensor outputs to the I/O modules activated via machine data:

- MD 10362: HW_ASSIGN_ANA_FASTIN (I/O addr. of I/O module)
- MD 10366: HW_ASSIGN_DIG_FASTIN (I/O addr. of I/O module)

For more information about I/O modules, see Section 3.1, page 3/TE1/3-39.

Test control direction

Proceed as follows to test the control direction of the clearance control function:

– Activate clearance control via a part program with CLC(1) (see Subsection 2.6.1, page 3/TE1/2-22)

– Generate an input voltage, e.g. via the following synchronized action:

```
N100  $AC_TIMER[1]=2.5
N110  ID = 1 EVERY $AC_TIMER[1] >= 2.5 DO $AC__TIMER[1] = 0
N120  ID = 2 WHENEVER $AC_TIMER[1] < 2.0 DO $A_OUTA[6] = 100000.0 *
      ($AC_TIMER[1] – 1.0)
N130  ID = 3 WHENEVER $AC_TIMER[1] >= 2.0 DO $A_OUTA[6] = 0.0
```

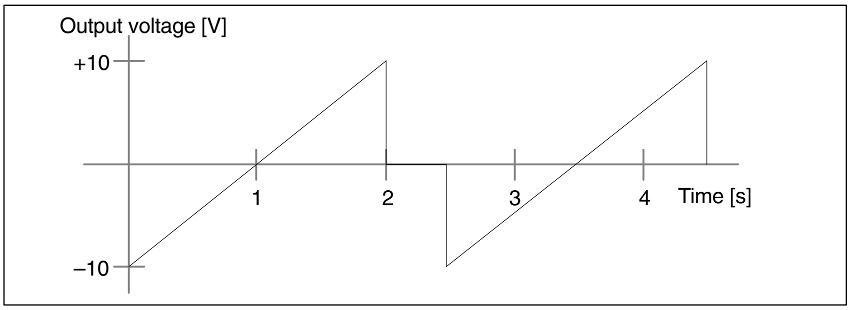


Fig. 2-10 Output voltage for synchronized action

The voltage specification for the analog output $A_OUTA[6] used in the synchronized action is subtracted from the clearance sensor input voltage by the clearance control function and therefore has the opposite polarity to the input signal.

Set the following machine data to induce the clearance control function to use analog output 6 ($A_OUTA[6]) as an additional input overlaid on the sensor:

- MD 10366: CLC_OFFSET_ASSIGN_ANAOUT = 6

Note

Before the clearance control function is activated for the first time, check that the entire working range enabled for clearance control:

- MD62505 CLC_SENSOR_LOWER_LIMIT
- MD62506 CLC_SENSOR_UPPER_LIMIT

is free of collisions.

An incorrect control direction can be corrected using one of the following methods:

- Reversing the polarity of the analog input
- Changing the sign of all values in the following machine data:
  - MD62511 CLC_SENSOR_VELO_TABLE_1
  - MD62513 CLC_SENSOR_VELO_TABLE_2

Sensor signal check

The quality of the analog input signal can be checked using one of the function-specific display data described in Section 2.7, page 3/TE1/2-33.

Function-spec. alarm texts

Function-specific alarm texts must first be integrated into the appropriate HMI data management before they can be displayed. A description of how to do this appears in Section 2.8, page 3/TE1/2-36.

Completion

A data backup is recommended once the start-up procedure has been completed.

(840D)

References:
/IAD/ SINUMERIK 840D Installation & Start-Up Guide 840D
Data Backup

(840Di)

References:
/HBI/ SINUMERIK 840Di Manual
User Data Backup/Series Installation and Start-Up

Note

A data backup is recommended once the start-up procedure has been completed.

2.6 Programming

2.6.1 Activating and deactivating clearance control (CLC)

Syntax

CLC(*Mode*)

Mode
- Format: Integer
- Value range: –1, 0, 1, 2, 3

CLC(...) is a procedure call and must therefore be programmed in a dedicated part program block.

Functionality

The following modes are available for activating/deactivating clearance control:

- CLC(1)

  Activation of the clearance control with compensation vector in the direction of the tool orientation

  The evaluation of the sensor collision signal is deactivated.

- CLC(2)

  Activation of the clearance control with compensation vector in the direction of the tool orientation

  The evaluation of the sensor collision signal is activated.

- CLC(3)

  Activation of the clearance control with programmed compensation vector

  The evaluation of the sensor collision signal is deactivated.

- CLC(0)

  Deactivation of clearance control without canceling the position offset

  If the clearance-controlled axes are still moving at the instant of deactivation due to the sensor signal, they are stopped. The workpiece coordinate system (WCS) is then synchronized with the corresponding standstill positions. An automatic preprocessor stop is executed.

- CLC(–1)

  Deactivation of clearance control with cancellation of the position offset

  If the clearance-controlled axes are still moving at the instant of deactivation due to the sensor signal, they are stopped. A position offset to the last programmed position is canceled automatically with the deactivation command.

RESET response

CLC(0) is executed implicitly on a RESET (NC RESET or end of program).

Parameterizable RESET response

The reset response of a 1D clearance control function can be defined via the channel-specific NCK OEM machine data:

- MD62524: CLC_ACTIVE_AFTER_RESET (reset response with active CLC)

The following response can be parameterized:

- MD62524: CLC_ACTIVE_AFTER_RESET = 0
  In the event of a RESET, the clearance control function responds as it does to deactivation with CLC(0) (see above: activation/deactivation modes).
- MD62524: CLC_ACTIVE_AFTER_RESET = 1
  The current state of the clearance control function is retained.

⚠

Caution

The channel-specific NCK OEM machine data:

- MD62524: CLC_ACTIVE_RESET (reset response with active CLC)

is only effective in conjunction with 1D clearance control. For 3D clearance control, CLC(0) is always effective in the event of a RESET.

Supplementary conditions

Please note the following supplementary conditions:

Continuous-path mode

Activating/deactivating clearance control (CLC(*Mode*)) during active continuous-path mode (G64/G64x) will induce a drop in velocity for path motions. To avoid voltage drops of this type, clearance control must be activated before a path section with constant path velocity. During the corresponding path section, if necessary, clearance control can be blocked and then re-enabled via the programmed gain factor for clearance control (CLC_GAIN).

Block change with exact stop

If exact stop is active at the end of the block (G60/G09 with G601/G602) the block change may be delayed due to axis movements induced by the clearance control sensor signal.

Sensor collision monitoring

Machine data MD 62504: CLC_SENSOR_TOUCHED_INPUT can be used to configure a digital input for an additional collision signal from the sensor. This collision monitor can be activated and deactivated block-synchronously through alternate programming of CLC(1)/CLC(2).

As a reaction to the sensor collision signal, the clearance control moves, irrespective of the feedrate override setting, at maximum velocity in the plus direction until it reaches the currently valid upper limit. Path motion is stopped simultaneously.
NC START can be used to resume machining.

3D clearance control and 5-axis transf.

If 3D clearance control is activated before the 5-axis transformation required for clearance control in the direction of the tool orientation has been activated, the clearance control function will be dependent on the active working plane (G17/G18/G19):

- G17: Direction of clearance control = Z
- G18: Direction of clearance control = Y

– G19: Direction of clearance control = X

- Activation of 5-axis transformation

  When 5-axis transformation is activated, the tool orientation specified by means of the rotary axis positions must tally with the control direction specified by the active working plane on activation of clearance control.

  If the tool orientation of the 5-axis transformation and the control direction of the clearance control function do not tally, the following CLC alarm will appear:

  - Alarm “75016 Channel *number* Block *number* CLC: Orientation changed with TRAFOOF”

- Deactivation of 5-axis transformation

  If 5-axis transformation is deactivated when clearance control is active, the last control direction before 5-axis transformation was deactivated is retained:

Tool radius compensation

3D clearance control can only be deactivated if no tool radius compensation is active in the channel at the instant of deactivation (G40).
If tool radius compensation is active (G41/G42), the following alarm appears:

- Alarm “75015 Channel *number* Block *number* CLC(0) with active TRC.”

Compensation vector

Actual position of the direction axes
If the clearance control is activated with a programmable compensation vector at a position of 0 on all 3 direction axes, a compensation vector cannot be calculated from this information. The following alarm is then displayed:

- Alarm “75019 Channel *number*, error ID: 1, angle 0.0”

Referencing of the direction axes
The direction axes must be referenced before clearance control is activated with programmable compensation vector CLC(3).

Interface signals of the direction axes
The following interface signals must be set for all 3 direction axes by the PLC user program, before clearance control is activated with programmable compensation vector CLC(3).

- DBB3x.DBX1.5 = 1 (position measuring system 1)
- DBB3x.DBX2.1 = 1 (servo enable)
- DBB3x.DBX21.7 = 1 (pulse enable)
  x = axis number

Switchover of clearance control
Direct switchover of clearance control from CLC(1) or CLC(2) to CLC(3) or vice-versa is not possible. Such switchovers are ignored without a checkback message. If a switchover is necessary, the clearance control must first be deactivated with CLC(0) or CLC(–1) and then activated in the desired mode.

Interpolation of the compensation vector
If the compensation vector is required to follow a non-linear workpiece surface, such as an arc, with respect to its orientation, this can be achieved by programming the direction axes.

Example
Orientation of the compensation vector perpendicular to a semi-circular workpiece surface. The programming of the traversing movement is not considered.

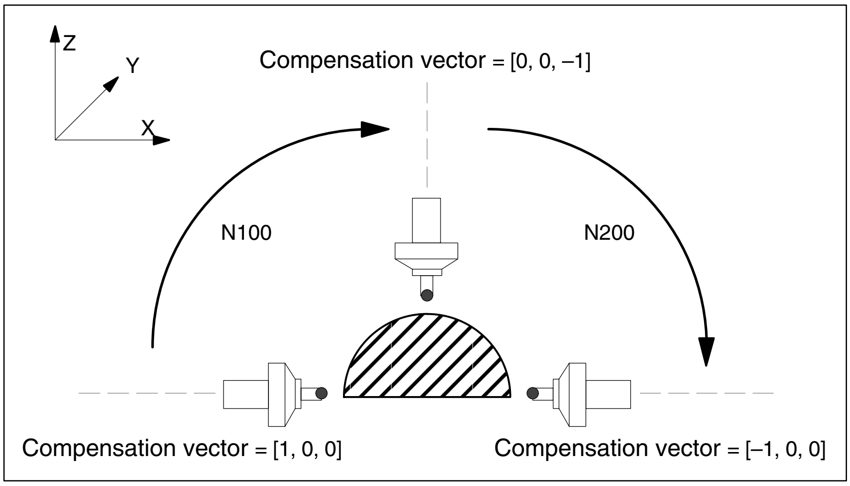


Fig. 2-11 Interpolation of the compensation vector

The compensation vector must be oriented by programming the direction axes at [1, 0, 0] before part program block N100. In part program block N100, the end position of the compensation vector is oriented by programming the direction axes at [0, 0, –1].

The intermediate values are generated by path interpolation of all axes programmed in the part program block:

- Geometry axes for the movement of the machining head
- Direction axes of the compensation vector

It is necessary to break the movement down into part program blocks N100 and N200, because an antiparallel orientation of the compensation vector of [1, 0, 0] at the start of the movement and [–1, 0, 0] at the end of the movement (semicircle) would otherwise result. In this case, the interpolator would interpolate only the X coordinate of the compensation vector, and the orientation of the compensation vector would remain unchanged.

Antiparallel orientation of the compensation vector
When an antiparallel orientation of the compensation vector is programmed in a part program block, the following alarm is displayed:

- Alarm “75018 Channel *number* Block *number* CLC in programmable direction, error ID: 1”

Further information about the interpolation of the compensation vector
The interpolation of the compensation vector is not a genuine vector interpolation, as described above, but results from the interpolation of the actual positions of the direction axes.

Consequently, if the compensation vector changes due to the workpiece contour, the interpolation of the direction axes is included in the path interpolation of the geometry axes. In order to minimize the impact of the direction axes on the path interpolation, it is recommended to configure the dynamic response of the direction axes at least equal to or greater (by a factor of approx. 10) than the dynamic response of the geometry axes.

In the case of a re-orientation (rotation) of the compensation vector, it is also necessary to note the ratio between the programmed traversing path and the configured dynamic response of the direction axes. The ratio should be chosen such that the programmed traversing path is not traversed in one or a small number of interpolation cycles, due to the dynamic response of the axis. This causes heavy loads on the machine and, in certain circumstances, may trigger axial alarms and abort part program execution.

Example

Rotation of the compensation vector and thus the machining head through 90°:

– Initial orientation: Parallel to coordinate axis X
– Target orientation: Parallel to coordinate axis Y

Bad programming of re-orientation:

– [1, 0, 0] –> [0, 1, 0]

Good programming of re-orientation:

– [100, 0, 0] –> [0, 100, 0]

Rotation of the workpiece coordinate system

As described above, the compensation vector always refers to the basic coordinate system (machine coordinate system). If the workpiece coordinate system (rotation, mirroring) is transformed to machine a workpiece in such a way that the coordinate axes of both coordinate systems are no longer parallel with the same orientation, a corresponding transformation must be carried out for the compensation vector.

Caution

If the workpiece coordinate system is transformed such that the coordinate axes of the basic and workpiece coordinate systems are no longer parallel with the same orientation, it is the sole responsibility of the user to ensure that an appropriate transformation of the compensation vector is carried out.

2.6.2 Closed-loop control gain (CLC_GAIN)

Syntax

CLC_GAIN = *Factor*

Factor

- Format: Real
- Value range: ≧ 0.0

CLC_GAIN is an NC address and can therefore be written together with other instructions in a part program block.

When a negative factor is programmed, the absolute value is used without an alarm output.

Functionality

The current closed-loop control gain for clearance control is produced by the active characteristic specified via machine data:

- MD62510 CLC_SENSOR_VOLTAGE_TABLE1
- MD62511 CLC_SENSOR_VELO_TABLE1

or

- MD62512 CLC_SENSOR_VOLTAGE_TABLE2
- MD62513 CLC_SENSOR_VELO_TABLE2

CLC_GAIN can be used to multiply the closed-loop control gain of the characteristic by a programmable factor.

Caution

Increasing the gain (CLC_GAIN > 1.0) may lead to oscillation in the controlled axes!

Instant of activation

The modified closed-loop control gain is effective in the part program block in which CLC_GAIN has been programmed or, if this block does not contain any executable instructions, in the next part program block with executable instructions.

Response to characteristic changeover

The programmed factor remains active even when the gain characteristic is changed over with CLC_SEL, i.e. it is immediately applied to the newly selected characteristic.

Response to CLC_GAIN=0.0

If the closed-loop control gain for clearance control is deactivated with CLC_GAIN=0.0, the CLC position offset present at the instant of deactivation is retained and is not changed. This can be used for example when laser-cutting sheet steel to “skip over” sections of sheet that are not to be machined without foundering.

If the tool orientation is changed when 3D clearance control is active and the closed-loop control gain has been deactivated (CLC_GAIN=0.0), the CLC offset vector is rotated simultaneously. This generally induces an offset in the CLC operating point on the workpiece surface (see Fig. 2-12).

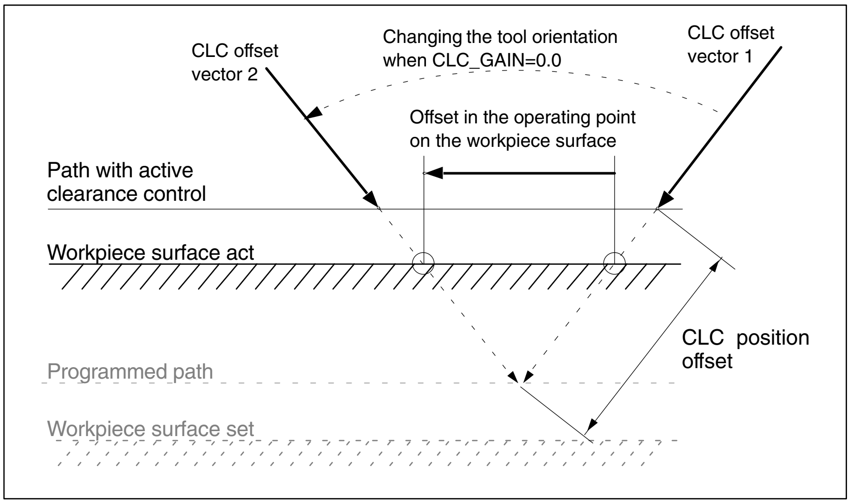


Fig. 2-12 Response of the CLC offset vector when CLC_GAIN=0.0

Reset

Within a part program, a modified gain factor must be reset by means of explicitly programming CLC_GAIN=1.0.

RESET response

CLC_GAIN=1.0 becomes effective after a POWER ON RESET, NC RESET or end of program.

2.6.3 Limiting the control range (CLC_LIM)

Syntax

CLC_LIM(*lower limit*, *upper limit*)

Lower limit, *upper limit*

Format and value range as machine data:

- MD62505 CLC_SENSOR_LOWER_LIMIT[n]
- MD62506 CLC_SENSOR_UPPER_LIMIT[n]

CLC_LIM(...) is a procedure call and must therefore be programmed in a dedicated part program block.

Functionality

The maximum control range for clearance control can be modified on a block-specific basis using CLC_LIM. The maximum programmable lower/upper limit is limited by the limit value preset in the relevant machine data:

- MD62505 CLC_SENSOR_LOWER_LIMIT[1]
- MD62506 CLC_SENSOR_UPPER_LIMIT[1]

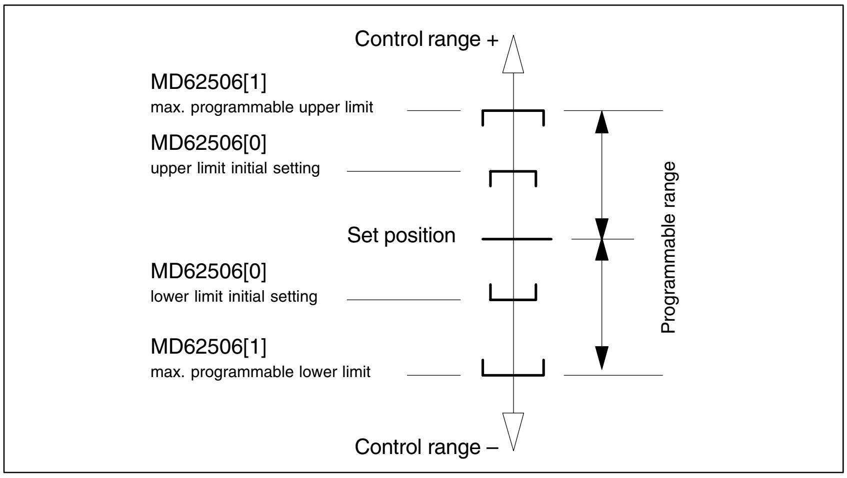


Fig. 2-13 Value range limits for lower and upper limit

The control range limit is effective in relation to the current programmed setpoint position of the axis. If the limits are changed so that the actual position is located outside the limit, the clearance control automatically effects travel back to the limit range.

Reset

Within a part program, a modified control range limit can be reset by explicitly programming CLC_LIM without a “CLC_LIM( )” argument. This reapplies the limits from the machine data:

- MD62505 CLC_SENSOR_LOWER_LIMIT[0]
- MD62506 CLC_SENSOR_UPPER_LIMIT[0]

RESET response

The default setting from the machine data becomes effective after POWER ON RESET, NC RESET and end of program.

Error messages

The following programming errors are displayed with an alarm:

- Programming more than 2 arguments
  - CLC alarm “75005 Channel *number* Block *number* CLC_LIM: general programming error”
- Programming arguments outside the permissible limits
  - CLC alarm “750010 Channel *number* Block *number* CLC_LIM Value greater than MD limit”

2.6.4 Direction-dependent traversing motion disable

Syntax

$A_OUT[*number*] = *enabling signal*

Number

Number of the parameterized digital output (see below: Parameterization)

- Format: Integer
- Value range: 1, 2, . . . max. number of digital outputs

Enabling signal

Enabling signal, can be inverted (see below: Parameterization)

- Format: Integer
- Value range: 0, 1

System variable $A_OUT[n] can be set block-synchronously in the part program or asynchronously via synchronized actions.

Functionality

Parameterizable digital outputs (system variable $A_OUT) can be used for direction-dependent disabling of the traversing motion (manipulated variable) induced via clearance control. As long as e.g. the negative traversing direction is disabled, the clearance-controlled axes will only travel in a positive direction due to the sensor signal.

This can be used for example when laser-cutting sheet steel to “skip over” sections of sheet that are not to be machined without foundering.

Parameterization

The following machine data is used to parameterize the digital outputs:

- MD62523: CLC_LOCK_DIR_ASSIGN_DIGOUT[n] “Assignment of digital outputs to interlock CLC motion”

  n = 0 –> Digital output for disabling the negative traversing direction

  n = 1 –> Digital output for disabling the positive traversing direction

Example

The following digital outputs are to be used:

- $A_OUT[3] to disable the negative traversing direction
- $A_OUT[4] to disable the positive traversing direction

Parameter settings in the machine data:

- MD62523: CLC_LOCK_DIR_ASSIGN_DIGOUT[0] = 3
- MD62523: CLC_LOCK_DIR_ASSIGN_DIGOUT[1] = 4

Effect:

- $A_OUT[3] = 0 –> Negative traversing direction enabled
- $A_OUT[3] = 1 –> Negative traversing direction disabled
- $A_OUT[4] = 0 –> Positive traversing direction enabled
- $A_OUT[4] = 1 –> Positive traversing direction disabled

Inversion of the evaluation

Enter the negative number of the digital output to evaluate the digital output signal with inversion:

Parameter settings in the machine data:

- MD62523: CLC_LOCK_DIR_ASSIGN_DIGOUT[0] = –3
- MD62523: CLC_LOCK_DIR_ASSIGN_DIGOUT[1] = –4

Effect:

- $A_OUT[3] = 0 –> Negative traversing direction disabled
- $A_OUT[3] = 1 –> Negative traversing direction enabled
- $A_OUT[4] = 0 –> Positive traversing direction disabled
- $A_OUT[4] = 1 –> Positive traversing direction enabled

2.6.5 Voltage offset, can be set on a block-specific basis (CLC_VOFF)

Syntax

CLC_VOFF = V*oltage offset*

V*oltage offset*
- Format: Real
- Unit: Volts
- Value range: No restrictions

CLC_VOFF is an NC address and can therefore be written together with other instructions in a part program block.

Functionality

CLC_VOFF can be used to preset a constant voltage offset for clearance control, which is subtracted from the input voltage of the clearance sensor. The programmed voltage offset therefore changes the setpoint distance between the workpiece and the clearance sensor or offsets the operating point for clearance control.

The quantitative effect of the voltage offset is dependent on the additional parameters for clearance control and can therefore not be standardized in a generally valid format.

Instant of activation

The voltage offset is effective in the part program block in which CLC_VOFF has been programmed or, if this block does not contain any executable instructions, in the next part program block with executable instructions.

Reset

Within a part program, a voltage offset must be reset by means of explicitly programming CLC_VOFF=0.0.

RESET response

CLC_VOFF =0.0 becomes effective after a power on reset, NC RESET or end of program.

2.6.6 Voltage offset definable by synchronized action

Syntax

$A_OUTA[*number*] = V*oltage offset*

Number

Number of the parameterized analog output (see below: Parameterization)
- Format: Integer
- Value range: 1, 2, . . .max. number of analog outputs

V*oltage offset*

As voltage offset for CLC_VOFF (Subsection 2.6.5, page 3/TE1/2-31).

Functionality

A parameterizable output (system variable $A_OUTA) can be used to apply a voltage offset for clearance control, which, like CLC_OFF, is subtracted from the input voltage of the clearance sensor.

The voltage offset can be modified in the interpolation cycle by programming the analog output within a synchronized action.

Parameterization

The following machine data is used to parameterize the analog output:

- MD62522: CLC_OFFSET_ASSIGN_ANAOUT "Modification of the setpoint distance by means of sensor signal override"

Example

An external voltage Uext is present at analog input $A_INA[3], which is to be overlaid on the sensor voltage as a continuously variable voltage offset e.g. for test or start-up purposes. $A_OUTA[2] is used as an analog output for the clearance control voltage offset.

Parameter setting for the analog output for clearance control voltage offset:

– MD62522: CLC_OFFSET_ASSIGN_ANAOUT = 2

The analog input $A_INA[3] is assigned to the clearance control analog output $A_OUTA[2] within a synchronized action:

ID=1 DO $A_OUTA[2] = $A_INA[3]

2.6.7 Selection of the active sensor characteristic (CLC_SEL)

Syntax

CLC_SEL(*characteristic number*)

Characteristic number

– Format: Integer
– Value range: 1, 2

CLC_SEL(...) is a procedure call and must therefore be programmed in a dedicated part program block.

Characteristic number = 2 selects characteristic 2. Any other value selects characteristic 1 without alarm.

Functionality

CLC_SEL can be used to switch between the sensor characteristics defined in the machine data.

Characteristic 1:

- MD 62510: CLC_SENSOR_VOLTAGE_TABLE_1
- MD 62511: CLC_SENSOR_VELO_TABLE_1

Characteristic 2:

- MD 62512: CLC_SENSOR_VOLTAGE_TABLE_2
- MD 62513: CLC_SENSOR_VELO_TABLE_2

RESET response

Characteristic 1 becomes effective after a power on reset, NC RESET or end of program.

2.7 Function-specific display data

The “clearance control” technological function provides specific display data for supporting start-up and for service purposes.

Application options

Application options for display data include for example:

- Determination of form variances and transient control errors via the variables for the maximum and minimum position offset/sensor voltage.
- Determination of the voltage noise detected by the A/D converter via the variables for the maximum and minimum sensor input voltage. This requires a constant clearance between the clearance sensor and the workpiece surface and the deactivation of clearance control via CLC_GAIN = 0.0.

The minimum and maximum values are detected in the position controller cycle.

Types of variable

The display data are available both as channel-specific GUD (Global User Data) and as OPI variables.

2.7.1 Channel-specific GUD variables

The “clearance control” technological function provides the following channel-specific GUD variable for HMI applications:

- SINUMERIK HMI Advanced
- SINUMERIK HMI Embedded

as a display data:

Table 2-1 Channel-specific GUD variables

GUD variables	Description	Unit	Access
CLC_DISTANCE[0]	Current position offset	mm	read only
CLC_DISTANCE[1]	Absolute minimum of position offset	mm	read/write
CLC_DISTANCE[2]	Absolute maximum of position offset	mm	read/write
CLC_VOLTAGE[0]	Current sensor input voltage	V	read only
CLC_VOLTAGE[1]	Absolute minimum of sensor input voltage	V	read/write
CLC_VOLTAGE[2]	Absolute maximum of sensor input voltage	V	read/write

Once the technological function has been started up successfully, the GUD variables listed are not displayed automatically on the HMI interface.

HMI Advanced

Proceed as follows to create and display the GUD variables in HMI Advanced.

1. Set password
   Enter the password for protection level 1: (machine manufacturer).

2. Activate the “definitions” display.

   Operating area switchover > Services > Data Selection

3. If no SGUD.DEF file is yet available:

   Operating area switchover > Services > Data admin > New...

   – Name: SGUD

   – Type: Global data/system

   Confirm with **OK**.

   This opens the file in the editor.

4. Edit the GUD variable definitions

```
DEF CHAN REAL CLC_DISTANCE[3]   ; Array of real, 3 elements
DEF CHAN REAL CLC_VOLTAGE[3]    ; Array of real, 3 elements
M30
```

5. Save the file and close the editor.

6. Activate the SGUD.DEF file.

The GUD variables for clearance control are now displayed under:

Operating area switchover > Parameters > User data > Channel user data

HMI Embedded

Proceed as follows to create and display the GUD variables in HMI Embedded.

1. Set password

   Enter the password for protection level 1: (machine manufacturer).

2. If no SGUD.DEF file is yet available:

   Operating area switchover > Services > Data admin > Programs/Data:
   Set cursor to definitions **> New...**

   – Name: SGUD

   – Type: DEF

   Confirm with **OK**.

   This opens the file in the editor.

3. Edit the GUD variable definitions

```
DEF CHAN REAL CLC_DISTANCE[3]   ; Array of real, 3 elements
DEF CHAN REAL CLC_VOLTAGE[3]    ; Array of real, 3 elements
M30
```

4. Save the file and close the editor.

5. Activate the SGUD.DEF file.

The GUD variables for clearance control are now displayed under:

Operating area switchover > Parameters > User data > Channel-spec. user data

SINUMERIK NCK

The new GUD variables, which are already being displayed, will only be detected by the RESU function and supplied with up-to-date values following an NCK POWER ON RESET.

Note

Once the GUD variables have been created, an NCK POWER ON RESET must be carried out in order for the clearance control function to update the GUD variables.

2.7.2 OPI variable

The “clearance control” technological function provides the following channel-specific OPI variables as display data for the HMI application:

– SINUMERIK HMI Advanced

as a display data:

Table 2-2 Channel-specific OPI variable

OPI variable	Description	Unit	Access
CLC[0]	Current position offset	mm	read only
CLC[1]	Absolute minimum of position offset	mm	read/write
CLC[2]	Absolute maximum of position offset	mm	read/write
CLC[3]	Current sensor input voltage	V	read only
CLC[4]	Absolute minimum of sensor input voltage	V	read/write
CLC[5]	Absolute maximum of sensor input voltage	V	read/write
CLC[6]	1. component of the standard-ized tool orientation vector	–	read only
CLC[7]	2. component of the standard-ized tool orientation vector	–	read only
CLC[8]	3. component of the standard-ized tool orientation vector	–	read only

Once the technological function has been started up successfully, the OPI variable is not available automatically.

OPI variable definition

Proceed as follows to define the OPI variables.

1. Create the CLC-specific definition file: **CLC.NSK**

   Note
   We recommend that you create the file in the \OEM directory rather than in the \MMC2 directory so that it is not overwritten when a new software version is installed.

2. Define the CLC-specific OPI variables.

   Add the following line to the CLC.NSK file:

   LINK(“CLC” ,200, 2 1 1 1 1F# /NC 5 0 1”,100)

3. Create/expand the user-specific definition file: **USER.NSK**

   See 1: Note

4. In file USER.NSK, supplement the call of the CLC-specific definition file: CLC.NSK. To do this, insert the following line:

```
CALL(CLC.NSK)
```

Using LinkItem

In order to use the OPI variables in a DDE control, the “LinkItem” property of the DDE control must be set in accordance with the following example:

```
label1.LinkItem = “CLC[u1,1,9](“ ”!d%15.4lf“ ”)”
```

The format string can be modified if necessary.

The following code lines provide an example of how the variables supplied by means of NCDDE access can be distributed on a field of labels:

```
For i = 0 To 8
    label2.Caption[ i ] =  Trim$(Mid$(label1.Caption, 1 + 15 * i, 15) )
Next
```

2.8 Function-specific alarm texts

The “clearance control” technological function supports the output of function-specific language-dependent alarm texts. The corresponding alarm texts must be created in language-specific alarm text files and declared to the HMI application.

References

A detailed description of how to incorporate new alarms appears in:

SINUMERIK **HMI Embedded**

/IAM/ IBN HMI/MMC
IM2 Installation and Start-Up HMI Embedded
Chapter: Alarm Texts and Help Files

SINUMERIK **HMI Advanced**

/IAM/ IBN HMI/MMC
IM4 Installation and Start-Up HMI Advanced
Chapter: Alarm Texts and Help Files

German alarm texts

Recommended German alarm texts:

075000 0 0 “Kanal %1 CLC: Incorrect MD configuration, error no: %2”
075005 0 0 “Kanal %1 Satz %2 CLC: general programming error”
075010 0 0 “Kanal %1 Satz %2 CLC_LIM Wert größer als MD-Grenze”
075015 0 0 “Kanal %1 Satz %2 CLC(0) bei aktiver WRK”
075016 0 0 “Kanal %1 Satz %2 CLC: Orientierung geändert bei TRAFOOF”
075020 0 0 “Kanal %1 CLC-Positionsversatz an unterer Begrenzung: %2”
075021 0 0 “Kanal %1 CLC-Positionsversatz an oberer Begrenzung: %2”
075025 0 0 “Kanal %1 CLC gestoppt wegen Sensorkopf berührt”

English alarm texts

Recommended English alarm texts:

075000 0 0 “Channel %1 CLC: machine data 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 during 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”

■

Notes

Supplementary Conditions 3

3.1 I/O modules

For A/D conversion, the analog output current of the clearance sensor must be connected to NC via an I/O module with analog input to the NC.

3.1.1 I/O modules (840D)

The analog I/O module (DMP compact module) is connected to the drive bus via an NCU terminal block.

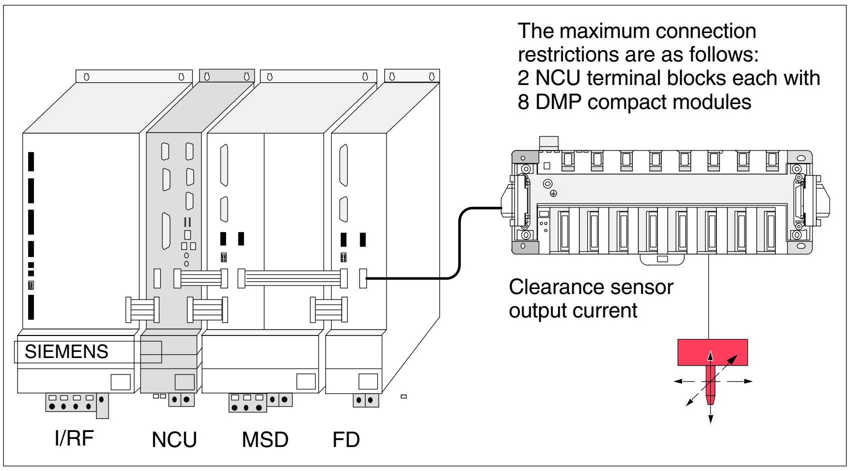


Fig. 3-1 Clearance sensor connection via analog DMP module

Suitable I/O modules

As the A/D conversion time directly affects the deadtime of the clearance control servo loop, only one I/O module may be used with low conversion time.

An I/O module suitable for clearance control is:

- DMP compact module 1I, NC analog
  Conversion time: 75 µs
  Order number (MRPD): 6FC5 211–0AA10–0AA0

I/O module connection

A description of how the I/O modules are connected appears in:

References: /PHD/ SINUMERIK 840D Configuration Manual (Hardware)
Chapter: Terminal Block
NCU Terminal Block 6FC5 211–0AA00–0AA0
DMP Compact Modules
DMP Compact Module 1E NC Analog 6FC5 211–0AA10–0AA0

3.1.2 I/O modules (840Di)

On the SINUMERIK 840Di, the analog I/O module is connected via PROFIBUS-DP.

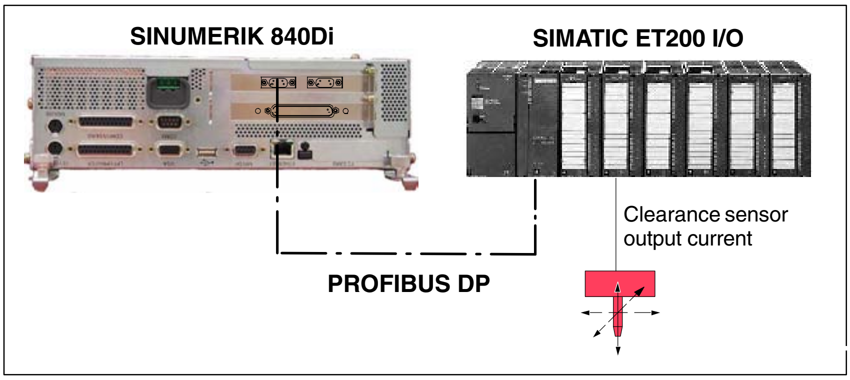


Fig. 3-2 Clearance sensor connection via analog S7 I/O module

Suitable I/O modules

As the A/D conversion time directly affects the deadtime of the clearance control servo loop, only one I/O module may be used with low conversion time.

A SIMATIC S7 module suitable for clearance control is:

- Analog I/O module 2 AI, U, high-speed for ET 200S
  Conversion time per channel (max. 2): 100 µs
  Cycle time for both channels: 1 ms
  Order number (MRPD): 6ES7 134-4FB50-0AB0

I/O module connection

A description of how the I/O modules are connected appears in:

References: /HBI/ SINUMERIK 840Di Manual
Chapter: SIMATIC S7 I/O Devices

3.1.3 External smoothing filters

If an external filter is to be interconnected to smooth the output voltage of the clearance sensor before the A/D conversion of the output voltage by the I/O module, please ensure that the resulting time constant is small in relation to the NC position controller cycle.

Note

It is better for the control if electromagnetic shielding is used to ensure a large signal-noise ratio than if smoothing filters are used in the signal path.

3.2 Function-specific boundary conditions

NC stop from PLC

If, in addition to the programmed path motion, the traversing movement of the clearance-controlled axes is also to be stopped when the NC stops, the “NC stop axes and spindles” interface signal must also be set in addition to the “NC stop” interface signal:

- DB21, DBB7.3 “NC stop”
- DB21, DBB7.4 “NC Stop axes and spindles”

Follow-up

If a clearance-controlled axis is to be switched as an alarm response or via the corresponding interface signal from the PLC in “follow-up” mode, setpoint output will cease for clearance control on this axis.

Travel without software limit switches

If the clearance-controlled axes are to travel without referencing (travel without software limit switches), values outside the traversing range used must be entered in the corresponding machine data for the axis-specific software limit switches:

- MD36100: POS_LIMIT_MINUS
- MD36110: POS_LIMIT_PLUS
- MD36120: POS_LIMIT_MINUS2
- MD36130: POS_LIMIT_PLUS2

Clearance control takes the machine data into account even if an axis is not being referenced.

Disabling digital/analog inputs

Neither the analog input for the input voltage of the clearance sensor nor the digital input used by the clearance control in the context of the “Lift fast with position controller cycle” special function can be controlled (disabled) via the PLC (DB10.DBB0/B10.DBB146).

See also the description of machine data:

- MD62508: CLC_SPECIAL_FEATURE_MASK, bits **4** and **5**

Section 4.1, page 3/TE1/4-46 onwards.

Gantry axes

Only one of the clearance-controlled axes may be configured as the master axis of a gantry grouping, defined via machine data:

MD37100: GANTRY_AXIS_TYPE (gantry axis definition).

Following axes in a gantry grouping may not be used in the context of clearance control.

Axis position display

The actual current axis position of a clearance-controlled axis as the sum of an interpolatory axis position and the current position offset of clearance control is not displayed in the main machine screen on a SINUMERIK standard HMI:

- SINUMERIK HMI Advanced
- SINUMERIK HMI Embedded

On the above HMIs, the actual current axis position is displayed in the service screen: **Operating Area Switchover** > **Diagnosis** > **Service Displays** > **Axis/ Spindle** as a "position actual value":

NC channels

The "clearance control" technological function is only available in the first NC channel, even on controls with more than one NC channel.

- Only channel axes in the first NC channel may be used as clearance-controlled axes.
- CLC part program commands may only be used in part programs processed in the first NC channel.

Conflict with other machine data

Machine data 30132: $MA_IS_VIRTUAL_AX = 1 may not be used.

Note

The "clearance control" technological function is only available in the **first** NC channel!

Computing time requirements

The additional computing time required for the "clearance control" technological function must be taken into account on control systems in which the cycle times set for the interpolator and position controller cycle have been substantially optimized in comparison with the default setting:

The additional computing time required comes into effect when clearance control is activated in the part program (CLC(x)). If the interpolation or position controller cycle is exceeded, the following alarm appears:

- Alarm: "4240 Computing time overflow at IPO or position controller level, IP *point in part program*"

and the processing of the part program is aborted.

Data Descriptions (MD, SD) 4

4.1 Machine data for clearance control

<table>
<tr><td>62500
MD number</td><td colspan="2">$MC_CLC_AXNO
Axis assignment for clearance control</td></tr>
<tr><td>Default setting: 0</td><td>Minimum input limit: –2</td><td>Max. input limit:
Maximum number of axes in channel</td></tr>
<tr><td>Changes effective after POWER ON</td><td>Protection level: 2 / 7</td><td>Unit: –</td></tr>
<tr><td>Data type: INT</td><td colspan="2"></td></tr>
<tr><td>Meaning:</td><td colspan="2">– 0: Deactivates the clearance control function
– x where x > 0:
Activates 1D clearance control for the channel axis with the axis number entered under x.
This axis must not be a modulo rotary axis.
– x where x < 0: Activates 3D clearance control.
In order for 3D clearance control to be activated, at least one of the two possible 5-axis transformations in the channel must be configured.
– –1
x = –1 selects the first 5-axis transformation configured in the 1st channel with $MC_TRAFO_TYPE_x (16 <=transtype <=149) for clearance control.
– –2
x = –2 selects the second 5-axis transformation configured in the 1st channel.
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.
It is permissible to configure 3-axis and 4-axis transformations (2D clearance control).
Restriction:
– Only one of the linear axes involved in the clearance control function may be configured as the master axis in a gantry grouping.
– No axis involved in the clearance control function may be configured as the slave axis of a gantry grouping.
– Incorrectly parameterized configurations are rejected with CLC alarm 75000 during POWER ON.</td></tr>
</table>

62502	$MC_CLC_ANALOG_IN	
MD number	Analog input for clearance control function	
Default setting: 1	Minimum input limit: 1	Maximum input limit: 8
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: INT		
Meaning:	The machine data defines the number of the analog input that is used for the clearance sensor. In contrast to the functions (synchronized actions) implemented in the interpolator, the clearance control input cannot be controlled via PLC interface DB10.DBW148 onwards.	

62504	$MC_CLC_SENSOR_TOUCHED_INPUT	
MD number	Input bit assignment for the sensor collision signal	
Default setting: 0	Minimum input limit: –40	Maximum input limit: 40
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: INT		
Meaning:	The machine data defines the number of the digital input that is used for collision monitoring. Precondition: The clearance sensor has a “sensor collision” signal. The digital inputs are numbered in the same way as the corresponding system variables: $A_IN[x], where x = number of digital input. E.g.: 3. input on 2nd input byte: $MC_CLC_SENSOR_TOUCHED_INPUT = 11 ; 3 + 1 * 8 Setting negative values inverts the corresponding input signal internally for processing (fail-safe method). For more information about sensor collision monitoring, see Section 2.4, page 3/TE1/2-14.	

62505 MD number	$MC_CLC_SENSOR_LOWER_LIMIT Lower motion limit of clearance control	
Default setting: –5.0, –10.0	Minimum input limit: –	Maximum input limit: 0.0
Changes effective after RESET	Protection level: 2 / 7	Unit: mm/inch
Data type: REAL		
Meaning:	The machine data comprises 2 field elements: – CLC_SENSOR_LOWER_LIMIT[**0**] 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. When the limit is reached, the PLC signal • DB21.DBB37.4 is set and CLC alarm 75020 is output. – CLC_SENSOR_LOWER_LIMIT[**1**] The second field element limits the value of the maximum programmable lower limit.	

62506 MD number	$MC_CLC_SENSOR_UPPER_LIMIT Upper motion limit of clearance control	
Default setting: 10.0, 40.0	Minimum input limit: 0.0	Maximum input limit: –
Changes effective after RESET	Protection level: 2 / 7	Unit: mm/inch
Data type: REAL		
Meaning:	The machine data comprises 2 field elements: – CLC_SENSOR_UPPER_LIMIT[**0**] The first field element of this machine data sets the upper limit for the deviation between the sensor-controlled machine position and the programmed position. When the limit is reached, the PLC signal • DB21.DBB37.5 is set and CLC alarm 75021 is output. – CLC_SENSOR_UPPER_LIMIT[**1**] The second field element limits the value of the maximum programmable upper limit.	

62508 MD number	$MC_CLC_SPECIAL_FEATURE_MASK Special functions and operating modes of the clearance control	
Default setting: 0	Minimum input limit: –	Maximum input limit: –
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: INT HEX format	840D: Applies as of SW: 5.3	
Meaning:	**Bit 0 and bit 1** Alarm reaction when CLC motion limits are reached: This machine data configures the alarm reaction when motion limits set in MD 62505 and MD 62506 or programmed with CLC_LIM are reached. Bit 0 = 0: Alarm 75020 does not stop program processing. This alarm can be acknowledged with the Cancel key. Bit 0 = 1 Alarm 75020 stops program processing at the lower limit. The alarm must be acknowledged with RESET. Bit 1 = 0 Alarm 75021 does not stop program processing. This alarm can be acknowledged with the Cancel key. Bit 1 = 1 Alarm 75021 stops program processing at the upper limit. The alarm must be acknowledged with RESET.	
Meaning:	**Bit 4** Operation as online tool length compensation in direction of orientation Bit 4 = 0 The clearance control function works normally. Bit 4 = 1 Unlike as for clearance control, the analog input does not set a velocity but directly applies an offset position. In this case, the ordinate of the selected sensor characteristic $MC_CLC_SENSOR_VELO_TABLE_x is interpreted in mm or inches instead of in mm/min (inch/min). This operating mode can be activated for test purposes and implementation of 3D tool length compensation. In this mode, the analog value is read in the position controller cycle rather than in the interpolator cycle. In this mode, analog values can be modified and/or specified via the PLC using DB10.DBW148 onwards. The input used must be activated via the following machine data: • MD10300: FASTIO_ANA_NUM_INPUTS	

Meaning:	**Bit 5:** Mode for rapid retraction in position control cycle Bit 5 = 0: The clearance control function works normally. Bit 5 = 1: The analog input is irrelevant. If the digital input configured with MD 62504: CLC_SENSOR_TOUCHED_INPUT is activated (possibly inverted), a retracting movement is started in the same position controller cycle corresponding to an analog signal setting of +10V during operation as "online tool length compensation" (see bit 4). The digital input signal that initiates the retracting movement 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 processed in the interpolator. This signal branch can be controlled by the PLC via standard signals DB10.DBB0 onwards.
Meaning:	**Bit 8:** Mode for alarm output if the lower movement limit has been reached Bit 8 = 0: Alarm 75020 is output. Bit 8 = 1: Alarm 75020 is not output if the alarm reaction when CLC motion limits are reached (Bit 0) has been configured without stop of the program execution: Bit 0 = 0
Meaning:	**Bit 9:** Mode for alarm output if the upper movement limit has been reached Bit 9 = 0: Alarm 75021 is output. Bit 9 = 1: Alarm 75021 is not output if the alarm reaction when CLC motion limits are reached (Bit 0) has been configured without stop of the program execution: Bit 1 = 0

Meaning:	**Bit 14:** Synchronization of the start position for single-axis clearance control. Bit 14 = 0: If the clearance control is only configured for one axis (MD 62500 $MC_CLC_AXNO), the current actual position is only synchronized as the start position of the next part program block for this axis when the clearance control is deactivated with CLC(0). Bit 14 = 1: If the clearance control is only configured for one axis (MD 62500 $MC_CLC_AXNO), the current actual positions are synchronized as the start positions of the next part program block for all axes when the clearance control is deactivated with CLC(0). This setting is only required for those applications where single-axis clearance control is used together with a 3/4/5-axis transformation (e.g. pipe cutting with rotating workpiece) and for the first motion block after CLC(0) in which an axis jump or alarm: “Channel %1 Axis %2 System error 550010” occurs in the CLC axis.

62510	**$MC_CLC_SENSOR_VOLTAGE_TABLE_1**	
MD number	Coordinate voltage of interpolation points sensor characteristic 1	
Default setting: –10.0, 10.0, 0.0, 0.0, 0.0, 0.0	Minimum input limit: –10.0	Maximum input limit: +10.0
Changes effective after RESET	Protection level: 2 / 7	Unit: Volts
Data type: REAL	Also 840D with SW 5.3 and higher: Maximum 10 interpolation points	
Meaning:	This machine data is used to define the voltage values for sensor characteristic 1. The associated velocity value should be entered under the same index **i** of machine data: • DM62511: CLC_SENSOR_VELO_TABLE_1[**i**]. In the simplest case, it is sufficient to define the characteristic using two interpolation points as a symmetrical straight line through zero: Example: – $MC_CLC_SENSOR_VOLTAGE_TABLE_1[0] = –10.0 ; V – $MC_CLC_SENSOR_VOLTAGE_TABLE_1[1] = 10.0 ; V – $MC_CLC_SENSOR_VELO_TABLE_1[0] = 500.0 ; mm/min – $MC_CLC_SENSOR_VELO_TABLE_1[1] = –500.0 ; mm/min All array elements of the machine data not used in the example must be assigned the value 0.0. If the defined sensor characteristic generates the incorrect control direction, i.e. once clearance control has been activated, the sensor “flees” from the workpiece, the control direction can be corrected either by reversing the polarity of the sensor signal on the I/O module or by changing the sign of the voltage values in the machine data. Notes for defining the sensor characteristic: – An interpolation point with velocity value 0 must not be positioned at the end of the table. – The characteristic must be monotone, i.e. the velocity over voltage values must either be exclusively ascending or exclusively descending. – The characteristic must not feature any jumps in the velocity waveform, i.e. it is not permitted to define different velocities for the same voltage value. – The characteristic must have at least two interpolation points. – No more than 5 interpolation points (3 on the 840D up to SW 5.3) with positive or negative velocity may be entered. – Characteristics in which the line does not pass exactly through the zero point may affect the clearance standardization set in the clearance sensor.	

62511 MD number	$MC_CLC_SENSOR_VELO_TABLE_1 Coordinate velocity of interpolation points sensor characteristic 1	
Default setting: 2000.0, –2000.0, 0.0, 0.0, 0.0, 0.0	Minimum input limit: –	Maximum input limit: –
Changes effective after RESET	Protection level: 2 / 7	Unit: mm / min inch / min
Data type: REAL	Also 840D with SW 5.3 and higher: Maximum 10 interpolation points	
Meaning:	This machine data is used to define the velocity values for sensor characteristic 1. The associated voltage value should be entered under the same index **i** of machine data: • MD 62510: CLC_SENSOR_VOLTAGE_TABLE_1[**i**]. For more information about defining characteristics, see the description of machine data MD62510 CLC_SENSOR_VOLTAGE_TABLE_1.	

62512 MD number	$MC_CLC_SENSOR_VOLTAGE_TABLE_2 Coordinate voltage of interpolation points sensor characteristic 2	
Default setting: –10.0, 10.0, 0.0, 0.0, 0.0, 0.0	Minimum input limit: –10.0	Maximum input limit: +10.0
Changes effective after RESET	Protection level: 2 / 7	Unit: Volts
Data type: REAL	Also 840D with SW 5.3 and higher: Maximum 10 interpolation points	
Meaning:	This machine data is used to define the voltage values for sensor characteristic 2. For more information about defining characteristics, see the description of machine data MD62510 CLC_SENSOR_VOLTAGE_TABLE_1.	

62513 MD number	$MC_CLC_SENSOR_VELO_TABLE_2 Coordinate velocity of interpolation points sensor characteristic 2	
Default setting: 2000.0, –2000.0, 0.0, 0.0, 0.0, 0.0	Minimum input limit: –	Maximum input limit: –
Changes effective after RESET	Protection level: 2 / 7	Unit: mm / min inch / min
Data type: REAL	Also 840D with SW 5.3 and higher: Maximum 10 interpolation points	
Meaning:	This machine data is used to define the voltage values for sensor characteristic 2. For more information about defining characteristics, see the description of machine data MD62510 CLC_SENSOR_VOLTAGE_TABLE_1.	

62516	$MC_CLC_SENSOR_VELO_LIMIT	
MD number	Velocity of the distance control movement	
Default setting: 100.0	Minimum input limit: 0.0	Maximum input limit: 100.0
Changes effective after RESET	Protection level: 2 / 7	Unit: Percent
Data type: REAL		
Meaning:	1D clearance control The machine data is used to define the maximum traversing velocity of the overlaid control movement as a percentage of the maximum residual axis velocity of the maximum value of the clearance-controlled axis: • MD32000: MAX_AX_VELO[x]2D/3D 2D/3D clearance control With the 2D and 3D clearance control variants, the reference value used is the maximum velocity of the slowest clearance-controlled axis multiplied by √2 or √3 respectively.	

62517	$MC_CLC_SENSOR_ACCEL_LIMIT	
MD number	Acceleration of clearance control motion	
Default setting: 100.0	Input limit: 0.0	Maximum input limit: 100.0
Changes effective after RESET	Protection level: 2 / 7	Unit: Percent
Data type: REAL		
Meaning:	1D clearance control The machine data is used to define the maximum acceleration of the overlaid control movement as a percentage of the maximum residual axis acceleration of the maximum value of the clearance-controlled axis: • MD32300: MAX_AX_ACCEL[x] 2D/3D clearance control With the 2D and 3D clearance control variants, the reference value used is the maximum acceleration of the slowest clearance-controlled axis multiplied by √2 or √3 respectively.	

62520 MD number	$MC_CLC_SENSOR_STOP_POS_TOL Positional tolerance for status message "Clearance control zero speed"	
Default setting: 0.05	Input limit: 0.0	Maximum input limit: –
Changes effective after RESET	Protection level: 2 / 7	Unit: mm/inch
Data type: REAL		
Meaning:	When clearance control is active, in order to achieve the exact stop condition (G601/G602), both the axes involved in the programmed traversing movement and the clearance-controlled axes must have achieved their exact stop conditions. The exact stop condition for clearance control is defined via a position window and a dwell time: • MD62520: CLC_SENSOR_STOP_POS_TOL • MD62521: CLC_SENSOR_STOP_DWELL_TIME If the clearance control function and/or the clearance-controlled axes for the parameterized dwell time are within the positional tolerance, the exact stop condition for clearance control has been met. Setting notes If the clearance control function is unable to remain within the parameterized position window during the corresponding dwell time, the following alarm will appear under certain circumstances: • Alarm "1011 Channel *channel number* system error 140002" In order to avoid an alarm or if an alarm is triggered, proceed as follows: 1. Switch on the clearance control function with the typical distance of the clearance sensor to a thin sheet metal. 2. Knock on the sheet such that the laser head performs visible compensation movements. When the compensation movement is finished, the sheet should no longer be touched. 3. If the interface signal "flickers": IS: DB3x, DBB60.7 (position reached with exact stop fine) after knocking or release of the process gas, modify the following machine data: • MD36010: STOP_LIMIT_FINE (increase) • MD62520: CLC_SENSOR_STOP_POS_TOL (increase) • MD62521: CLC_SENSOR_STOP_DWELL_TIME (reduce) The changes to the machine data will not become effective until an NCK Power on Reset has been performed. The clearance control may therefore have to be activated again when the NC has powered up.	

62521 MD number	$MC_CLC_SENSOR_STOP_DWELL_TIME Dwell time for status message “Clearance control zero speed”	
Default setting: 0.1	Minimum input limit: 0.0	Maximum input limit: –
Changes effective after RESET	Protection level: 2 / 7	Unit: s
Data type: REAL	Measures for avoiding alarm 1011	
Meaning:	This machine data is used to define the dwell time for achieving the exact stop condition for clearance control. Enter the corresponding positional tolerance in machine data: • MD62520: CLC_SENSOR_STOP_POS_TOL For more information about the exact stop condition for clearance control, see the description of machine data MD62520: CLC_SENSOR_STOP_POS_TOL.	
Related to	The dwell time set must not exceed the maximum waiting time set in machine data MD36020 POSITIONING_TIME for achieving the exact stop condition.	

62522 MD number	$MC_CLC_OFFSET_ASSIGN_ANAOUT Modification of the setpoint distance by means of sensor signal override	
Default setting: 0	Minimum input limit: –8	Maximum input limit: 8
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: INT		
Meaning:	The machine data defines the number of the analog output whose output value is subtracted from the clearance sensor input voltage. The analog inputs are numbered in the same way as the corresponding system variables: $A_OUTA[x], where x = number of analog output. The analog output can be used both block-synchronously from a part program and synchronously via a synchronized action via variable $A_OUTA[n].	

62523 MD number	**$MC_CLC_LOCK_DIR_ASSIGN_DIGOUT** Assignment of the digital outputs for disabling the CLC movement	
Default setting: 0.0	Minimum input limit: –40	Maximum input limit: 40
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: INT		
Meaning:	The machine data comprises 2 field elements: – CLC_LOCK_DIR_ASSIGN_DIGOUT[**0**] The first field element is used to define the digital output via which the negative direction of movement for clearance control can be blocked. – CLC_LOCK_DIR_ASSIGN_DIGOUT[**1**] The second field element is used to define the digital output via which the positive direction of movement for clearance control can be blocked. Enter a negated output number to invert evaluation of the switching signal. Example: Digital output 1 ($A_OUT[**1**]) is to block negative direction of movement, digital output 2 ($A_OUT[**2**]) is to block positive direction of movement: – MD62523: CLC_LOCK_DIR_ASSIGN_DIGOUT[0] = **1** – MD62523: CLC_LOCK_DIR_ASSIGN_DIGOUT[1] = **2** The corresponding system variables can be used block-synchronously in the part program or asynchronously via synchronized actions to activate/deactivate the blocking of each direction of movement: – Blocking of negative direction of movement ON/OFF $A_OUT[1] = 1 / 0 – Blocking of positive direction of movement ON/OFF $A_OUT[2] = 1 / 0 Inversion of switching signal – MD62523: CLC_LOCK_DIR_ASSIGN_DIGOUT[0] = **–1** – Blocking of negative direction of movement **ON** / OFF $A_OUT[1] = **0** / 1	

62524 MD number	**$MC_CLC_ACTIVE_AFTER_RESET** Clearance control remains active after RESET	
Default setting: 0	Minimum input limit: –	Maximum input limit: –
Modification effective after NEW_CONF	Protection level: 2 / 7	Unit: –
Data type: BOOL		
Meaning:	**1D clearance control** This machine data is used to parameterize the RESET characteristics (end of program RESET or NC RESET) of **1D** clearance control. – CLC_ACTIVE_AFTER_RESET = 0 On RESET, clearance control is deactivated in the same way as the part program command CLC(0). – CLC_ACTIVE_AFTER_RESET = 1 Clearance control maintains its current state on RESET. **3D clearance control** The machine data is not effective for the 3D clearance control. In this case, a RESET always deactivates clearance control.	

62525 MD number	**$MC_CLC_SENSOR_FILTER_TIME** PT1 filtering time constant of sensor signal	
Default setting: 0.0	Minimum input limit: 0.0	Maximum input limit: –
Changes effective immediately	Protection level: 2 / 7	Unit: s
Data type: REAL		
Meaning:	This machine data is used to parameterize the time constant of the PT1 filter for clearance control (corresponds to an RC element). The PT1 filter can be used to dampen the higher-frequency noise components in the input signal of the clearance sensor. The effect of the filter can be monitored via the function-specific display data (see Section 2.7, page 3/TE1/2-33). A setting of zero deactivates the filter completely. **Note** Each additional time constant in the servo loop reduces the maximum possible servo loop dynamics.	

62528 MD number	**$MC_CLC_PROG_ORI_AX_MASK** Axis mask of the direction axes	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 0C0000000H
Changes effective immediat.	Protection level: 2 / 7	Unit: –
Data type: DWORD		
Meaning:	Each bit of the axis mask refers, by way of its bit index n, to the channel axis [n+1]. 3 bits, and only 3 bits, can be set to correspond to the 3 direction axes of the compensation vector. The bits are evaluated in ascending order. The first channel axis configured in this way corresponds to the X coordinate of the compensation vector. The second channel axis corresponds to the Y coordinate, etc.	

62529	$MC_CLC_PROG_ORI_MAX_ANGLE	
MD number	Limit angle	
Default setting: 45.0	Minimum input limit: 0.0	Maximum input limit: 180.0
Changes effective immediately	Protection level: 2 / 7	Unit: Degrees
Data type: REAL		
Meaning:	Permissible limit angle between tool orientation and the CLC direction	

62530	$MC_CLC_PROG_ORI_ANGLE_AC_PARAM	
MD number	Index of display variable for the current difference angle.	
Default setting: –1	Minimum input limit: –1	Maximum input limit: 20000
Changes effective immediately	Protection level: 2 / 7	Unit: –
Data type: INT		
Meaning:	Index n of system variable $AC_PARAM[n] in which the current difference angle between the tool orientation and CLC direction is output.	

■

Signal Descriptions 5

5.1 Signals to channel

DB21, ... **DBB1.4** Data Block	**CLC stop** Signal(s) to channel (PLC–> NCK)		
Edge evaluation: No		Signal(s) updated: Cyclic	Signal(s) valid: Also 840D SW 3.6 and higher
Signal state 1 or signal transition 0 ——> 1	Clearance control is deactivated in the same way as the part program command CLC_GAIN=0.0.		
Signal state 0 or signal transition 1 ——> 0	Clearance control is enabled.		

DB21, ... **DBB1.5** Data Block	**CLC_Override** Signal(s) to channel (PLC–> NCK)		
Edge evaluation: No		Signal(s) updated: Cyclic	Signal(s) valid: Also 840D SW 3.6 and higher
Signal state 1 or signal transition 0 ——> 1	The channel-specific override DB21.DBB4 is effective for clearance control. Override settings < 100% reduces the velocity limit preset in machine data: • MD62516: CLC_SENSOR_VELO_LIMIT for the overlaid movement. Override settings > 100% apply the limitation from the machine data.		
Signal state 0 or signal transition 1 ——> 0	The maximum velocity of the control motion is not dependent on the override setting		
Application example	The difference for the operator is particularly dependent on whether the sensor motion is stopped or not with a 0 override.		
Related to	Channel-specific override setting DB21.DBB4 and DB21. DBB6.7		

5.2 Signals from channel

<table>
<tr><td>DB21, ...
DBB37.3
Data Block</td><td colspan="2">CLC is active
Signal(s) from channel (NCK–> PLC)</td></tr>
<tr><td>Edge evaluation: No</td><td>Signal(s) updated: Cyclic</td><td>Signal(s) valid: Also 840D SW 3.6 and higher</td></tr>
<tr><td>Signal state 1 or signal transition 0 ––––> 1</td><td colspan="2">Clearance control is activated.</td></tr>
<tr><td>Signal state 0 or signal transition 1 ––––> 0</td><td colspan="2">Clearance control is deactivated.</td></tr>
</table>

<table>
<tr><td>DB21, ...
DBB37.4–5
Data Block</td><td colspan="2">CLC motion has stopped
Signal(s) from channel (NCK–> PLC)</td></tr>
<tr><td>Edge evaluation: No</td><td>Signal(s) updated: Cyclic</td><td>Signal(s) valid: Also 840D SW 3.6 and higher</td></tr>
<tr><td>Signal state 1 or signal transition 0 ––––> 1</td><td colspan="2">The traversing movement of the clearance-controlled axes based on clearance control is at a standstill. The following conditions will set the signal:
– The standstill conditions set in the following machine data are met:
• MD62520: CLC_SENSOR_STOP_POS_TOL
• MD62521: CLC_SENSOR_STOP_DWELL_TIME
– Programming of CLC_GAIN=0.0
– PLC interface signal: DB21.DBB1.4 “Stop CLC motion”
Note
The “CLC motion at standstill” signal is only set if bits 4 and 5 are set at the same time. If only one of the two bits is set –> see below.</td></tr>
<tr><td>Signal state 0 or signal transition 1 ––––> 0</td><td colspan="2">Clearance control generates traversing movements directly in the clearance-controlled axes.
While the axes are moving as a result of clearance control, axial interface signals DB3x.DBB30.6/7 “Position reached exact stop coarse/fine” cannot be set.</td></tr>
<tr><td>Related to ....</td><td colspan="2">DB3x.DBB30.6/7 “Position reached, exact stop coarse/fine”</td></tr>
</table>

<table>
<tr><td>DB21, ...
DBB37.4
Data Block</td><td colspan="2">CLC motion at lower motion limit
Signal(s) from channel (NCK–> PLC)</td></tr>
<tr><td>Edge evaluation: No</td><td>Signal(s) updated: Cyclic</td><td>Signal(s) valid: Also 840D SW 3.6 and higher</td></tr>
<tr><td>Signal state 1 or signal transition 0 ––––> 1</td><td colspan="2">The traversing movement of the clearance-controlled axes based on clearance control has been stopped at the upper movement limit set in
• MD62505: CLC_SENSOR_LOWER_LIMIT
or programmed with CLC_LIM(..).
Note
If the DB21.DBB37.5 signal is set at the same time –> see above “CLC motion at standstill” signal.</td></tr>
<tr><td>Signal state 0 or signal transition 1 ––––> 0</td><td colspan="2">Clearance control has left the lower limitation.</td></tr>
</table>

DB21, ... DBB37.5 Data Block	CLC motion at upper motion limit Signal(s) from channel (NCK–> PLC)	
Edge evaluation: No	Signal(s) updated: Cyclic	Signal(s) valid: Also 840D SW 3.6 and higher
Signal state 1 or signal transition 0 ——> 1	The traversing movement of the clearance-controlled axes based on clearance control has been stopped at the upper movement limit set in • MD62506: CLC_SENSOR_UPPER_LIMIT or programmed with CLC_LIM(..). **Note** If the DB21.DBB37.4 signal is set at the same time –> see above "CLC motion at standstill" signal.	
Signal state 0 or signal transition 1 ——> 0	Clearance control has left the upper limitation.	

■

Notes

6

Examples

– No examples available –

■

Notes

Data Fields, Lists

7

7.1 Alarms

12550	**Channel %1 block %2 name %3 not defined or option not installed**
Explanation	Parameter: • %1 = channel number • %2 = block number, label • %3 = source symbol; in this case: CLC If the “clearance control” technological function is not available or not activated, the function-specific part program commands are rejected as invalid syntax.
Reaction	Interruption of part program interpretation, alarm signal in PLC interface
Remedy	Install and activate compile cycle.
Delete criterion	NC RESET

<table>
<tr><th>75000</th><th>Channel %1 CLC: Incorrect MD configuration, error no: %2</th></tr>
<tr><td>Explanation</td><td>Parameter:
• %1 = channel number
• %2 = error number

Error number:
• –1
The interpolation points of one of the two sensor characteristics are not strictly monotone ascending or descending.
• –2
One of the two sensor characteristics has less than 2 valid interpolation points.
• –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.
• –4
The digital input set for sensor collision monitoring:
– MD 62504 $MC_CLC_SENSOR_TOUCHED_INPUT
is not activated on the control
– MD 10350 $MN_FASTIO_DIG_NUM_INPUTS
• –5
A fast input was not assigned to the special “lift fast with position controller cycle” function:
– MD 62504 $MC_CLC_SENSOR_TOUCHED_INPUT
• –6
The axis selected for clearance control is not active in the channel:
– MD 62500 $MC_CLC_AXNO
• –7
The 5-axis transformation selected for clearance control:
– MD 62500 $MC_CLC_AXNO
is not configured in the channel:
– MD 24100 $MC_TRAFO_TYPE_x
• –8
More than one of the axes involved in the clearance control is a leading axis in a gantry grouping MD 37100 $MA_GANTRY_AXIS_TYPE.
• –9
More than one of the axes involved in the clearance control is a following axis in a gantry grouping:
– MD 37100 $MA_GANTRY_AXIS_TYPE
• –10
In the Export version, CLC can only be activated if the “number of interpolating axes” option has been set to a value less than 4.
• –11
Either three axes or no axes may be configured for clearance control with programmable compensation vector CLC(3):
– MD 62528 $MC_CLC_PROG_ORI_AX_MASK
If three axes are configured, they must also be assigned to the channel:
– MD 20070 $MC_AXCONF_MACHAX_USED</td></tr>
<tr><td>Reaction</td><td>Dropout of signal mode group ready, alarm signal in PLC interface</td></tr>
<tr><td>Remedy</td><td>Correct the relevant machine data.</td></tr>
<tr><td>Delete criterion</td><td>NCK POWER ON RESET</td></tr>
</table>

75005	**Channel %1 block %2 CLC: General programming error**
Explanation	Parameter: • %1 = channel number • %2 = block number, label Possible errors: • The activation/deactivation command for clearance control "CLC(..)" accepts only the values 2, 1, 0 and –1 as call parameters. • 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 part program interpretation. Alarm signal in PLC interface.
Remedy	Modify part program. Configure digital input for collision evaluation in MD if necessary.
Delete criterion	NC RESET

75010	**Channel %1 block %2 CLC_LIM value higher than MD limit**
Explanation	Parameter: • %1 = channel number • %2 = block number, label One of the limitations for the position offset of the clearance control programmed with CLC_LIM(...,...) is greater than the value permitted in the corresponding machine data: • MD62505 $MC_CLC_SENSOR_LOWER_LIMIT[1] or • MD62506 $MC_CLC_SENSOR_UPPER_LIMIT[1]
Reaction	Abortion of part program interpretation. Alarm signal in PLC interface.
Remedy	Modify part program. Raise limitation in appropriate machine data if necessary.
Delete criterion	NC RESET

75015	**Channel %1 block %2 CLC(0) with active TOC**
Explanation	Parameter: • %1 = channel number • %2 = block number, label 3D clearance control has been switched off with CLC(0) while tool radius compensation is still active (G41/G42). 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 part program interpretation. Alarm signal in PLC interface.
Remedy	Modify part program: • switch off active G41/G42 before CLC(0) OR • do not switch off clearance control, but just "freeze" temporarily (CLC_GAIN=0.0) OR • cancel the position offset mechanically with CLC(–1).
Delete criterion	NC RESET

75016	**Channel %1 block %2 CLC: Orientation changed with TRAFOOF**
Explanation	Parameter: • %1 = channel number • %2 = block number, label Possible cause: • 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. • The transformation has been switched off temporarily (TRAFOOF) while clearance control is still active. When 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 part program interpretation. Alarm signal in PLC interface.
Remedy	Modify part program: • Do not switch on clearance control until transformation is already active or make sure that the required conditions relating to orientation are observed.
Delete criterion	NC RESET

75018	**Channel %1 block %2 CLC in programmable direction, error ID: %3**
Explanation	Parameter: • %1 = channel number • %2 = block number, label • %3 = error number Error cause: • Error ID: **0** CLC(3) was programmed but – the option bit is not enabled or – the configuration of the direction axes contains an error • Error ID: **1** The currently programmed compensation vector is antiparallel (in the opposite direction) to the last programmed compensation vector.
Reaction	Abortion of part program interpretation. Alarm signal in PLC interface.
Remedy	Correct the machine data (0) or the part program (1).
Delete criterion	NC RESET

75019	**Channel %1, error ID: %2, angle %3**
Explanation	Parameter: • %1 = channel number • %2 = error number • %3 = angle between compensation vector and tool orientation Error cause: • Error ID: **1** and angle: **0.0** The CLC direction is [0, 0, 0]. A compensation vector cannot be generated. • Error ID: **2** and angle: ***Angle*** The angle between the current tool orientation and the compensation vector is greater than the configured maximum angle: – MD 62529 $MC_CLC_PROG_ORI_MAX_ANGLE
Reaction	Abortion of part program execution. Alarm signal in PLC interface.
Remedy	Increase the permissible limit angle or change the part program.
Delete criterion	NC-RESET

75021	**Channel %1 CLC position offset at upper limit %2**
Explanation	Parameter: • %1 = channel number The position offset due to the overlaid movement has reached the limit set with the machine data: • MD62506 $MC_CLC_SENSOR_UPPER_LIMIT or programmed with CLC_LIM(...,...).
Reaction	Depending on setting in bit 1 of machine data: • MD62508 $MC_CLC_SPECIAL_FEATURE_MASK – Bit 0 = 0: Alarm display only, no internal reaction – Bit 0 = 1: Stop programmed motion, NC start disable
Remedy	Check position and form of the workpiece. If necessary, program further limits.
Delete criterion	Depending on setting in bit 0 of machine data: • MD62508 $MC_CLC_SPECIAL_FEATURE_MASK – Bit 0 = 0: CANCEL – Bit 0 = 1: NC RESET

75025	**Channel %1 CLC stopped since sensor tip touched**
Explanation	Parameter: • %1 = channel number Sensor head collision monitoring has responded.
Reaction	A retraction motion to the upper limitation of the position offset • MD62506 $MC_CLC_SENSOR_UPPER_LIMIT starts. The maximum velocity and acceleration reserves available in the control direction are used for this purpose. The velocity override setting has no effect on this retraction motion. The path motion is stopped simultaneously.
Remedy	Use NC START to continue the part program. The overlaid movement reverts to the control clearance.
Delete criterion	NC START or NC RESET

7.2 Machine data

7.2.1 Drive-specific parameters (840Di)

Number	Name/identifier	Reference
Drive parameter (SIMODRIVE 611D; POSMO SI, CD, CA)		
1502	SPEED_FILTER_1_TIME[n] / Time constant, setpoint speed filter 1	
1503	SPEED_FILTER_2_TIME[n] / Time constant, setpoint speed filter 2	

7.2.2 Drive-specific machine data (840D)

Number	Name/identifier	Reference
Drive machine data (SIMODRIVE 611D)		
1502	SPEED_FILTER_1_TIME[n] / Time constant, setpoint speed filter 1	/DD2/
1503	SPEED_FILTER_2_TIME[n] / Time constant, setpoint speed filter 2	/DD2/

7.2.3 NC-specific machine data

Number	Name/identifier	Reference
General ($MN_ ...)		
10300	FASTIO_ANA_NUM_INPUTS / Number of active analog NCK inputs	A2
10350	FASTIO_DIG_NUM_INPUTS / Number of active digital NCK input bytes	A2
10362	HW_ASSIGN_ANA_FASTIN / Hardware assignment of external analog NCK inputs: 0...7	A2
10380	HW_UPDATE_RATE_FASTIO / Update cycle of synchronously clocked external NCK input/output modules	A2
10382	HW_LEAD_TIME_FASTIO / Pretrigger time of synchronously clocked external NCK inputs/outputs Terminal block: 0...3	A2
10384	HW_CLOCKED_MODULE_MASK / Synchronous processing of individual external input/output modules Terminal block: 0...3	A2
10712	NC_USER_CODE_CONF_NAME_TAB / List of renamed NC identifiers	/PA/
Channel-specific ($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
28254	MM_NUM_AC_PARAM / number of parameters for synchronized actions	S7
Axis-specific ($MA_...)		
32070	CORR_VELO / Axis velocity for handwheel, external zero offsets, SA clearance control	H1, K2, W4
32410	AX_JERK_TIME / Time constant for axial jerk filter	B2
32610	VELO_FFW_WEIGHT / Feedforward control factor for velocity feedforward control	K3
36000	STOP_LIMIT_COARSE / Exact stop coarse	B1
36010	STOP_LIMIT_FINE / Exact stop fine	B1
36040	STANDSTILL_DELAY_TIME/Delay time zero speed monitoring	A3
36060	STANDSTILL_VELO_TOL Max./velocity for axis/spindle stopped	A2
36750	AA_OFF_MODE/Value calculation mode for axial position override	S5
Channel-specific machine data for clearance control ($MC_ ...)		
62500	CLC_AXNO/Axis assignment for clearance control	

62502	CLC_ANALOG_IN /Analog input for clearance control	
62504	CLC_SENSOR_TOUCHED_INPUT/Assignment of an input bit for the "sensor collision" signal	
62505	CLC_SENSOR_LOWER_LIMIT/Lower motion limit of clearance control	CLC_LIM()
62506	CLC_SENSOR_UPPER_LIMIT/Upper motion limit of clearance control	CLC_LIM()
62508	CLC_SPECIAL_FEATURE_MASK/Special functions and operating modes of clearance control	
62510	CLC_SENSOR_VOLTABE_TABLE_1/Coordinate voltage of interpolation points sensor characteristic 1	CLCGAIN
62511	CLC_SENSOR_VELO_TABLE_1/Coordinate velocity of interpolation points sensor characteristic 1	
62512	CLC_SENSOR_VOLTABE_TABLE_2/Coordinate voltage of interpolation points sensor characteristic 2	
62513	CLC_SENSOR_VELO_TABLE_2/Coordinate velocity of interpolation points sensor characteristic 2	
62516	CLC_SENSOR_VELO_LIMIT/Velocity of clearance control motion	MAX_AX_ VELO
62516	CLC_SENSOR_ACCEL_LIMIT/Acceleration of clearance control motion	MAX_AX_ ACCEL
62520	CLC_SENSOR_STOP_POS_TOL/Positional tolerance for status message "Clearance control zero speed"	
62521	CLC_SENSOR_STOP_DWELL_TIME/Wait time for status message "Clearance control zero speed"	
62522	CLC_OFFSET_ASSIGN_ANAOUT/Modification of the setpoint clearance via sensor signal override (SW 5.3 and higher)	
62523	CLC_LOCK_DIR_ASSIGN_DIGOUT/Assignment of the digital outputs for CLC movement deactivation (SW 5.3 and higher)	
62524	CLC_ACTIVE_AFTER_RESET/Clearance control remains active after RESET (SW 5.3 and higher)	
62525	CLC_SENSOR_FILTER_TIME/Time constant of PT1 filter of sensor signal	
62528	CLC_PROG_ORI_AX_MASK / Progr. orientation vector: Axis mask	
62529	CLC_PROG_ORI_MAX_ANGLE / Progr. orientation vector: Maximum difference angle	
62530	CLC_PROG_ORI_ / Progr. orientation vector: Index of the $AC_PARAM variables for the output of the current difference angle	

7.3 Interface signals

DB no.	Bit, byte name		Reference
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	

■

Notes

SINUMERIK 840D sl/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

In SW 4.3, the “analog axis” function can be used to control up to 8 of the available NC axes with an analog drive (e.g.: SIMODRIVE 611A) via a +/– 10 V 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 part 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:

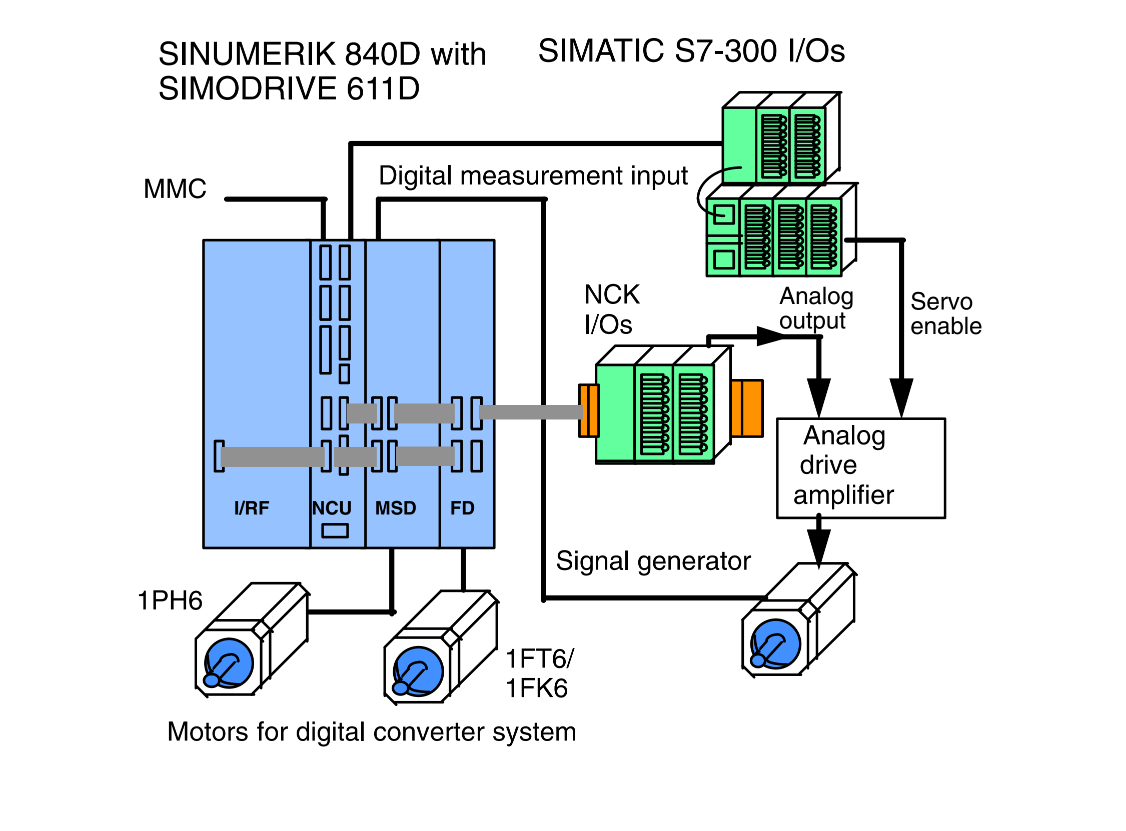


Fig. 2-1 Hardware configuration of an analog axis

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 so 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.

Machine data MD 63530: ANALOG_AXIS is used to declare an NC axis as an 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.

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 run-up.

If the same DMP module is assigned to more than one analog axis, 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.: synchronized actions) as this would interfere with the servo output. DMP modules are assigned by 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 terminal block into which the analog output module is inserted must be set to clock-synchronous operation.
  This is achieved by setting the bit corresponding to the module slot on the terminal block (e.g.: 5. 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 routed to the analog drive amplifier (e.g.: SIMODRIVE 611A), which controls the motor.

Controller 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. Machine data 30100 – 30130 are used for this purpose.

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	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. Bits set to 1 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 +/– 8 V 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
8 V at motor speed of 6000 rev/min	MD 32250: RATED_OUTVAL = 80

4 V 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 with machine data MD 36700: DRIFT_ENABLE = 1. 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” is set 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 so 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.

■

Notes

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 see Section “File and Program Management” in the document “SINUMERIK 840D/810D 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 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 text:
   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

Failure to take appropriate precautions **can** have undesirable consequences.

The functions activated by the option data trigger the corresponding compile cycles. The compile cycles can significantly change the behavior of the control and can create hazardous situations via access to the NC.
Before a compile cycle is activated, appropriate safety precautions to prevent potential damage to machinery and personal injury must be taken (you may need to take action to safeguard against incorrect parameter settings or programming in the compile cycles).

4.2 Machine data for the analog axis function

63530 MD number	**$MA_ANALOG_AXIS** Configuration of an analog axis	
Default setting: 0	Minimum input limit: 0	Maximum 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. Input value = 0: Axis is a digital axis Input value > 0: Axis is one of the 3 possible analog axes. The value to be entered here is the number of the DMP module on the NCU terminal block that must output an analog +/– 10 V speed setpoint to the drive amplifier of the axis.	

■

5 Signal Descriptions

No separate signals to the PLC are provided for the analog axis function.

6 Examples

6.1 General start-up of a compile cycle function

Requirement

- 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 step is identical to the procedure followed for a standard upgrade to a more recent software version and likewise requires the static (battery-backed) control system memory to be erased. To avoid the consequential loss of all data in the SRAM, back up the SRAM before performing the operation. For a detailed description, please see 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, these must be saved in a separate archive file:
   Press the “Data out” softkey and select the required data under “NC active data”:
   “Measuring system compensations”,
   “Sag/angularity compensation” and
   “Quadrant error compensation”.
   Save this data by pressing the “Archive...” softkey and specifying a 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 the number “6” does not appear, 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 activate the compile cycle.

Option data for compile cycles

- Set the option for compile cycle application 2
- **Reboot.** Machine data 63530 should now appear 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/TK1_UK.COM.

GUD

Set up the “ANALOG_AXIS_VOLTAGE” GUD 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 +/– 8 V.

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	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

Data Fields, Lists 7

7.1 Alarms

Detailed explanations of the alarms, which may occur, appear in

References: /DA/, Diagnostics Guide

or in the Online help.

7.2 Machine data

Number	Identifier name	Reference
General ($MN_ ...)		
10310	FASTIO_ANA_NUM_OUTPUTS Number of active NCK outputs	A4
10364	HW_ASSIGN_ANA_FASTOUT Hardware assignment of external analog NCK outputs: 0...7	A4
10380	HW_UPDATE_RATE_FASTIO Update cycle of synchronously clocked external NCK input/output modules	A4
10384	HW_CLOCKED_MODULE_MASK Synchronous processing of individual external input/output modules. Terminal block: 0...3	A4
Axis-specific ($MA_...)		
30100	CTRLOUT_SEGMENT_NR Setpoint assignment drive type	G2
30110	CTRLOUT_MODULE 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 sl/840D/840Di/810D Description of Functions Special Functions (Part 3)

Speed/Torque Coupling, Master-Slave (TE3)

■

Brief Description 1

As of SW 6

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.

Differences compared to previous solution (up to SW 5.x)

- If traversing is programmed for a slave axis that has already been linked, an alarm is issued.
- The setpoint position of the coupled slave 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 part 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 see Chapter 3 for more information about the differences.

As of SW 6.4

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.

Up to SW 5.x

The speed/torque coupling function (master-slave) is used for mechanically-coupled axes that are driven by two separate motors.
This function was available up to SW 5 only via 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 controller to ensure 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.

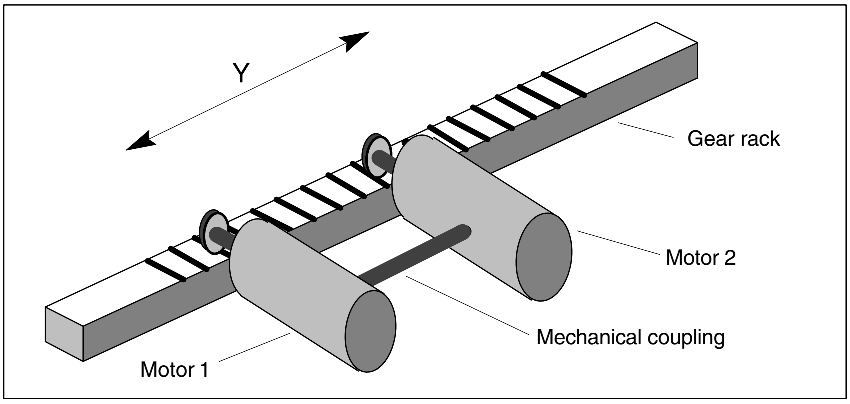


Fig. 2-1 Permanent mechanical coupling

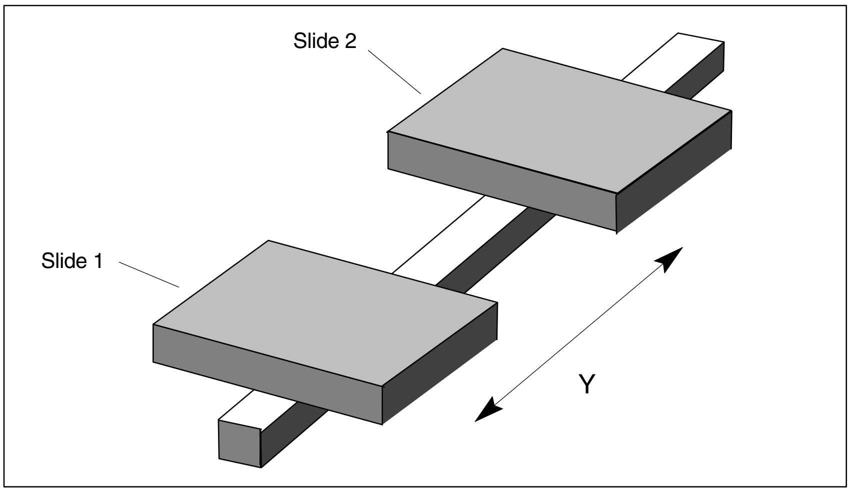


Fig. 2-2 Slides (linear motor) for temporary coupling

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.

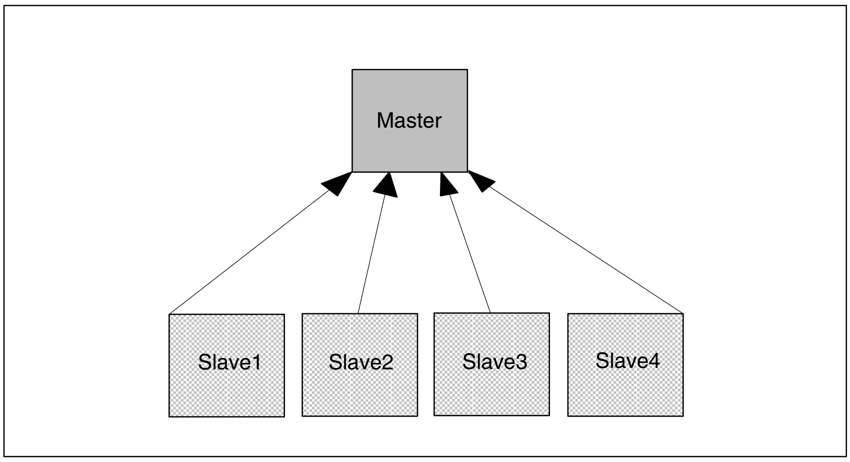


Fig. 2-3 Four coupling relationships with the same master axis

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.

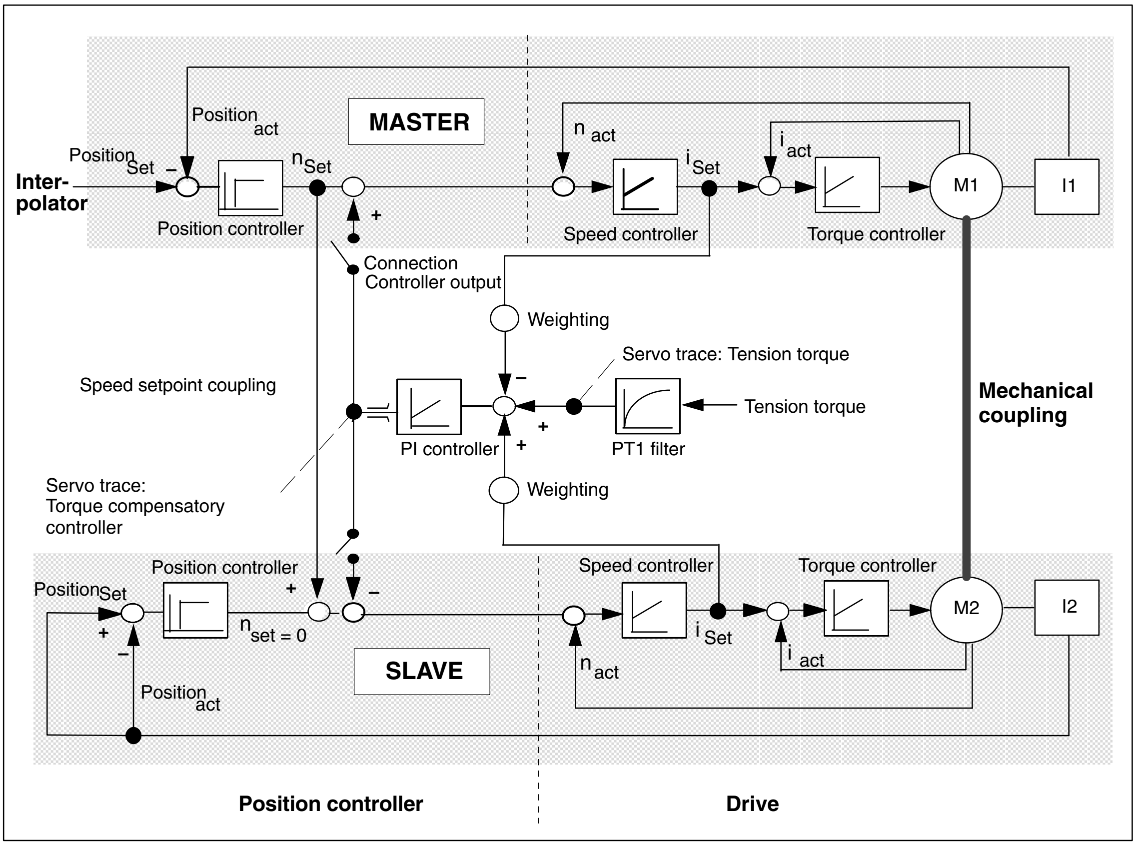


Fig. 2-4 Control structure

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 part program dynamically. This type of configuration can change the static configuration but does not have any reverse effect on the associated machine data.

The instruction
MASLDEF (slv1, slv2, ..., master axis)
assigns one or more slave axes to a master axis,

MASLDEL (slv1, slv2, ...)
cancels the assignment of the slave axes to the master axis and also disconnects the current coupling (similar to MASLOF).

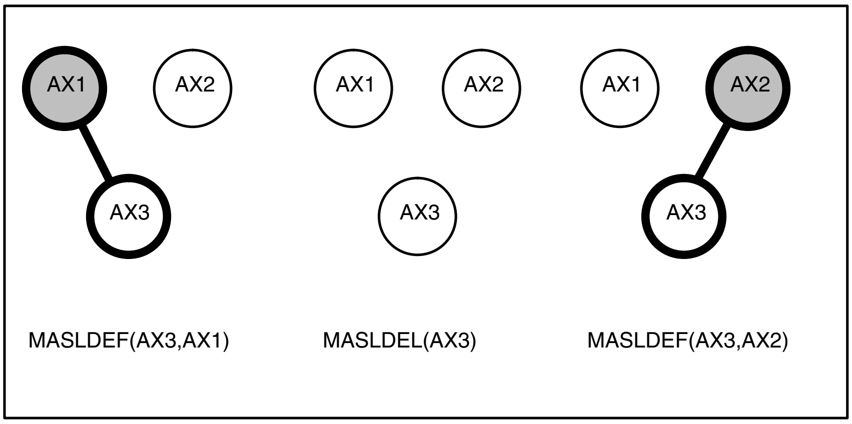


Fig. 2-5 Varying configuration of master axis

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 end of part program.

Note

To implement a standard assignment in each reset, you can add the corresponding MASLDEF and MASLDEL instructions 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 filter is entered in MD 1252: 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.

Setting MD 37256: MS_TORQUE_CTRL_MODE = 3 or
MS_TORQUE_CTRL_P_GAIN = 0, deactivates torque compensatory control.

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 rated 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 controller can be switched on and off directly via the PLC interface signal DB31, ... DBX24.4.
You must set MD 37255: MS_TORQUE_CTRL_ACTIVATION=1. The activated status can be read back in DB31, ... DBX96.4. MD 37254: MS_TORQUE_CTRL_MODE is then only used for configuring the torque distribution.

2.1.5 Tension torque

By 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 rated 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.

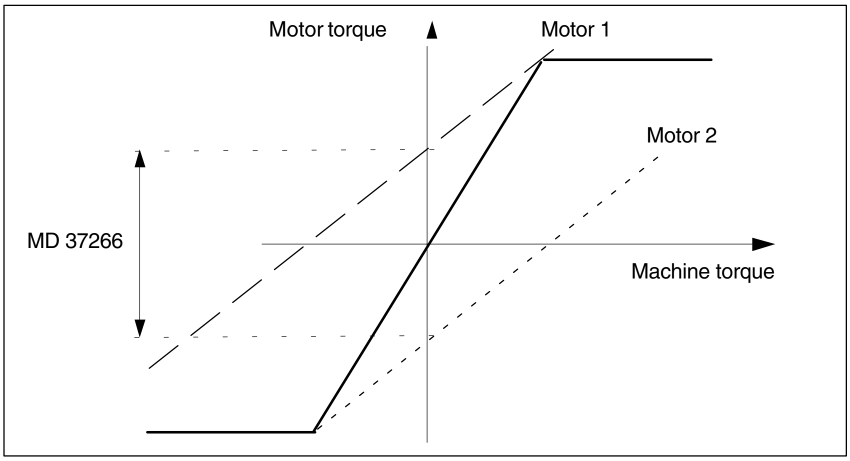


Fig. 2-6 Resulting tension torque

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.

Only one temporary master-slave coupling configured via a machine data (MD 37262: MS_COUPLING_ALWAYS_ACTIVE = 0) can be closed and disconnected with axial PLC interface signal "Master/slave on" (DB31, ... DBX24.7) and within a part program using the commands
MASLON(slaveAxis1, slaveAxis2, ...) and
MASLOF(slaveAxis1, slaveAxis2, ...).

The setpoint status of the coupling always corresponds to the last specification made.

The current coupling status can be read back in the slave axis via PLC interface signal "Master/slave coupling active" (DB31, ... DBX96.7).

In the part program and from the synchronized 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 instruction MASLOFS(Slv1, Slv2, ...)
can be used too disconnect the coupling in the same way as MASLOF and decelerate the slave spindle automatically.

Note

A permanent coupling
(MD 37262 MS_COUPLING_ALWAYS_ACTIVE=1)
does not require explicit activation.

For more information about how on a permanent master-slave coupling for the slave axis, PresetOn can be used to synchronize the actual value to the same value as the master axis, see

References: /PGA/, Chapter 13 "Master-Slave Grouping"

Example

For an example of how to configure the master-slave coupling between AX1=master and AX2=slave, please see:

Section 6.1. Speed/Torque Coupling (SW 6 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 instruction is retained after a mode change, RESET or end of part program.
- 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.

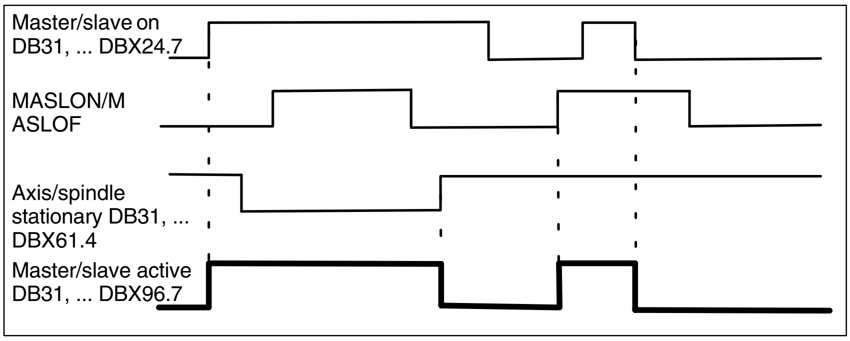


Fig. 2-7 Activation procedure

Block stepping is halted for MASLON until the coupling has actually been closed. During this time, the associated channel operating message "Master-slave switchover inactive" is displayed on the MMC/HMI.

Activation/ deactivation during 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 motion

The coupling procedure at different speeds is divided into two phases.

Phase 1

Closure of the coupling is requested with interface signal IS "Master/slave on" (DB31, ... DBX24.7). The slave spindle accelerates or decelerates along the ramp with the dynamic response available to it until it reaches the setpoint speed of the master spindle.

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 spindle(s) is used to generate the synchronism signals.

IS "Speed tolerance coarse" (DB31, ... DBX96.3) and
IS "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 "Tolerance coarse" signal can be used to implement a PLC monitoring function that checks a coupled group for loss of speed synchronism. The "Tolerance fine" signal can be used to derive the time for mechanical closure of the coupling and to activate the torque compensatory controller directly.

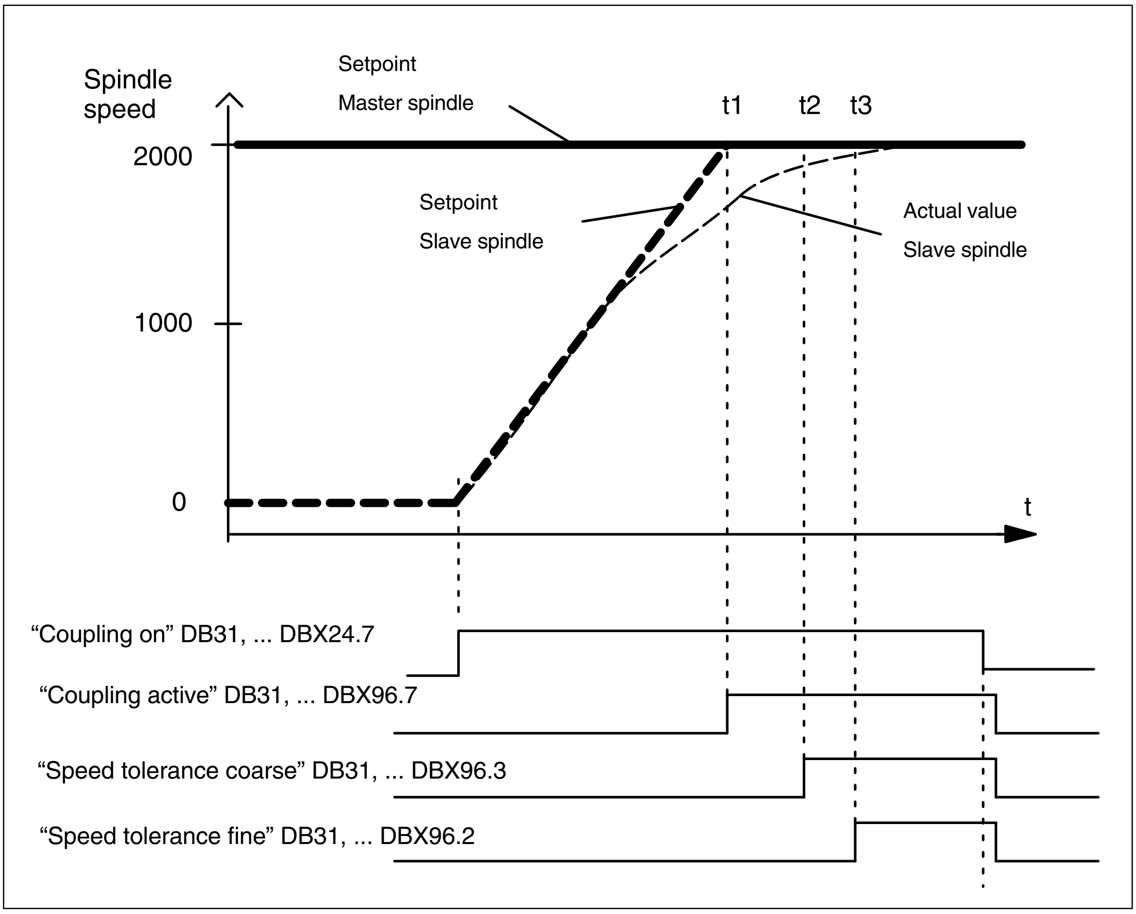


Fig. 2-8 Coupling procedure between two spindles with different speeds

Deactivation during motion

An active coupling is disconnected using the MASLOF instruction.

This instruction 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.

You can use the MASLOFS instruction to decelerate slave spindles automatically when disconnecting the coupling. For axes and spindles in positioning mode, the coupling is still only disconnected at standstill.

Note

The implicit preprocessor stop is omitted for MASLON and MASLOF. The missing preprocessor stop means that the $P system variables of the slave spindle do not supply updated values until reprogrammed.

Coupling characteristics (SW 6.5 and higher)

For spindles in speed control mode, the coupling characteristics of the MASLON, MASLOF, MASLOFS, MASLDEL instructions and the PLC with IS “Master/Slave ON” (DB31, ... DBX24.7) is defined explicitly via

MD 37263: MS_SPIND_COUPLING_MODE.

MD 37263 = 0

Coupling and disconnection take place only at standstill.

The current coupling state is retained until all axes involved have actually come to a standstill. The MASLOFS and MASLOF instructions are identical; the slave spindle is not decelerated automatically.

MD 37263 = 1

Coupling and disconnection takes place immediately and therefore during motion.

During coupling, the slave spindles are accelerated automatically to the current speed of the master spindle.

On disconnection, the slave spindles rotating at this time retain their speeds until next speed programming. However, a slave spindle disconnected with MASLOFS decelerates automatically.

2.1.8 Axial interface signals

When a master/slave coupling is requested, the PLC axis enables “Servo enable” (DB31, ... DBX2.1) and “Pulse enable” (DB31, ... DBX21.7) of the slave axis are derived directly from the specifications of 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 controller enable signals for the individual axes are not canceled until MD 36620: SERVO_DISABLE_DELAY_TIME has expired.

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 “Current controller active” (DB31, ... DBX61.7) or “Speed controller active” (DB31, ... DBX61.6) drive status signals are missing in the master or slave axis, the PLC interface signal “Master/slave active” (DB31, ... DBX96.7) is reset in the slave axis at standstill. When the master and slave axes return to closed-loop control mode, IS Master/Slave active (DB31, ... DBX96.7) is set on the slave axis.

With IS (DB31, ... DBX24.4), the torque compensatory controller is activated by the PLC. The status of the torque compensatory controller can be read from IS “Master/slave comp. contr. active” (DB31, ... DBX96.4).

Note

If the coupling is closed, the slave axis operates in speed control mode; status signal “Position controller active” (DB31, ... DBX61.5) is not enabled.

2.1.9 Axial monitoring functions

With the exception of speed setpoint and actual velocity monitoring, axial monitoring functions such as contour and standstill monitoring are not active due to the lack of a position controller.
Position control parameters such as gain factor, feedforward control and balancing may therefore be set to different value in the master and slave axis without triggering a response from the monitoring functions.

To achieve the same braking response for all coupled axes in the event of a fault, the same alarm reaction is applied to the entire coupling grouping when the coupling is active.

When correcting fault states, 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 closed master-slave coupling, MD 37268: MS_TORQUE_WEIGHT_SLAVE should be set to a low value in the slave axis. Traversing of a coupled-motion slave axis is not prevented by the torque compensatory controller in this case.

Reference point approach

If the coupling is closed, only the master axis can be referenced. Referencing of slaves axes is suppressed. The referencing requirement does not have to be explicitly canceled for the slave axis in order to do this. The referencing status of coupled slave axes remains unchanged. The slave axis position is generally not the same as the master axis position. This difference in position is not significant. If the coupling is not closed, each axis can be referenced separately as usual.

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. Disconnecting the coupling during an axis reversal error will generate an incorrect actual value for the slave axis.

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 drive. 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 does not have to be activated explicitly. The current settings of the master axis apply. The speed feedforward value of the master axis is already incorporated in the speed setpoint of the slave axis. If torque feedforward control is active, the load-side torque feedforward value of the master axis is also applied in the slave drive.
In coupled mode, the mechanical ratios change.
Settings applied axially must be modified accordingly. All coupled drives should have the same speed control dynamics.

Gantry

If one master-slave relationship is defined on each side of the gantry grouping to increase the gain, only the leading axis or following axis may be operated as master axis.

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.

Weight counterbalance

The additional torque for the electronic weight counterbalance MD 32460: TORQUE_OFFSET is computed in the slave axis irrespective of its coupling status.

Gear stage change with active master-slave coupling

An automatic gear stage change in a coupled slave spindle is not possible and can only be implemented indirectly using the master spindle. The point in time at which the gear stage is changed is then derived from the master spindle. The oscillating motion of the coupled slave spindle is generated implicitly via the oscillating motion of the master spindle.

Unlike the master spindle, the associated parameter set must be selected explicitly in the coupled slave spindle.
In order to enable parameter sets to be specified, machine data

MD 35590: PARAMET_CHANGE_ENABLE must be set to the value 2.

In the event of a gear stage change for the master/slave spindle, the associated parameter set index can be activated by the PLC via the VDI interface.

Note

For more information about gear stage change and parameter sets for changes in spindle mode, see:

References: /FB1/, S1, “Spindles” Chapter 2
/FB1/, A2, “Various Interface Signals”
Interface signals from/to axis/spindle

Axis container

If a coupled slave axis is configured in an axis container, alarm “4025 Switch axis container %3 not permitted: Master-slave active channel %1 Axis %2” is output. The axis container may not be advanced because the coupling is active.

SW 6.4 and higher

In the event that masters change, dynamic configuration can be used to match the relevant spindle the master spindle following a rotation of the axis container. Both master and slave spindles can be container spindles.

For a coupling to be closed after container rotation using a different spindle in each case, the old coupling must be disconnected before the rotation, the configuration deleted and the new coupling closed after the rotation.

Example for a cyclic coupling sequence (Position=3/Container=CT1)

```
MASLDEF(AUX,SPI(3))          ; S3 Master for AUX
MASLON(AUX)                  ; Coupling ON for AUX
M3=3 S3=4000                 ; Processing ...
MASLDEL(AUX)                 ; Delete configuration and
                             ; disconnect coupling
AXCTSWE(CT1)                 ; Container rotation
```

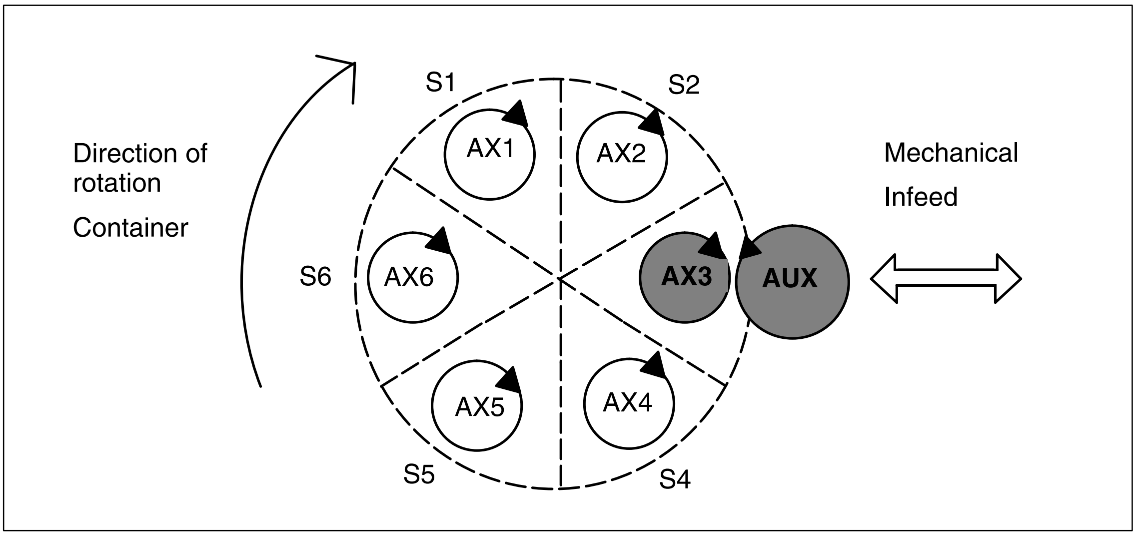


Fig. 2-9 Coupling between container spindle S3 and auxiliary motor AUX (prior to rotation)

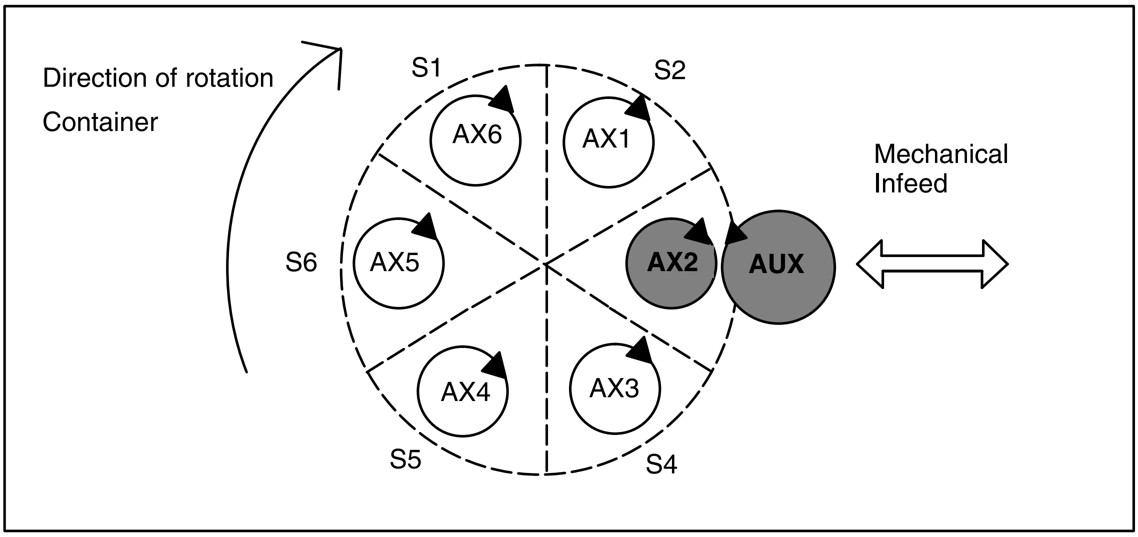


Fig. 2-10 Coupling between container spindle S3 and auxiliary motor AUX (after to rotation)

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 detection of the slave software limit switch equals the distance required by the master axis to brake the coupling.

The master axis controls the movement away from the limit switch, since the coupling cannot be disconnected until the cause of the alarm has been eliminated.

Block search

The SERUPRO “block search with calculation” function can be used without restriction in combination with a permanent master-slave coupling if 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.
- Once the block search has been completed, the associated axis positions and speeds must be modified subsequently by the user via a system ASUB (asynchronous subroutine) “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 changed
      AND ($AA_MASL_STAT[Y]<>0))       ; during the block search and
                                       ; the current status is coupled.
N30 MASLOF(Y)                          ; Open coupling
N40 SUPA Y=$AA_IM[X]–$P_SEARCH_MASLD[Y]
                                       ; Cancel position offset
                                       , on 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)      ; The coupling status of the
      AND ($AA_MASL_STAT[SPI(2)]==0))     ; 2nd spindle changed
                                          ; during the block search and
                                          ; the current status is open.
N30 M2=$P_SEARCH_SDIR[2]                  ; Update direction of rotation
N40 S2= $P_SEARCH_S[2]                    ; Update speed
N50 ENDIF
N60 ENDIF
N70 REPOSA
```

- In order that the PROGEVENT.SPF subroutine can start automatically, the following machine data must be parameterized accordingly:

– MD 11450 SEARCH_RUN_MODE = H02
– MD 11602 ASUP_START_MASK = H03
– MD 11604 ASUP_START_PRIO_LEVEL = 100

For more application examples, see **Chapter 6**.

Note

For more information about event-driven program calls and block searches in program test mode (SERUPRO), please see:

References: /FB/, K1, Mode Group, Channel, Program Operation

2.1.11 Compatibility of SW 6.4 with earlier versions

Implicit preprocessor stop

The implicit preprocessor stop 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.

Up to SW 6.4	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 at which configuration alarm 26031 is output changes from the time at which the control starts up to the time at which an attempt is made to close the coupling. The alarm is acknowledged with a reset.

2.1.12 Supplementary conditions in 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.

In the coupled state, the acceleration of spindles at the current limit may not provide an adequate adjustment reserve for the torque compensatory controller in order to maintain the desired distribution of torque between master and slave.

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 (up to SW 5.x)

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, otherwise the two motors would work in opposition.

Master-slave operation possible only with digital 611D drives.

One of the two drives, the "master", is programmed, while the other drive, the "slave", is linked via the speed setpoint coupling.

This function essentially consists of:

- A speed setpoint coupling and
- A torque controller 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 NC loses the position reference of the slave axis. 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 as it would cause the controllers between the master and slave to work in opposition.

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, a configurable tension torque can be applied to the torque controller via machine data.

The master and slave axes do not have to be programmed in the same channel.
The speed setpoint is coupled in the position controller 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.

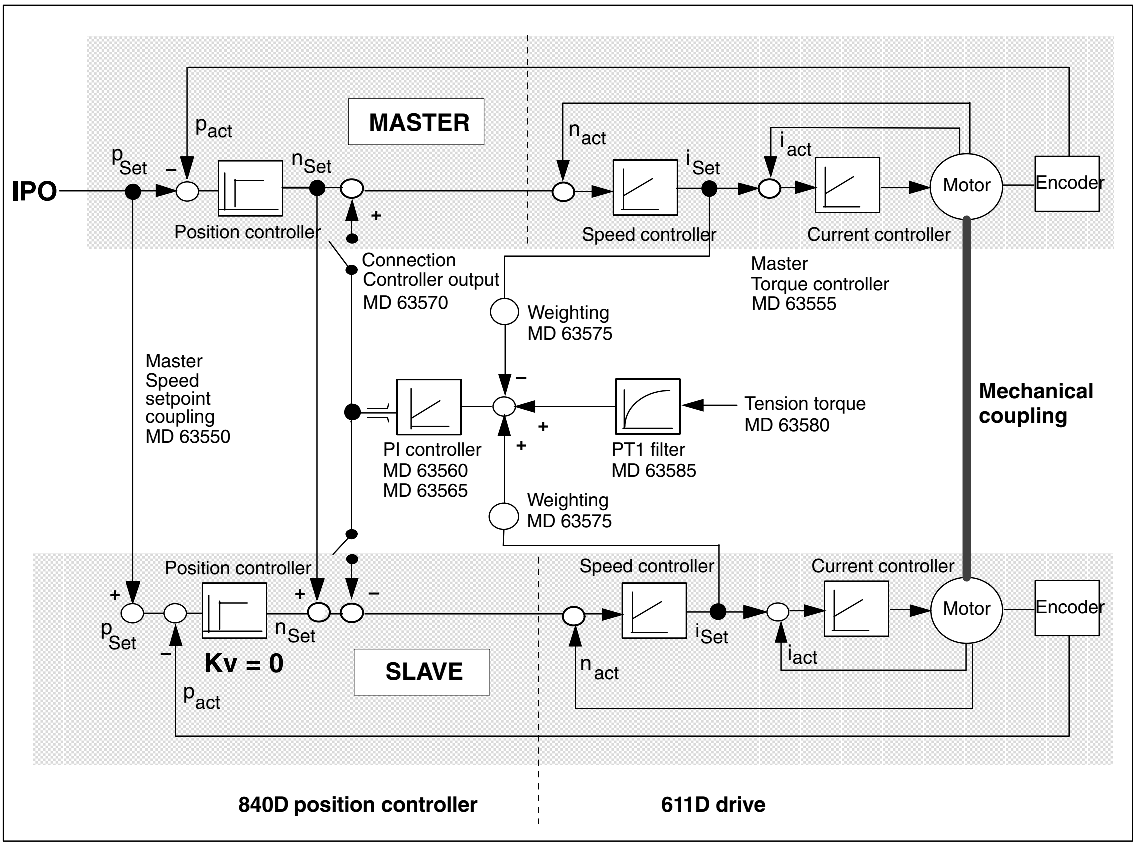


Fig. 2-11 Control structure

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. 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 speed setpoint 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 speed setpoint 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.

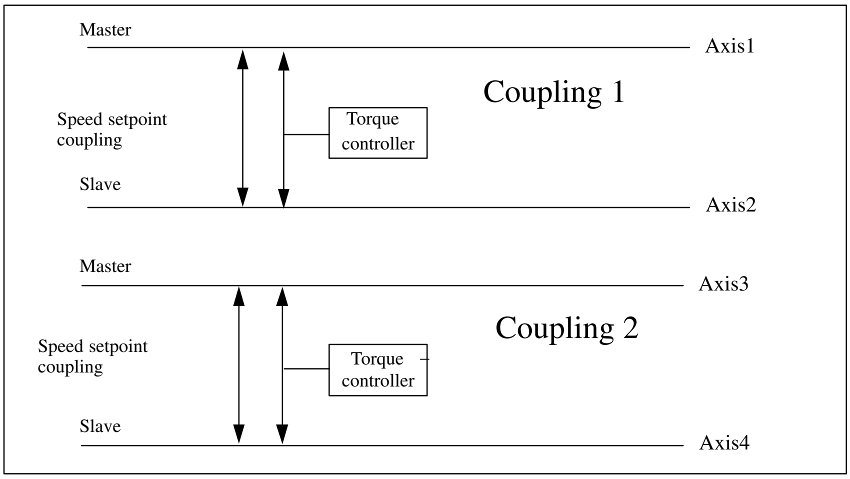


Fig. 2-12 Independent master-slave couplings

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. Please note:

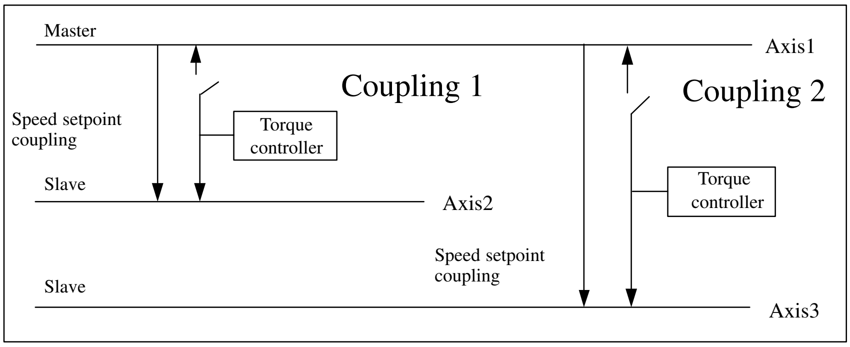


Fig. 2-13 One master, two slaves

The torque controller for coupling 1 attempts to maintain the same torque between axis 1 and axis 2 by writing a speed setpoint to axis 1 and axis 2. The torque controller for coupling 2 also tries to maintain the same torque between axis 1 and axis 3. Both controllers would write speed setpoints to axis 1.

In order to ensure a stable system, both controllers must be parameterized so 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 controllers now try to match the torque of the slave axis to the torque of the master axis, without adding speed setpoints 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 basic display and the actual value is frozen. If a coupling is always active, i.e. the slave axis is never traversed individually, we recommend that this axis is displayed as the last axis in the automatic basic display. This is achieved by entering this axis as the last axis in the channel (MD 20070: AXCONF_MACHAX_USED).

Several channels

The master axis and slave axis do not have to be programmed in the same channel. Multi-channel couplings are possible for several active channels.

Axis replacement

Although provision can be made for replacing axes between channels (MD 30550: AXCONF_ASSIGN_MASTER_CHAN), this is subject to 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 and 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 axes. 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 inverted for one of the drives with MD 32100: MOTION_DIR. In this case, the speed setpoint and the output of the torque controller are calculated correctly; there is no need to set further machine data.

Different motor speeds

The 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 speed feedforward control is active in the master axis, 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.

Computing time load

Each master-slave coupling places a load on the position control level and the interpolation level. The table shows the computing time required depending on the NCU hardware.

NCU	Position control	Interpolator level
572	1st coupling 0.120 ms each additional coupling + 0.050 ms	1st coupling 0.100 ms each additional coupling + 0.020 ms
573	1st coupling 0.040 ms each additional coupling + 0.020 ms	1st coupling 0.030 ms each additional coupling + 0.010 ms

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 speed setpoint 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 controller” is present.

All axes of the mode group follow on; the alarms can only be reset with POWER ON.

2.2.4 Torque controller

The torque controller between master and slave ensures even torque distribution 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 setpoint speed nset, which is applied to the master and slave axes.

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 speed setpoint nset by multiplying the torque difference Mdiff by a gain factor Kp. The resulting speed setpoint is added to the master and slave axes.

nset = Mdiff * Kp

The P gain Kp of the torque compensatory controller has the dimension [(mm/min)/Nm].

The reset time is entered in the axial MD 63560: MS_TORQUE_CTRL_P_GAIN as a percentage value of the following ratio:
Maximum drive velocity [mm/min]/Rated torque [Nm].

The maximum drive velocity is the content 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 controller.

Example:	
Maximum drive velocity of the slave axis	30000 mm/min
Motor rated torque of the slave axis	10 Nm
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 the 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 compensatory controller I_TIME. Ki can only be calculated if the gain factor of the P controller Kp 0. The I controller is not active unless the P component is also activated.

Ki = 1/ position controller cycle * I_TIME * Kp

The reset time is entered in the axial MD 63565: MS_TORQUE_CTRL_I_TIME in seconds.
The default setting 0 deactivates the I component if the P component already ensures appropriate torque distribution.

Limiting the controller output

The MD 63600: MS_MAX_CTRL_VELO can be used 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. The default is 100%. The limit works in both a positive and a negative direction.

Interconnecting the torque controller output

You can use an additional MD 63570: MS_TORQUE_CTRL_MODE to connect the output of the torque controller 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 speed setpoint coupling applies.

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 compensatory controller in order to enable 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 and the torque of the master axis is weighted with the difference between the MD and 100%.

Example:
The slave axis is to produce 30% of the overall torque.
70% is to be supplied by the master axis.
MS_TORQUE_WEIGHT_SLAVE = 30

Tension

An axial MD 63580: MS_TENSION_TORQUE can be used to connect a constant tension torque as an input to the torque controller. This tension torque is injected continuously and ensures that both of the coupled drives are under tension.
The slave axis MD is the important setting 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 to suit individual machining situations.

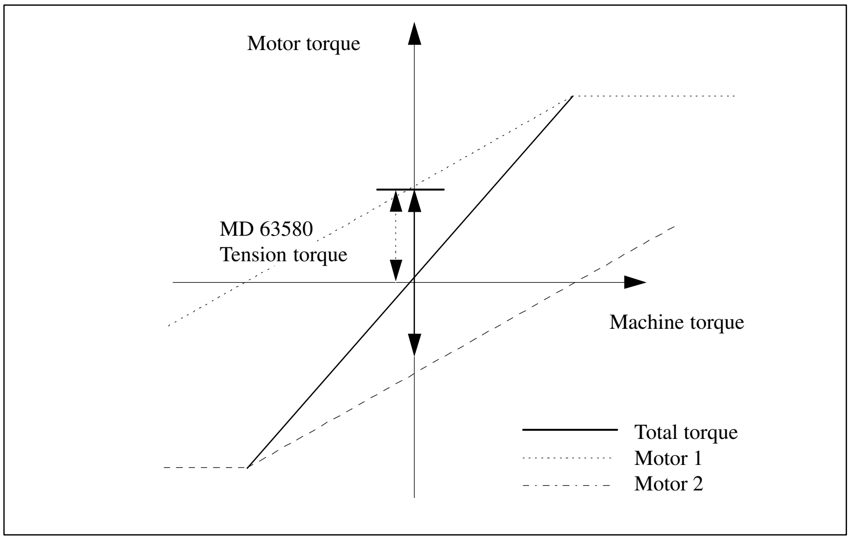


Fig. 2-14 Tension torque

PT1 filter

The tension torque is supplied 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 speed setpoint at the torque controller output when the controller is operated without an I component. The PT1 filter is configured using MD 63585: MS_TENSION_TORQ_FILTER_TIME. The time is entered in seconds. Enter 0 to deactivate the PT1 filter.

Note

The function ensures distribution of the torque-producing currents (Iq) and not distribution of the torques.
This means that torque distribution is also assured on FSD synchronous motors (no field weakening). In contrast, however, only current distribution is assured on MSD asynchronous motors in the field-weakening range. Torque distribution is assured only on motors of the same type operating simultaneously at the same speed. If MSD motors are not operated in the field-weakening range, 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 axes are put into operation individually in a master-slave coupling, whereby the individual axis takes the full load, the P component in the speed controller must then be halved in the two axes. This is the only way to ensure that overshoot is avoided 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 axes 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. In the event of axis replacement, 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

The 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. Once 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 part 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 part program is terminated with M30 or RESET.

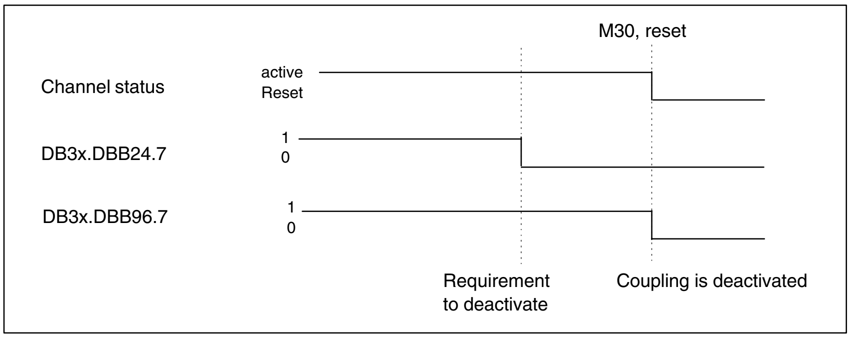


Fig. 2-15 Deactivating a coupling

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 via the PLC (DB3x.DBB24.7).

DB3x.DBB96.7	“State of 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 stationary”

Since the coupling is processed in the position controller 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 and 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 starts up, the slave axis is synchronized with the NC so that the slave axis can be traversed from this position.
The motion of the slave axis is always displayed in the “Dialog” menu, “Service display” softkey (even in the coupling is active).

Traversing the slave axis

A slave axis in an active coupling must not be traversed by the part 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 part program (G74).

If a master-slave coupling can be activated via a PLC signal (DB3x,DBB24.7), the master and slave axes are referenced individually in the “not active” coupling state.

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 the NC can start up without having to reference the slave axis.

Response in the event of an error

In the event of error conditions for alarms with alarm reaction "Follow-up in master and/or slave", each axis is decelerated 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 both axes have returned to the "control active" state following a channel reset.

■

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.

"Virtual axis"

In connection with "master-slave", the use of the function: "Virtual axis":
MD30132: $MA_IS_VIRTUAL_AX (axis is virtual axis)
is not permitted.

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 information

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 (up to SW 5.x)

- 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 coupled slave 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 part program processing.
- 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. So cascading is 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 (up to SW 5.x)

3.2.1 Axis replacement

Axes can only be exchanged between channels subject to the following restrictions:
In order to activate or deactivate a coupling, the slave and master axis channels must be in the RESET state. 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. Please note:
On the slave axis, the actual value in the “Diagnosis” menu under the “Service” softkey 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 speed setpoints in [mm/min] or [rpm] of the P controller and the I controller of a torque controller 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 see Section “File and Program Management” in the document “SINUMERIK 840D/810D/ Programming Guide Advanced”.

Proceed as follows:

Create SGUD:

– “Services” menu

– If the “Definitions” directory does not appear, select definitions using the “Data selection” softkey

– Open the Definitions directory

– “Manage data” softkey

– “New” softkey

– Create file
Name: SGUD
File type: Select global data/system

– OK

– The file opens. Enter the following lines:

```
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]
M30
```

– Close file and load

File: Create “Initial.ini”:

– Menu: Services > “Manage data” softkey > “New” softkey

– Create new directory type “NC data backup” and in this create the file: “Initial.ini”

Name: initial
Type: Initialization program

– OK

– The file is opened. Enter the following line:

```
M17
```

– Close file and 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 torque controller output 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
Content:
[OemSignalList]

Item0	= Type := Title,	Signalindex := – 1,	Unit := No
Item5	= Type := Signal,	Signalindex := 200,	Unit := Torque|Force
Item10	= Type := Signal,	Signalindex := 201,	Unit := Torque|Force
Item15	= Type := Signal,	Signalindex := 202,	Unit := NcSpeed

File: \ oem \ language \ Ibsvt_gr.ini
Content:
[OemComboBoxItemNames]

Item0	=	“MASTER-SLAVE”
Item5	=	“Master torque”
Item10	=	“Slave torque”
Item15	=	“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

In order to increase the resolution of the signals, the data is displayed in the following units:
Torques 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 data is to be displayed:

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.

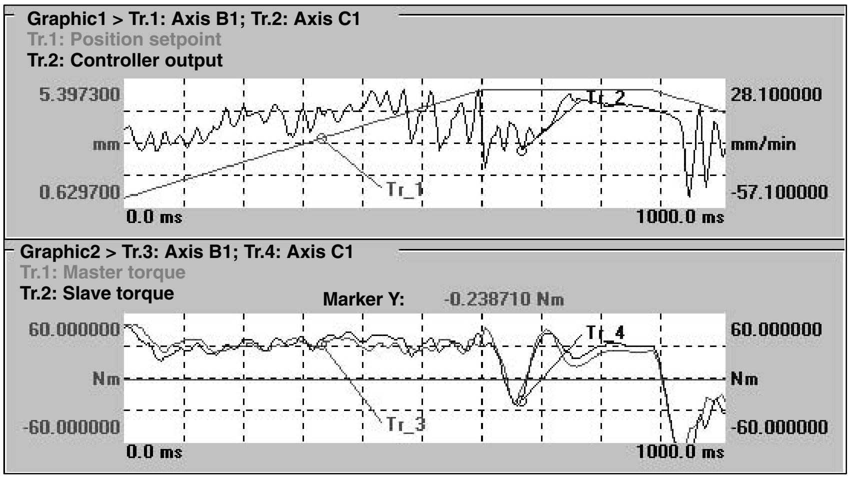


Fig. 3-1 Example of a measurement with 4 measured values

3.2.6 Controller data to analog output

Machine data MD 63595 TRACE_MODE Bit0 can be used to activate the output of controller data to an analog input.
The following data are output at the analog output on 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

Referenced to the rated torque, the torques are normalized to 8 V; referenced to the max. velocity of the slave axis, the torque controller output (in mm/min) is normalized to 8 V.

MD 10364 HW_ASSIGN_ANA_FASTOUT is used to specify the slots used by the analog converter on the terminal block.

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: in TF_GR.COM:

075150 0 0 “Slave axis %1 and master axis for speed setpoint 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 for speed/torque coupling (SW 6 and higher)

37250 MD number	**MS_ASSIGN_MASTER_SPEED_CMD** Machine axis number of master axis for speed setpoint coupling	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 31
Changes effective after POWER ON	Protection level: 2	Unit: –
Data type: DWORD	Applies as of 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 MD number	**MS_ASSIGN_MASTER_TORQUE_CTR** Master axis number for torque control	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 31
Changes effective after POWER ON	Protection level: 2	Unit: –
Data type: DWORD	Applies as of 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. Uniform torque distribution is ensured by the torque compensatory controller. In the default setting = 0, the same master axis is used for torque distribution as for the speed setpoint coupling MS_ASSIGN_MASTER_SPEED_CMD.	
Related to	MD 37250 MS_ASSIGN_MASTER_SPEED_CMD MD 37254 MS_TORQUE_CTRL_MODE MD 37256 MS_TORQUE_CTRL_P_GAIN MD 37258 MS_TORQUE_CTRL_I_TIME MD 37260 MS_TORQUE_WEIGHT_SLAVE	

37254 MD number	MS_TORQUE_CTRL_MODE Connection of torque control output		
Default setting: 0	Minimum input limit: 0	Maximum input limit: 3	
Change effective: Immediately	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies as of SW: 6.1		
Meaning:	The output of the torque compensatory controller is applied when torque distribution is active: 0: Master and slave axes 1: Slave axis 2: Master axis 3: None of the axes		
Special cases, errors, ...	for		
Related to	MD 37250 MS_ASSIGN_MASTER_SPEED_CMD MD 37252 MS_ASSIGN_MASTER_TORQUE_CTR MD 37255 MS_TORQUE_CTRL_ACTIVATION		
Additional references	Tab. 2.2 Combination options for machine data		

37255 MD number	MS_TORQUE_CTRL_ACTIVATION (SW 6.4 and higher) Activate torque compensatory control		
Default setting: 0	Minimum input limit: 0	Maximum input limit: 1	
Changes effective after NEW_CONF	Protection level: 2 / 7	Unit: –	
Data type: BYTE	Applies as of SW: 6.4		
Meaning:	The torque compensatory controller can be switched on and off either via MD 37254 or via the PLC (DB31, ... DBX24.4). In the case of activation/deactivation via the PLC, MD 37254 is used only to interconnect the torque compensatory controller. Value 0: Activation/deactivation via MD 37254 Value 1: Activation/deactivation via DB31, ... DBX24.4		
Related to	MD 37254: MS_TORQUE_CTRL_MODE		

37256 MD number	MS_TORQUE_CTRL_P_GAIN Gain factor of torque compensatory controller		
Default setting: 0	Minimum input limit: 0	Maximum input limit: 100	
Changes effective after NEW_CONF	Protection level: 2/7	Unit: %	
Data type: DOUBLE	Applies as of SW: 6.1		
Meaning:	Gain factor of the torque compensatory controller The gain factor is entered as a percentage of the maximum load-side axis velocity of the slave axis in relation to the rated torque. The maximum axis velocity is derived from MD32000 and the maximum rated torque from the product of drive machine data MD1113: TORQUE_CURRENT_RATIO and MD 1118: MOTOR_STANDSTILL_CURRENT.		
Related to	MD 37254 MS_TORQUE_CTRL_MODE MD 37258 MS_TORQUE_CTRL_I_TIME MD 32000 MAX_AX_VELO		

37258 MD number	MS_TORQUE_CTRL_I_TIME Reset time for torque compensatory controller		
Default setting: 0	Minimum input limit: 0	Maximum input limit: 100	
Changes effective after NEW_CONF	Protection level: 2/7	Unit: s	
Data type: DOUBLE	Applies as of SW: 6.1		
Meaning:	Reset time for torque compensatory control The reset time does not become active until the P gain factor > 0.		
Related to	MD 37254 MS_TORQUE_CTRL_MODE MD 37256 MS_TORQUE_CTRL_P_GAIN MD 32000 MAX_AX_VELO		

37260 MD number	**MS_MAX_CTRL_VELO** Limitation of torque compensatory control				
Default setting: 0		Minimum input limit: 0		Maximum input limit: 100	
Changes effective after NEW_CONF			Protection level: 2/7		Unit: %
Data type: DOUBLE				Applies as of SW: 6.1	
Meaning:	Torque compensatory controller limitation The speed setpoint calculated by the torque compensatory controller is limited. The possible limitation (as a percentage) is referenced to MD 32000 MAX_AX_VELO of the slave axis.				
Related to	MD 37254 MS_TORQUE_CTRL_MODE MD 37256 MS_TORQUE_CTRL_P_GAIN MD 37258 MS_TORQUE_CTRL_I_TIME MD 32000 MAX_AX_VELO				

37262 MD number	**MS_COUPLING_ALWAYS_ACTIVE** Master/slave coupling active after power ON				
Default setting: 0		Minimum input limit: 0		Maximum input limit: 1	
Changes effective after NEW_CONF			Protection level: 2/7		Unit: –
Data type: Byte				Applies as of SW: 6.1	
Meaning:	POWER ON response of a master-slave coupling. 0: Temporary coupling The coupling is activated/deactivated via the PLC interface signals and language commands. 1: Permanent coupling The coupling is activated permanently via this machine data. The PLC interface signals and language commands have no effect.				
Related to	MD 37252 MS_ASSIGN_MASTER_TORQUE_CTR MD 37250 MS_ASSIGN_MASTER_SPEED_CMD				

37263 MD number	**MS_SPIND_COUPLING_MODE** Coupling characteristics of a spindle				
Default setting: 0		Minimum input limit: 0		Maximum input limit: 1	
Changes effective after NEW_CONF			Protection level: 2/7		Unit: –
Data type: Byte				Applies as of SW: 6.5	
Meaning:	Defining the coupling characteristics of speed-controlled spindles. 0: The coupling is only made/released at standstill. 1: The coupling is made/released immediately, i.e. even during motion. This machine data MD 37263 acts on all coupling operations of speed-controlled spindles made via PLC using IS “Master/slave on” (DB31, ... DBX24.7) or part program instructions MASLON, MASLOF, MASLOFS and MASLDEL.				
Related to	–				

37264 MD number	**MS_TENSION_TORQUE** Master-slave tension torque				
Default setting: 0		Minimum input limit: –100		Maximum input limit: 100	
Changes effective IMMEDIATELY			Protection level: 2/7		Unit: –
Data type: PERCENT				Applies as of SW: 6.1	
Meaning:	You can enter a constant tension torque between the master and slave axis as a percentage of the rated drive torque of the slave axis				
Related to	MD 37252 MS_ASSIGN_MASTER_TORQUE_CTR MD 37266 MS_TENSION_TORQ_FILTER_TIME				

37266 MD number	**MS_TENSION_TORQ_FILTER_TIME** Filter time constant master-slave tension torque		
Default setting: 0	Minimum input limit: 0	Maximum input limit: 100	
Changes effective IMMEDIATELY	Protection level: 2/7	Unit: s	
Data type: DOUBLE	Applies as of SW: 6.1		
Meaning:	The tension torque between the master and slave axes can be applied via a PT1 filter. Any changes in MD 37264 are then executed with the filter time constant. The filter is inactive by default and any torque changes take effect without filtering.		
Related to	MD 37264 MS_TENSION_TORQUE		

37268 MD number	**MS_TORQUE_WEIGHT_SLAVE** Weighting of the torque value for the slave axis		
Default setting: 50	Minimum input limit: 0	Maximum input limit: 100	
Changes effective after NEW_CONF	Protection level: 2/7	Unit: _	
Data type: PERCENT	Applies as of 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. On motors with the same rated torque, 50/50 torque distribution is recommended. The torque of the master axis is calculated implicitly as 100% – MD37268.		
Related to	MD 37252 MS_ASSIGN_MASTER_TORQUE_CTR MD 37266 MS_TENSION_TORQ_FILTER_TIME		

37270 MD number	**MS_VELO_TOL_COARSE (from SW 6.4)** Master-slave velocity tolerance “coarse”		
Default setting: 10.0	Minimum input limit:	Maximum input limit:	
Changes effective after NEW_CONF	Protection level: 2/7	Unit: %	
Data type: DOUBLE	Applies as of SW: 6.4		
Meaning:	Tolerance window “coarse” for the differential velocity between master and slave. The PLC interface signal DB31, ... DBX96.3 is set for a velocity differential within the tolerance window. The machine data is specified as a percentage (%) of MD 32000: MAX_AX_VELO.		
Related to	MD 32000: MAX_AX_VELO		

37272 MD number	**MS_VELO_TOL_FINE (from SW 6.4)** Master-slave velocity tolerance “fine”		
Default setting: 10.0	Minimum input limit:	Maximum input limit:	
Changes effective after NEW_CONF	Protection level: 2/7	Unit: %	
Data type: DOUBLE	Applies as of SW: 6.4		
Meaning:	Tolerance window “fine” for the differential velocity between master and slave. The PLC interface signal DB31, ... DBX96.2 is set for a velocity differential within the tolerance window. The machine data is specified as a percentage (%) of MD 32000: MAX_AX_VELO.		
Related to	MD 32000: MAX_AX_VELO		

37274 MD number	**MS_MOTION_DIR_REVERSE (from SW 6.4)** Invert master-slave direction of travel		
Default setting: 0	Minimum input limit: 0	Maximum input limit: 1	
Changes effective after NEW_CONF	Protection level: 2/7	Unit: –	
Data type: BYTE	Applies as of SW: 6.4		
Meaning:	The direction of travel of the slave axis is to be inverted in coupled state. 1: Invert direction of travel 0: Direction of travel unchanged		
Related to	MD 32100: AX_MOTION_DIR		

4.2 Machine data of speed/torque coupling (up to SW 5.x)

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:

- Option data

Warning

Failure to take appropriate precautions **can** have undesirable consequences.

The functions activated by the option data trigger the corresponding compile cycles. The compile cycles can significantly change the behavior of the control and can create hazardous situations via access to the NC.
Before a compile cycle is activated, appropriate safety precautions to prevent potential damage to machinery and personal injury must be taken (you may need to take action to safeguard against incorrect parameter settings or programming in the compile cycles).

63550 MD number	**$MA_MS_ASSIGN_MASTER_SPEED_CMD** Configuration of a master/slave coupling	
Default setting: 0	Minimum input limit: 0	Maximum 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 speed setpoint coupling. This configures a master-slave coupling. Master and slave axis need not be programmed in the same channel.	

63555	**$MA_MS_ASSIGN_MASTER_TORQUE_CTRL**	
MD number	Configuration of a master/slave coupling	
Default setting: 0	Minimum input limit: 0	Maximum 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 speed setpoint coupling. This applies to most cases. Master and slave axis need not be programmed in the same channel.	

63560	**$MA_MS_TORQUE_CTRL_P_GAIN**	
MD number	P gain of the torque control	
Default setting: 0,0	Minimum input limit: 0	Maximum input limit: 100,0
Changes effective after RESET	Protection level: 2 / 7	Unit: %
Data type: DOUBLE		
Meaning:	The P controller calculates a speed setpoint nset by multiplying the torque differential Mdiff by the P gain Kp. nset = Mdiff * Kp The dimension of the P gain is [(mm/min)/Nm]. A percentage value of the following ratio is entered: Maximum drive velocity [mm/min]/Rated torque [Nm] The data of the slave axis are relevant for a torque control.	

63565	**$MA_MS_TORQUE_CTRL_I_TIME**	
MD number	Reset time I controller of the torque control	
Default setting: 0,0	Minimum input limit: 0,0	Maximum input limit: 100,0
Changes effective after RESET	Protection level: 2 / 7	Unit: s
Data type: DOUBLE		
Meaning:	The I controller calculates a speed setpoint by multiplying the sum of the torque differential Mdiff by the I gain. nset = Mdiff * Ki 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. Ki = 1/ position controller cycle * I_TIME * Kp The reset time is input in seconds.	

63570	$MA_MS_TORQUE_CTRL_MODE	
MD number	Connection of the torque control output	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 2
Modifications take immediate effect	Protection level: 2 / 7	Unit: –
Data type: INT		
Meaning:	This machine data enables the speed setpoint calculated in the torque control to be freely connected to the master and slave axes. Even if the speed setpoint is not applied to the axis, the torque control calculates a speed setpoint. Meaning: 0: Switch controller output to master and slave 1: Switch controller output only to slave 2: Switch controller output only to master 3: Controller is deactivated, only the speed setpoint coupling is active	

63575	$MA_MS_ASSIGN_MASTER_SPEED_CMD	
MD number	Weighting of the current torque values	
Default setting: 50,0	Minimum input limit: 0,0	Maximum 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 by a factor of MD minus 100%. Example: 30% of the total torque should be assigned to the slave axis. 70% is supplied to the master axis. $MA_MS_TORQUE_WEIGHT_SLAVE = 30	

63580	$MA_MS_TENSION_TORQUE	
MD number	Tension torque	
Default setting: 0,0	Minimum input limit: –100,0	Maximum 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. The value to be input is a percentage of the rated torque of the slave axis. 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 part program.	

63585	$MA_MS_TENSION_TORQ_FILTER_TIME	
MD number	Time constant of the PT1 filter for tension torque	
Default setting: 0,0	Minimum input limit: 0,0	Maximum 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 measured in seconds. If the tension torque changes, the torque continues to build up. Entering zero completely deactivates the filter.	

63590	$MA_MS_COUPLING_ALWAYS_ACTIVE	
MD number	Master-slave coupling active after POWER ON	
Default setting: 0	Minimum input limit: 0	Maximum 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. Value 1: As soon as the conditions for activation of a coupling are met following a POWER ON, the coupling is activated. It can no longer be released, i.e. it is no longer possible to operate the drives separately. Modifying the PLC signal at axis DB3x.DBB24.7 has no effect. Value 0: The coupling can be activated via the PLC signal at axis DB3x.DBB24.7.	

63595	$MA_TRACE_MODE	
MD number	Activate/deactivate master-slave trace	
Default setting: 0	Minimum input limit: 0	Maximum 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 Bit 0: 0: No trace active 1: Analog trace active: From this coupling, the torque of the master axis, slave axis and controller output is output to analog outputs on the terminal block. Bit 1: 0: Open and close coupling only in RESET channel state. 1: Open and close coupling without RESET channel state. Bit 2: 0: Open coupling if master or slave axis are in follow-up mode and the “axis stationary” signal = 1. 1: Open coupling if master or slave axis are not in closed-loop control mode and “axis stationary” signal = 1.	

63600 MD number	$MA_MS_MAX_CTRL_VELO Limit value for controller output	
Default setting: 100	Minimum input limit: 0	Maximum 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. The default setting is 100%.	

■

Notes

Signal Descriptions 5

5.1 Axisspecific signals

DB31 – DB61 **DBX24.4** Data block	**Activate torque compensatory controller** Signal(s) from axis/spindle (PLC –> NCK)
Edge evaluation: Yes	Signal(s) updated: Cyclic
Signal state 1 or edge change 0 –> 1	Torque compensatory controller is to be activated The following conditions must be met for activation: – Difference “fine” reached (DB3x.DBX96.2)
Signal state 0 or edge change 1–> 0	Torque compensatory controller is to be deactivated.

DB31 – DB61 **DBX24.7** Data block	**Activate master-slave coupling** Signal(s) from axis/spindle (PLC –> NCK)
Edge evaluation: Yes	Signal(s) updated: Cyclic
Signal state 1 or edge change 0 –> 1	Torque compensatory controller is to be activated
Signal state 0 or edge change 1–> 0	Master-slave coupling is to be deactivated. The following conditions must be met for activation and deactivation: – Master and slave axis in position control (DB3x.DBB61.7) – Master and slave axis at standstill (DB3x.DBB61.4) – The channels of the master and slave axis are in the “RESET” state (DB2x.DBB35.7) If one condition is not met, the coupling is not activated or deactivated. No alarm appears and the status of the coupling remains the same. If all the conditions are met at a later point in time, the coupling will be activated or deactivated depending on the state of the signal. The signal at the slave axis of a coupling is relevant.

DB31 – DB61 **DBX96.2** Data block	**Differential speed "Fine"** Signal(s) from axis spindle (NCK –> PLC)
Edge evaluation: No	Signal(s) updated: Cyclic
Signal state 1 or edge change 0 –> 1	The differential speed lies in the range defined by MD 37272: MS_VELO_TOL_FINE.
Signal state 0 or edge change 1–> 0	The differential speed has not reached the range defined in MD 37272: MS_VELO_TOL_FINE.

DB31 – DB61 **DBX96.3** Data block	**Differential speed "Coarse"** Signal(s) from axis spindle (NCK –> PLC)
Edge evaluation: No	Signal(s) updated: Cyclic
Signal state 1 or edge change 0 –> 1	The differential speed lies in the range defined by MD 37270: MS_VELO_TOL_FINE.
Signal state 0 or edge change 1–> 0	The differential speed has not reached the range defined in MD 37270: MS_VELO_TOL_COARSE.

DB31 – DB61 **DBX96.4** Data block	**Status of the torque compensatory control** Signal(s) from axis spindle (NCK –> PLC)
Edge evaluation: No	Signal(s) updated: Cyclic
Signal state 1 or edge change 0 –> 1	Torque compensatory control is active.
Signal state 0 or edge change 1–> 0	Torque compensatory controller is not active. The signal at the slave axis of a coupling is relevant.

DB31 – DB61 **DBX96.7** Data block	**Status of the master-slave coupling** Signal(s) from axis spindle (NCK –> PLC)
Edge evaluation: No	Signal(s) updated: Cyclic
Signal state 1 or edge change 0 –> 1	Master-slave coupling is active.
Signal state 0 or edge change 1–> 0	Master-slave coupling is not active. The signal at the slave axis of a coupling is relevant.

Examples 6

6.1 Speed/torque coupling

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 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	Both axes are mechanically coupled to one another.
3. Request to close the coupling	PLC interface signal “Master/slave on” DB32, ... DBX24.7 is set.
4. Read back coupling state	When the axis is at a standstill, the coupled slave axis sets PLC interface signal “Master/slave active” DB32, ... DBX96.7 and clears “Position controller active” DB32, ... DBX61.5. Wait for checkback signal.
5. Move master-slave grouping	The master axis is moved.

6.1.3 Close/separate coupling via part program

This application allows you to close or separate a master-slave coupling between the machine axes AX1=Master axis and AX2=Slave via the part program.

Preconditions

- A configured master axis MD 37250 0.
- Activation of a master-slave coupling via MD 37262 = 0.
- The coupling is open.

Part program

```
N10 G0 AX1=0 AX2=0; Approach coupling position. Each of the axes
                    moves to 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 master axis and slave axis.
                    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	PLC interface signal “Servo enable” DB31, ... DBX2.1 is set for both axes.
3. Evaluate checkbacks	Connect the PLC interface signals of the master axis with AND: – DB31, ... DBX61.7 “Current controller active” – DB31, ... DBX61.6 “Speed controller active” – DB31, ... DBX61.5 “Position controller active”
	Connect the PLC interface signals of the slave axis with AND: – DB32, ... DBX61.7 “Current controller active” – DB32, ... DBX61.6 “Speed controller active” – DB32, ... DBX96.7 “Master/slave active”
4. Release brakes	If the result of the AND operations on the master and slave axes is 0, the brake may be released.

■

7 Data Fields, Lists

7.1 Interface signals

DB number	Bit, byte	Name	Reference
Axis/spindle-specific			
DB3x.	DBX24.4	"Activate torque compensatory controller" (SW 6.4 and higher)	
DB3x.	DBX24.7	"Activate master-slave coupling"	
DB3x.	DBX96.2	"Differential speed Fine" (SW 6.4 and higher)	
DB3x.	DBX96.3	"Differential speed Coarse" (SW 6.4 and higher)	
DB3x.	DBX96.4	"State of torque compensatory controller" (SW 6.4 and higher)	
DB3x.	DBX96.7	"State of master-slave coupling"	

7.2 NC machine data

7.2.1 Speed/torque coupling (SW 6 and higher)

Number	Identifier	Name	Reference
Axis/channelspecific($MA_ ...)			
37250	MS_ASSIGN_MASTER_SPEED_CMD	Machine axis number of 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 (from SW 6.4)	
37256	MS_TORQUE_CTRL_P_GAIN	Gain factor of torque compensatory controller	
37258	MS_TORQUE_CTRL_I_TIME	Reset time for torque compensatory controller	
37260	MS_MAX_CTRL_VELO	Limitation of torque compensatory control	
37262	MS_COUPLING_ALWAYS_ACTIVE	Master/slave coupling active after power ON	
37263	MS_SPIND_COUPLING_MODE	Coupling characteristics of a spindle, SW 6.5 and higher	
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 (from SW 6.4)	

7.2.2 Speed/torque coupling (up to SW 5.x)

Number	Identifier	Name	Reference
Axis/channelspecific ($MA_ ...)			
34110	REFP_CYC_NR	NC start without referencing the axis	R1
36620	SERVO_DISABLE_DELAY_TIME	Cutout delay servo 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

Detailed explanations of the alarms, which may occur, appear in

References: /DA/, Diagnostics Guide

or in the Online help.

7.4 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 and the coupling status:

Identifier	Meaning	Reference
Axis/channelspecific($MA_ ...)		
$P_SEARCH_MASLC[slave axis identifier]	This variable registers a change in the coupling status during the SERUPRO block search in SW 6.2 and higher.	PGA1
$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.	PGA1
$AA_MASL_STAT[slave axis identifier]	This variable outputs the current coupling status in SW 6 and higher. 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.	PGA1/ FBSY

■

Notes

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

Handling Transformation Package (TE4)

■

1 Brief Description

Functionality

The handling transformation package has been designed for use on **manipulators** and **robots**. The package is a type of modular system, which enables the customer to configure the transformation for his machine by setting machine data (provided that the relevant kinematics are included in the handling transformation package).

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 orientation programming, the entry of tool parameters and transformation calls.

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/Base center point coordinate system
WS	Workpiece coordinate system
WZ	Tool coordinate system
x_3, y_3, z_3	Coordinates of first wrist 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.

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 a tool (milling cutter, laser beam) on a 5-axis machine to be oriented in any desired relation to the workpiece 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, which are stored in machine data.

Kinematic categories

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 to produce a right-handed coordinate system.
Translation is always entered with reference 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: 3. 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.

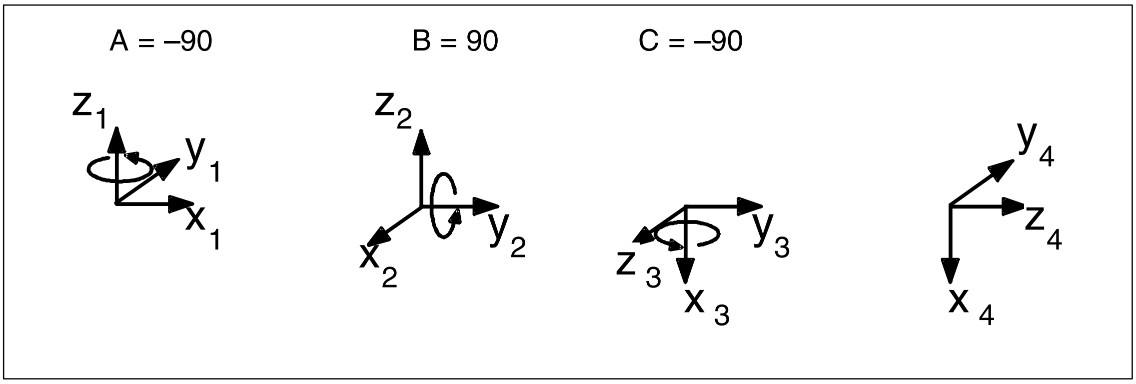


Fig. 2-1 Example of rotation through RPY angles

2.2.3 Definition of a joint

A joint is a linear or rotary axis.
The basic axis identifiers are defined on the basis of the arrangement and sequence of the individual joints. These are described by identifying letters (S, C, R, N), which are explained below.

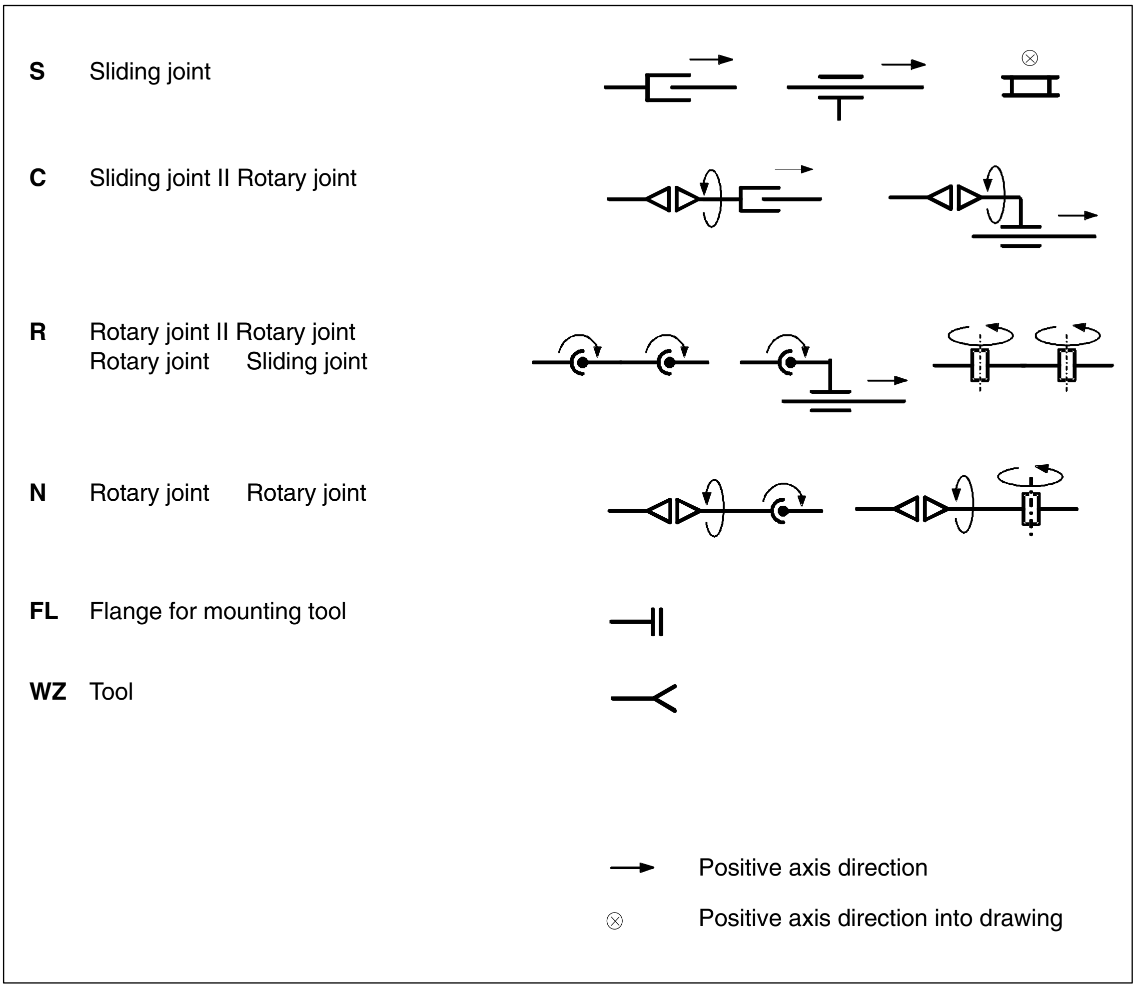


Fig. 2-2 Joint identifying letters

2.3 Configuration of a kinematic transformation

In order to ensure that the kinematic transformation can convert the programmed values into axis motions, it must have access to some information about the mechanical construction of the machine. This information is stored in machine data:

- Axis assignments
- Geometry information

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 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 part 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.

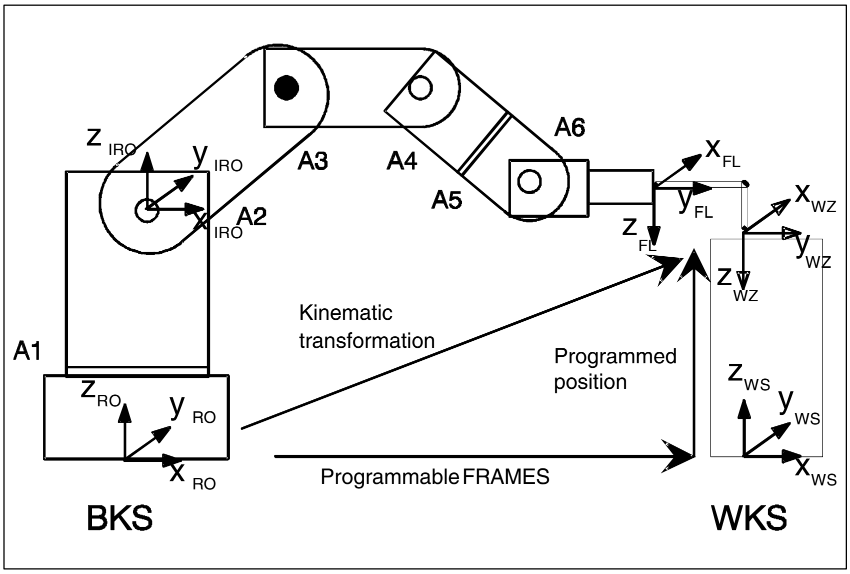


Fig. 2-3 Closed kinematic loop illustrated by the example of a robot

---

Note

For more detailed information about coordinate systems, please see:
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	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	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	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	MD 62604: TRAFO6_WRIST_AXES specifies the hand type. The term wrist axes generally refers to axes four to six.
MD 62610 TRAFO6_TFLWP_POS MD 62611 TRAFO6_TFLWP_RPY	The frame T_FL_WP links the last hand coordinate system with the first flange coordinate system.

These data are described in more detail in subsequent sections.

Basic axes included in every transformation

MD 62603
TRAFO6_MAIN_AXES

The first 3 axes included in the transformation are generally referred to as the “basic axes”. They must always be mutually parallel or perpendicular. Each of the following basic axis arrangements has its own special identifier (see Subsection 2.2.3). The MD 62603: TRAFO6_MAIN_AXES is used to enter the basic axis identifier.

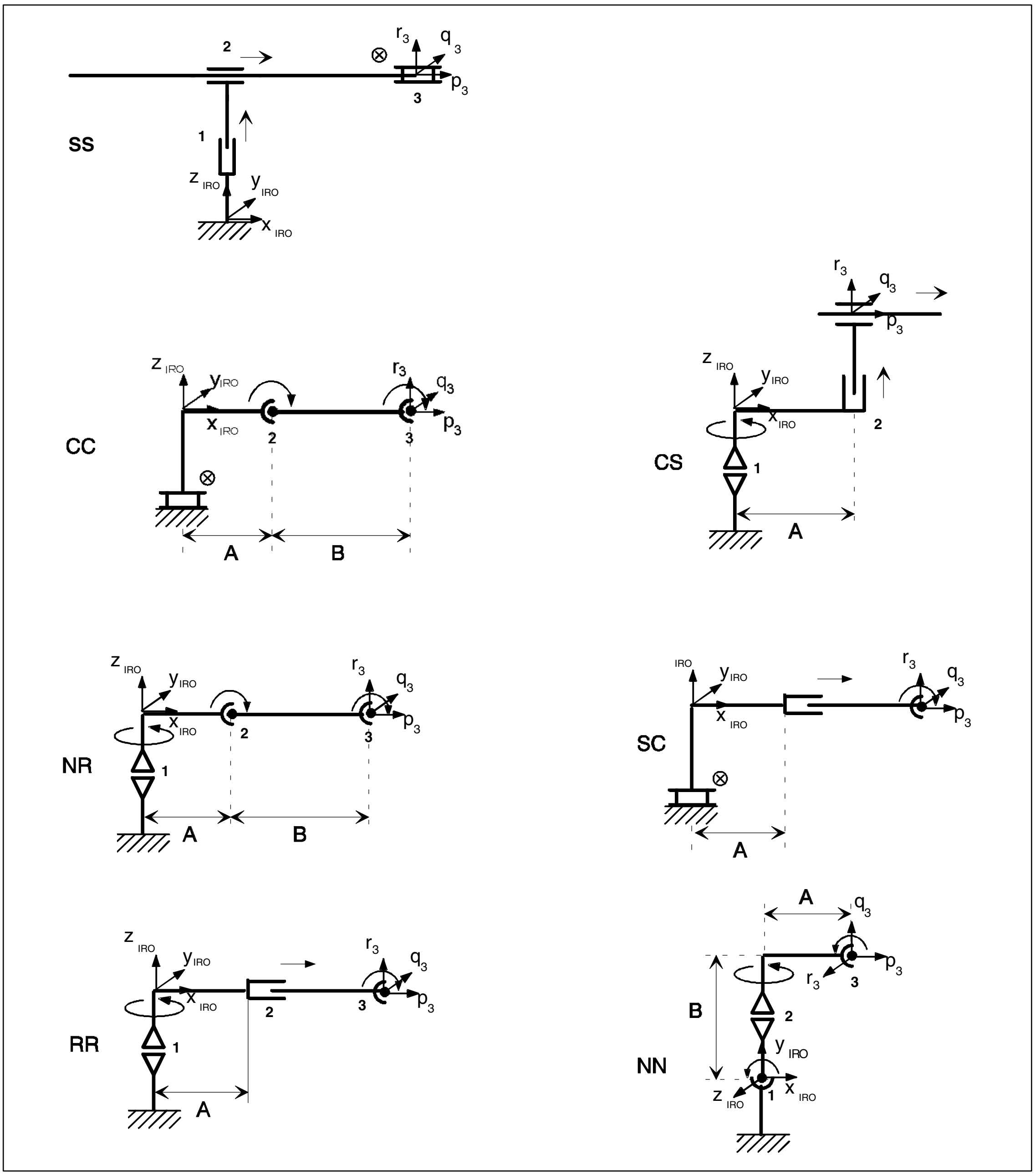


Fig. 2-4 Overview of basic axis configurations

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))

Wrist axes included in every transformation

MD 62604
TRAFO6_WRIST_
AXES

The fourth axis and all further axes are generally referred to as “wrist axes”. The handling transformation package can only identify hands with rotary axes. The wrist axis identifier for three-axis hands is entered in MD 62603: TRAFO6_MAIN_AXES. In the case of hands with fewer than three axes, the identifier for a beveled hand with elbow or a central hand is entered in MD 62603: TRAFO6_MAIN_AXES. The current software supports only wrist axis types beveled hand with elbow or central hand.

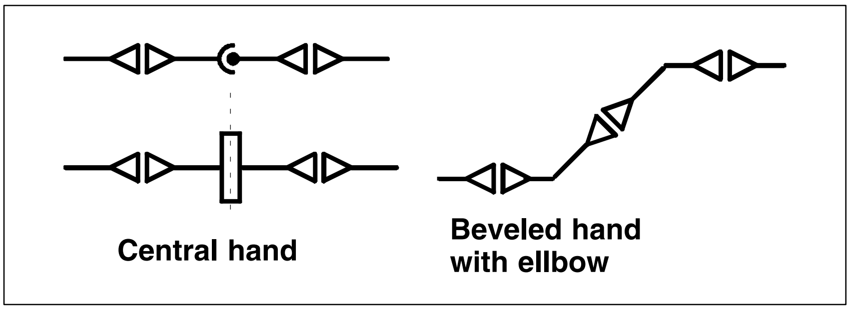


Fig. 2-5 Overview of wrist axis configurations

Parameterization of wrist 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 wrist axes intersect at one point. All parameters must be set as shown in Table 2-1.

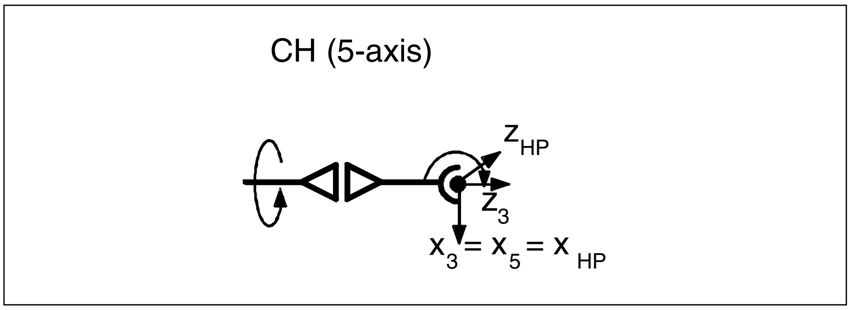


Fig. 2-6 Central hand

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, und a_4 are available for this type of hand, as shown in Table 2–1.

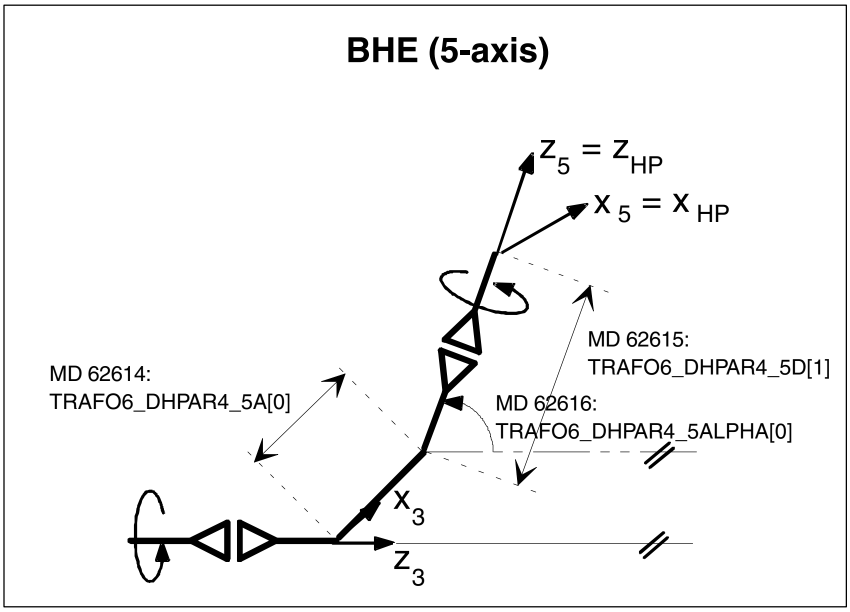


Fig. 2-7 Beveled hand with elbow (5-axis)

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	[α_4, 0.0]

Link frames

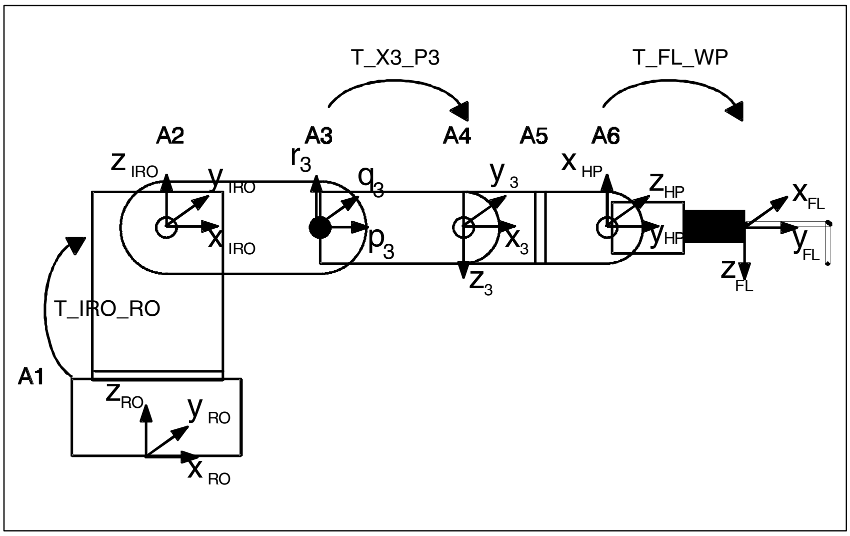


Fig. 2-8 Link frames

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 more detailed information about FRAMES, please see:
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 wrist axis (x3_y3_z3 coordinate system). The p3_q3_r3 coordinate system is included in the kinematic diagrams for the basic axis arrangements.
The z3 axis is always on the 4th axis.

Depending on the number of axes to be included in the transformation, frame T_X3_P3 is subject to certain restrictions relating to the hand and basic axes:

- For 5-axis kinematics, frame T_X3_P3 can be freely selected in the following cases:
  - If the basic axes are of the SS type.
  - If the basic axes are of the CC, CS or SC type, the transformation must either include a central hand (ZEH) or the 4th axis must be positioned in parallel to the last rotary basic axis.
  - If the basic axes are of the NR or RR type, the transformation must either include a central hand (ZEH) or the 4th axis must be positioned in parallel to the last basic rotary axis and an X flange must intersect the 5th axis.
  - If the basic axes are of the NN type, the transformation must include a central hand.
- With 4-axis kinematics, the z3 axis must always be parallel/anti-parallel or perpendicular to the last basic axis.

T_FL_WP

Frame T_FL_WP links the flange with the last internal coordinate system predefined with the handling transformation package (hand-point coordinate system).
For kinematics with fewer than 6 axes, this frame is subject to certain restrictions, which are explained with the relevant kinematics.

Other configuring data

Number of transformed axes

MD 62605 TRAFO6_NUM_AXES
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 used for the purpose of kinematic conversion. 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 with a wrist axis parallel to the last rotary basic 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

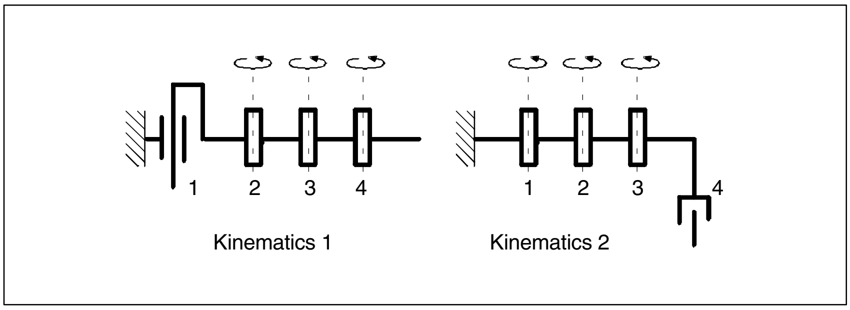


Fig. 2-9 Rearrangement of axes (example 1)

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 wrist axes are involved in the transformation.

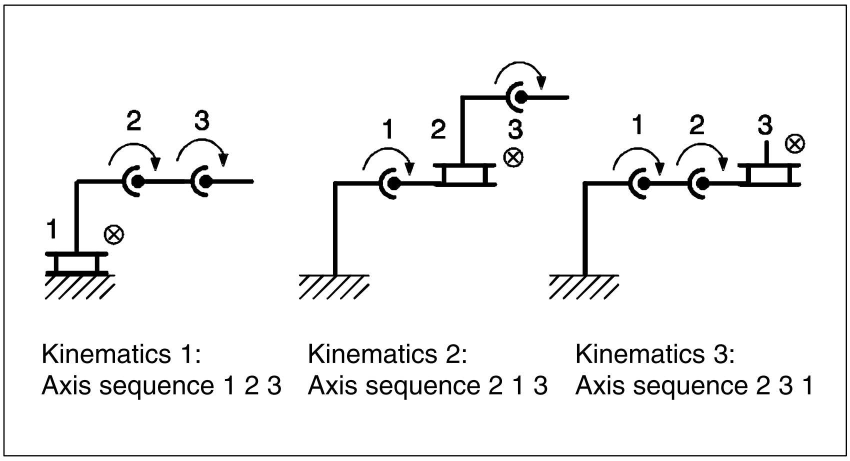


Fig. 2-10 Rearrangement of axes (example 2)

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, MD62618: TRAFO6_AXES_DIR[ ] must be set to **–1** for the relevant axis if the direction is to be inverted, or otherwise to **+1**.

Adapting the zero points of the 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 zero points, the deviation between the mathematical zero point and the adjustment 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. The mathematical zero point for axis 2 is 90°. 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 –90° as a mathematical zero point.

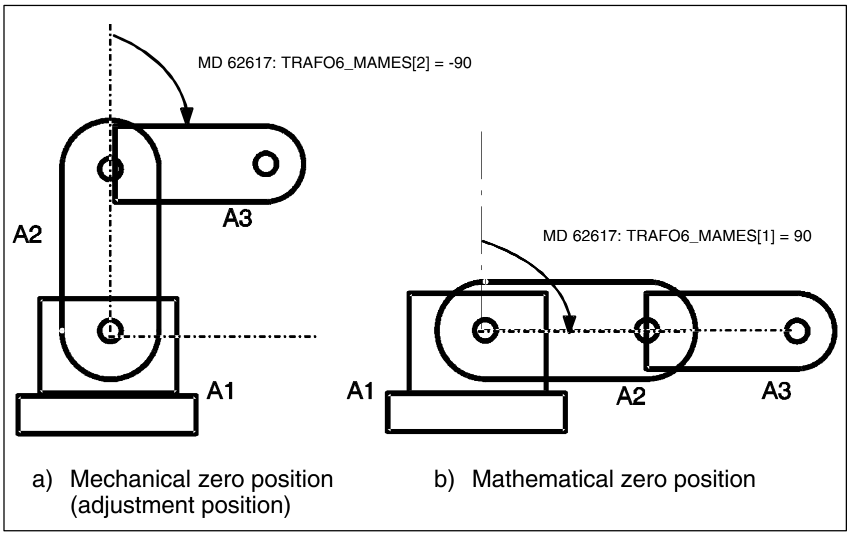


Fig. 2-11 Matching mathematical and mechanical zero points

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

Separate velocities are introduced for the Cartesian motion components for traversing the axes with G00 and active transformation.
For traversal with G01 or G02, the path velocity is specified using path feedrate F.

MD 62629
TRAFO6_VELCP

MD 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

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

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: A angle
Index i = 1: B angle
Index i = 2: C angle

MD 62632
TRAFO6_ACCORI

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: A angle
Index i = 1: B angle
Index i = 2: C angle

2.4 Descriptions of kinematics

The following descriptions of kinematics for transformations involving 2 to 5 axes explain the general configuring procedure first before describing how the machine data need to be configured, using a configuring example for each kinematic type. These examples do not include all possible lengths and offsets. The direction data refer to the positive directions of traversal and rotation for the transformation. The axis positions correspond to their zero position for the relevant transformation.

2.4.1 3-axis kinematics

3-axis kinematics normally possess 3 translational degrees of freedom, but do not have a degree of freedom for orientation. In other words, they only include basic axes.

Configuring

The procedure for configuring a 3-axis kinematic is as follows:

1. Enter “Standard” kinematic category in MD 62600: TRAFO6_KINCLASS.
2. Set the number of axes for transformation in MD 62605: TRAFO6_NUM_AXES = 3.
3. Compare the basic axes with the basic axes contained in the handling transformation package. –> Enter the 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 wrist 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 the directions of rotation of axes with the directions defined in the handling transformation package and correct in MD 62618: TRAFO6_AXES_DIR.
8. Enter the mechanical zero offset in MD 62617: TRAFO6_MAMES.
9. Enter the 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 is stored 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 kinematics

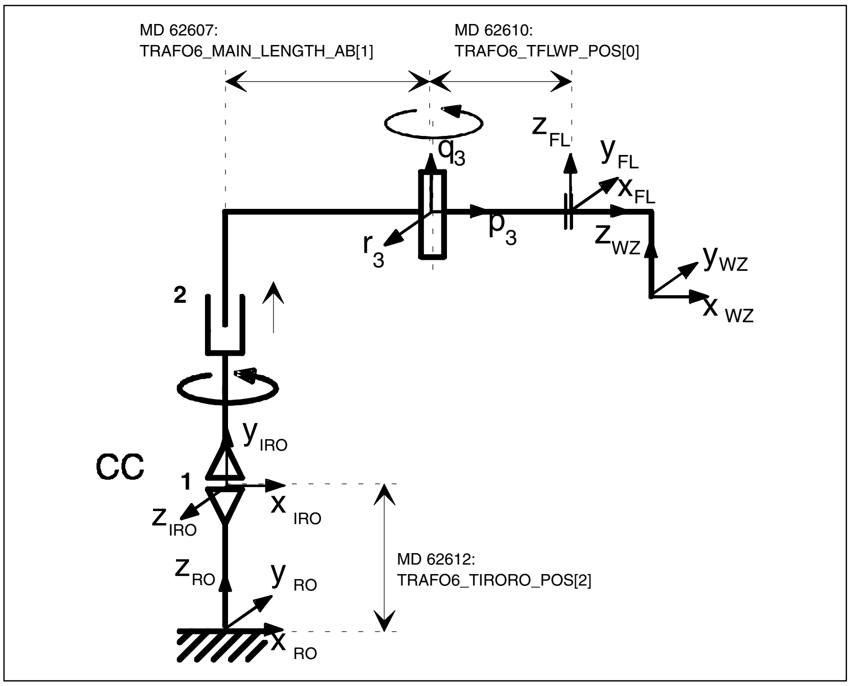


Fig. 2-12 3-axis CC kinematics

Table 2-4 Configuration data for 3-axis CC kinematics

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 Configuration data for 3-axis CC kinematics

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 kinematics

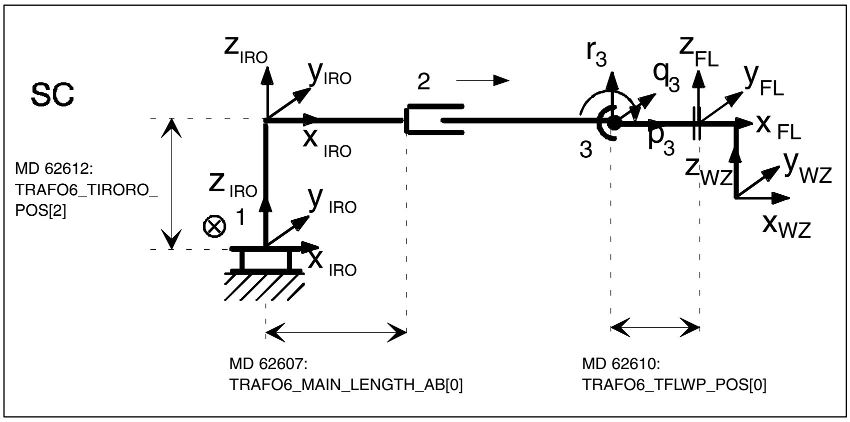


Fig. 2-13 3-axis SC kinematics

Table 2-5 Configuration data for 3-axis SC kinematics

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 kinematics

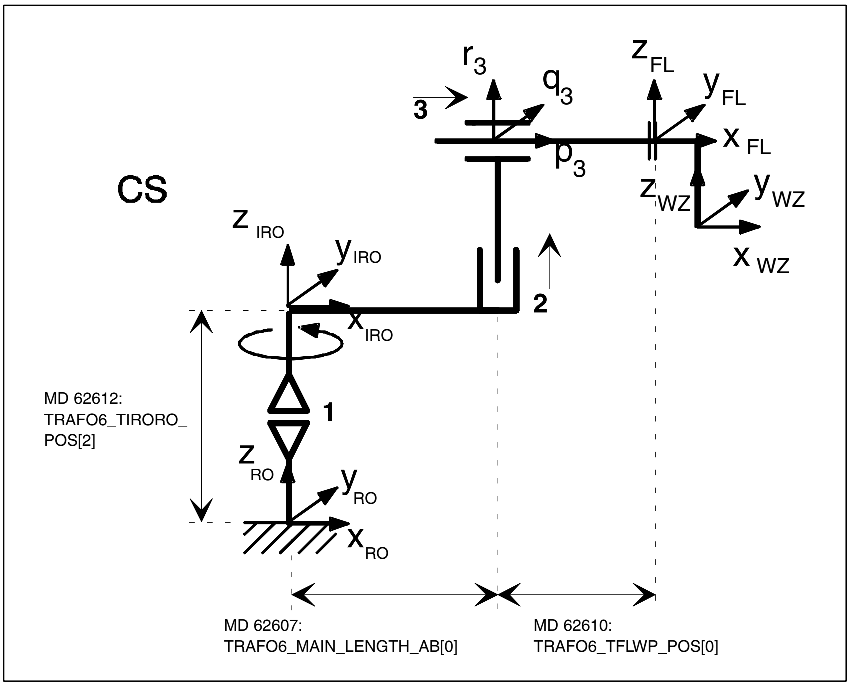


Fig. 2-14 3-axis CS kinematic

Table 2-6 Configuration data for 3-axis CS kinematics

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

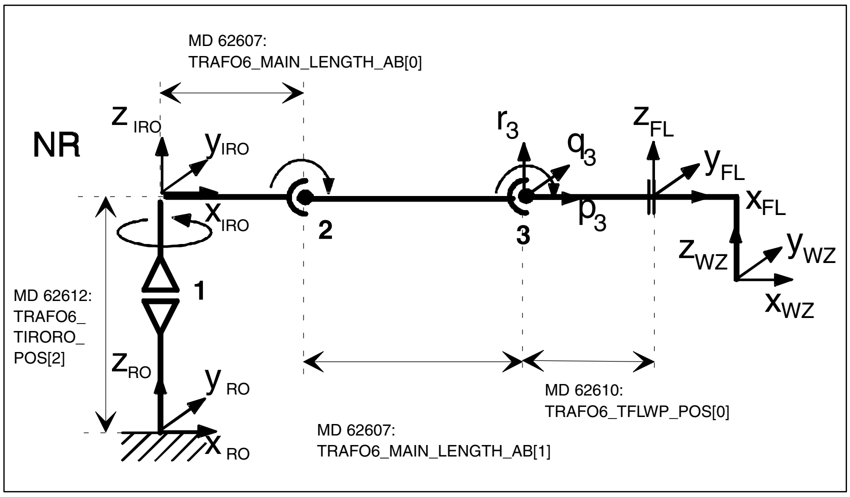


Fig. 2-15 3-axis NR kinematics

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 kinematics

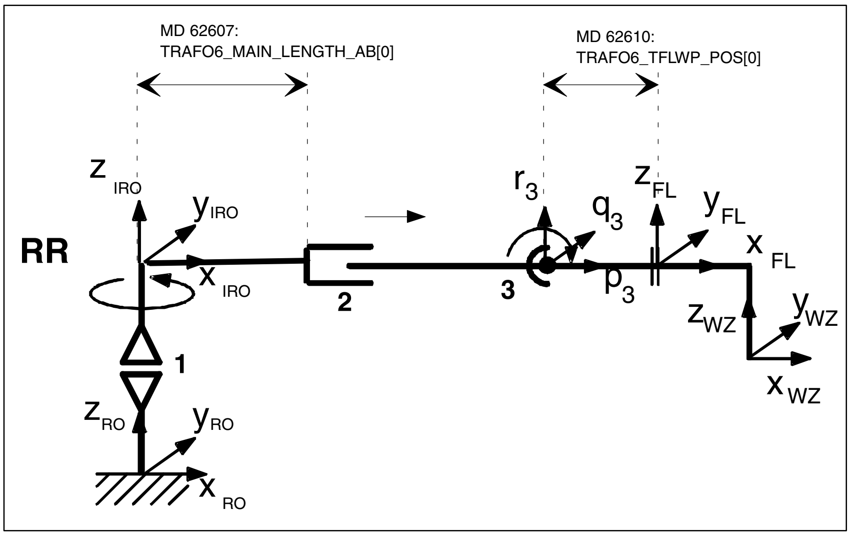


Fig. 2-16 3-axis RR kinematics

Table 2-8 Configuration data for 3-axis RR kinematics

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	[300.0, 0.0]
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 kinematics

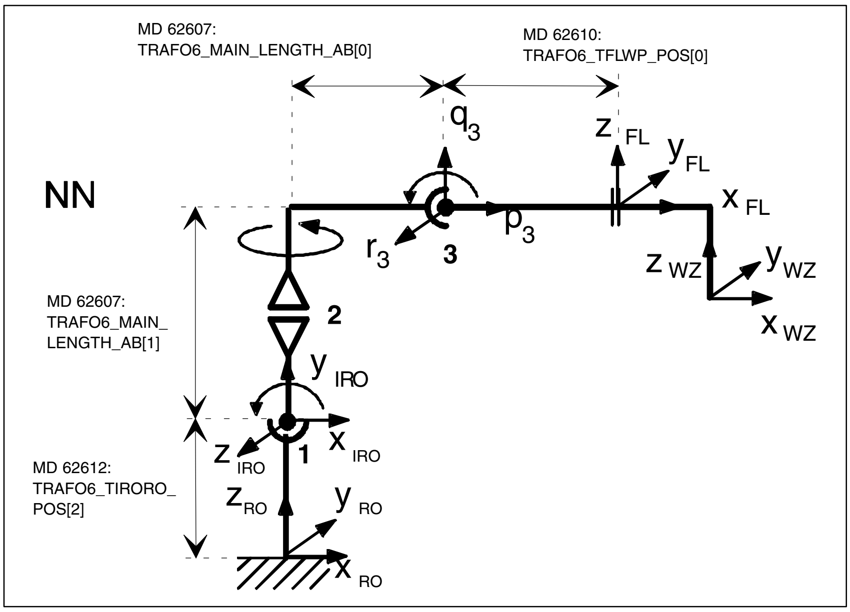


Fig. 2-17 3-axis NN kinematics

Table 2-9 Configuration data for 3-axis NN kinematics

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 to 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 the number of axes for transformation in MD 62605: TRAFO6_NUM_AXES = 4.
3. Compare the basic axes with the basic axes contained in the handling transformation package.
–> Enter the 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 wrist axis identifier (i.e. no hand in this case).
6. Enter in MD 62606: TRAFO6_A4PAR whether axis 4 runs 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 the directions of rotation of axes with the directions defined in the handling transformation package and correct in MD 62618: TRAFO6_AXES_DIR.
9. Enter the mechanical zero offset in MD 62617: TRAFO6_MAMES.
10. Enter the 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 is stored 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 is stored in MD 62610: TRAFO6_TFLWP_POS and the rotation in MD 62611: TRAFO6_TFLWP_RPY.

SCARA kinematics

4-axis CC kinematics

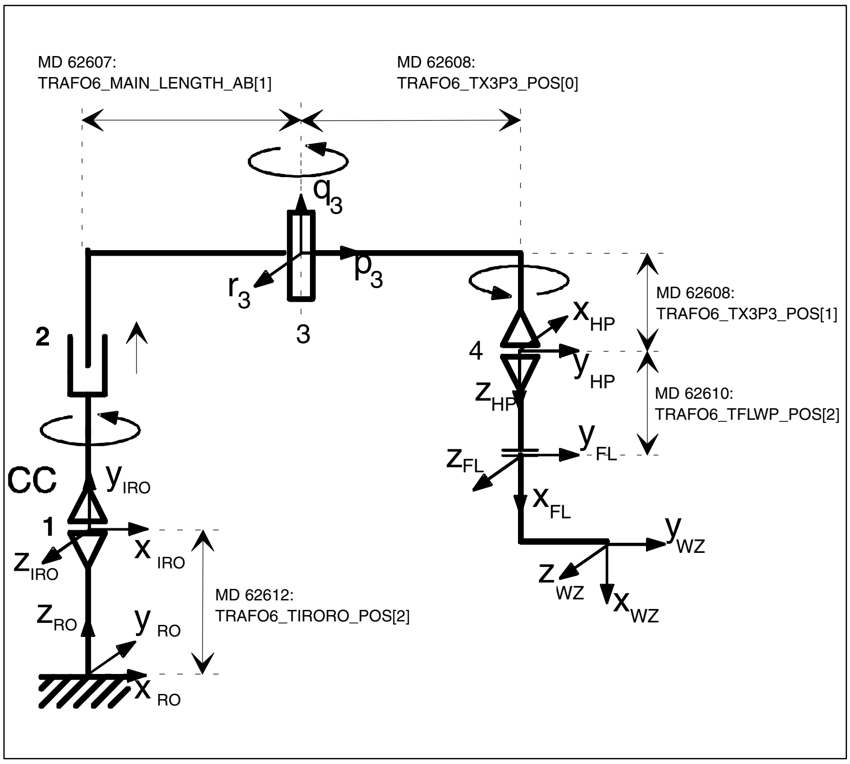


Fig. 2-18 4-axis CC kinematics

Table 2-10 Configuration data for 4-axis CC kinematics

Machine data	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, 1, 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	[0.0, 300.0]

Table 2-10 Configuration data for 4-axis CC kinematics

Machine data	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	[0.0, 0.0, 200.0]
MD 62611: TRAFO6_TFLWP_RPY	[0.0, –90.0, 0.0]

4-axis SC kinematics

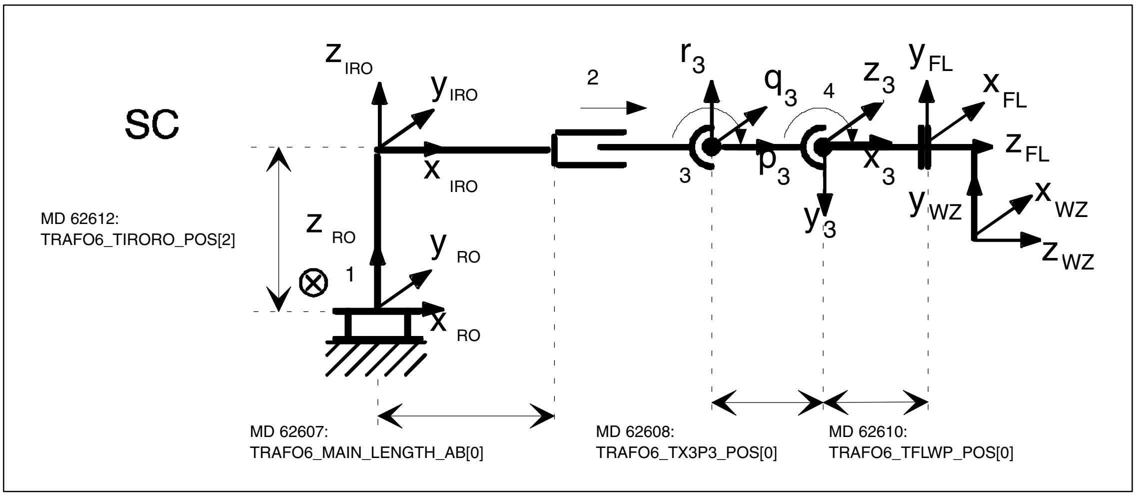


Fig. 2-19 4-axis SC kinematics

Table 2-11 Configuration data for 4-axis SC kinematics

Machine data	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, 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	[300.0, 0.0]
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 Configuration data for 4-axis SC kinematics

Machine data	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 kinematics

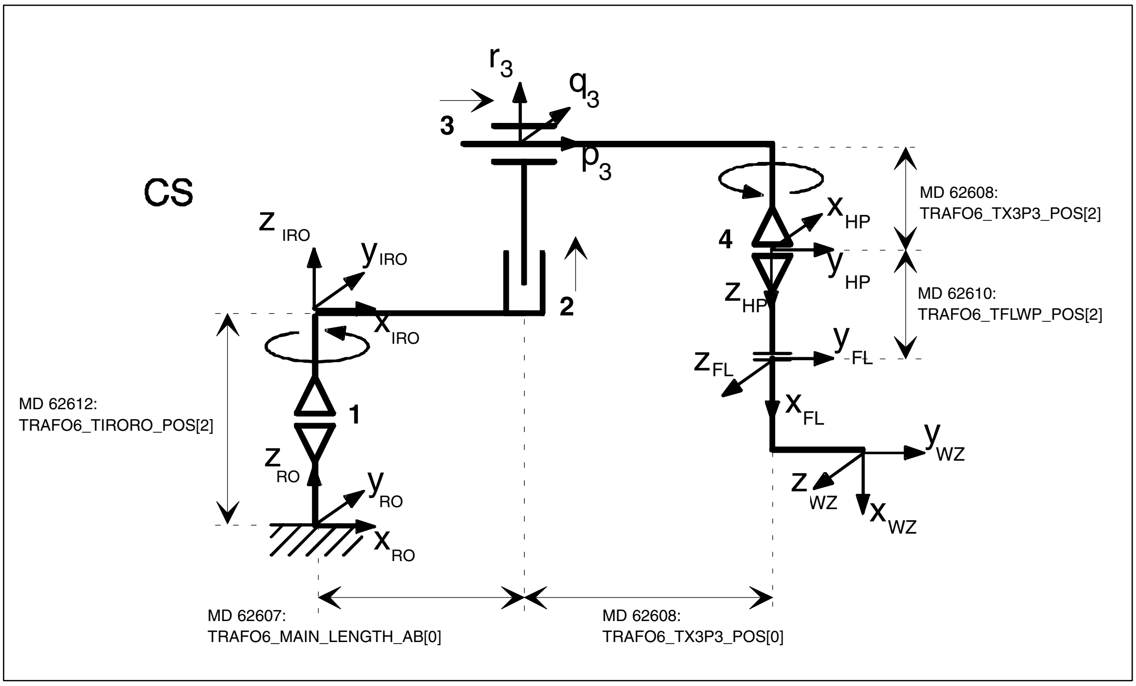


Fig. 2-20 4-axis CS kinematic

Table 2-12 Configuration data for 4-axis CS kinematics

Machine data	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 for 4-axis CS kinematics

Machine data	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

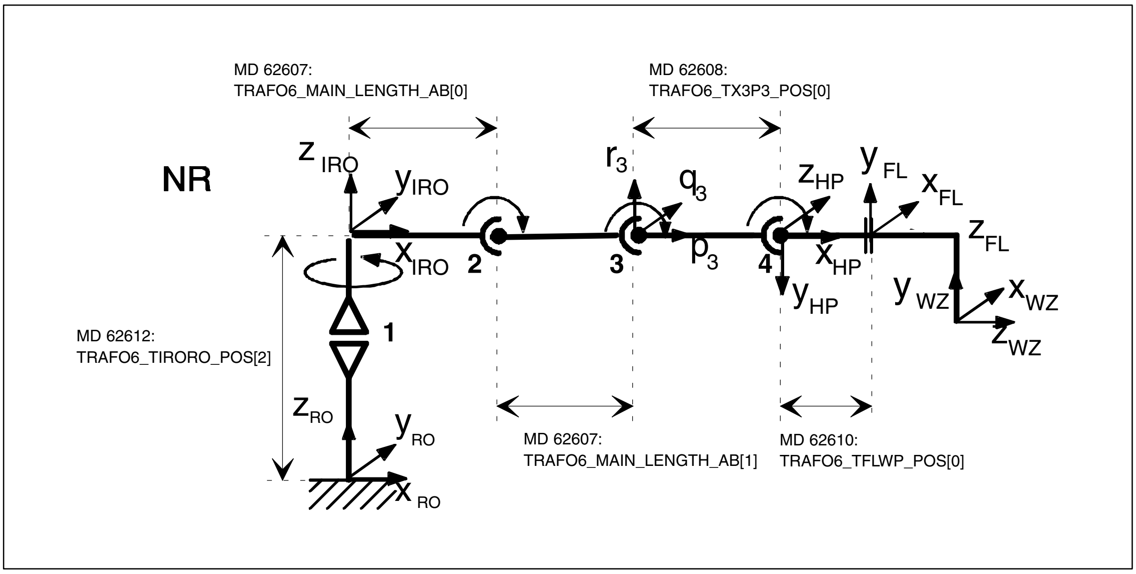


Fig. 2-21 4-axis NR kinematics

Table 2-13 Configuration data 4-axis NR kinematic

Machine data	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 data	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:
   – 4. Axis parallel to the 3rd axis: 2-dimensional tool is possible [X, 0.0, Z]
   – 4. Axis perpendicular to the 3rd axis: Only 1-dimensional tool is possible [X, 0.0, 0.0]
4. There are restrictions for the tool as far as 5-axis Scara kinematics are concerned:
   – 4. Axis perpendicular to the 3rd axis: 1-dimensional tool is possible [X, 0.0, 0.0]
5. Two successive basic axes must be parallel or orthogonal.
6. The 4th axis must only be mounted in a parallel or orthogonal way to the last basic axis.

Configuring

The procedure for configuring a 5-axis kinematic is as follows:

1. Enter “Standard” kinematic category in MD 62600: TRAFO6_KINCLASS.
2. Set the number of axes for transformation in MD 62605: TRAFO6_NUM_AXES = 5.
3. Compare the basic axes with the basic axes contained in the handling transformation package.
   –> Enter the 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 wrist axes. If axis 4 and 5 intersect, a central hand (ZEH) is present. In all other case, the ID for beveled hand with elbow (BHE) must be entered in MD 62604: TRAFO6_WRIST_AXES.
6. Enter in MD 62606: TRAFO6_A4PAR whether axis 4 runs 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 the directions of rotation of axes with the directions defined in the handling transformation package and correct in MD 62618: TRAFO6_AXES_DIR.
9. Enter the mechanical zero offset in MD 62617: TRAFO6_MAMES.
10. Enter the 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 is stored in MD 62608: TRAFO6_TX3P3_POS and the rotation in MD 62609: TRAFO6_TX3P3_RPY.
13. Specification of wrist 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 is stored in MD 62610: TRAFO6_TFLWP_POS and the rotation in MD 62611: TRAFO6_TFLWP_RPY.

SCARA kinematics

5-axis CC kinematics

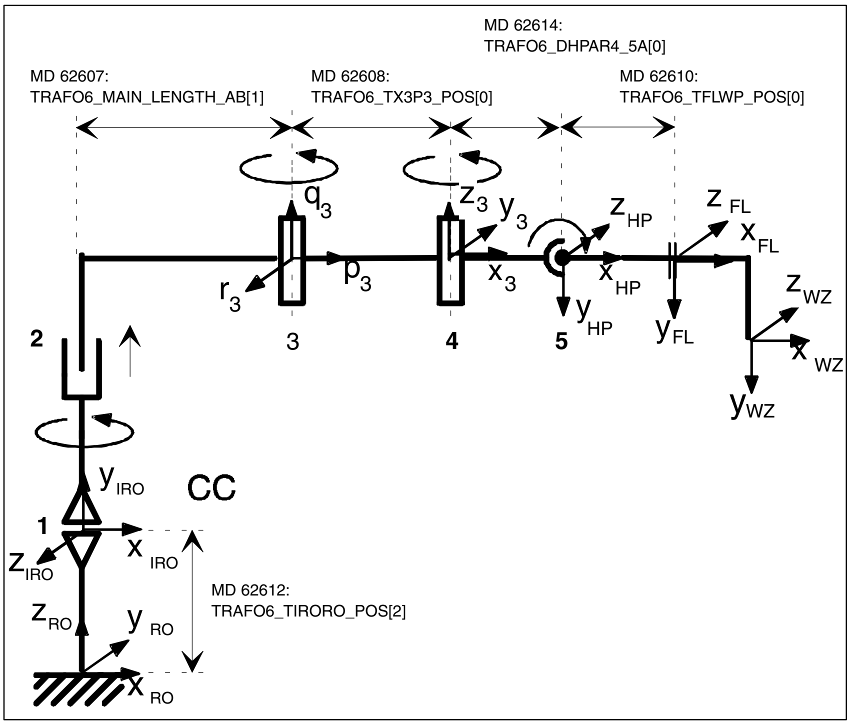


Fig. 2-22 5-axis CC kinematics

Table 2-14 Configuration data for 5-axis CC kinematics

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	[0.0, 500.0]
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**

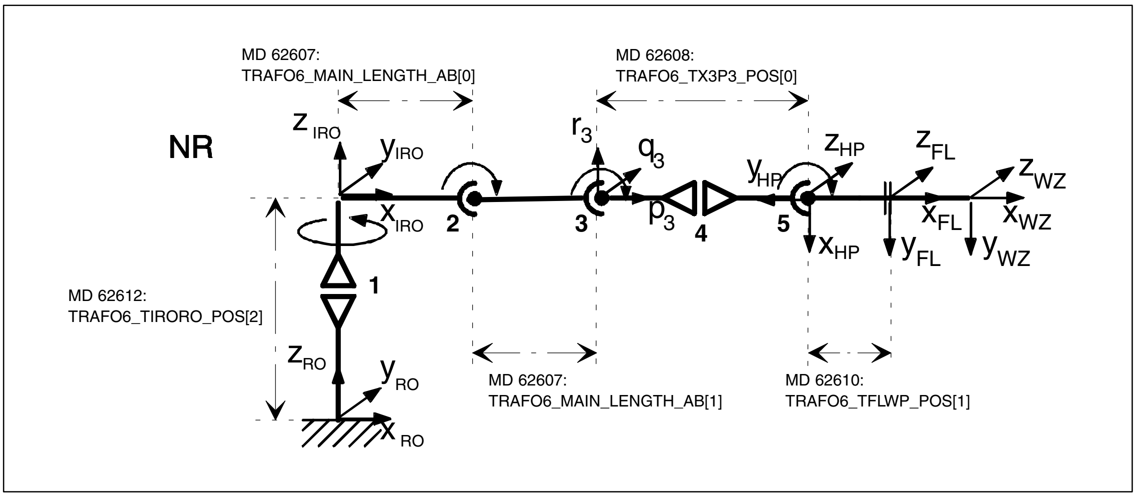


Fig. 2-23 5-axis NR kinematics

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	[500.0, 0.0, 0.0]
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 must be set for these kinematics. MD 62602: TRAFO6_SPECIAL_KIN specifies the type of special kinematic.

Special 2-axis SC kinematics

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.

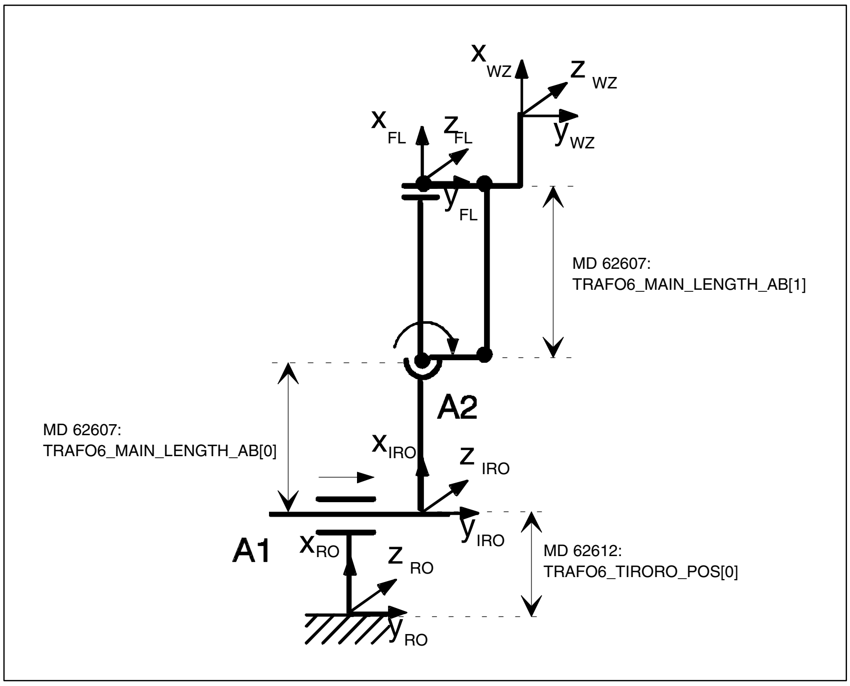


Fig. 2-24 Special 2-axis SC kinematic

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 kinematics

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.

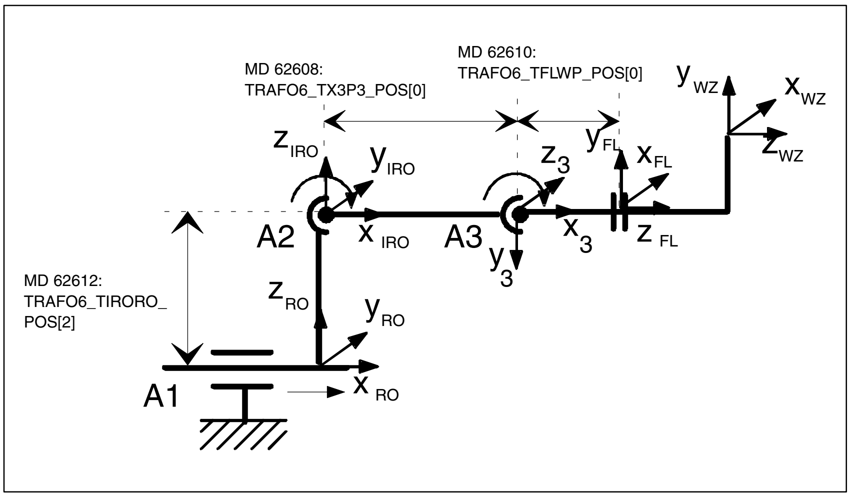


Fig. 2-25 Special 3-axis SC kinematic

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	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 4-axis SC kinematics

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.

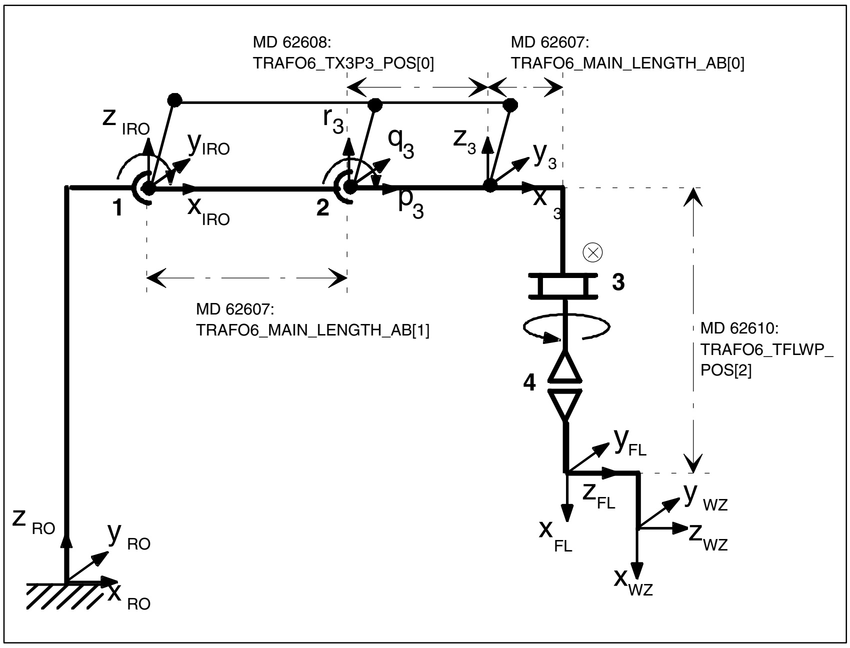


Fig. 2-26 Special 4-axis SC kinematic

Table 2-18 Configuring data for a special 4-axis SC kinematic

Machine data	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. Its identifier is MD 62602: TRAFO6_SPECIAL_KIN = 5.

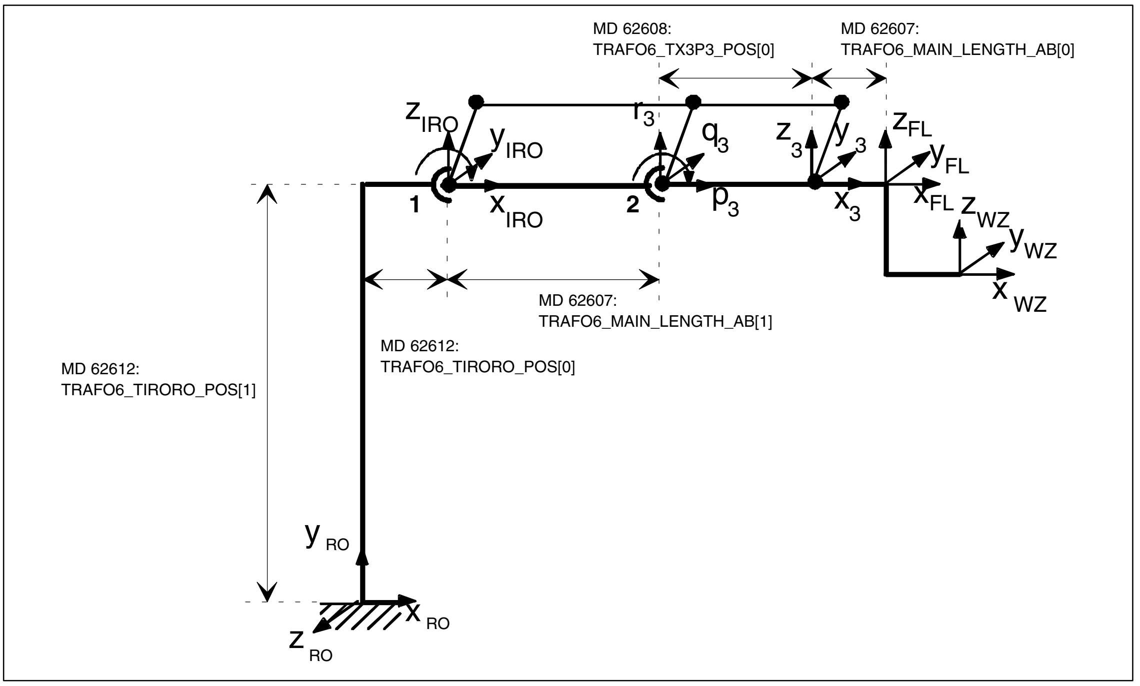


Fig. 2-27 Special 2-axis NR kinematics

Table 2-19 Configuration data for special 2-axis NR kinematics

Machine data	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

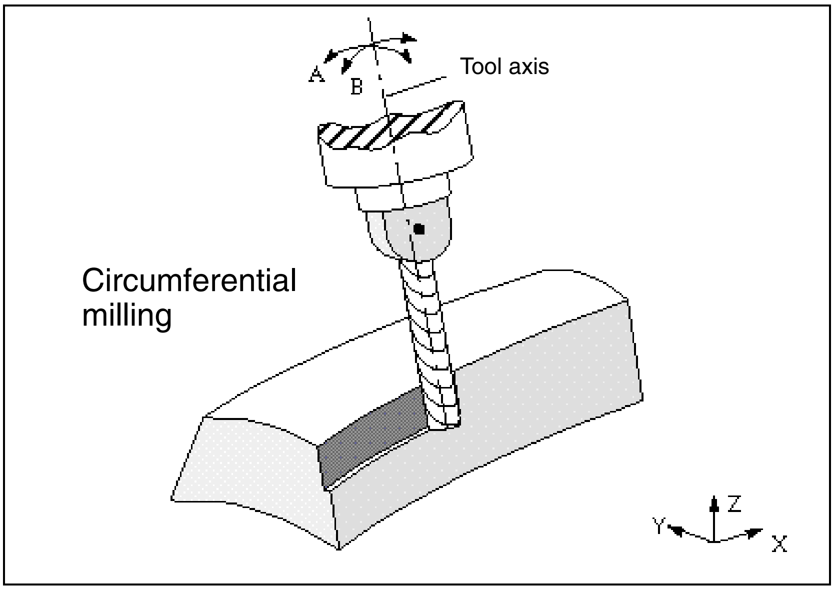


Fig. 2-28 Machining of workpieces with 5-axis transformation

Programming

Three possible methods can be used to program the orientation of the tool:

- Directly as “orientation axes” A, B and C in degrees
- Via Euler or RPY angles in degrees with A2, B2, C2
- Using direction vectors A3, B3, C3.

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

Euler or RPY

The MD 21100: ORIENTATION_IS_EULER can be used to switch between Euler and RPY entry.

!

Important

It is not possible to program using Euler angles, RPY angles or direction vectors for kinematics involving fewer than 5 axes. In such cases, only one degree of freedom is available for orientation. This orientation angle can only be programmed with “Orientation axis angle” “A”.

Orientation reference

A tool orientation at the start of a block can be transferred to the block end in the workpiece coordinate system only using the ORIWKS command.

ORIWKS command

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

ORIMKS command

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

!

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 the NC language commands ORIWKS and ORIMKS.

ORIMKS is the initial setting (SW version 2 and higher). The MD 20150: GCODE_RESET_VALUES can be used to change the initial setting:
GCODE_RESET_VALUES [24] = 1 ⇒ ORIWKS is the initial setting
GCODE_RESET_VALUES [24] = 2 ⇒ ORIMKS is the initial setting
GCODE_RESET_VALUES [24] = 3 ⇒ ORIPATH

When ORIPATH is active, the orientation is calculated from the lead and side angles relative to the path tangent and surface normal vector.

Improper tool orientation

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

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

Alarm 12130 "Illegal tool orientation" is output when Euler angles and direction 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, 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 direction vectors:

N50 A3=1 B3=0 C3=0

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

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

error message 12240 "Channel X block Y tool orientation xx defined more than once" is displayed and the NC part program stops.

2.5.1 Programming orientation for 4-axis kinematics

Tool orientation for 4-axis kinematic

4-axis kinematics possess only one degree of freedom for orientation. When the orientation is programmed using RPY angles, Euler angles or direction vectors, it is not generally possible to guarantee that the specified orientation can be approached. If used at all, this type of orientation programming is only suitable for certain types of kinematic, i.e. those which feature an invariance in orientation angles relative to the basic axes. This is the case for example for SCARA kinematics.
This is why, for kinematics with 4 axes, the orientation may only be programmed via “orientation angle” **A**. This angle corresponds to the RPY angle C according to the robotics definition, i.e. one rotation about the Z-RO axis, as illustrated in Fig. 2-29.

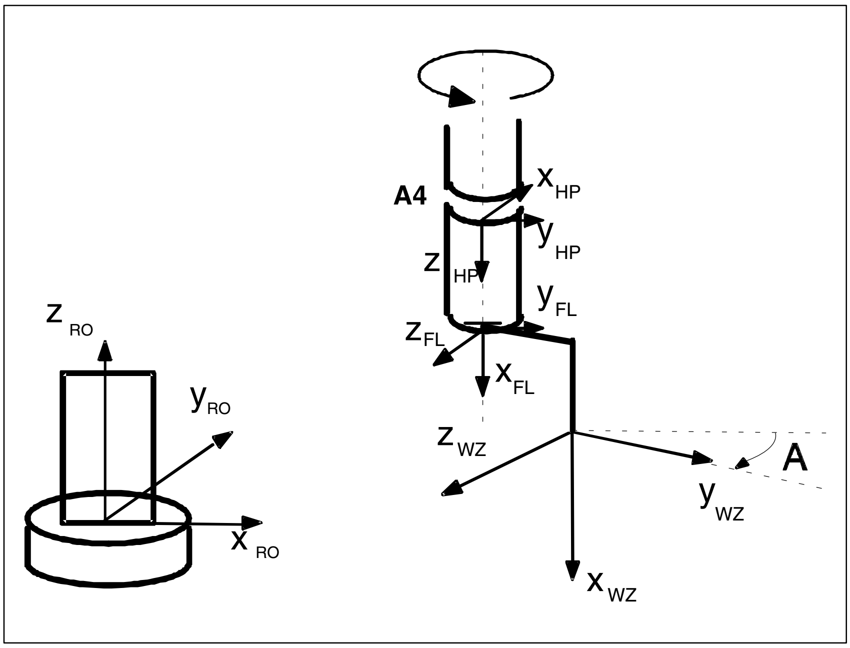


Fig. 2-29 Orientation angle for 4-axis kinematic

2.5.2 Programming orientation for 5-axis kinematics

Tool orientation for 5-axis kinematics

For 5-axis kinematics, when programming via orientation vector, it is assumed that the orientation vector corresponds to the x component of the tool.

When programming via orientation angle (RPY angle according to robotics definition), the x component of the tool is considered as the initial point for rotations.

For this purpose, the vector in the x tool direction, as shown in Fig. 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.

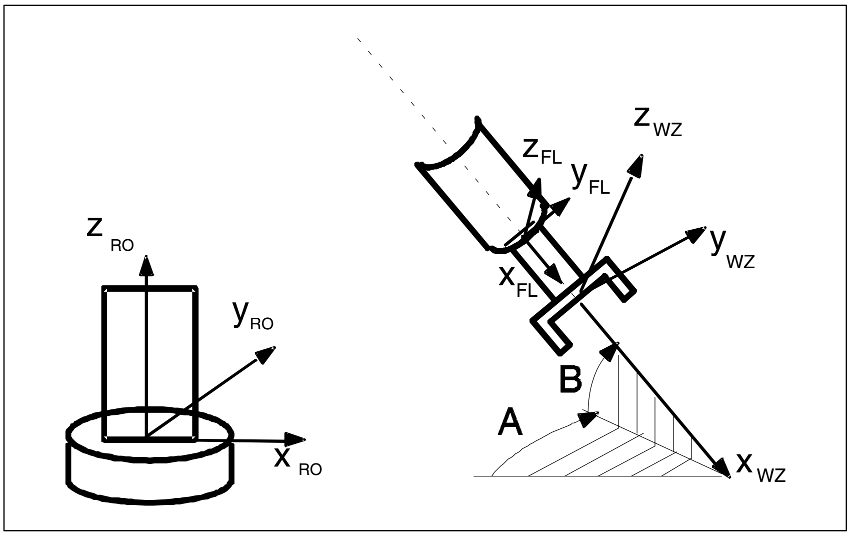


Fig. 2-30 Orientation angle for 5-axis kinematic

In SW 5.3 and higher, it is possible to define orientation axes for the handling transformation package.

!

Important

For more information, see the Description of Functions, Special Functions F2 (Part 3), Section 2.6 Orientation Axes and the “Programming Guide Advanced”, Subsection 7.1.4, “Orientation Axes”.

2.6 Singular positions and how they are handled

The calculation of the machine axes to a preset position, i.e. position with orientation, is not always clear. Depending on the machine kinematic, there may be positions with an infinite number of solutions. These positions are called “singular”.

Singular positions

- A singular position is, for example, characterized by the fact that the **fifth axis is positioned at 0**°. In this case, the singular position does not depend on a specified orientation. In this position, the fourth axis is not specified, i.e. 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 **180**°. These positions are called **leveling/diffraction singularity**.
- Another singular position exists for articulated arm kinematics if the hand point is above the rotary axis of axis 1. This position is called **overhead singularity**.

Extreme velocity increase

If the path runs in the proximity of a pole (singularity), one axis or several axes may traverse at a very high velocity. In this case, alarm 10910 “Extreme axis velocity increase” is triggered.

Behavior at pole

The unwanted behavior of fast compensating movements can be improved by reducing the velocity in the proximity of a pole. Traveling through the pole with active transformation is usually not possible.

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 more information, see
References: /PGA/, Programming Guide Advanced,
Chapter "5-Axis Machining"

Deactivation

The currently active transformation is deactivated by means of TRAFOOF or TRAFOOF().

!

Important

When deactivating the "handling transformation package" transformation, a preprocessing stop and a preprocessing synchronization are 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/EOP

The control behavior in terms of transformation following 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 Defines the initial setting for the active transformation following end of part program or RESET in accordance with the value set in MD 20140: TRAFO_RESET_VALUE with the following meaning:

- 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 part program.

2.8 Actual-value display

MCS machine coordinate system

The machine axes are displayed in mm/inch and/or degrees in MCS display mode.

WCS workpiece coordinate system

If the transformation is active, the tool tip (TCP) is specified in mm/inch and the orientation by the RPY angles A, B and C in WCS display mode. The tool 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 not active, the axes are displayed with the channel axis identifiers. If not, 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. For a kinematic as illustrated in Fig. 2–23, only a 1-dimensional tool with length in the x direction is permitted.
The direction of the tool depends on the initial setting of the machine, which is specified with G codes G17, G18 and G19. The tool lengths refer to the zero position specified by G17. This zero position should not be modified in the program.

Example

An example of a 2-dimensional tool mounted on a 5-axis Scara is described below (see Fig. 2.22). Type 100 (cutting tool) is specified as the 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
```

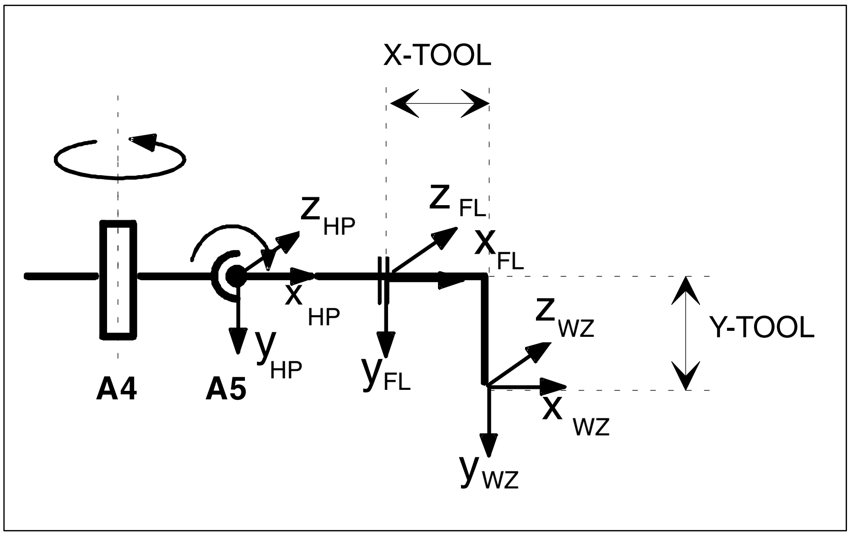


Fig. 2-31 Tool length programming

2.10 Cartesian PTP travel with handling transformation package

As from software package 5.3 it is possible to use the function Cartesian PTP travel with the handling transformation package. For this purpose, MD 24100: TRAFO_TYPE_1 must be set to **4100**.

!

Important

For more information, see the Description of Functions Special Functions F2 (Part 3) Section 2.7 “Cartesian PTP Travel” and the “Programming Guide Advanced”, Subsection 7.1.5, Cartesian PTP Travel.

■

3 Supplementary Conditions

3.1 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”

3.2 Functional restrictions

NCU 572.2

The handling transformation package can be utilized on NCU 572.2 hardware only on condition that is has been specifically enabled for the customer.

Clearance control

The handling transformation package cannot be operated together with the technology function: “clearance control”, as generally the three basic axes are not arranged perpendicular to one another.

Travel to fixed stop

The handling transformation package cannot be operated in conjunction with the “travel to fixed stop” function.

Several transformations

The handling transformation package can only be activated once per channel.

Tool programming

Tools can only be parameterized by specifying tool lengths. It is not possible to program an orientation for the tool.

Programming of the orientation

The programming possibilities of the orientation depend on the number of axes available on the machine:

Number < 5:

- Orientation axis angle

Number 5:

- Orientation axis angle
- Orientation vector

Singularities

A pole cannot be crossed when a transformation is active. Singular positions can cause axis overloads. The feedrate is not automatically adjusted. The user must reduce the feedrate appropriately at the relevant points.

■

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 Channelspecific machine data

62600 MD number	**TRAFO6_KINCLASS** Kinematic category	
Default setting: 1	Minimum input limit: 1	Maximum input limit: 2
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: DWORD		
Meaning:	The following kinematic categories can be specified: • Standard transformation: 1 • Special transformation: 2	
Restriction:	See Section 2.1	

62601 MD number	**TRAFO6_AXES_TYPE[n]** Axis type for transformation [axis no.]: 0...5	
Default setting: 1, 1, 1, 3, 3, 3	Minimum input limit: 1	Maximum 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. The following types of axis can be specified: • Linear axis: 1 • Rotary axis: 3 (4)	
Restriction:	See Subsection 2.3.2	

62602 MD number	**TRAFO6_SPECIAL_KIN** Special kinematic type	
Default setting: 1	Minimum input limit: –	Maximum input limit: –
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: DOUBLE		
Meaning:	This machine data defines the type of special kinematics. The following types of special kinematics are available: • 5-axis articulated arm with coupling between axis 2 and axis 3: 1 • 2-axis SCARA with mechanical coupling to tool: 3 • 3-axis SCARA with degrees of freedom X, Y, A: 4 • 2-axis articulated arm with coupling between axis 1 and axis 2: 5 • 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 MD number	TRAFO6_MAIN_AXES Basic axis identifier	
Default setting: 1	Minimum input limit: 1	Maximum 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 basic axes are generally the first 3 axes. The following basic axis arrangements are provided: • SS (Portal): 1 • CC (SCARA): 2 • NR (articulated arm): 3 • SC (SCARA): 4 • RR (articulated arm): 5 • CS (SCARA): 6 • NN (articulated arm): 7	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62604 MD number	TRAFO6_WRIST_AXES Wrist axis identifier	
Default setting: 1	Minimum input limit: 1	Maximum 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. The following types of hand are provided: • No hand: 1 • Central hand: 2 • Beveled hand with elbow: 6	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62605 MD number	TRAFO6_NUM_AXES Number of transformed axes	
Default setting: 3	Minimum input limit: 2	Maximum 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 included in the transformation. Packages 2.3 (810D) and 4.3 (840D) support kinematics with a maximum of 5 axes.	
Restriction:	See Subsection 2.3.2	

62606 MD number	**TRAFO6_A4PAR** Axis 4 is parallel/anti-parallel to last basic axis	
Default setting: 0	Minimum input limit: 0	Maximum 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. This machine data is only relevant for kinematics with more than 3 axes. • Axis 4 is parallel/anti-parallel: 1 • Axis 4 is not parallel: 0	
Restriction:	See Subsection 2.3.2	

62607 MD number	**TRAFO6_MAIN_LENGTH_AB[n]** Basic axis lengths A and B, n = 0...1	
Default setting: 0.0, 0.0	Minimum input limit: –	Maximum 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 defined specifically for each basic axis type. • n = 0: Basic axis length A • n = 1: Basic axis length B	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62608 MD number	**TRAFO6_TX3P3_POS[n]** Attachment of hand [position component], n = 0...2	
Default value: 0.0, 0.0, 0.0	Minimum input limit: –	Maximum 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. • Index 0: x component • Index 1: y component • Index 2: z component	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62609 MD number	TRAFO6_TX3P3_RPY[n] Attachment of hand [rotation component], n = 0...2	
Default value: 0.0, 0.0, 0.0	Minimum input limit: –	Maximum 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. • Index 0: Rotation through RPY angle A • Index 1: Rotation through RPY angle B • Index 2: Rotation through RPY angle C	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62610 MD number	TRAFO6_TFLWP_POS[n] Frame between wrist point and flange coordinate system (position component), n = 0...2	
Default value: 0.0, 0.0, 0.0	Minimum input limit: –	Maximum 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: • Index 0: x component • Index 1: y component • Index 2: z component	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62611 MD number	TRAFO6_TFLWP_RPY[n] Frame between wrist point and flange coordinate system (rotation component), n = 0...2	
Default value: 0.0, 0.0, 0.0	Minimum input limit: –	Maximum 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: • Index 0: Rotation through RPY angle A • Index 1: Rotation through RPY angle B • Index 2: Rotation through RPY angle C	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62612 MD number	TRAFO6_TIRORO_POS[n] Frame between base center point and internal coordinate system (position component), n = 0...2	
Default value: 0.0, 0.0, 0.0	Minimum input limit: –	Maximum 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: • Index 0: x component • Index 1: y component • Index 2: z component	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62613 MD number	TRAFO6_TIRORO_RPY[n] Frame between base center point and internal coordinate system (rotation component), n = 0...2	
Default value: 0.0, 0.0, 0.0	Minimum input limit: –	Maximum 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: • Index 0: Rotation through RPY angle A • Index 1: Rotation through RPY angle B • Index 2: Rotation through RPY angle C	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62614 MD number	TRAFO6_DHPAR4_5A[n] Parameter A for configuring the hand, n = 0...1	
Default setting: 0.0, 0.0	Minimum input limit: –	Maximum 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. • n = 0: Transition from axis 4 to 5 • n = 1: Transition from axis 5 to 6	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62615 MD number	**TRAFO6_DHPAR4_5D[n]** Parameter D for configuring the hand, n = 0...1	
Default setting: 0.0, 0.0	Minimum input limit: –	Maximum input limit: –
Changes effective after POWER ON	Protection level: 2 / 7	Unit: mm
Data type: DOUBLE		
Meaning:	This machine data defines the length d. • n = 0: Transition from axis 4 to 5 • n = 1: Transition from axis 5 to 6	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62616 MD number	**TRAFO6_DHPAR4_4ALPHA[n]** Parameter ALPHA for configuring the hand, n = 0...1	
Default setting: –90.0, 90.0	Minimum input limit: –	Maximum input limit: –
Changes effective after POWER ON	Protection level: 2 / 7	Unit: Degrees
Data type: DOUBLE		
Meaning:	This machine data defines the angle a. • n = 0: Transition from axis 4 to 5 • n = 1: Transition from axis 5 to 6	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62617 MD number	**TRAFO6_MAMES[n]** Offset between mathematical and mechanical zero point [axis no.]: 0...5	
Default setting: 0.0, 0.0, 0.0, 0.0, 0.0, 0.0	Minimum input limit: –	Maximum input limit: –
Changes effective after POWER ON	Protection level: 2 / 7	Unit: Degrees
Data type: DOUBLE		
Meaning:	An offset can be entered in this machine data in order to match the mechanical zero point of a rotary axis and the mathematical zero point defined by the transformation.	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62618 MD number	TRAFO6_AXES_DIR[n] Matching of physical and mathematical direction of rotation [axis no.]: 0...5	
Default setting: 1, 1, 1, 1, 1, 1	Minimum input limit: –1	Maximum 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. • +1: Direction of rotation is identical • –1: Direction of rotation is different	
Restriction:	See Subsection 2.3.2	
Figure		

62619 MD number	TRAFO6_DIS_WRP Mean distance between wrist point and singularity	
Default setting: 10.0	Minimum input limit: –	Maximum 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. **Not functional!**	
Restriction:		
Figure		

62620 MD number	TRAFO6_AXIS_SEQ Rearrangement of axes	
Default setting: 1, 2, 3, 4, 5, 6	Minimum input limit: 1	Maximum 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 in order to convert a kinematic to a standard kinematic.	
Restriction:	See Subsection 2.3.2	
Figure	See Subsection 2.3.2	

62621 MD number	TRAFO6_SPIN_ON Configuration includes triangular or trapezoidal spindles	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 1
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: DWORD		
Meaning:	This machine data defines whether the configuration includes triangular spindles or trapezoidal connections. • 0: Not included • 1: Included This function is currently not supported. **$MC_TRAFO6_SPIN_ON must be set to** 0**.** **Machine data** 62622 **to** 62628 **are therefore not functional!**	
Restriction:		
Figure		

62622 MD number	TRAFO6_SPIND_AXIS[n] Axis controlled by triangular spindle, n = 0...2	
Default setting: 0, 0, 0	Minimum input limit: 0	Maximum input limit: 5
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: DWORD		
Meaning:	This machine data defines which axis is controlled by a triangular spindle. The configuration can include a maximum of 3 triangular spindles. • n = 0: 1. triangular spindle • n = 1: 2. triangular spindle • n = 2: 3. triangular spindle	
Restriction:		
Figure		

62623 MD number	TRAFO6_SPINDLE_RAD_G[n] Radius G for triangular spindle, n = 0...2	
Default value: 0.0, 0.0, 0.0	Minimum input limit: –	Maximum 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 MD number	TRAFO6_SPINDLE_RAD_H[n] Radius H for triangular spindle, n = 0...2	
Default value: 0.0, 0.0, 0.0	Minimum input limit:	Maximum 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 MD number	TRAFO6_SPINDLE_SIGN[n] Sign for triangular spindle, n = 0...2	
Default setting: 1, 1, 1	Minimum input limit: –1	Maximum 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 MD number	TRAFO6_SPINDLE_BETA[n] Angular offset for triangular spindle, n = 0...2	
Default value: 0.0, 0.0, 0.0	Minimum input limit: –	Maximum 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 MD number	TRAFO6_TRP_SPIND_AXIS[n] Axes driven via trapezoidal spindle, n = 0...1	
Default setting: 0, 0	Minimum input limit: 0	Maximum input limit: 5
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: DWORD		
Meaning:	This machine data defines which axes are driven via a trapezoidal connection. • n = 0: Axes driven via trapezoidal • n = 1: Coupling axis	
Restriction:		
Figure		

62628 MD number	TRAFO6_TRP_SPIND_LEN[n] Trapezoid lengths, n = 0...3	
Default value: 0.0, 0.0, 0.0, 0.0	Minimum input limit: –	Maximum input limit: –
Changes effective after POWER ON	Protection level: 2 / 7	Unit: mm/inches
Data type: DOUBLE		
Meaning:	This machine data defines the lengths of the trapezoidal connection.	

62629 MD number	TRAFO6_VELCP[n] Cartesian velocity [no.]: 0...2	
Default value: 10000.0, 10000.0, 10000.0	Minimum input limit: –	Maximum 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. • n = 0: Velocity in x direction • n = 1: Velocity in y direction • n = 2: Velocity in z direction	
Restriction:	See Subsection 2.3.2	

62630 MD number	TRAFO6_ACCCP[n] Cartesian acceleration [no.]: 0...2	
Default value: 2.0, 2.0, 2.0	Minimum input limit: –	Maximum 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. • n = 0: Acceleration in x direction • n = 1: Acceleration in y direction • 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 value: 10.0, 10.0, 10.0	Minimum input limit: –	Maximum input limit: –
Changes effective immediately	Protection level: 2 / 7	Unit: rpm
Data type: DOUBLE		
Meaning:	This machine data can be set to specify a velocity for orientation angles for traversing blocks containing G0. • n = 0: Velocity angle A • n = 1: Velocity angle B • 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 value: 1.0, 1.0, 1.0	Minimum input limit: –	Maximum 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. • n = 0: Acceleration angle A • n = 1: Acceleration angle B • 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	Minimum input limit: –	Maximum input limit: –
Changes effective immediately	Protection level: 2 / 7	Unit: %
Data type: DOUBLE		
Meaning:	**Not functional!**	

■

Signal Descriptions 5

5.1 Channelspecific signals

DB21-DB28 **DBB232** Data Block	Number of active G function of G function group 25 (tool orientation reference) Signal(s) from channel (NCK–> PLC)
Edge evaluation:	Signal(s) updated:
Meaning 1	ORIWKS: The tool orientation is defined in the workpiece coordinate system (WCS) and is thus not dependent on the machine kinematics.
Meaning 2	ORIMKS: The tool orientation is defined in the machine coordinate system and is thus dependent on the machine kinematics. This is the default setting in 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 **DBX317.6** Data Block	PTP traversal active Signal(s) from channel (NCK–> PLC)
Edge evaluation: Yes	Signal(s) updated:
Signal state 1 (or signal transition 0 –> 1)	PTP traversal active
Signal state 0 (or signal transition 1 –> 0)	CP traversal active
Signal irrelevant for ...	No handling transformations active
Further references	FB Special Functions, F2

DB21-DB28 **DBX33.6** Data Block	Transformation active Signal(s) from channel (NCK–> PLC)
Edge evaluation: Yes	Signal(s) updated:
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	FB Special Functions, F2

DB21-DB28 **DBX29.4** Data Block	Activate PTP traversal Signal(s) to channel (PLC–> NCK)
Edge evaluation: Yes	Signal(s) updated:
Signal state 1 (or signal transition 0 –> 1)	Activate PTP traversal
Signal state 0 (or signal transition 1 –> 0)	Activate CP traversal
Signal irrelevant for ...	No handling transformations active
Further references	FB Special Functions, F2

■

Examples 6

6.1 General information about start-up

Note

In SW 6.4 and higher, 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.

MMC SW 3.5 or higher is required.
For the **810D**, an NCK Jeida card supporting “handling transformation package” kinematic transformation P2.3 and higher) must be available and for the **840D**, an NCK-OEM Jeida card (technology card 2 or higher).

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 (840D only)

As the first step in installing a compile cycle function on the **840D**, the original card inserted in the NCU must be replaced by the technology card.
This step is identical to the procedure followed for a standard upgrade to a more recent software version and likewise requires the static (battery-backed) control system memory to be erased. To avoid the consequential loss of all data in the SRAM, back up the SRAM before performing the operation. For a detailed description, please see 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, these must be saved in a separate archive file:
   Press the “Data out” softkey and select the required data under “NC active data”:
   “Measuring system compensations”,
   “Sag/angularity compensation” and
   “Quadrant error compensation”.
   Save this data by pressing the “Archive...” softkey and specifying a 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 run-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 the number “6” does not appear, an error has occurred:
– Incorrect PC card (e.g. card for NCU2 in NCU3 hardware)
– Card hardware defective

3. Copy back SRAM (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 traversal* 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:
    – Wrist 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 62617: TRAFO6_MAMES.
13. Enter the Cartesian velocities and acceleration rates.

■

Data Fields, Lists 7

7.1 Interface signals

DB number	Bit, byte	Name	Refer-ence
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 traversal	F2

7.2 NC machine data

Number	Identifier	Name	Refer-ence
General ($MN_ ...)			
10620	EULER_ANGLE_NAME_TAB[n]	Name of Euler angle	R1
19410	TRAFO_TYPE_MASK, bit 4	Option data for OEM transformation	A2
Channelspecific ($MC_ ...)			
21100	ORIENTATION_IS_EULER	Angle definition for orientation programming	F2
21110	X_AXIS_IN_OLD_X_Z_PLANE	Coord. system for automatic FRAME def.	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	Assignm. of geometry axes to channel axes	F2
62600	TRAFO6_KINCLASS	Kinematic category	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	Wrist axis identifier	Subs. 2.3.2
62605	TRAFO6_NUM_AXES	Number of transformed axes	Subs. 2.3.2
62606	TRAFO6_A4PAR	Axis 4 is 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

Detailed explanations of the alarms, which may occur, appear in

References: /DA/, Diagnostics Guide

or in the Online help.

■

SINUMERIK 840D sl/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 part 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 part program and only permits 1:1 couplings.

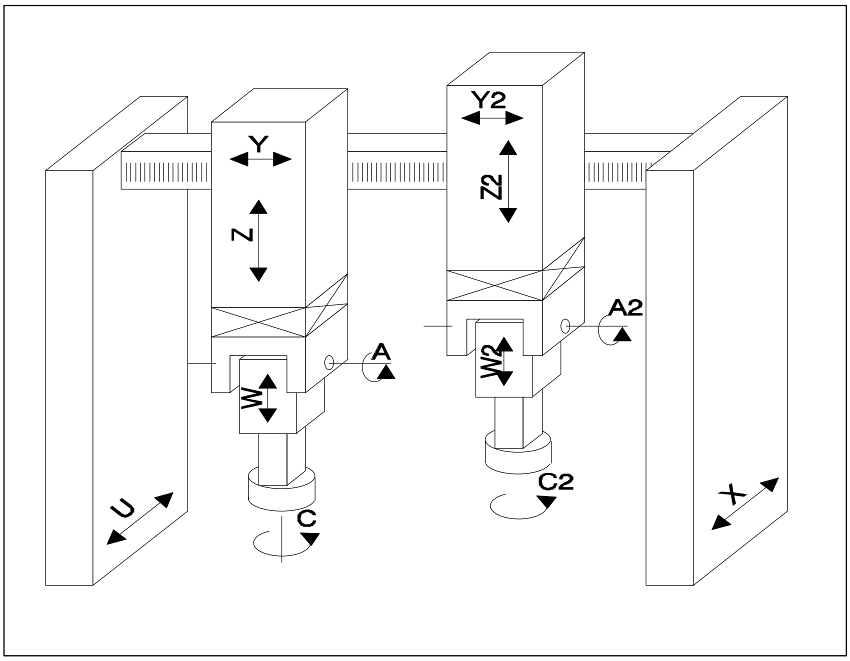


Fig. 2-1

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 part program commands.

2.2 Description of MCS coupling functions

2.2.1 Defining coupling pairs

A CC_Master axis is assigned to a CC_slave axis by means of axial machine data **MD 63540: CC_MASTER_AXIS**. Axis assignments to a coupling can only be altered when the coupling is deactivated.

A CC_Slave axis is displayed in axial VDI-Out byte **DB3x DBB97 bit0**.

Prerequisite

- 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_COPON([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

An axial machine data **MD 63541: CC_POSITION_TOL** is used to define a monitoring window. The absolute difference between the actual values of CC_Slave axis and CC_Master axis must never be greater than this value. Alarm 70010 is output if the tolerance window is violated.

The monitoring function is not active:

- If the machine data is set to 0.
- If the coupling is switched off.
- If axis/spindle inhibit is set for one of the axes.
- If an axis is in follow-up mode.
- For the 1:–1 coupling.

If the offset stored at the instant of coupling activation changes when 1:1 coupling is active, the change is indicated by NC => PLC VDI-SS **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 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 a ProtecSlave axis (PSlave) by means of 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

Axial machine data **MD 63544: CC_COLLISION_WIN** on the PSlave axis is used to indicate the minimum clearance between PSlave and PMaster. 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 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 stored 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

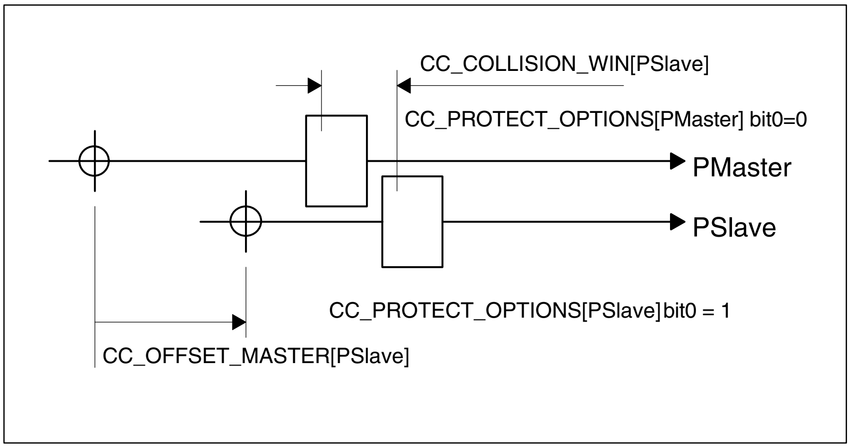


Fig. 2-2 Configuring example

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 part program: X=0.0 X2=20.0

N50 G0 X100 X2=90 ; the monitoring alarm is activated because X and X2 are interpolating together: For this reason, the acceleration rate of X2 < maximum acceleration.

Remedy:

- N50 G0 POS[X]=100 POS[X2]=90 or
- switch the monitoring function off

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.

Even if a machining head is being used in production, coupling should be active! This is essential primarily if only the second head (Y2....) is being used. “Axis/spindle inhibit” must then be set axially (PLC –> NCK) for the “parked” head.

Note

When an axis/spindle inhibit is active, a part program can be executed if this axis is not operating under position control.

Spindle functionalities

Since an MCS coupling cannot be activated for spindles, other types of 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 synchronized 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.

■

3

Supplementary Conditions

Validity

The function is configured only for the first channel.

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 the corresponding compile cycles. The compile cycles can significantly change the behavior of the control and can create hazardous situations via access to the NC.
Before a compile cycle is activated, appropriate safety precautions to prevent potential damage to machinery and personal injury must be taken (you may need to take action to safeguard against incorrect parameter settings or programming in the compile cycles).

Note

The machine data, which contain the parameters for the coupling, can only be modified when the coupling is not active.

4.2 Channel-specific machine data

28090 MD number	**NUM_CC_BLOCK_ELEMENTS** Number of block elements for compile cycles.	
Default setting: 0	Minimum input limit: 0	Maximum input limit:
Changes effective after POWER ON	Protection level:	Unit: –
Data type: DWORD		
Meaning:	Number of block elements for compile cycles. Dynamic memory is reserved. MN_NUM_CC_BLOCK_ELEMENTS = 1 (or higher)	

28100 MD number	**NUM_CC_BLOCK_USER_MEM** Total size of usable block memory for compile cycles	
Default setting: 256	Minimum input limit: 0	Maximum 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. Dynamic memory is reserved. The memory is allocated in blocks of 128 bytes.	

4.3 Axis-specific OEM machine data

63540 MD number	**CC_MASTER_AXIS** Specifies the CC_Master axis assigned to a CC_Slave axis	
Default setting: 0	Minimum input limit: 0	Maximum 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. The machine axis number is entered here. The channel-specific machine data 20070 MC_AXCONF_MACHAX_USED[n–1] and 20080 MC_CHANAX_NAME_TAB[n–1] can be used to determine the machine axis and axis name. **Notice:** CC_Master and CC_Slave must be of the same axis type (i.e. both linear or both rotary). CC_Master and CC_Slave must not be a spindle. CC_Master and CC_Slave must not be exchange axes. If the two axes have different dynamic responses, it is advisable to make the weaker of the two the CC_Master axis. The machine data may be altered only when the coupling is switched off.	

63541 MD number	**CC_POSITION_TOL** Monitoring window (valid only for CC_Slave axes)	
Default setting: 0	Minimum input limit: 0	Maximum input limit: ∞
Changes effective after RESET	Protection level:	Unit: –
Data type: DOUBLE		
Meaning:	Monitoring window (valid only for CC_Slave axes) The difference between the actual values of CC_Slave axis and CC_Master axis must never leave the monitoring window, otherwise an alarm will be generated. The following applies: d = | act [CC_Master]– (act[CC_Slave] + Offset) | d <= MD63541 Offset = Difference in position between CC_Master and CC_Slave when coupling is activated. A setting of 0 deactivates the monitoring function.	

63542 MD number	CC_PROTEC_MASTER Specifies the PMaster axis assigned to a PSlave axis	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 8
Changes effective after RESET	Protection level:	Unit: –
Data type: INT		
Meaning:	If the value (n) is greater than 0, the axis is a PSlave axis. The machine data specifies the associated PMaster axis. The channel-specific machine data MD 20070: MC_AXCONF_MACHAX_USED[n–1] and MD 20080: MC_CHANAX_NAME_TAB[n–1] can be used to define the machine axis and the axis name. **Notice:** PMaster and PSlave must be of the same axis type (i.e. both linear or both rotary).	

63543 MD number	CC_PROTEC_OPTIONS	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 7
Changes effective after RESET	Protection level:	Unit: –
Data type: INT		
Meaning:	**Bit0 – Bit3 for PMaster and PSlave** Bit0 = 1 Retract in PLUS Bit1 = 1 Factor 1.2 for maximum braking acceleration rate Bit2 = 1 Monitoring can be activated even if axis is not referenced. Bit3 = 1 Reverse retraction direction if axis is master axis **Bit4 – Bit7 for PSlave only** Bit4 = 1 Monitoring continuously active. (otherwise switch ON/OFF by PLC) Bit5 Not used Bit6 Not used Bit7 Display active protection in DBx DBX66.0.	

63544 MD number	CC_COLLISION_WIN Collision protection window	
Default setting: 1.0	Minimum input limit: –	Maximum 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 MD number	**CC_OFFSET_MASTER**	
Default setting: 0.0	Minimum input limit: –	Maximum input limit: –
Changes effective after RESET	Protection level:	Unit: –
Data type: INT		
Meaning:	Zero point offset between PSlave and PMaster. Only the value for the PSlave axis is used.	

■

Notes

Signal Descriptions 5

5.1 Axis-specific VDI OUT signals

DB31 - 61 **DBX66.0** Data Block	**Activate monitor**	
Edge evaluation: No	Signal(s) updated:	Signal(s) valid from SW: 5.1
Signal state 1	Monitoring is active. This display must be activated in MD 65543: CC_PROTECT_OPTIONS for the PSlave axis. **Note**: Conflicts may occur in connection with customer-specific compile cycles.	
Signal state 0	Monitor is not active.	

DB31 - 61 **DBX97.0** 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. The associated CC_Master axis can be found in the machine data.	
Signal state 0	Axis is not a CC_Slave axis.	

DB31 - 61 **DBX97.1** 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 example	Displayed only for the CC_Slave axis.	

DB31 - 61 **DBX97.2** 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 example	Displayed only for the CC_Slave axis.	

DB31 - 61 **DBX97.3** 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 following activation This bit is set to 1 if a particular event (SW/HW limit switch on CC_Slave axis) causes a change in the offset between CC_Master and CC_Slave stored when the coupling was activated.	
Signal state 0	No new offset since activation	
Signal irrelevant for	The bit is not set in the RESET phase.	
Application example	Displayed only for the CC_Slave axis.	
Additional references		

5.2 Axis-specific VDI IN signals

DB31 - 61 **DBX24.2** 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 example	Evaluated only on the CC_Slave axis.	

DB31 - 61 **DBX24.3** 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 example	Evaluated only on the PSlave axis.	

■

Examples 6

6.1 General start-up of a compile cycle function

Note

In SW 6.4 and higher, 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.

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 step is identical to the procedure followed for a standard upgrade to a more recent software version and likewise requires the static (battery-backed) control system memory to be erased. To avoid the consequential loss of all data in the SRAM, back up the SRAM before performing the operation.

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 the "Data out" softkey
   - Select from the "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 see 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 the number ”6” does not appear, one of the following errors may have 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...”.
- 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	Refer- ence
Axis/channelspecific($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

File: MBDDE.INI

If file: “C:\OEM\MBDDE.INI” does not exist, please create it. Open the file in the editor and insert the section: [Text Files].

In file: “C:\OEM\MBDDE.INI”, in section: [Text Files], add an entry for the alarm text files of the function:

- CZYK=C:\OEM\TF_

Alarm text files

Create the language-specific alarm text files:

- C:\OEM\TF_xx.COM
  xx = GR for German, UK for English

Alarm text file: TF_GR.COM

Enter the following alarm texts in the German alarm text file:

075050 0 0 “Channel %1, incorrect MD configuration. Error no. %2”

075051 0 0 “Channel %1 block %2 CC_COPON CC_COPOFF error no. %3”

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 axes not at standstill axis %2”

075070 0 0 “Channel %1 incorrect machine data for collision protection %2”

075071 0 0 “Channel %1 collision monitoring axis %2”

Alarm descriptions A detailed description of the individual alarms is given below:

75050 **Channel %1 incorrect MD configuration. Error no. %2**

Explanation: Incorrect configuration in MD $MA_CC_MASTER_AXIS

Error no. %2 is displayed as a decimal value of the bit-coded error mask:

$2^1 = 2$: This or the CC_Master axis is a spindle.
$2^2 = 4$: No coupling between rotary and linear axes.
$2^3 = 8$: Axes must not be exchange axes.

Reaction

Remedy: Check machine data.

Delete criterion: RESET

75051 **Channel %1 block %2 CC_COPON CC_COPOFF error code %3**

Explanation: Error in the interpretation of CC_COPON or CC_COPOFF

%3 = 1	Wrong argument programmed
%3 = 10	An axis for which a coupling has not been defined has been programmed in CC_COPON(x).
%3 = 20	Too many arguments
%3 = 100	Internal error
%3 = 200	Internal error

Reaction: Interpreter stop

Remedy: Correct part program.

Delete criterion: RESET

75060 **Channel %1 tolerance window exceeded 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 the dynamic settings of the axes involved.
Check the mechanics of the axes.

Delete 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.

Delete criterion — RESET

75062 **Channel %1 axes not in standstill axis %2**

Explanation — The CC_Master and/or CC_Slave axis(es) were not at standstill when the coupling was switched on.

%2 = 1	No valid master axis has been parameterized.
%2 = 2	When the coupling was activated, the CC_Master and/or CC_Slave axis was not stationary.
%2 = 3	When the coupling was activated, one axis was not active in this channel.

Reaction — Coupling cannot be activated.

Remedy — Input G601 for path axes or enter a STOPRE before the CC_COPON command.

Delete criterion — RESET

75070 **Channel %1 incorrect machine data for collision protection %2**

Explanation — Incorrect machine data for collision protection.

Reaction — Interpreter stop

Remedy — Correct the machine data.
Both axes must be either rotary axes or linear axes!

Delete 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

Delete criterion — RESET

Detailed explanations of all further alarms, which may occur, appear in

References: /DA/, Diagnostics Guide

or in the Online help.

■

Notes

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

Retrace Support (TE7)

■

1 Brief Description

Function

The “continue machining - retrace support” technological function (“RESU” in the remainder of this document) supports the retracing of uncompleted 2-dimensional machining processes such as laser cutting, water jet cutting, etc.

In the event of a fault during the machining process, e.g. loss of the laser, RESU can be used even by machine operators who do not have specific knowledge of the active part program to interrupt machining and travel back along the contour from the interruption point to a program continuation point necessary for machining purposes.

The machine operator initiates the retrace at the program continuation point. As part of the retrace process, an implicit block search takes place along the contour with calculation followed by repositioning on the contour and automatic retracing of the part program machining process.

The retrace option is selected and deselected in advance using part program commands within the machining program. The program continuation point can be selected at any position within the contour ranges specified in this way.

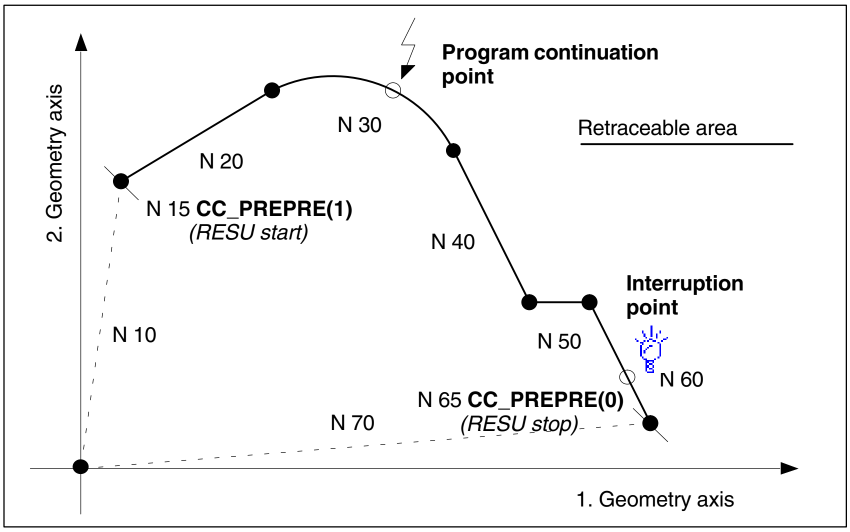


Fig. 1-1 Programmed contour with program continuation and interruption points

Precise retracing of contours is possible on all programmed contours comprising straight and circular elements. During retracing, other programmed contour elements such as splines or automatically inserted non-linear contour elements (circle, parable, etc. e.g. through tool radius compensation) are mapped as straight lines through the start and end points of the corresponding contour element, thereby preventing precise retracing of contours.

Function code

The code for the “retrace support” technological function for function-specific identifiers of program commands, machine data, etc. is:

- RESU= REtrace SUpport

Restrictions

The “retrace support” technological function is subject to the following restrictions:

- The technological function is only available in the **1st channel** of the NC.

Note

The technological function is only available in the **1st channel** of the NC.

Compile cycle

The “retrace support” technological function is a compile cycle.

System-specific availability and instructions on how to use compile cycles are described in:

(840D)

References: /FB3/ Description of Functions Special Functions
Installation of Compile Cycles TE0

(840Di)

References: /HBI/ SINUMERIK 840Di Manual
NC Installation and Start-Up With HMI Advanced,
Loadable Compile Cycles

■

Detailed Description 2

2.1 Description of functions

In order to be able to resume interrupted machining at a specific point in a part program, a block search can be carried out using the “Block search with calculation on contour” standard function. However, this requires detailed knowledge of the part program in order to be able to enter the block number of the part program block required for the block search (i.e. the number of the block the search needs to locate).

The “Continue machining - Retrace Support” technological function (referred to as RESU below) supports the continuation of the machining operation by means of an implicit block search with calculation on the contour without the machine operator requiring knowledge of the part program block required.

Continue machining might be required for example in a laser cutting application if the laser is lost during the machining operation and machining needs to resume at the point at which it was interrupted.

RESU provides the following automatic subfunctions to support program continuation:

– Function-specific reverse travel along the contour to the required program continuation point
– Automatic identification of the part program block associated with the program continuation point
– Block search with calculation on the contour for the part program block identified
– Repositioning on the contour at the program continuation point
– Continuation of part program machining

In order to be able to approach the required program continuation point exactly, it is possible to switch several times between reverse and forward travel along the contour during the continue machining process.

RESU is activated by programming the function-specific part program command CC_PREPRE(1). In the context of RESU, only the contour range between the RESU start CC_PREPRE(1) and the interruption point (NC STOP) can be retraced.

Once RESU has been launched, all part program blocks in which traversing movements are programmed are logged by RESU for possible subsequent reverse travel. Contour ranges for which continue machining is irrelevant can be removed from the log using RESU stop CC_PREPRE(**0**).

Contour ranges not logged are bridged by straight lines between the starting and end point during reverse/forward travel.

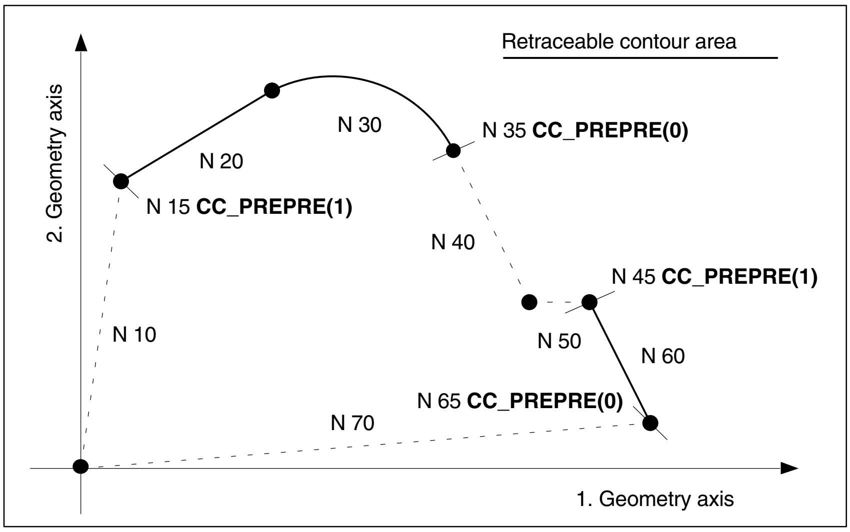


Fig. 2-1 Retraceable contour areas

Restrictions

RESU is subject to the following application restrictions:

1. Program continuation/reverse travel is only possible for part program blocks, which contain contour areas of the 1st and 2nd geometry axes.
2. RESU can only be used in the 1st NC channel.

2.1.1 Definition of terms

Interruption point

The interruption point is the point of the contour at which the traversing movement comes to a standstill following an NC STOP and reverse travel is activated.

Program continuation point

The program continuation point is the point of the contour at which reverse travel terminates and program continuation is activated.

Retraceable contour area

Retraceable contour areas comprise motion blocks from the 1st and 2nd geometry axes, which are programmed in the part program between the RESU start CC_PREPRE(**1**) and RESU stop CC_PREPRE(**0**) commands. See Fig. 2-1, page 3/TE7/2-6.

2.1.2 Functional sequence (principle)

Functional sequence

The principle sequence of the RESU function between the interruption point, program continuation point and continuation of part program processing is described below:

1. Prerequisites
   A part program with motion blocks in the 1st and 2nd geometry axis as well as the part program command for the RESU start has been started in the 1st channel.

2. Interrupt part program processing
   Part program processing/traversing movement may be interrupted any number of motion blocks after RESU start by NC STOP.

3. Select reverse travel
   Reverse travel is selected using a PLC interface signal.
   Interface signal: DB21, DBX0.1 = 1

4. Reverse travel
   NC START generates reverse travel along the contour in the first two geometry axes of the channel. Instead of the current machining program, RESU selects the automatically generated RESU main program. For more information about RESU programs, see Section 2.4, page 3/TE7/2-17.

5. End reverse travel
   Once the required program continuation point on the contour has been reached, reverse travel is ended using NC STOP.

6. Select forward travel (optional)
   Use the PLC interface signal to deselect reverse travel and activate forward travel.
   Interface signal: DB21, DBX0.1 = 0

7. Forward travel (optional)
   NC START generates forward travel along the contour in the first two geometry axes of the channel.

8. End forward travel (optional)
   Once the required program continuation point on the contour has been reached, forward travel is ended using NC STOP.

9. Retrace support
   Retrace support is activated using the PLC interface signal.
   Interface signal: DB21, DBX0.2 = 1. For retrace support, RESU automatically selects the original machining program and launches a block search with calculation as far as the program continuation point.

10. Continue part program processing
    Part program processing continues at the program continuation point in accordance with the “Block search with calculation” standard function when two NC START commands are set one after the other.

    The first NC START command processes the action blocks. Once the last action block has been reached (DB21, DBX32.7 = 1), the continuation ASUB: CC_RESU_BS_ASUP.SPF is activated.

    For more information about the ASUB, see Section 2.4, page 3/TE7/2-17.

    The second NC START command processes the approach block before part program processing is resumed.

Note

Points 3. to 8. can be repeated as often as required.

Following retrace support, a new reverse travel is possible up to a maximum of the last program continuation point. See Subsection 2.1.3, page 3/TE7/2-9.

Signal chart for interface signals

The principle sequence of the RESU function is illustrated in Fig. 2-2 as a signal chart of the interface signals involved:

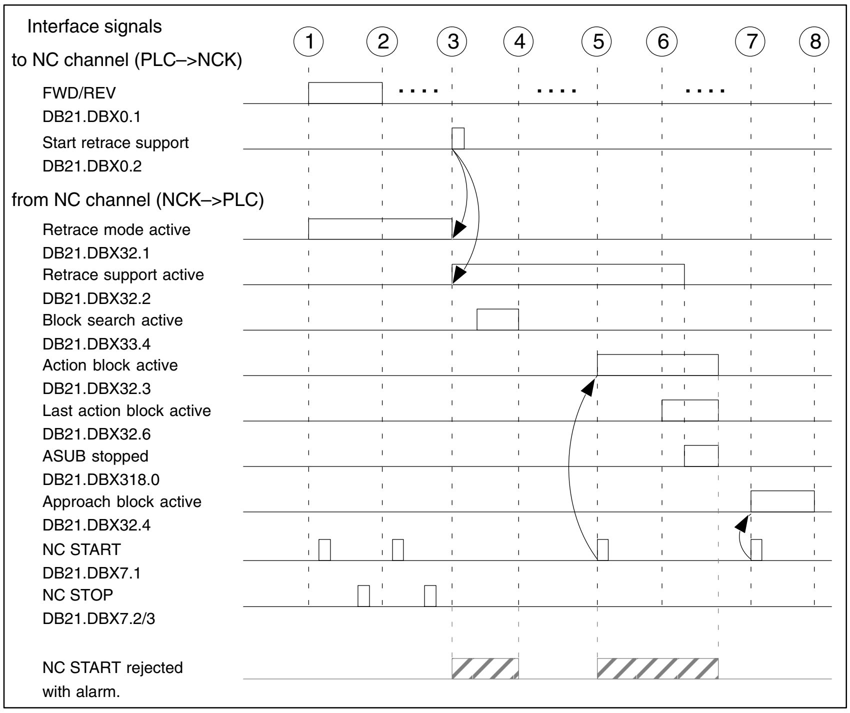


Fig. 2-2 Signal chart

Legend

1. Reverse travel is initiated
2. Forward travel is initiated (optional)
3. Retrace support is initiated (block search)
4. Search target (target block) located
5. 1st NC START –> Action blocks are output
6. Last action block is active

   When the last action block is activated, the RESU ASUB CC_RESU_BS_ASUP.SPF is launched
7. 2nd NC START –> Return travel to approach block for program continuation point
8. Part program processing (target block) resumed

2.1.3 Retraceable contour area

In the event of multiple retrace support operations within a single contour range, reverse travel along the contour is only ever possible up to the last program continuation point. On the first reverse travel following RESU start, travel as far back as the start of the contour range is possible. (See Fig. 2-3. For the purpose of simplicity, the interruption point (**U**) is identical in both cases.)

1. Reverse travel

   Before the first reverse travel, travel as far back as the start of the first contour element (N20) is possible following RESU start (N15) ($W1_{max}$). If for example reverse travel goes as far back as program continuation point W1, W1 will define the maximum RESU range for any further reverse travel following retrace support and forward travel.

2. Reverse travel

   The 2nd reverse travel can now only travel as far back as the last program continuation point $W2_{max} =$ W1. If for example reverse travel goes as far back as program continuation point W2, the maximum RESU range is restricted further.

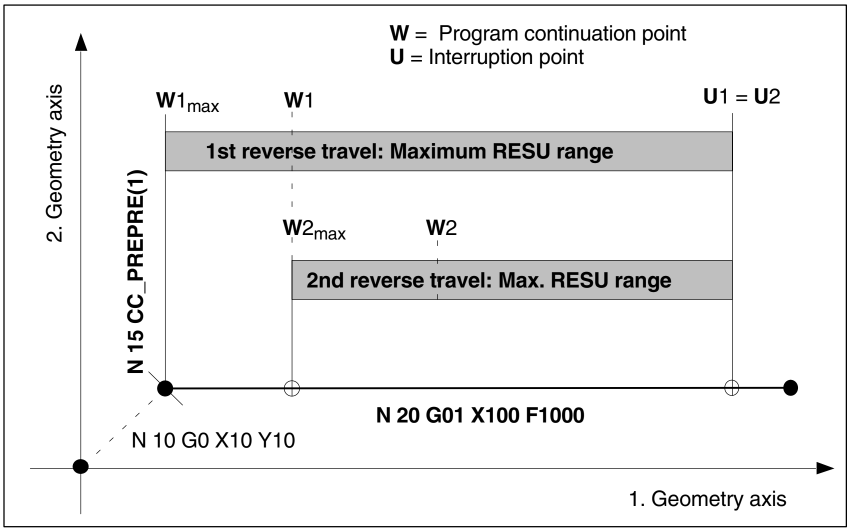


Fig. 2-3 Maximum retraceable contour area

2.2 Startup

Compile cycle

Before starting up the technological function, make sure that the corresponding compile cycle has been loaded and activated.

(840D)

References: /FB3/ Description of Functions Special Functions
Installation of Compile Cycles (TE0)

(840Di)

References: /HBI/ SINUMERIK 840Di Manual
NC Installation and Start-Up With HMI Advanced,
Loadable Compile Cycles

2.2.1 Activating the technological function

The technological function is activated via the following machine data:

- MD60900+x: CC_ACTIVE_IN_CHAN_RESU[0], Bit 0 = 1

Note

The technological function is only available in the **1st channel** of the NC.

2.2.2 Memory configuration: Block memory

Memory configuration

RESU requires additional data in the NCK-internal block memory. The following memory-configuring channel-specific machine data must be parameterized:

- MD28090: MM_NUM_CC_BLOCK_ELEMENTS (number of block elements for compile cycles (DRAM)) = x + **4** [1)]
- MD28100: MM_NUM_CC_BLOCK_USER_MEM (size of block memory for compile cycles (DRAM) in kBytes) = x + **20** [1)]

1) See Note.

Note

The values indicated must be entered in addition to the existing machine data value x.

2.2.3 Memory configuration: Heap memory

Memory requirements

RESU requires compile cycles heap memory for the following function-specific buffers:

1. Block buffer

   The larger the block buffer (see Fig. 2-5, page 3/TE7/2-17), the more part program blocks can be traversed in reverse. 32 bytes are required per part program block. The block buffer can be parameterized directly.

2. Block search buffer

   The block search buffer contains the information required for processing subroutine searches in the context of RESU. 180 bytes are required for each subroutine. The block search buffer requires at least 2160 bytes (12 subroutine calls with 180 bytes each). The block search buffer cannot be parameterized directly. The size of the block search buffer is displayed via a function-specific GUD variable.

   For information about how to create the GUD variable, see Subsection 2.6.1, page 3/TE7/2-25.

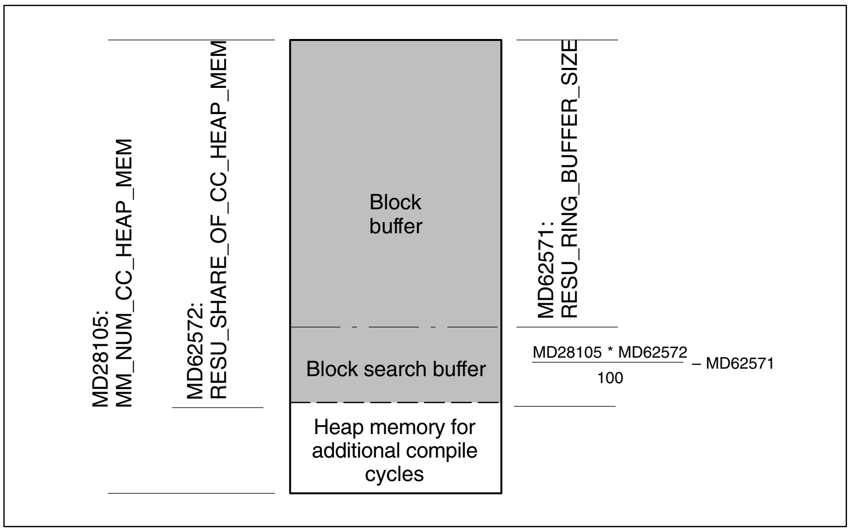


Fig. 2-4 Compile cycles heap memory allocation

Memory configuration

By default, RESU requires the following compile cycles heap memory:

- MD28105: MM_NUM_CC_HEAP_MEM (heap memory in KB for compile cycles (DRAM)) = x + **50**

  Note: The values indicated must be entered in addition to the existing machine data value x.

- MD62571: RESU_RING_BUFFER_SIZE (size of block buffer/ring buffer in part program blocks) = **1000**

- MD62572: RESU_SHARE_OF_CC_HEAP_MEM (RESU share of total heap memory) = **100**

Error messages

The block search buffer requires at least 2160 bytes (corresponding to 12 subroutine calls with 180 bytes each). Otherwise, the following alarm will be generated during NC run-up:

- Alarm “75600 Channel 1 retrace support: Incorrect MD configuration, error no. 5”

If the block search buffer is not big enough during operation, the following alarm appears:

- Alarm “75606 Channel 1 retraceable contour shortened”

2.2.4 RESU main program memory area

The following machine data can be used to set the archive for the RESU main progam CC_RESU.MPF (see Subsection 2.4.1, page 3/TE7/2-17):

- MD62574: RESU_SPECIAL_FEATURE_MASK, Bit 1 = **0**
  Archive = Dynamic memory area of NC (DRAM)
  (default)
- MD62574: RESU_SPECIAL_FEATURE_MASK, Bit 1 = **1**
  Archive = Static memory area of NC (SRAM)

RESU creates the RESU main program: CC_RESU.MPF (see Subsection 2.4.1, page 3/TE7/2-17) in the dynamic memory area of the NC (DRAM).

DRAM memory configuration

If the RESU main program is created in the dynamic memory area of the NC, the available dynamic memory area of the NC must be enlarged:

- MD18351: MM_DRAM_FILE_MEM_SIZE = x + **100**

  Note: The values indicated must be entered in addition to the existing machine data value x.

Error messages

If the RESU main program is created in the dynamic memory area of the NC (DRAM) but no DRAM memory is requested:

– MD18351: MM_DRAM_FILE_MEM_SIZE = 0

the following alarm will be generated during NC run-up:

- Alarm “75604 Channel 1 reverse travel not possible, error no. 2”

SRAM memory configuration

If the RESU main program is created in the dynamic memory area of the NC, it is retained even after a POWER OFF. However, as RESU regenerates the RESU main program every time the retrace support function is used, this parameter setting is not recommended.

2.2.5 RESU subroutines archive

The RESU-specific subroutines:

- INI program: CC_RESU_INI.SPF
- END program CC_RESU_END.SPF
- Retrace support ASUB CC_RESU_BS_ASUP.SPF
- RESU ASUB CC_RESU_BS_ASUP.SPF

can be archived as user or manufacturer cycles.

User cycles

The machine data default:

- MD62574: RESU_SPECIAL_FEATURE_MASK, bit 2 = **0**

archives the RESU-specific subroutines by default as user cycles.

Manufacturer cycles

Set machine data:

- MD62574: RESU_SPECIAL_FEATURE_MASK, bit 2 = **1**

to archive the RESU-specific subroutines as manufacturer cycles.

Series startup

The first time the NC starts up after the technological function has been activated, the RESU-specific subroutines are archived with their default content as user cycles due to the default setting in machine data MD62574: RESU_SPECIAL_FEATURE_MASK, bit 2 = 0.

If the setting is then changed to specify that the RESU-specific subroutines must be archived as manufacturer cycles, the RESU-specific subroutines already created as user cycles are retained even after a new run-up and must be deleted.

Machine data:

- MD62574: RESU_SPECIAL_FEATURE_MASK, bit 3 = **1**

can be set to support series start-up so that RESU-specific subroutines available as user cycles are deleted during NC run-up without prompting.

2.2.6 ASUB enable

The following machine data must be set for the start enable for the RESU-specific ASUB CC_RESU_ASUP.SPF while the channel is in the NC STOP state:

- MD11602: ASUP_START_MASK, bit 1 = **1**

  Note: The values indicated must be entered in addition to the existing machine data value x (x logically ORed with '**H01**').

- MD11604: ASUP_START_PRIO_LEVEL = **1**

Error messages

If an ASUB enable is not parameterized during NC STOP, the following alarm will appear during NC run-up:

- Alarm “75600 Channel 1 retrace support: Incorrect MD configuration, error no. 6”

2.2.7 PLC user program

The following functionality is necessary for the sequential coordination of the RESU function in the PLC user program:

–	IF	DB21, DBX32.2	“Retrace support active” == 1
	THEN	DB21, DBX0.1	“Forward/Reverse” = 0
		DB21, DBX0.2	“Start retrace support” = 0
–	IF	DB11, DBX0.7	“Mode group RESET” == 1
	OR	DB21, DBX7.7	“RESET” == 1
	THEN	DB21, DBX0.1	“Forward/Reverse” = 0
		DB21, DBX0.2	“Start retrace support” = 0

The following signals should be reset for safety reasons:

–	IF	DB21, DBX0.2	“Start retrace support” == 1
	THEN	DB21, DBX0.1	“Forward/Reverse” = 0
–	IF	DB21, DBX0.1	“Forward/Reverse” == 1
	THEN	DB21, DBX0.2	“Start retrace support” = 0

Programming example

The following program extract implements the changes described above:

```
U   DB21.DBX32.2   // IF     Retrace support active” == 1
R   DB21, DBX0.1   // THEN   “Forward/Reverse” = 0
R   DB21, DBX0.2   //        “Start retrace support” = 0
O   DB11, DBX0.7   // IF     Mode group RESET” == 1
O   DB21, DBX7.7   // OR     “Reset” == 1
R   DB21, DBX0.1   // THEN   “Forward/Reverse” = 0
R   DB21, DBX0.2   //        “Start retrace support” = 0
U   DB21, DBX0.2   // IF     “Start retrace support” == 1
R   DB21, DBX0.1   // THEN   “Forward/Reverse” = 0
U   DB21, DBX0.1   // IF     “Forward/Reverse” == 1
R   DB21, DBX0.2   // THEN   “Start retrace support” = 0
```

2.2.8 Function-specific alarm texts

Function-specific alarm texts must first be integrated into the appropriate HMI data management before they can be displayed. A description of how to do this appears in Section 2.7, page 3/TE7/2-27.

2.3 Programming

2.3.1 RESU start/stop/Reset (CC_PREPRE)

Syntax

CC_PREPRE(*Mode*)

Mode
- Format: Integer
- Value range: –1, 0, 1

CC_PREPRE(...) (Prepare Retrace) is a procedure call and must therefore be programmed in a separate part program block.

Functionality

The following modes are available for starting/stopping/resetting the RESU function:

- **CC_PREPRE(1)**

  Starts the logging of the motion blocks.

  The information required for reverse travel is logged on a block-specific basis in a RESU-internal block buffer. The traversing information refers to the 1st and 2nd geometry axes of the channel:

  - MD20050: AXCONF_GEOAX_ASSIGN_TAB[x]; where x = 0 and 1

  Or, if transformation is active:

  - MD24120: TRAFO_GEOAX_ASSIGN_TAB[x]; where x = 0 and 1

- **CC_PREPRE(0)**

  Stops the logging of the motion blocks.

  Can also be used to remove contour ranges that are not relevant from the log.

  Contour ranges removed in this way are bridged by a straight line between the starting and end point during reverse travel.

- **CC_PREPRE(–1)**

  Deactivates logging of the motion blocks and deletes the function-internal block buffer. Contour ranges located before the instant of deactivation of the part program are therefore no longer available for RESU.

RESET response

In the event of one of the following types of RESET:
- NCK POWER ON RESET (warm start)
- NC-RESET
- End of program (M30)

CC_PREPRE(–1) is executed implicitly.

Error messages

The following programming errors are detected and displayed with alarms:

– Invalid mode programmed:
  - RESU alarm “75601 Channel *number* Block *number* Invalid parameter for CC_PREPRE( )”
– More than one parameter programmed:
  - Alarm “12340 Channel *number* Block *number* Too many parameters”
– RESU technological function not available
  The technological function is not available. The compile cycle may not have been loaded or has not been activated:
  - Alarm “12340 Channel *number* Block *number* Name CC_PREPRE not defined or option not available”

2.4 RESU-specific part programs

RESU uses the following automatically generated part programs described in Subsection 2.1.2, page 3/TE7/2-7, which can be modified to some extent:

- Main program: CC_RESU.MPF
- INI program: CC_RESU_INI.SPF
- END program CC_RESU_END.SPF
- Retrace support ASUB CC_RESU_BS_ASUP.SPF
- RESU ASUB CC_RESU_BS_ASUP.SPF

Fig. 2-5 provides an overview of the internal structure of the technological function and the relationship between the various part programs.

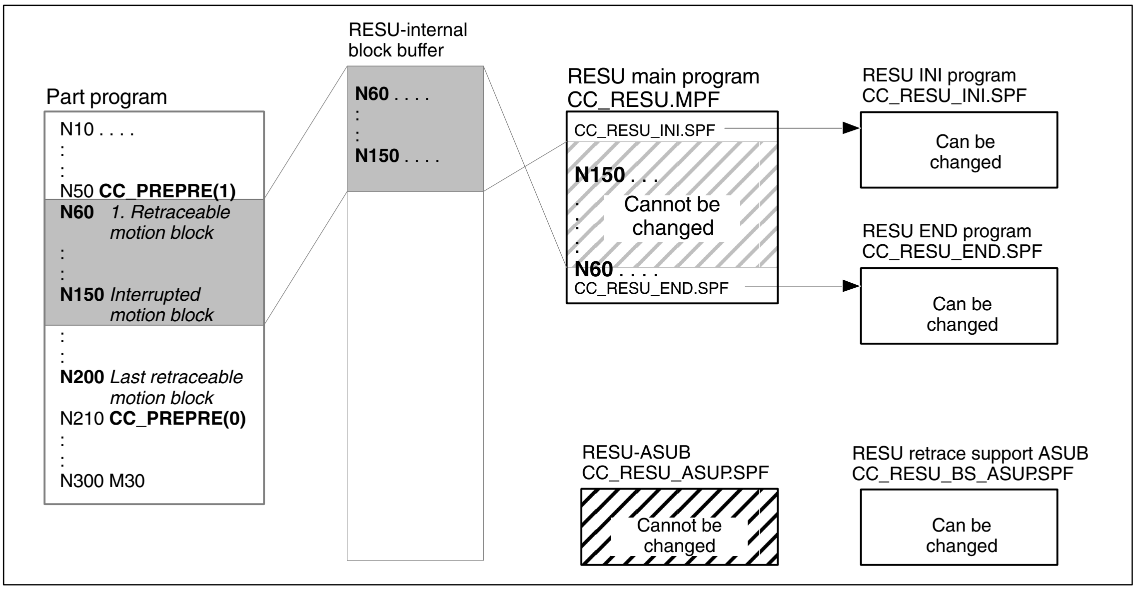


Fig. 2-5 RESU program structure

2.4.1 Main program (CC_RESU.MPF)

In addition to the calls for the RESU-specific subroutines, the RESU main program CC_RESU.MPF contains the motion blocks generated from the motion blocks logged in the block buffer for reverse/forward travel along the contour. The program is always regenerated by the RESU function if, once the part program has been interrupted, the status of the interface signal:

- DB21, DBX0.1 “Reverse/Forward”

changes.

Note

CC_RESU.MPF may <u>not</u> be changed. User-specific modifications must be made in the corresponding RESU-specific subroutine.

Error message

By default, RESU generates motion blocks for the entire retraceable contour range logged in the block buffer. If there is not enough memory space for all motion blocks to be generated in the parameterized memory area of the RESU main program (see Subsection 2.2.4, page 3/TE7/2-12), RESU reduces the number of motion blocks generated.

The missing memory and/or reduction in the number of motion blocks generated is indicated by an alarm:

- RESU alarm "75608 Channel *number* NC memory limit reached, RAM type *type*"

RESU main program in SRAM

If the RESU main program is created in the part program memory (SRAM), the following system alarm appears at the same time as the RESU alarm:

- Alarm "6500 NC memory limit reached"

Note

If the number of motion blocks generated is reduced due to insufficient memory, the entire retraceable contour can still be retraced for retrace support. To do this, proceed as follows:

1. Travel back to the end of the RESU main program.
2. Two-time change of the interface signal:
   – DB21, DBX0.1 "Reverse/Forward"

Using the current position as a new interruption point, this enables RESU to generate a new RESU main program.

Subsequently, travel is possible as far as the end of the retraceable contour range or, if the limits have changed, as far as the starting point of the last motion block that can be generated. The procedure described can be repeated as many times as required both for reverse and forward travel.

2.4.2 INI program (CC_RESU_INI.SPF)

The RESU-specific subroutine CC_RESU_INI.SPF contains the default settings required for reverse travel:

– Metric input system: G71
– Absolute dimensions: G90
– Deactivation of the configurable zero offsets/frames: G500
  See supplementary conditions Subsection 3.2.7, page 3/TE7/3-32
– Deactivation of the active tool offsets: T0
  See supplementary conditions Subsection 3.2.8, page 3/TE7/3-33
– Deactivation of the tool radius compensation G40
– Traversing velocity: F200

CC_RESU_INI.SPF has the following content by default:

```
PROC CC_RESU_INI
    G71 G90 G500 T0 G40 F200

    ;Existing system frames are deactivated

    ;Actual value and scratching
    if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H01'
    $P_SETFRAME = ctrans()
    endif

    ;External work offset
    if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H02'
    $P_EXTFRAME = ctrans()
    endif

    ;Toolholder
    if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H04'
    PAROTOF
    endif

    if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H08'
    TOROTOF
    endif

    ;Tool reference points
    if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H10'
    $P_WPFRAME = ctrans()
    endif

    ;Cycles
    if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H20'
    $P_CYCFRAME = ctrans()
    endif

    ;Transformations
    if $MC_MM_SYSTEM_FRAME_MASK B_AND 'H40'
    $P_TRAFRAME = ctrans()
    endif

    ; Bit mask for global basic frames
    $P_NCBFRMASK = 0

    ;Bit mask for channel-specific basic frames
    $P_CHBFRMASK = 0

    ;Programmable frame
    $P_PFRAME = ctrans()
M17
```

Note

CC_RESU_INI.SPF may not be changed.

CC_RESU_INI.SPF may not contain any RESU part program commands CC_PREPRE(x).

Caution

In changing the content of the RESU-specific subroutine CC_RESU_INI.SPF, the user (machine manufacturer) accepts responsibility for the correct sequence of the technological function.

2.4.3 END program (CC_RESU_END.SPF)

The RESU-specific subroutine CC_RESU_END.SPF must stop reverse travel once the end of the retraceable contour is reached. If the RESU function is parameterized appropriately, this scenario will not arise under normal circumstances.

CC_RESU_END.SPF has the following content by default:

```
PROC CC_RESU_END
    M0
M17
```

Note

CC_RESU_END.SPF may not be changed.

CC_RESU_END.SPF may <u>not</u> contain any RESU part program commands CC_PREPRE(x).

Caution

In changing the content of the RESU-specific subroutine CC_RESU_END.SPF, the user (machine manufacturer) accepts responsibility for the correct sequence of the technological function.

2.4.4 Retrace support ASUB (CC_RESU_BS_ASUP.SPF)

The RESU-specific ASUB CC_RESU_BS_ASUP.SPF causes the NC to travel to the current path point when retrace support is activated:

– Reapproach next point on path: RMN
– Approach along line on all axes: REPOSA

CC_RESU_BS_ASUP.SPF has the following content by default:

```
PROC CC_RESU_BS_ASUP SAVE
    RMN
    REPOSA
M17
```

Note

CC_RESU_BS_ASUP.SPF may be changed.

User-specific modifications must be inserted <u>before</u> the part program block RMN.

Caution

In changing the content of the RESU-specific subroutine CC_RESU_BS_ASUP.SPF, the user (machine manufacturer) accepts responsibility for the correct sequence of the technological function.

2.4.5 RESU-ASUP (CC_RESU_ASUP.SPF)

The RESU-specific ASUB CC_RESU_ASUP.SPF is required internally by the function. The ASUB is launched if the RESU interface signal: DB21, DBX0.1 (Forward/Reverse) is inverted in the NC STOP state.

CC_RESU_ASUP.SPF has the following content:

```
PROC CC_RESU_ASUP
    ; siemens system asub – do not change
    G4 F0.001
    M0
    REPOSA
M17
```

Note

CC_RESU_ASUP.SPF may not be changed.

2.5 Retrace support

Retrace support describes the entire operation, from the point at which retrace support is launched via:

– interface signal "Start retrace support" DB21, DBX0.2 = 1

to the continuation of part program processing on the programmed contour.

Requirement

In order for retrace support to function, the retrace mode, launched by means of the request for reverse travel, must be active in the channel:

– Interface signal "Retrace mode active" DB21, DBX32.1 == 1

See Fig. 2-2, page 3/TE7/2-8. Signal charts between points 1 and 3.

Subfunctions

The two essential subfunctions of retrace support are the standard NC functions:

– Block search with calculation on contour
– Repositioning on contour via shortest route (REPOS RMN)

2.5.1 Block search with calculation on contour

The block search with calculation on contour launched implicitly by the RESU function as part of retrace support serves the following purposes:

– Sets the program pointer to the part program block of the part program for repositioning using reverse/forward travel
– Calculates the axis positions on the basis of the programmed motion blocks from the start of the part program to the target block
– Collates the instructions programmed from the start of the part program to the target block, which are executed in the action block. These include:
  – Auxiliary functions
  – Tool change
  – Spindle functions
  – Feedrate programming

All part program instructions, which are not executed in the action block but are required for retrace support in the part program, must be entered manually in the RESU-specific retrace support ASUB CC_RESU_BS_ASUP.SPF, e.g.:

  – Synchronized actions
  – M functions

References

A complete description of the "Block search" function appears in:

/FB1/ Description of Functions Basic Machine
Mode Group, Channel, Program Operation (K1)
Program Test

2.5.2 Repositioning

Following the end of the last action block (last motion block before repositioning), NC START launches the output of the approach block for repositioning all channel axes programmed in the part program as far as the target block.

Geometry axes

In the approach block, the 1st and 2nd geometry axes in the channel take the shortest route along the contour to the program continuation point.

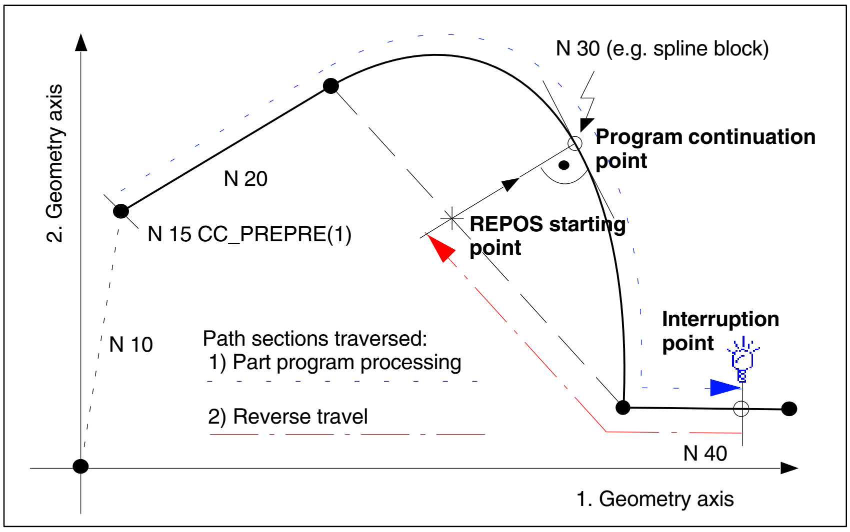


Fig. 2-6 Retraceable contour ranges and REPOS

Other channel axes

All other channel axes programmed in the part program travel to the relevant position calculated in the block search.

2.5.3 Temporal conditions affecting NC START

NC START must be initiated twice by the machine manufacturer as part of retrace support. The following conditions must be met:

1. NC START for output of action blocks

   The block search must be completed
   –> Interface signal: DB21.DBX33.4 == 0

2. NC START for output of approach block

   The RESU ASUB CC_RESU_BS_ASUP must be completed

   –> Interface signal: DB21.DBX318.0 == 1

For more information, see the signal chart for “NC START rejected with alarm” in Fig. 2-2, page 3/TE7/2-8.

2.5.4 Block search from last main block

As mentioned above, a block search with calculation on contour is carried out as part of retrace support. Even if the most powerful NCUs are used, this can lead to computing times of several minutes for very large part programs until the target block is reached. This delay time can be significantly reduced by using the block search from last main block.

Functionality

For retrace support with block search from last main block, the search for the target block takes place in 2 stages:

1. Block search without calculation from start of machining program to last main block before target block. Subroutines are ignored during this search, i.e. it takes place exclusively in the main program.
2. Block search with calculation on contour from main block to target block. This block search does not ignore subroutines.

Requirement

In order that a search from last main block can be used for retrace support, at least one main block must be programmed after the RESU start CC_PREPRE(1).

Main block

All instructions required for processing the subsequent section of the part program must be programmed in one main block.

A main block number comprising the character "**:**" and a positive integer number (block number) must be used to identify main blocks.

A complete description of how to use main blocks appears in:

References: /PG/ Programming Guide Fundamentals
NC Programming Fundamentals
Language Elements of Programming Language

Activation

The following RESU-specific machine data is used to activate the block search from last main block:

- MD62575: RESU_SPECIAL_FEATURE_MASK_2, bit 0
  - Bit 0 = 0: Retrace support with block search with calculation on contour
  - Bit 0 = 1: Retrace support with block search from last main block

Supplementary conditions

In order that a new retrace support operation can take place following a retrace support operation with block search from last main block, the RESU start CC_PREPRE(1) must be programmed in the retrace support ASUB CC_RESU_BS_ASUP.SPF:

Example:

```
PROC CC_RESU_BS_ASUP SAVE
    ;(synchronized actions, M functions, etc. required for retrace support)
    CC_PREPRE(1)
    RMN
    REPOSA
M17
```

2.6 Function-specific display data

2.6.1 Channel-specific GUD variables

RESU provides the following channel-specific GUD variable for HMI applications:

– SINUMERIK HMI Advanced

– SINUMERIK HMI Embedded

as a display data:

Table 2-1 Channel-specific GUD variables

GUD variables	Description	Unit	Access
CLC_RESU_LENGTH_BS_BUFFER	Size of block search buffer	Byte	read only

Once the technological function has been started up successfully, the GUD variables listed are not displayed automatically on the HMI interface.

HMI Advanced

Proceed as follows to create and display the GUD variables in HMI Advanced.

1. Set password
   Enter the password for protection level 1: (machine manufacturer).
2. Activate the “definitions” display
   Operating area switchover > Services > Data Selection
3. If no SGUD.DEF file is yet available:
   Operating area switchover > Services > Data admin > New...
   – Name: SGUD
   – Type: Global data/system
   Confirm with **OK**.
   This opens the file in the editor.
4. Edit the GUD variable definitions
```
DEF CHAN REAL CLC_RESU_LENGTH_BS_BUFFER
M30
```
5. Save the file and close the editor
6. Activate the SGUD.DEF file

The GUD variables for clearance control are now displayed under:

Operating area switchover > Parameters > User data > Channel user data

HMI Embedded

Proceed as follows to create and display the GUD variables in HMI Embedded.

1. Set password
   Enter the password for protection level 1: (machine manufacturer).

2. If no SGUD.DEF file is yet available:
   Operating area switchover > Program > Definit.data > New
   – Name: SGUD
   – Type: DEF
   Confirm with **OK**.
   This opens the file in the editor.
3. Edit the GUD variable definitions
```
DEF CHAN REAL CLC_RESU_LENGTH_BS_BUFFER
M30
```
4. Save the file and close the editor
5. Activate the SGUD.DEF file

The GUD variables for clearance control are now displayed under:

Operating area switchover > Parameters > User data > Channel-spec. user data

SINUMERIK NCK

The new GUD variable, which is already being displayed, will only be detected by the RESU function and supplied with an up-to-date value following an NCK POWER ON RESET.

Note

Once the GUD variables have been created, an NCK POWER ON RESET must be carried out in order for the RESU function to update the GUD variables.

2.7 Function-specific alarm texts

The RESU function supports the output of function-specific language-dependent alarm texts. The corresponding alarm texts must be created in language-specific alarm text files and declared to the HMI application.

References

A description of how to incorporate new alarms appears in:

SINUMERIK **HMI Embedded**

/IAM/ IBN HMI/MMC
IM2 Installation and Start-Up HMI Embedded
Chapter: Alarm Texts and Help Files

SINUMERIK **HMI Advanced**

/IAM/ IBN HMI/MMC
IM4 Installation and Start-Up HMI Advanced
Chapter: Alarm Texts and Help Files

German alarm texts

Recommended German alarm texts:

075600 0 0 “Kanal %1 Retrace Support: falsche MD-Konfiguration, Fehler-Nr. %2”
075601 0 0 “Kanal %1 Satz %2 Ungültiger Parameter bei CC_PREPRE( )”
075604 0 0 “Kanal %1 Rückwärtsfahren nicht möglich, Fehler-Nr. %2”
075605 0 0 “Kanal %1 Retrace Support: interner Fehler, Fehler-Nr. %2”
075606 0 0 “Kanal %1 retrace fähige Kontur wurde verkürzt”
075607 0 0 “Kanal %1 Wiederaufsetzen nicht möglich”
075608 0 0 “Kanal %1 NC-Speichergrenze erreicht, RAM-Typ %2”
075609 0 0 “Kanal %1 RESU-Achse, falsche Achskonfig., Achs-Typ %2, Satz %3”
075610 0 0 “Kanal %1 RESU, NC-START nicht möglich”

English alarm texts

Recommended English alarm texts:

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 was shortened”
075607 0 0 “Channel %1 program continuation not possible”
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”

■

Notes

Supplementary Conditions 3

3.1 Function-specific boundary conditions

3.1.1 Within subroutines

Clear retrace support within subroutines depends on whether the subroutine call is made outside or inside a program loop.

Outside

Clear retrace support is possible if a subroutine is called outside a program loop.

Inside

Clear retrace support may not be possible if a subroutine is called inside a program loop.

See Subsection 3.1.2, page 3/TE7/3-29.

Number of passes P

Subroutine repetitions using number of passes P are taken into account for retrace support. This means that retrace support is performed in the part program with the correct reference to the part program block and number of passes P to the program continuation point of the contour.

3.1.2 Within program loops

NC high-level language

In NC high-level language, program loops can be programmed using:

- LOOP ENDLOOP
- FOR ENDFOR
- WHILE ENDWHILE
- REPEAT UNTIL
- CASE/IF-ELSE-ENDIF in conjunction with GOTOB

If retrace support is performed within program loops, the retrace support is always effective in the first loop run.

If the program continuation point on the programmed contour is the result of a loop run that is not equal to the first loop run, this may under certain circumstances result in significant contour deviations during subsequent machining processes.

Warning

If the program continuation on the programmed contour is the result of a loop run that is not equal to the first loop run, this may under certain circumstances result in significant contour deviations during subsequent machining processes, posing a risk of personal injury and damage to the machine.

3.1.3 Full circles

In full circles, the block starts and ends at the same contour point. As in such cases it is impossible to make a clear distinction, retrace support on a contour point of this type is always based on the point at which the block starts. The first part program block following retrace support is then the circular block.

In order to avoid the circular block being traversed following retrace support, a contour point shortly before the end of the circular block should be selected as the program continuation point.

3.1.4 Automatically generated contour elements

The automatic generation of non-linear/circular contour elements by the NC takes place e.g. during programming of the following NC functions in the part program:

– RND
– G641/G642
– Tool radius compensation

For reverse/forward travel as part of RESU, these contour elements can be replaced by straight lines between the start and end of the block.

3.2 Supplementary conditions for standard functions

3.2.1 Axis replacement, 1st and 2nd geometry axis

As long as RESU is active, the first two geometry axes in the channel may not be transferred to another channel via axis replacement (RELEASE(x)/GET(x)).

RESU is active:

– Start: Part program command CC_PREPRE(1)
– End: Program end or part program command CC_PREPRE(–1)

3.2.2 Channel axes

Channel axes other than the 1st and 2nd geometry axes on the channel are not affected by RESU.

If traversing movements in other channel axes are required for retrace support and/or reverse travel, these can either be set manually by the machine operator or programmed as a traversing block in the RESU-specific subroutine CC_RESU_INI.SPF.

Warning

Throughout the continue machining operation in the context of the RESU technological function, the machine operator must ensure that the associated traversing movements remain free of collisions.

3.2.3 Block numbers

The RESU-specific subroutines:

– CC_RESU_INI.SPF
– CC_RESU_END.SPF

and their subroutines must not contain any block numbers.

The following alarm appears in the event of an error:

- Channel “75604 reverse travel not possible, error no. *number*”

3.2.4 Block search

Block search with calculation

RESU is subject to the following supplementary conditions in the context of the block search with calculation (on contour/at end of block) standard function:

– The last CC_PREPRE(x) RESU part program command run during the block search is effective in the target block.
– The retraceable contour range starts with the REPOS approach block.

Block search without calculation

CC_PREPRE(x) RESU part program commands are not effective during block searches without calculation.

3.2.5 Transformations

RESU can also be used for active kinematic transformation (e.g. 5-axis transformation) subject to restrictions, as the traversing movements of the first two geometry axes on the channel are recorded in the basic coordinate system (BCS) and therefore before the transformation.

Transformation change

While RESU is active, no transformation changes must take place and transformation must not be activated/deactivated.

RESU is active:

– **Start**: Part program command CC_PREPRE(1)
– **End**: Program end or part program command CC_PREPRE(–1)

References

You will find a complete description of the transformations in:

/FB2/ Description of Functions Extended Functions
Kinematic Transformation M1

/FB3/ Description of Functions Special Functions
Handling Transformation Package TE4

3.2.6 Compensation

RESU can be used in conjunction with compensations as the traversing movements of the first two geometry axes on the channel are recorded in the basic coordinate system (BCS) and therefore before the compensation.

References

You will find a complete description of the compensations in:

/FB2/ Description of Functions Extended Functions
Compensations K3

3.2.7 Frames

RESU can be used in conjunction with frames.

However, as the traversing movements of the first two geometry axes on the channel are recorded in the basic coordinate system (BCS) and therefore after the frames have been taken into account, the frame offsets must be deactivated during retrace support (reverse/forward travel).

The frame offsets are deactivated during retrace support via the standard default settings of the RESU-specific subroutine. See Subsection 2.4.2, page 3/TE7/2-18.

References

You will find a complete description of the frames in:

/FB1/ Description of Functions Basic Machine
Axes, Coordinate System, Frames,
Actual-Value System for Workpiece K2

3.2.8 Tool offsets

RESU can be used in conjunction with tool offsets.

However, as the traversing movements of the first two geometry axes on the channel are recorded in the basic coordinate system (BCS) and therefore after the frames have been taken into account, the tool offsets must be deactivated during retrace support (reverse/forward travel).

The tool offsets are deactivated during retrace support via the standard default settings of the RESU-specific subroutine CC_RESU_INI.SPF. See Subsection 2.4.2, page 3/TE7/2-18.

Contour deviations

Specific instances of tool radius compensation, e.g. compensation on outside corners G450 DISC=x, may generate contour deviations between the contour traversed during retrace support and the contour programmed in the machining program.

Contour deviations are always generated if tool radius compensation produces contour elements that are non-linear or circular. For example, G450 DISC=x, where x > 0 produces parabolic or hyperbolic contour elements.

References

You will find a complete description of tool offsets in:

/FB1/ **Description of Functions Basic Machine**
Tool Offset W1

■

Notes

Data Description (MD, SD) 4

4.1 RESU-specific machine data

62571 MD number	$MC_RESU_RING_BUFFER_SIZE Size of circular buffer (block buffer)	
Default setting: 1000	Minimum input limit: 10	Maximum input limit: 1000000
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: DWORD	Applies as of SW:	
Meaning:	The block buffer contains the geometric information for the part program. The value entered in the machine data corresponds to the number of recordable part program blocks (32 bytes/part program block). The size of the block buffer corresponds to the number of retraceable blocks.	

62572 MD number	$MC_RESU_SHARE_OF_CC_HEAP_MEM RESU share of parameterizable heap memory	
Default setting: 100.0	Minimum input limit: 1.0	Maximum input limit: 100.0
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: DOUBLE	Applies as of SW:	
Meaning:	The size of the heap memory, which is available to all active compile cycles, is parameterized using the following general machine data: • MD28105: MM_NUM_CC_HEAP_MEM The RESU machine data can be used to limit the maximum heap memory share available for RESU.	

62573 MD number	$MC_RESU_INFO_SA_VAR_INDEX[1] Indices of the synchronized action variables	
Default setting: –1	Minimum input limit: –1	Maximum input limit: 10000
Changes effective after POWER ON	Protection level: 2 / 7	Unit: –
Data type: DWORD	(840D) Applies from SW: 6.4	
Meaning:	Not used. The machine data may not be used.	

62574 MD number	**$MC_RESU_SPECIAL_FEATURE_MASK** Additional RESU features	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 0F (hex)
Changes effective after **POWER ON**	Protection level: 2 / 7	Unit: –
Data type: DWORD	Applies as of SW:	
Meaning:	**Bit 0** Not used. May not be used.	
Meaning:	**Bit 1** Bit 1 = 0: (default setting) (recommended setting) The RESU main program CC_RESU.MPF is stored in the dynamic memory area of the NC (DRAM). Bit 1 = 1: The RESU main program CC_RESU.MPF is stored in the battery-backed part program memory (SRAM).	
Meaning:	**Bit 2** Bit 2 = 0: (default) The RESU-specific subroutines: – CC_RESU_INI.SPF – CC_RESU_END.SPF – CC_RESU_BS_ASUP.SPF – CC_RESU_ASUP.SPF are stored as user cycles. Bit 2 = 1: (recommended setting) The RESU-specific subroutines (see above) are created as manufacturer cycles.	
Meaning:	**Bit 3** Bit 3 = 0: (default) No effects (see below Bit 3 = 1). Bit 3 = 1: (recommended setting if bit 2 = 1) If the RESU-specific subroutines (see above) have been created as manufacturer cycles but RESU-specific subroutines are available as user cycles during NC run-up, these are deleted without a prompt for the user.	

62575 MD number	**$MC_RESU_SPECIAL_FEATURE_MASK_2** Additional RESU features	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 01 (hex)
Changes effective after **RESET**	Protection level: 2 / 7	Unit: –
Data type: DWORD	Applies as of SW:	
Meaning:	**Bit 0** Bit 0 = 0: (default setting) (recommended setting) For the purpose of retrace support, a block search with calculation is used on the contour, starting at the beginning of the part program. Bit 0 = 1: 2 different types of block search are used in order to accelerate retrace support: – From the start of the part program to the last main block: Block search without calculation – From the last main block to the current part program block: Block search with calculation on the contour	

■

Signal Descriptions 5

5.1 Interface signals

5.1.1 Signals to channel

DB21, ... **DBX0.1** Data Block	**Reverse/Forward** Signal(s) to channel (PLC –> NCK)	
Edge evaluation: Yes	Signal(s) updated:	Also (840D) valid in: SW 5.3 and higher
Signal state 1 or signal transition 0 –> 1	Activate reverse travel. The RESU main program CC_RESU.MPF is generated from the traversing blocks recorded in the RESU-internal block buffer in order to initiate travel back along the contour on the next NC START.	
Signal transition 1 –> 0	Activate forward travel. The RESU main program CC_RESU.MPF is generated from the traversing blocks recorded in the RESU-internal block buffer in order to initiate travel forwards along the contour on the next NC START.	
Signal state 0	No meaning	
Signal irrelevant for	RESU technological function not loaded or not activated.	

DB21, ... **DBX0.2** Data Block	**Start retrace support** Signal(s) to channel (PLC –> NCK)	
Edge evaluation: No	Signal(s) updated:	Also (840D) valid in: SW 5.3 and higher
Signal state 1	Start retrace support: The original machining program is reselected and a block search is carried out to locate the program continuation point.	
Signal state 0	No meaning	
Signal irrelevant for	The NC is not in Retrace mode or RESU is not active.	

5.1.2 Signals from channel

DB21, ... **DBX32.1** Data Block	**Retrace mode active** Signal(s) from channel (NCK–>PLC)	
Edge evaluation: No	Signal(s) updated:	Also (840D) valid in: SW 5.3 and higher
Signal state 1	The “Retrace mode active” signal is active as long as the control is in Retrace mode. This is the case from initial activation of the “reverse/forward” signal until activation of the “start retrace support” signal.	
Signal state 0	The machining program is executed.	
Signal irrelevant for	RESU technological function not active.	

DB21, ... **DBX32.2** Data Block	**Retrace support active** Signal(s) from channel (NCK–>PLC)	
Edge evaluation: No	Signal(s) updated:	Also (840D) valid in: SW 5.3 and higher
Signal state 1	The “retrace support active” signal is set as soon as signal state 1 is detected for the “start retrace support” signal. The “retrace support active” signal is reset at the end of retrace support once the last action block has been completed.	
Signal state 0	Retrace support not active	
Signal irrelevant for	RESU technological function not active.	

■

6 Examples

– No examples available –

■

Notes

7 Data Fields, Lists

7.1 Machine data

7.1.1 NC-specific machine data

Number	Identifier	Name	Reference
General ($MN_ ...)			
11602	ASUP_START_MASK	Ignore stop reasons if an ASUB is running	
11604	ASUP_START_PRIO_LEVEL	Indicates the ASUB from which ASUB_START_MASK is operative.	
18351	MM_DRAM_FILE_MEM_SIZE	Size of the memory for files in the DRAM of the passive file system (in KB)	

Number	Identifier	Name	Reference
Channel-specific ($MC_ ...)			
20050	AXCONF_GEOAX_ASSIGN_TAB	Assignment geometry - channel axis	
24120	TRAFO_GEOAX_ASSIGN_TAB_1	GEO/channel axis assignment of Transformation 1	
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 in KB	
28105	MM_NUM_CC_HEAP_MEM	Heap memory in KB for compile cycle applications (DRAM)	
62571	RESU_RING_BUFFER_SIZE	Size of ring buffer (RESU-internal block buffer)	
62572	RESU_SHARE_OF_CC_HEAP_MEM	Share of total CC_HEAP_MEM	
62573	RESU_INFO_SA_VAR_INDEX	Indices of the synchronized action variables	
62574	RESU_SPECIAL_FEATURE_MASK	Additional RESU features	
62575	RESU_SPECIAL_FEATURE_MASK_2	Additional RESU features	

7.2 Alarms

7.2.1 System alarms

Detailed explanations of the alarms, which may occur, appear in

References: /DA/, Diagnostics Guide

or in the Online help.

7.2.2 RESU-specific alarms

75600	**Channel %1 retrace support: Incorrect MD configuration, error code %2**
Explanation	Errors were detected in the RESU-specific machine data during NC start-up: **Error no. = 4** The following machine data must be modified/increased: – $MC_MM_NUM_CC_BLOCK_ELEMENTS – $MC_MM_NUM_CC_BLOCK_USER_MEM **Error no. = 5** Too little compile cycle heap memory is available. The following machine data must be modified: – $MC_RESU_RING_BUFFER_SIZE – $MC_RESU_SHARE_OF_CC_HEAP_MEM – $MC_MM_NUM_CC_HEAP_MEM **Error no. = 6** The following machine data must be modified: – $MN_ASUP_START_MASK – $MN_ASUP_START_PRIO_LEVEL **Error no. = 11** The following machine data have not been parameterized: – $MC_AXCONF_GEOAX_NAME_TAB[n] – $MN_INTERMEDIATE_POINT_NAME_TAB[n] – $MN_IPO_PARAM_NAME_TAB[n] **Error no. = 13** 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.
Reaction	Alarm display, mode group not ready, motion stop, no NC START possible, alarm signal at PLC interface
Remedy	**Error no. = 4, 5, 6, 11** Correct the machine data or assign values. **Error no. = 13** 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.
Program continuation	Switch the control OFF - ON.

75601	**Channel %1 block %2 invalid argument for CC_PREPRE()**
Explanation	Only values –1, 0, 1 are valid for the parameter.
Reaction	Alarm display, interpreter stop, alarm signal at PLC interface
Remedy	Correct part program
Program continuation	Clear the alarm with the RESET key. Restart part program.

75604	**Channel %1 reverse travel not possible, error no. %2**
Explanation	Reverse travel is not possible because the following error was detected: **Error no. = 1** The current part program block contains a block number. No block numbers may be programmed in the RESU-specific subroutines CC_RESU_INI.SPF and CC_RESU_END.SPF as these are relevant internally. **Error no. = 2** cc_resu.mpf cannot be generated as no DRAM is available **Error no. = 4** The selected retrace support block contains a block number. No block numbers may be programmed in the RESU-specific subroutines CC_RESU_INI.SPF and CC_RESU_END.SPF as these are relevant internally.
Reaction	Alarm display, alarm signal at PLC interface, no NC START possible
Remedy	**Error no. = 1** Remove all block numbers from subroutines cc_resu_ini.spf and cc_resu_end.spf and their subprogams. **Error no. = 2** Assign the desired value to machine data $MN_MM_DRAM_FILE_MEM_SIZE. **Error no. = 4** Remove all block numbers from subroutines cc_resu_ini.spf and cc_resu_end.spf and their subprogams.
Program continuation	Clear the alarm with the RESET key. Restart part program.

75605	**Channel %1 retrace support: Internal error, error code %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.
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.
Program continuation	Clear the alarm with the RESET key. Restart part program.

75606	**Channel %1 retraceable contour shortened**
Explanation	The block search buffer is full. This reduces the length of the retraceable contour.
Reaction	Alarm display
Remedy	The alarm has no effect on the current part program machining operation. However, if the alarm occurs more frequently, you should increase the size of the memory enabled for the NC. Machine data: – $MC_RESU_RING_BUFFER_SIZE – $MC_RESU_SHARE_OF_CC_HEAP_MEM – $MC_MM_NUM_CC_HEAP_MEM
Program continuation	Clear the alarm with the cancel key. No further operator action required.

75607	**Channel %1 Machining cannot be continued**
Explanation	The block search has been terminated with an error. Possible causes: – Incorrect operating mode –> JOG-AUTO instead of AUTO
Reaction	Alarm display, alarm signal at PLC interface
Remedy	Switch control to AUTO mode.
Program continuation	Clear the alarm with the cancel key. Start Continue Machining again.

75608	**Channel %1 NC memory limit reached, RAM type %2**
Explanation	Insufficient memory for complete generation of RESU main program CC_RESU.MPF. RAM type = 1: The part program memory on the NC (SRAM) is full. RAM type = 2: The enabled DRAM memory on the NC is full.
Reaction	Alarm display The RESU main program CC_RESU.MPF is filled and correctly terminated, i.e. a subroutine call CC_RESU_END is inserted with part program terminator M30 at the end, as long as there is sufficient memory available. The RESU main program does not comprise the entire retraceable contour.
Remedy	This alarm has no effect on current machining operations. However, the cause should be remedied if the alarm occurs frequently: RAM type = 1: 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 part programs that are no longer needed, or reduce the size of the ring buffer. RAM type = 2: Modify the size of the DRAM part program memory or reduce the size of the ring buffer.
Program continuation	Clear the alarm with the cancel key. No further operator action required.

75609	**Channel %1 RESU axis, incorrect axis config., axis type %2, block %3**
Explanation	The 1st or 2nd geometry axis is not being traversed as a geometry axis in the part program block displayed. For example, the following program settings were made: N20 POS[X] = 10 Axis type: Value not relevant
Reaction	Alarm display, interpreter stop, no NC START possible, motion stop, alarm signal at PLC interface
Remedy	Correct part program, e.g.: 1. Do not traverse the geometry axis as a positioning axis: N20 X = 10 2. Deactivate RESU temporarily: N19 CC_PREPRE(0) N20 POS[X] = 10 ; The X axis is to be traversed as a POS ; axis, therefore do not record this block. N21 CC_PREPRE(1) N22 G1 X200 If the alarm continues to be displayed and/or a part program block is indicated in the alarm in which the axis concerned is not actually programmed, “positioning axis” may still be saved as the axis type. Remedy: N30 X=IC(0) Programming the incremental distance as 0 mm ensures that axis X reverts to a geometry axis.
Program continuation	Clear the alarm with the RESET key. Restart part program.

75610	**Channel %1 RESU, NC START not possible.**
Explanation	While RESU is active, NC START is not permitted in certain cases: • Retrace support started (DB21, DBX0.2) and block search still active ((DB21, DBX33.4) • Following the 1st “official” NC START at the end of the block search, before the last action block has been completed.
Reaction	Alarm display, alarm signal at PLC interface
Remedy	Press NC START again.
Program continuation	Wait briefly, then cancel the alarm with NC START and continue machining.

■

Notes

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

Cycle-Clock-Independent Path-Synchronous Signal Output (TE8)

■

1 Brief Description

Function

The “cycle-independent path-synchronous switching signal output” technological function is used to output a digital signal dependent on the following states within the 1st channel on the NC:

1. Rapid traverse G00: active/inactive
2. Programmed feedrate threshold: undershot/exceeded

The activation/selection of which of the two options should control the output of the signal can be programmed via a part program command.

Function code

The code for the “cycle-independent path-synchronous switching signal output” technological function for function-specific identifiers of program commands, machine data, etc. is:

- HSLC = High Speed Laser Cutting

I/Os

The on-board I/O of the relevant NC module can be used as the digital I/O via which the switching signal is output.

(840D)

The switching signal can only be output via one of the 4 on-board digital outputs on the NCU module.

(840Di)

The switching signal can only be output via one of the 4 on-board digital outputs on the MCI board extension module (option).

Availability

The “cycle-independent path-synchronous switching signal output” technological function is available for the following systems:

- SINUMERIK 840D
- SINUMERIK 840Di

Restrictions

The “cycle-independent path-synchronous switching signal output” technological function is subject to the following restrictions:

- The technological function is only available in the **1st channel** of the NC.

Note

The technological function is only available in the **1st channel** of the NC.

Compile cycle

The "cycle-independent path-synchronous switching signal output" technological function is a compile cycle.

System-specific availability and instructions on how to use compile cycles are described in:

(840D) **References:** /FB3/ Description of Functions Special Functions
Installation of Compile Cycles TE0

(840Di) **References:** /HBI/ SINUMERIK 840Di Manual
NC Installation and Start-Up With HMI Advanced,
Loadable Compile Cycles

■

Detailed Description 2

2.1 Description of functions

The description of the how the technological function works is based on the example of HSLC (High-Speed Laser Cutting).

Calculating the switching positions

During high-speed laser cutting, e.g. as used to manufacture perforated sheets, it is absolutely essential to switch the laser beam on/off exactly at the programmed setpoint positions during the machining process. In order to minimize programming overheads, the switching positions are calculated by the technological function using the velocity of the geometry axes programmed in the part program block.

The following criteria define the setpoint position programmed in the part program block (end of block position) as a switching position:

1. A GO edge change
2. Overshoot/undershoot of a freely programmable velocity threshold

G0 edge change

If G0 (rapid traverse) is active in a part program block (programmed or modal), the switching signal is deactivated. Conversely, if G0 (rapid traverse) is not active in a part program block, the switching signal is activated. The G0 edge change marks the programmed end of block position of the previous block as the switching position.

See the examples of positions X30 on G0 edge change from N10 to N20 and X100 on G0 edge change from N30 to N40 in Fig. 2-1.

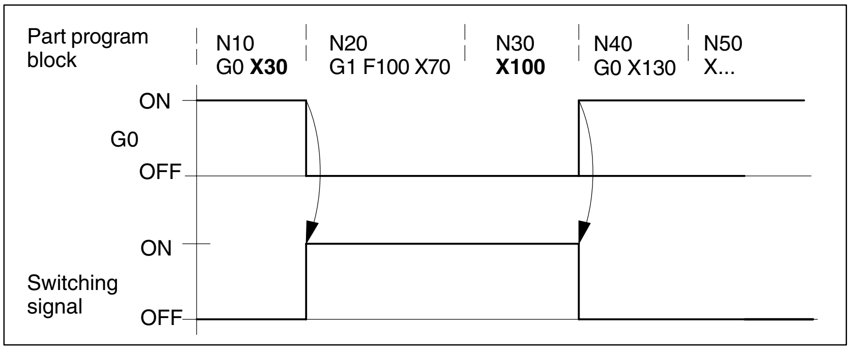


Fig. 2-1 Switching signal states and G0 edge change

Freely progr. threshold value

A freely programmable velocity threshold value is used to define the setpoint velocity programmed in the part program block at and above which the switching signal is activated/deactivated.

If the setpoint velocity programmed in the part program is higher than the programmed threshold value, the switching signal is deactivated. If the setpoint velocity is at/below the threshold value, the switching signal is activated. The edge change marks the programmed end of block position of the previous block as the switching position.

See the examples of positions X30 on edge change from N10 to N20 and X70 on edge change from N20 to N30 in Fig. 2-1.

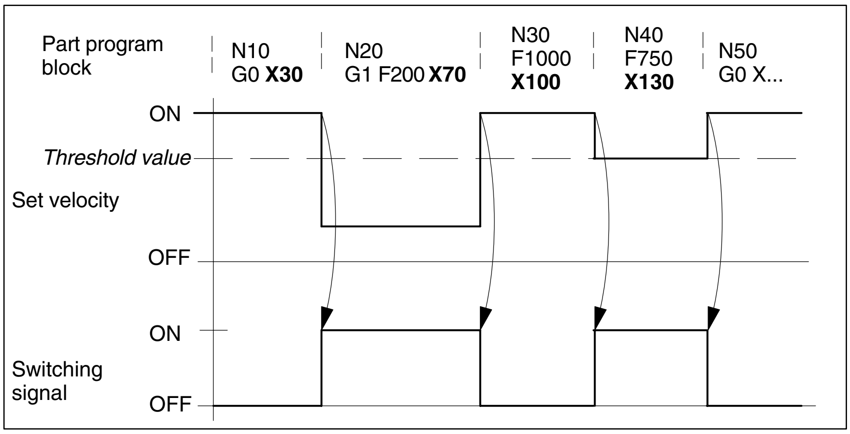


Fig. 2-2 Switching signal states with freely progr. threshold value

Note
G0 always deactivates the switching signal, regardless of the threshold value.

Calculating switching instants

In order for the switching to be as precise as possible at the switching positions calculated, the control calculates the positional difference between the actual position of the geometry axes involved and the switching difference in every position controller cycle.

If the positional difference is less than 1.5 position controller cycles, it is converted into a temporal difference taking into account the current path velocity and acceleration rate of the geometry axes.

With the temporal difference specified, a hardware timer is started, which triggers the switching signal at exactly the instant calculated in advance regardless of the position controller cycle.

Approached switching position

If a switching position is not reached exactly, e.g. in continuous-path mode and travel in more than one geometry axis, switching takes place at the instant at which the positional difference between the actual position of the geometry axes involved and the programmed switching position increases again.

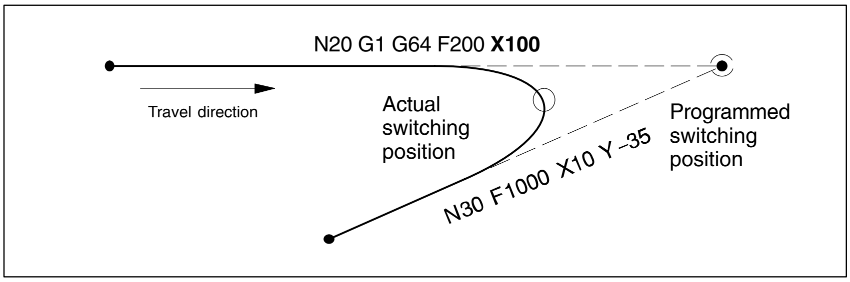


Fig. 2-3 Switching position offset for continuous-path mode

Switching position offset

A positional offset of the switching position can be programmed for both switching operations:

- Negative offset distance = Lead
  A negative offset distance offsets the switching position before the setpoint position programmed in the part program block.
  If an excessively large negative offset distance is programmed, i.e. the setpoint has already been exceeded by the time the edge is detected, the signal is switched immediately.
- Positive offset distance = Follow-up
  A positive offset distance offsets the switching position after the setpoint position programmed in the part program block.

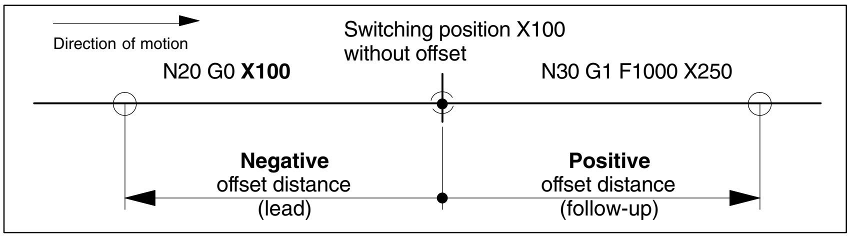


Fig. 2-4 Switching position offset

Path reference

The offset distance is a positional data, which refers to the programmed path. For the purpose of simplicity, linear motion is assumed. Curves in the path are not ignored.

Behavior with single block and G60

Due to the internal motion logic, negative offset distances (lead) have no effect when used with the following standard functions:

- Single block
- Exact stop at block end (G60)

Note

Negative offset distances (lead) have no effect when used with the “single block” and “exact stop at block end (G60)” standard functions.

Behavior in the event of an interruption in the part program

Following an interruption in the part program (NC-STOP) and subsequent change to JOG mode, the technological function is deactivated and/or switching signals cease to be output.

The technological function is only restarted and/or switching signals output once the system has reverted to AUTOMATIC mode and the part program has been resumed (NC START).

2.2 Startup

Compile cycle

Before starting up the technological function, make sure that the corresponding compile cycle has been loaded and activated.

(840D)

References: /FB3/Description of Functions Special Functions
Installation of Compile Cycles (TE0)

(840Di)

References: /HBI/SINUMERIK 840Di Manual
NC Installation and Start-Up With HMI Advanced,
Loadable Compile Cycles

2.2.1 Activating the technological function

The technological function is activated via the following machine data:

- MD60900+x: CC_ACTIVE_IN_CHAN_HSLC[0], Bit 0 = 1

Note

The technological function is only available in the **1st channel** of the NC.

2.2.2 Configuring the memory

Memory configuration

The technological function requires additional data in the NCK-internal block memory. The following memory-configuring channel-specific machine data must be parameterized:

- MD28090: MM_NUM_CC_BLOCK_ELEMENTS (number of block elements for compile cycles (DRAM)) = x + **1** 1)
- MD28100: MM_NUM_CC_BLOCK_USER_MEM (size of block memory for compile cycles (DRAM) in kBytes) = x + **10** 1)

1) See Note.

Note

The values indicated must be entered <u>in addition</u> to the existing machine data value x.

2.2.3 Parameterizing the digital on-board outputs

Digital output

A digital output from the local I/O is required for the switching signal. The following machine data must be used to parameterize at least one digital output byte:

- MD10360: FASTIO_DIG_NUM_OUTPUTS (Number of active digital output bytes) 1

(840D)

A complete description of how to parameterize a digital output on a SINUMERIK 840Di appears in:

References: /HBI/ SINUMERIK 840Di Manual
NC Installation and Start-Up with HMI Advanced,
Digital and Analog I/Os

(840D) and (840Di)

A complete description of the digital outputs appears in:

References: /FB2/ Description of Functions Extended Functions
Digital and Analog NCK I/Os (A4)

2.2.4 Parameterizing the switching signal

Output number of switching signal

Once the compile cycle has started up, the following function-specific machine data appears in the channel-specific machine data:

- MD62560: $MC_FASTON_NUM_DIG_OUTPUT
  Number of digital output of switching signal

The number of the digital output n (where n = 1, 2, ... 4) via which the switching signal is to be output must be entered in the machine data.

Features of the switching signal

In the context of the general features of digital output signals, the switching signal features are as follows:

- The number of the digital output can be changed from within the part program.
- The digital output can be disabled from within the PLC user program.
- The number of the digital output cannot be changed from within the PLC user program.

Deactivating the switching signal

Entering the number of the digital output n = 0 deactivates the function. No message or alarm is output.

Effect on other output signals

The hardware-timer-controlled output of the switching signal at the parameterized output delays the signal output for the other digital on-board outputs, e.g. due to synchronized actions, by 2 IPO cycles.

Note

The output of the switching signal delays the signal output of the other digital on-board outputs by 2 IPO cycles.

2.3 Programming

2.3.1 Activation (CC_FASTON)

Syntax

CC_FASTON(<*Parameter 1*>, <*Parameter 2*> [, <*Parameter 3*>])

CC_FASTON( ) is a procedure call and must therefore be programmed in a dedicated part program block.

Functionality

The parameters for the CC_FASTON( ) procedure have the following meaning:

- Parameter 1
  Length of the offset distance for activation of the switching signal. The parameter unit corresponds to the setting preset via machine data:
  - MD10240: $MN_SCALING_SYSTEM_IS_METRIC
- Parameter 2
  Length of the offset distance for deactivation of the switching signal. The parameter unit corresponds to the setting preset via machine data:
  - MD10240: $MN_SCALING_SYSTEM_IS_METRIC
- Parameter 3
  This parameter is optional.
  – If the parameter is not indicated in the procedure call, the G0 edge change is used as the switching criterion.
  – If the parameter is indicated in the procedure call, it contains as a switching criterion the velocity setpoint value, which, when undershot or exceeded, activates/deactivates the switching signal.

Programming example

```
DEF REAL DIFFON          = –0.08
DEF REAL DIFFOFF         = 0.08
DEF REAL FEEDTOSWITCH    = 20000

CC_FASTON( DIFFON, DIFFOFF, FEEDTOSWITCH )
```

Changing parameters

The parameters for the CC_FASTON( ) procedure can be modified at any time during the execution of the part program. To do this, enter the procedure call again with the new parameter values. The switching criterion (G0 edge change/ velocity threshold value) may also be changed.

Reset response

A RESET (NC RESET or end of program) deactivates the function.

Supplementary conditions

The following supplementary condition must be observed:

Continuous-path mode

Although the CC_FASTON( ) procedure call must be programmed in a dedicated part program block, this will not lead to a drop in velocity while continuous-path mode is active (G64, G641,...).

2.3.2 Deactivation (CC_FASTOFF)

Syntax

CC_FASTOFF( )

CC_FASTOFF( ) is a procedure call and must therefore be programmed in a dedicated part program block.

Functionality

The CC_FASTOFF( ) procedure call is used to deactivate the function.

Supplementary conditions

The following supplementary condition must be observed:

Continuous-path mode

Although the CC_FASTON( ) procedure call must be programmed in a dedicated part program block, this will not lead to a drop in velocity while continuous-path mode is active (G64, G641,...).

2.4 Function-specific alarm texts

The function supports the output of function-specific language-dependent alarm texts. The corresponding alarm texts must be created in language-specific alarm text files and declared to the HMI application.

References

A description of how to incorporate new alarms appears in:

SINUMERIK **HMI Embedded**

/IAM/ IBN HMI/MMC
IM2 Installation and Start-Up HMI Embedded
Chapter: Alarm Texts and Help Files

SINUMERIK **HMI Advanced**

/IAM/ IBN HMI/MMC
IM4 Installation and Start-Up HMI Advanced
Chapter: Alarm Texts and Help Files

German alarm texts

Recommended German alarm texts:

075500 0 0 “Kanal %1 falsche Konfiguration der Funktion: Clock-independent switching signal output”

English alarm texts

Recommended English alarm texts:

075500 0 0 “Channel %1 Clock-independent switching signal output function incorrectly configured”

■

Notes

Supplementary Conditions 3

3.1 Function-specific boundary conditions

3.1.1 Geometry axes

The switching positions can only be determined via the programmed traversing movements of the geometry axes in the 1st channel. The following alarm appears if there are no geometry axes in the 1st channel:

- Alarm: “75500 channel *channel number*, incorrect function configuration: Clock-independent switching signal output”

3.2 Supplementary conditions for standard functions

3.2.1 Block search

If a block search is carried out for a part program block designed to activate the technological function following a CC_FASTON( ) procedure call, the switching signal is activated with the next traversing movement. One of the specific consequences of this is to initiate travel along the contour from the start position of the geometry axes back to the program continuation point with an activated switching signal.

Example

- Standard sequence
  If part program machining is executed as standard, the switching signal is activated for the first time at the beginning of part program block N60.

Part program

```
N10 G0 X0 Y0
N20 G1 F1000 X10 Y10
N30 G0 X50
N40 CC_FASTON(–0.4, 0.4)
N50 Y50
N60 G1 X100 Y100
```

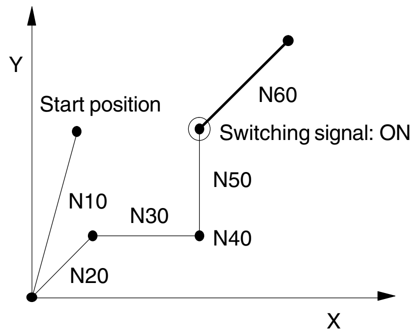


Fig. 3-1 Switching signal for part program machining operation

- Sequence following block search
  If a block search is executed for the block end point of part program block N60, the switching signal is activated on reaching the start position of the geometry axes.

Part program

```
N10 G0 X0 Y0
N20 G1 F1000 X10 Y10
N30 G0 X50
N40 CC_FASTON(–0.4, 0.4)
N50 Y50
N60 G1 X100 Y100
```

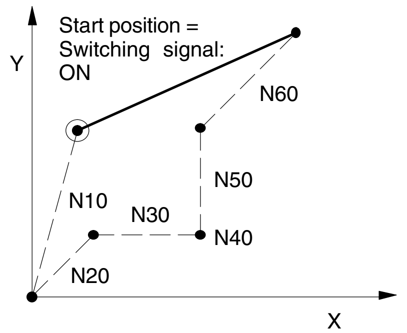


Fig. 3-2 Switching signal after block search

Switching signal output in REPOS block

The user (machine manufacturer) must take appropriate measures, e.g. disable the switching signal, in order to suppress the activation of the switching signal in the REPOS block in the constellation described above.

Note

It is part of the general responsibility of the user (machine manufacturer) to suppress the output of the switching signal during repositioning, e.g. after a block search.

References

You will find a complete description of the block search in:

/FB2/ **Description of Functions Basic Machine**
Mode Group, Channels, Program Operation K1
Program Test

3.2.2 Transformations

The function will only run correctly with deactivated transformation. There is no monitoring function.

References

You will find a complete description of the transformations in:

/FB2/ **Description of Functions Extended Functions**
Kinematic Transformation M1

/FB3/ **Description of Functions Special Functions**
Handling Transformation Package TE4

3.2.3 Compensation

All compensations are taken into account.

References

You will find a complete description of the compensations in:

/FB2/ **Description of Functions Extended Functions**
Compensations K3

3.2.4 Tool radius compensation (TRC)

As part of tool radius compensation, control-internal part program blocks (compensation blocks) are inserted into the part program. Referenced to the switching signal output, a compensation block is always added to the next programmed part program block.

References

You will find a complete description of tool radius compensation in:

/FB1/ **Description of Functions Basic Machine**
Tool Radius Compensation W1
Tool Radius Compensation

3.2.5 Continuous-path mode (ADIS)

A part program block inserted into the part program internally by the control system in continuous-path mode with programmable smoothing characteristics (G641 ADIS) is added to the previous part program block with reference to the original switching position. This causes the switching signal not to be switched until the start of the new part program block.

References

You will find a complete description of continuous-path mode with programmable smoothing characteristics (ADIS) in:

/FB1/ **Description of Functions Basic Machine**
Continuous-Path Mode, Exact Stop and LookAhead B1

3.2.6 Software cam

As the hardware timer is also used for the “software cams” function, it cannot be used at the same time as the “clock-independent switching signal output with software cams” function.

The following alarm appears in the event of an error:

- Alarm: “75500 channel *channel number*, incorrect function configuration: Clock-independent switching signal output”

References

You will find a complete description of the software cams in:

/FB2/ **Description of Functions Extended Functions**
Software Cams, Position Switching Signals N3

■

Notes

Data Descriptions (MD, SD) 4

4.1 Function-specific machine data

62560 MD number	**$MC_FASTON_NUM_DIG_OUTPUT** Configuration of the switching signal output	
Default setting: 0	Minimum input limit: 0	Maximum input limit: 4
Changes effective after POWER ON	Protection level: 2/7	Unit: –
Data type: Byte		
Meaning:	The number of the digital on-board output (1...4) assigned to the switching signal is specified via machine data. 0 deactivates the output of the switching signal.	

■

Notes

Signal Descriptions 5

– No signal descriptions available –

■

Notes

6 Examples

– No examples available –

■

Notes

Data Fields, Lists

7

7.1 NC machine data

7.1.1 General machine data

Number	MD identifier	Name	Reference
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	

7.1.2 Function-specific machine data

Number	MD identifier	Name	Reference
Channel-specific ($MC_...)			
62560	FASTON_NUM_DIG_OUTPUT	Number of on-board digital output for switching signal	

7.2 Alarms

7.2.1 System alarms

Detailed explanations of the alarms, which may occur, appear in

References: /DA/, Diagnostics Guide

or in the Online help.

7.2.2 Function-specific alarms

75500	**Channel % 1 incorrect function configuration: Clock-independent switching signal output**
Explanation	%1 = channel number The following configurations generate this alarm: • No geometry axes defined in the 1st NC channel • Option: Software cam set.
Reaction	The function cannot be activated.
Remedy	Change the configuration.
Program continuation	Trigger NC reset.

■

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

Preprocessing (V2)

■

Notes

1 Brief Description

Preprocessing

The programs stored in the directories for standard and user cycles can be preprocessed to reduce runtimes.

Preprocessing is activated via machine data.

Standard and user cycles are preprocessed when the power is switched on, i.e. as an internal control function, the part program is translated (compiled) into a binary intermediate code optimized for processing purposes.

All program errors that can be corrected by means of a compensation block are detected during preprocessing. In addition, when the program includes branches and check structures, a check is made to ensure that the branch destinations are present and that 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:

- To optimize the runtimes of part programs with high-level language components (branches, check structures, motion-synchronous actions)
- CPU time intensive part programs (e.g. stock removal cycles)
- Faster processing of time-critical sections (e.g. program continuation after preprocessing stop during rapid deletion of distance to go, or return stroke, or in the tool change cycle).

■

Notes

Detailed Description 2

2.1 General functionality

General information

- **SW 3.2 and higher**
  Standard and user cycles can be preprocessed.
  The processing time of part programs can then be reduced without restricting the control functionality.
  If machine data MD 10700: PREPROCESSING_LEVEL is set accordingly, standard and user cycles are preprocessed.
  Preprocessing is program-specific. Preprocessed part programs and programs interpreted in ASCII format can be mixed.
  Preprocessing reduces idle times.

- **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.
  These bits can be set to select cycle preprocessing specifically for individual directories (e.g. user cycles).

- **SW 5 and higher**
  Bit 5 added to MD 10700: PREPROCESSING_LEVEL. This allows you to select preprocessing for user cycles from the _N_CMA_DIR directory.

- **SW 6.4 and higher**
  Bit 5 added to MD 10700: PREPROCESSING_LEVEL. This allows selective preprocessing of the specific individual user cycles that have the command PREPRO after the PROC instruction.

- **SW 7.1 and higher**
  Precompiled cycles are stored by default in the **DRAM**.
  Bit 6 added to MD 10700: PREPROCESSING_LEVEL. This can be used to specify that compiled programs currently stored in the DRAM, which no longer have sufficient space in the DRAM, may be stored in the SRAM.

Functionality

The programs stored in the directories for standard and user cycles are preprocessed when the power is switched on, i.e. the part program is translated (compiled) into an intermediate binary code optimized for processing purposes. The compilation is processed when called.

Runtime optimization

The preprocessing function is primarily suited for optimizing the runtimes of part programs with high-level language components (branches, check structures, motion-synchronous actions).

While branches and check structures are invalidated by a search through all blocks (block start) when part programs are interpreted in ASCII format (active as default), a branch is made directly to the destination block in a preprocessed part program.

The runtime differences between branches and check structures are thus eliminated.

Preprocessing runtime example:
Runtime reduced by 30% on 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: PRE-PROCESSING_LEVEL**, bit 1 is set.

0: No preprocessing

Bit 0=0: Call description of cycles is not known as standard. Cycles must be declared as external before the cycle call like normal subroutines. This setting is useful if no cycles with call parameters are being used.

Bit 0=1: The cycle call description is generated during control POWER ON. All user cycles (_N_CUS_DIR directory) and Siemens cycles (_N_CST_DIR directory) with transfer parameters can be called without external declaration. Changes to the cycle call interface only take effect on POWER ON.

The following machine data must be set:
$MN_MM_NUM_MAX_FUNC_NAMES
$MN_MM_NUM_MAX_FUNC_PARAM

Bit 1=1: All cycles are preprocessed in a compilation optimized for processing during control POWER ON. All user cycles (_N_CUS_DIR directory) and standard cycles (_N_CST_DIR directory) are executed at high speed. Changes to the cycle programs do not take effect until the next POWER ON.

As of SW 3.5

Bit 2=1: During control run-up the standard cycles in directory _N_CST_DIR are preprocessed in a compilation optimized for processing.

Bit 3=1: During control run-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 command PREPRO in the PROC instruction line. Files in directories that are not marked by bits 1 – 4 are not preprocessed (SW 6.4 and higher) If the bit is Bit 0, preprocessing is controlled exclusively according to Bits 0–4.

As of SW 7.1

Bit 6 = 0:
The compilation is stored in the **DRAM** if sufficient memory is available. If sufficient memory is not available, preprocessing is aborted. Dimensioning of the DRAM with MD 18351: MM_DRAM_FILE_MEM_SIZE.

Bit 6 = 1:

The compilation for programs for which there is not sufficient space in the DRAM is stored in the SRAM. Error messages are output for programs for which there is not sufficient compilation space in the SRAM.

The areas occupied by the compilation in the DRAM are visible to the user.

Bit combinations are permissible.

Compilation

Subroutines located in the directories for standard cycles: _N_CST_DIR, _N_CMA_DIR and user cycles: _N_CUS_DIR (extension _SPF) and, if applicable, subroutines marked with PREPRO, are compiled. The name of the compilation corresponds to the original cycle with extension _CYC.

Note

Program changes to precompiled programs do not take effect until the next power ON!

Access 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 requirements

The memory requirement for compiled cycles is approximately factor 2 in addition to ASCII part programs.

The memory requirements for variables defined in the part programs are 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 subroutine 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 machine data
MD 28020: MM_NUM_LUD_NAMES_TOTAL.

Example:

```
PROC NAMES                    ; 1 name
DEF INT VARIABLE, FIELD[2]    ; 2 names
START:                        ; 1 name, only for preprocessing
FOR VARIABLE = 1 TO 9         ; 1 name, only for preprocessing
G1 F10 X=VARIABLE*10–56/86EX4+4*SIN(VARIABLE/3)
ENDFOR                        ; 1 name, only for preprocessing
M17
```

In order to execute this program normally, 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.

SW 7.1 and higher

Preprocessed programs/cycles are stored in the **DRAM**. The space required for each program must be flashed over unmodified as outlined above. Tailoring to the assignment of locations in the SRAM is only required if bit 6 in MD 10700: PREPROCESSING_LEVEL Bit 6 has been set to 1. In this case, the program compilations for which there is insufficient space in the DRAM are stored in the SRAM.

Examples of appropriate machine data settings appear in Section 6.2.

2.3 Program call

Overview

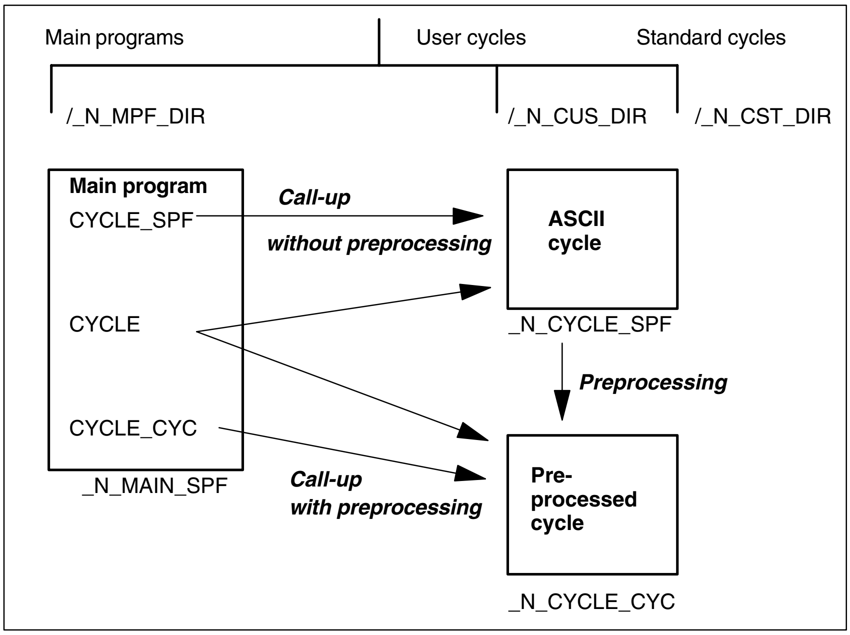


Fig. 2-1 Generation and call of preprocessed cycles without parameter

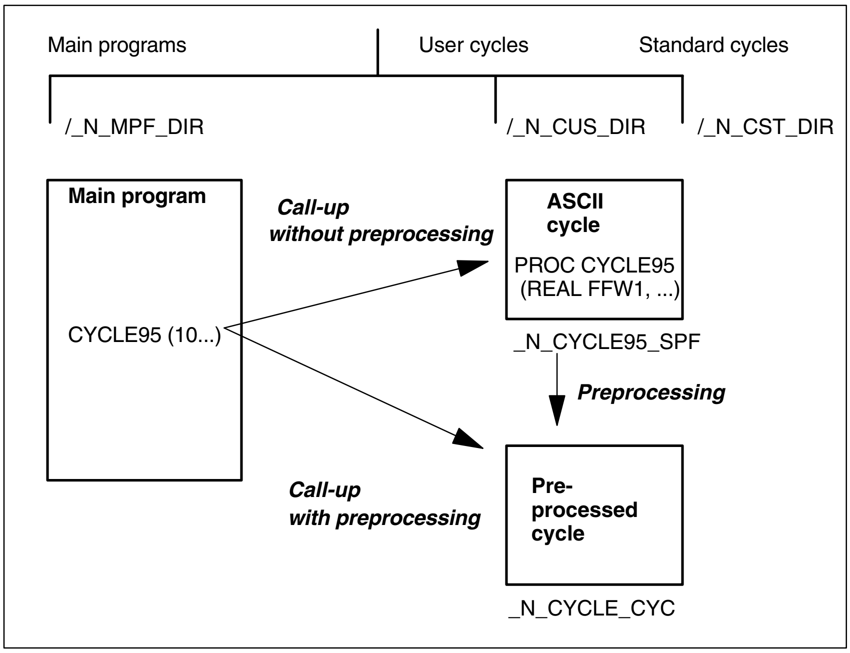


Fig. 2-2 Generation and call of preprocessed cycles with parameter

Call

- Compiled cycle:
  A compiled cycle is called in exactly the same way as a normal subroutine.

  Example: CYCLE

- Preprocessing is activated:
  The compiled cycle is called instead of the ASCII cycle.

  – If the subroutine is called explicitly with extension _SPF, then the ASCII cycle is called even if a compilation is available.

    Example: CYCLE_SPF ;ASCII cycle call

  – If the subroutine is called explicitly with extension _CYC, then the preprocessed cycle is called if available. An error message is output if no compilation is available.

    Example: CYCLE_CYC ;Preprocessed cycle call

  – If Bit 5 is set and a file that is not marked with PREPRO called explicitly with the extension _CYC, an error message is issued with Alarm 14011.

- If a subroutine is called without explicit extension, an attempt is first made to load the program. If this is not possible (not marked with PREPRO), an attempt is made to load the SPF program.

- The change to an external language mode with G291 is rejected and an alarm issued. When the pre-compiled cycle is called, an explicit change is made to the Siemens language mode.

- When the subroutine is called, it is checked whether the compiled file is older than the cycle. If so, the compile file is deleted and an alarm issued. The user must 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 is set, all programs that do not start with the PREPRO PROC instruction are not precompiled.

Syntax check

All program errors that can be corrected by means of a compensation block are detected during preprocessing. In addition, when the program includes branches and check structures, a check is made to ensure that the branch destinations are present and that 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 part program.

There are no restrictions on the calculation of measured process variables and in the reaction to signals from the process and other channels (override, deletion of distance to go, motion-synchronous actions, channel coordination, interrupt processing, etc.).

Axis identifiers

Part programs are compiled independently of channels. For this reason, the geometry and channel identifiers set via
MD $MC_AX CONF_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 channels and
- Independently of the axis identifiers defined on the machine.

The axes to be traversed are addressed indirectly via machine data or transferred as parameters:

- Indirect axis programming:
  - IF $AA_IM[**AXNAME($MC_AXCONF_CHANAX_NAME_TAB[4])**] > 5
    ; This branch 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:
  - Cycles definition
    PROC BOHRE(AXIS DRILLING AXIS)
    WHILE $AA_IW[**DRILLING AXIS**] > –10
    G1 G91 F250 **AX[DRILLING AXIS]** = –1
    ENDWHILE
  - Call from main program
    DRILL(Z)

■

Notes

Supplementary Conditions 3

Availability of the “pre-processing” function

The function is an option and is 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 from SW 3.2 and SW 3.5 or predefined in more detail at file level (SW 6.4 and higher) with the following existing machine data:

10700 MD number	**PREPROCESSING_LEVEL** Program preprocessing level				
Default setting: 1		Minimum input limit: 0		Maximum input limit: 15	
Changes effective after POWER ON			Protection level: 2/2		Unit: –
Data type: BYTE			Applies as of SW: 3.2		
Meaning:	Preprocessing describes the compilation of cycles/programs. Once they have been preprocessed, programs/cycles can be used in the part program without an additional EXTERN declaration. A number of different preprocessing levels are possible: **Bit 0= 0**: No preprocessing **Bit 0= 1**: The cycle call description is generated during control run-up, i.e. all the programs in directories _N_CUS_DIR and _N_CST_DIR can be called in the part program without an EXTERN declaration. Any changes made to the parameter interface of a cycle in the control do not take effect until the next POWER ON. **Bit 1= 1**: During control run-up, all cycles are preprocessed into a compilation optimized for processing. All user cycles (_N_CUS_DIR directory) and standard cycles (_N_CST_DIR directory) are processed at high speed. Program changes to cycle programs do not take effect until the next POWER ON. **Bit 2= 1**: During control run-up, the Siemens cycles in the _N_CST_DIR directory are preprocessed in a compilation optimized for processing (SW 3.5 and higher). **Bit 3= 1**: During control run-up, the user cycles in the _N_CUS_DIR directory are preprocessed in a compilation optimized for processing (SW 3.5 and higher). **Bit 4 =1**: Preprocessing of user cycles from directory _N_CMA_DIR (SW 5 and higher). **Bit 5 =1**: Preprocessing of user cycles only by marking with PREPRO in the PROC instruction (SW 6.4 and higher). **Bit 6 =1**: (SW 7.1 and higher). Precompiled programs are stored in the DRAM. Compilations for which there is insufficient space in the DRAM are transferred to the SRAM if this bit is set. Memory is required for preprocessing cycles. Memory usage can be optimized by means of selective preprocessing (directories/PREPRO): – Runtime-critical cycles are grouped in a single directory. – The remaining cycles are located in a different directory. – Use bit 5 to mark only critical cycles. – SW 7.1 and higher: The precompiled cycles are in the DRAM.					
Additional references	/PA/, “Programming Guide Fundamentals” (EXTERN Declaration)				

4.1.2 Channelspecific machine data

Memory requirements

The memory space required for variables defined in the part 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 subroutine 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

■

Notes

Example 6

6.1 Preprocessing individual files

```
PROC PART1 PREPRO                ; Preprocessing if bit 5 = 1
                                 ; in PREPROCESSING_LEVEL
N1000 DEF INT COUNTER
N1010 DESTINATION: 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 DESTINATION
N1070 M30

PROC PART2
N2000 DEF INT VARIABLE, FIELD[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 PART1
N30 G0 X10 Y10 Z10
N40 PART2
N50 G0 X100 Y100
N60 PART3
N70 G0 X10 Y10
N80 M30
```

Example constellations:
a) Bit 5 = 1
$MN_PREPROCESSING_LEVEL=45 ; Bit 0, 2, 3,5

PART1 subroutine is precompiled, the call description is generated.
PART2 subroutine is not precompiled but the call description is generated.

b) Bit 5 = 0
$MN_PREPROCESSING_LEVEL=13 ; Bit 0, 2, 3,

Both subroutines are precompiled, the call description is generated.

c)
Example for an **invalid** subroutine, active compilation:

PROC SUB1 PREPRO

G291 ; <—— Alarm during compilation, G291 not possible
G0 X0 Y0 Z0
M17

6.2 Preprocessing in the DRAM

Machine data for preprocessing only in the DRAM with selective selection:

; Bit 5 = 1	Selective program selection
; Bit6 = 0	No diversion to SRAM if DRAM full

N30 $MN_MM_DRAM_FILE_MEM_SIZE = 800 ; Reserve space

N40 $MN_PREPROCESSING_LEVEL = 63 ; Bit 0–5 = 1

M17

Machine data for preprocessing in the DRAM, with option to use the SRAM and with selective selection:

; Bit 5 = 1	Selective program selection
; Bit6 = 1	Diversion to SRAM if DRAM full

N30 $MN_MM_DRAM_FILE_MEM_SIZE = 800 ; Reserve space

N40 $MN_PREPROCESSING_LEVEL = 127 ; Bit 0–6 = 1

M17

■

Data Fields, Lists

7

7.1 Machine data

Number	Identifier	Name	Reference
General ($MN_ ...)			
10700	PREPROCESSING_LEVEL	Program preprocessing level	
18242	MM_MAX_SIZE_OF_LUD_VALUE	Maximum field size of LUD variables	S7
Channelspecific ($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

Detailed explanations of the alarms, which may occur, appear in

References: /DA/, Diagnostics Guide

or in the Online help.

■

Notes

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

3D Tool Radius Compensation (W5)

■

Notes

1 Brief Description

Why 3D TRC?

3D tool radius compensation is used to machine contours with tools that can be controlled in their orientation independently of the tool path and shape.

Note

This description is based on the specifications for 2D tool radius compensation.

References: /FB/, W1, "Tool Compensation"

How $2^1/_2$ D - 3D TRC differ

- 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 circumferential milling operations.

  While the orientation of the machining surface is not constant when other tools are used, it is determined by the contour and cannot thus be controlled independently of it.

- With **3D TRC**, surfaces with variable orientation are generated.

  The prerequisite for 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.

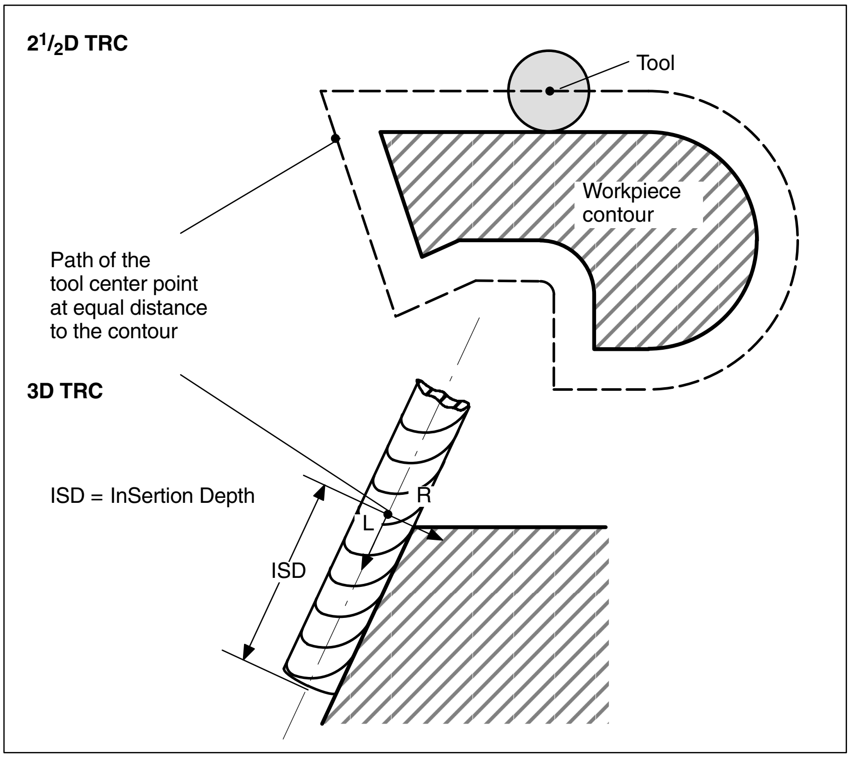


Fig. 1-1 $2^1/_2$ D, 3D tool radius compensation

The parameters for the operation shown in Fig. 1-2 “Face milling” are described in detail in Section 2.2.

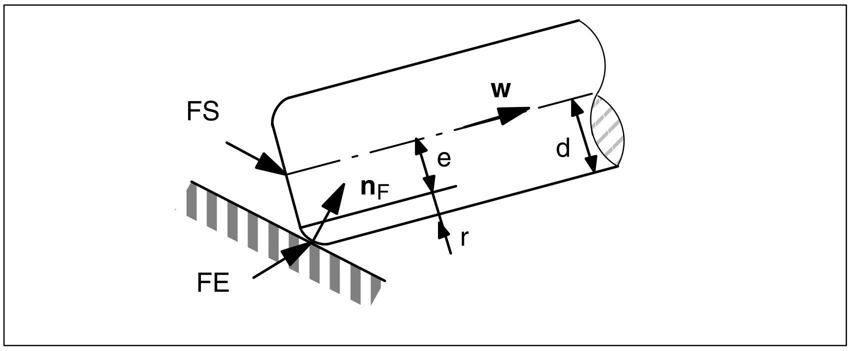


Fig. 1-2 Face milling

Orientation

With 3D TRC, a distinction must be drawn between

- 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 for peripheral milling.

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

Detailed Description 2

The following section provides a detailed function description of 3D tool radius compensation with respect to

- Peripheral milling and
- Face milling

item.

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 outside corners. The only decisive factor is the radius at the tool contact point.

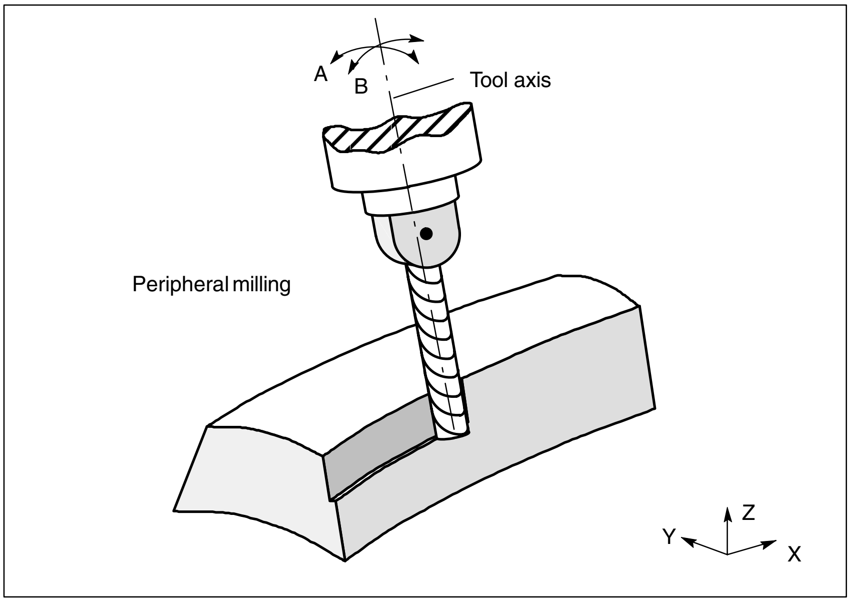


Fig. 2-1 Peripheral milling

Insertion depth (ISD)

The ISD (InSertion Depth) program command is used to program the tool insertion depth for peripheral milling operations. This makes it 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.

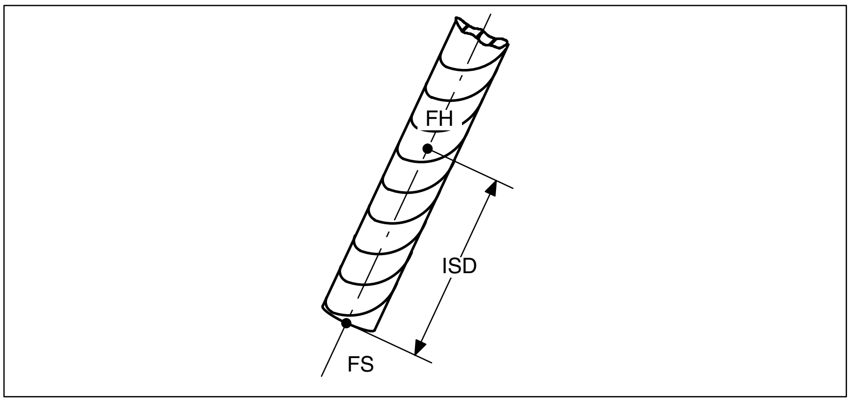


Fig. 2-2 Insertion depth

2.1.1 Corners for peripheral milling

Outside corners/ inside corners

Outside corners and inside corners must be treated separately. The terms inside corner and outside corner are dependent on the tool orientation. When the orientation changes at a corner, for example, the corner type may change **while** machining is in progress. Whenever this occurs, the machining operation is aborted with an error message.

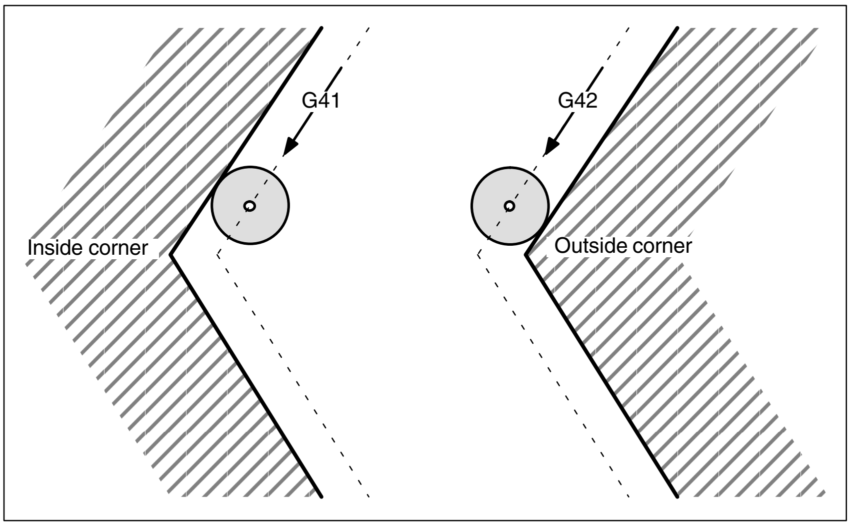


Fig. 2-3 Corner type

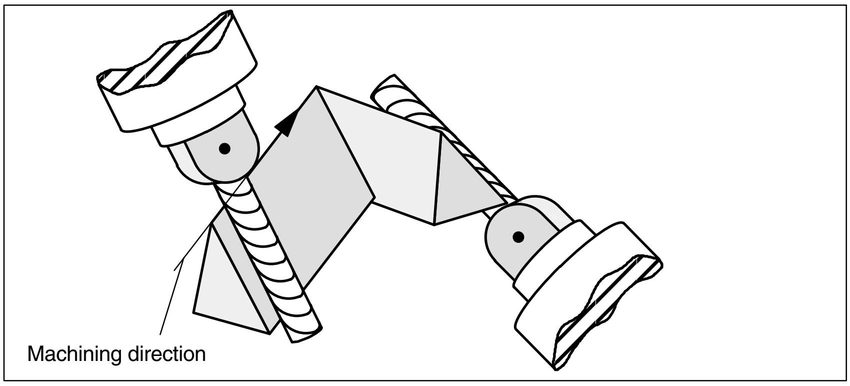


Fig. 2-4 Change of corner type during machining

2.1.2 Behavior at outside corners

Similar to the procedure for 21/2D tool radius compensation, a circle is inserted at outside corners (G450) or the point of intersection of the offset curves is approached (G451).
For transitions that are almost tangential, the procedure is identical whether G450 or G451 is active (limit angle can be set via MD). Conversely, if G451 is active, a circle is also inserted (procedure as for G450) if there is no intersection or if the corner angle exceeds a specific value (MD).
If there is a change in orientation between the two traversing blocks, a circle is always inserted.

G450

Outside corners are treated as if they were circles with a 0 radius. The tool radius compensation acts on these circles in the same way as on any other programmed path.

The circle plane extends from the final tangent of the first block to the start tangent of the second block.

The orientation can be changed during block transition.

A change in orientation between two programmed blocks is executed either before the circle block or in parallel to it. Circles are always inserted. The DISC command is not evaluated.

Programming

- **ORIC** Change in orientation and path motion in parallel (**ORI**entation Change **C**ontinuously)
- **ORID** Change in orientation and path motion in succession (**ORI**entation Change **D**iscontinuously)

The ORIC and ORID program commands are used to determine whether changes in orientation programmed between two blocks are executed before the inserted circle block is processed or at the same time.

When the orientation needs to be changed at outside corners, the change can be implemented in parallel to interpolation or separately from the path motion. When ORID is programmed, the inserted blocks are executed first without a path motion (blocks with changes in orientation, auxiliary function outputs, etc.). The circle block is inserted immediately in front of the second of the two traversing blocks which form the corner.

ORIC

If ORIC is active and there are two or more blocks with changes in orientation (e.g. A2= B2= C2=) programmed between the traversing blocks, then the inserted circle block is distributed among these intermediate blocks according to the absolute changes in angle.

Change in orientation

The method by which the orientation is changed at an outside corner is determined by the program command that is active in the first traversing block of an outside corner.

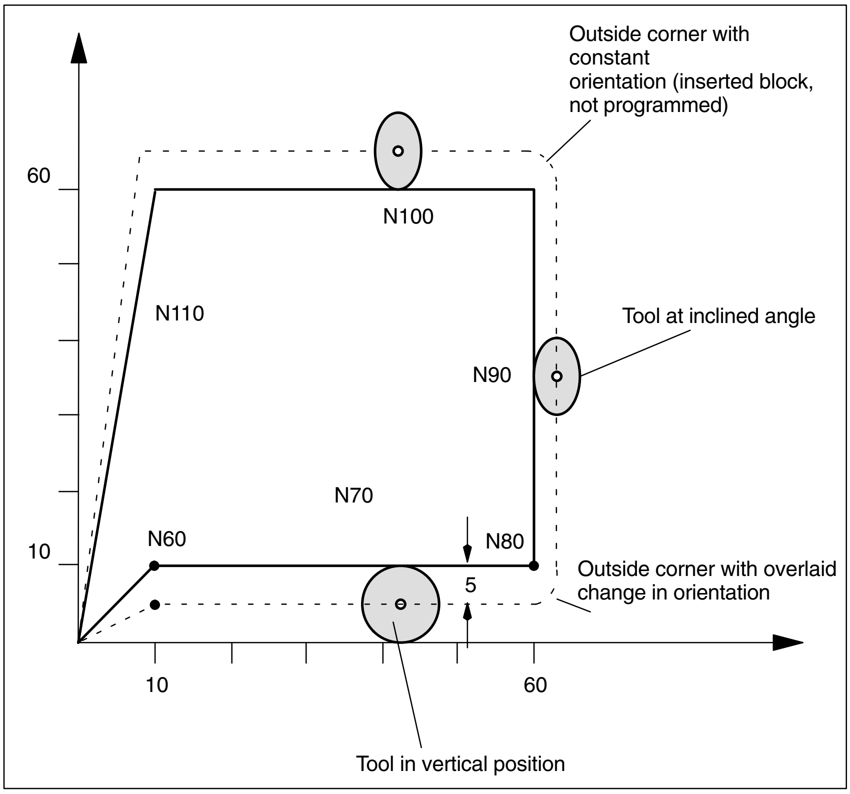


Fig. 2-5 **ORIC** Change in orientation and path motion in parallel

Example:

```
N10 A0 B0 X0 Y0 Z0 F5000
N20 T1 D1                      ;Radius=5
N30 TRAORI(1)                  ;Selection of transformation
N40 CUT3DC                     ;Selection of 3D TRC
N50 ORIC
N60 G42 X10 Y10                ;Selection of TRC
N70 X60
N80 A3=1 B3=0 C3=1             ;Change in orientation at the outside corner
formed by N70 and N90 Y60      ;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 the outside 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.

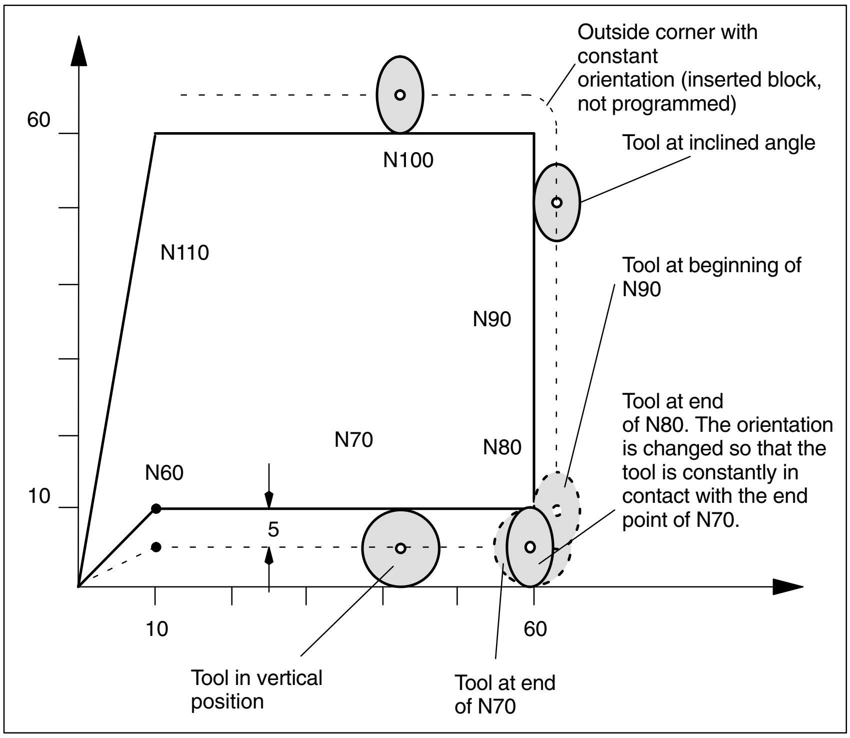


Fig. 2-6 **ORID**, change in orientation and path motion as successive actions

Example:

```
N10 A0 B0 X0 Y0 Z0 F5000
N20 T1 D1                  ;Radius=5
N30 TRAORI(1)              ;Selection of transformation
N40 CUT3DC                 ;Selection of 3D TRC
N50 ORID
N60 G42 X10 Y10            ;Selection of TRC
N70 X60
N80 A3=1 B3=0 C3=1         ;Change in orientation at the outside corner
N90 Y60                    ;formed by N70 and N90
N100 X10
N110 G40 X0 Y0
N120 M30
```

Note

The DISC command 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 inside corners

Collision monitoring

With the 3D compensation function, only adjacent traversing blocks are taken into account in the calculation of intersections.
Path segments must be long enough to ensure that the contact points of the tool do not cross the block limits into other blocks when the orientation changes at an inside corner.

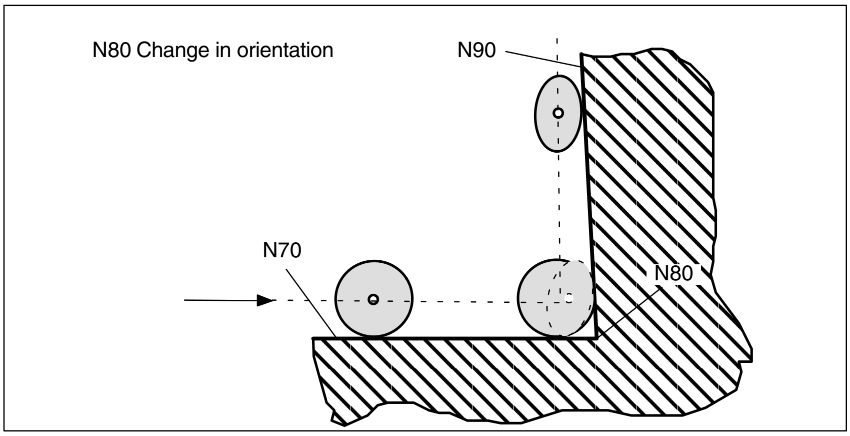


Fig. 2-7 The contact points of the tool must not cross the limits of block N70 or N90 as a result of the change in orientation in block N80

Example:

```
N10 A0 B0 X0 Y0 Z0 F5000
N20 T1 D1                        ;Radius=5
N30 TRAORI(1)                    ;Selection of transformation
N40 CUT3DC                       ;Selection of 3D TRC
N50 ORID
N60 G42 X10 Y10                  ;Selection of TRC
N70 X60
N80 A3=1 B3=0 C3=1               ;Change in orientation at the inside 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 inside corner so that it is constantly in contact with the two blocks forming the corner.

When the orientation changes in a block that is one of the two blocks forming the inside corner, then it is no longer possible to adhere to the programmed relationship between path position and associated orientation. This is because the orientation must reach its end value even though the path end position is not reached. This response is identical to the response of synchronized axes with $2^1/_2$ D tool radius compensation.

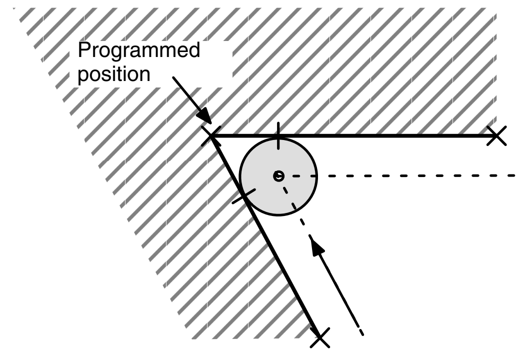


Fig. 2-8 Path end position and change in orientation at inside corners

Change in insertion depth

Generally speaking, the contour elements that form an inside corner are not positioned on the plane perpendicular to the tool. This means that the contact points between the two blocks and the tool are at different distances from the tool tip.

This means: the insertion depth (ISD) changes abruptly from the 1st to the 2nd block at an inside corner.

To ensure that this difference in depth is not an abrupt step change, it is distributed continuously among the blocks involved during interpolation. The depth-compensating motion is executed in the current tool direction.

This solution prevents the contour from being violated by cylindrical tools
if the length of the tool prevents the cutter contact point on the lateral surface of the cutter leaving the range in which machining is possible.

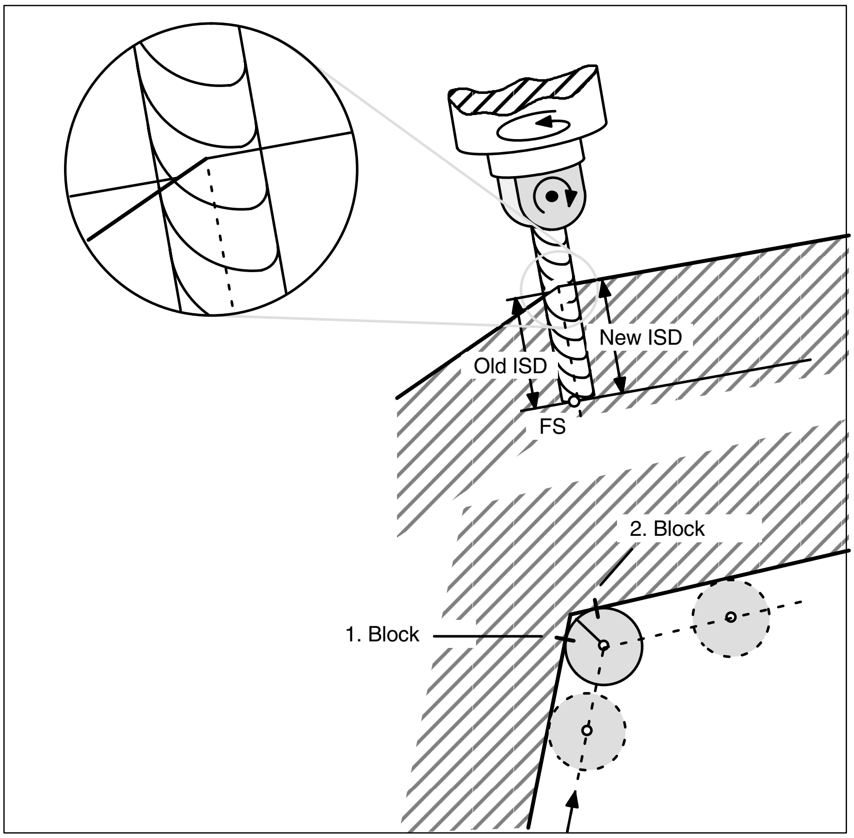


Fig. 2-9 Change in insertion depth

Example of inside corners

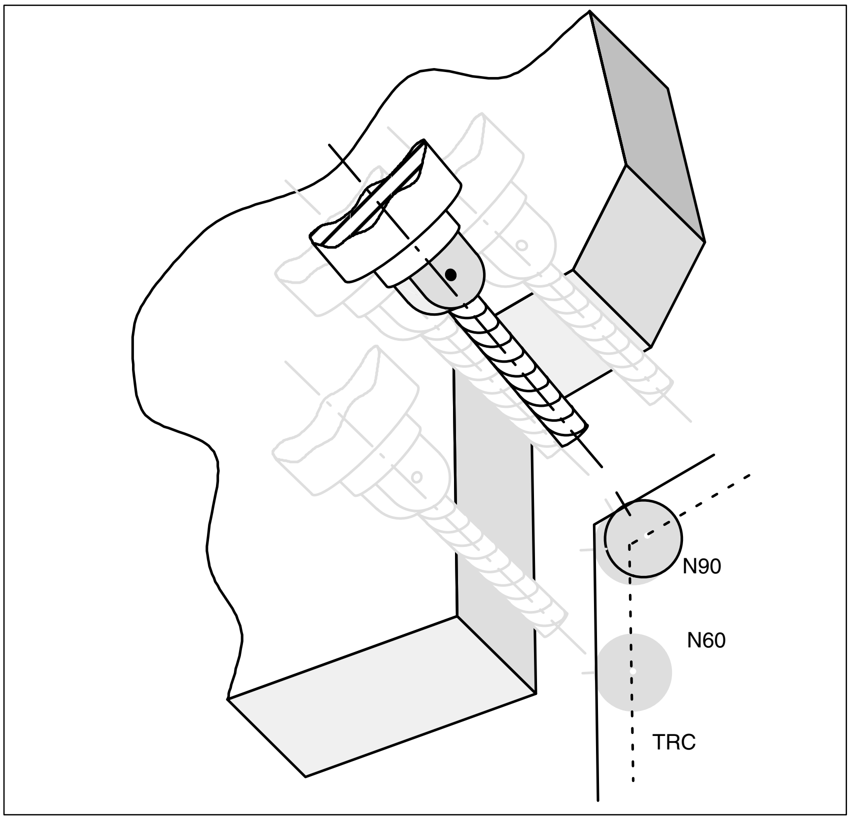


Fig. 2-10 Change in orientation at an inside corner

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 the inside corner
                             ;formed by N70 and N90 X60 Y90
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.

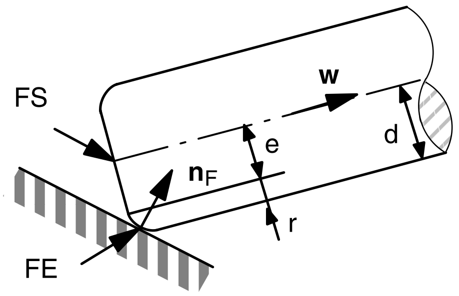


Fig. 2-11 Face milling with a torus

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 (bevel cutter), for example, have an identical machining action since only the section at the tool tip is taken into account. The only difference between these tools is that the tool shape is represented differently (dimensions).

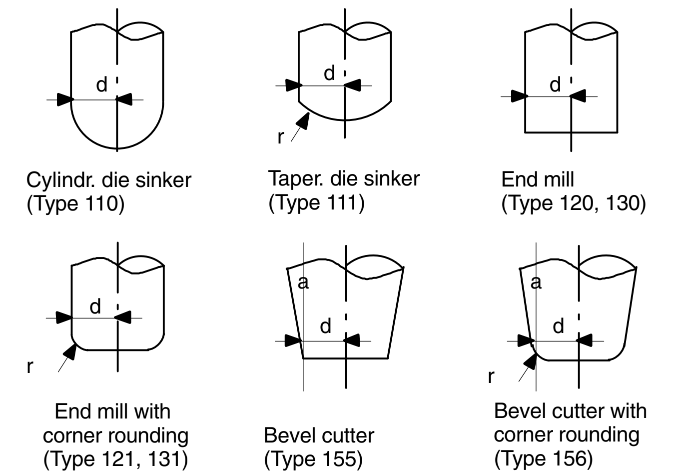


Fig. 2-12 Tool types for face milling

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 (inside corner), then the interpolation range of the surface normal vector is reduced accordingly, i.e. the end value of the surface normal vector is not reached as it would be with other interpolation quantities such as, for example, the position of an additional synchronized axis.

In addition to the usual methods of programming orientation, it is also possible to refer the tool orientation to the surface normal vector and path tangent vector using the addresses LEAD (lead or camber angle) and TILT (side angle). The lead angle is the angle between the tool orientation and the surface normal vector. The side angle is the angle between the path tangent and the projection of the tool vector into the surface to be machined. Specification of the angle relative to the surface normal is merely an additional option for programming tool orientation at the block end. It does not imply that the lead and side angles reach their programmed values before the path end point is reached.

The final tool orientation is calculated from the path tangent, surface normal vector, lead angle and side angle at the block end. This orientation is always implemented by the end of the block, particularly in cases where the block is shortened (at an inside corner). If the omitted path section is not a straight line in a plane, the lead and side angles generally deviate from their programmed values at the path end point. This is because the orientation has changed relative to the surface normal vector or path tangent vector when the absolute orientation of the tool is the same as at the original path end point.

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 γ 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.

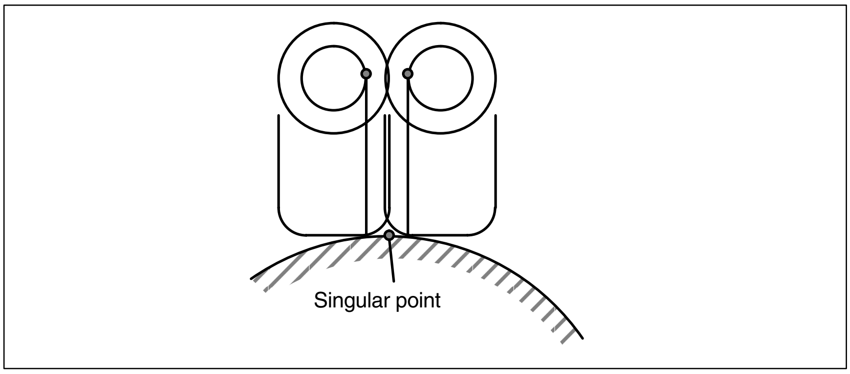


Fig. 2-13 Change in the machining point on the tool surface close to a point in which surface normal vector and tool orientation are parallel.

The problem is basically solved as follows: If the angle δ between the surface normal vector $\mathbf{n}_F$ and tool orientation “**w** is smaller than a limit value (machine data) δ_{min}, then the side angle γ 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 δ now becomes 0, i.e. the sign of lead angle β changes, the machining point moves from its current position to the opposite side of the tool. This movement is executed in an inserted linear block.

The machining operation is aborted with an alarm if an attempt is made to machine within the illegal angular range for the side angle γ (i.e. $\delta < \delta_{min}$ and $\gamma \times \quad 0$).

The insertion of linear blocks makes it necessary to split the original blocks at the singular points. The partial blocks created in this way are treated as if they were original, which means, for example, that a concave path containing a singularity is treated like an inside corner, i.e. there is no contour violation. Each new partial block must contain at least one tool contact point since this is always calculated on the basis of adjacent traversing blocks.

Singularities do not just occur at isolated points, but along whole curves. This is the case, for example, if the curve to be interpolated is a plane curve (i.e. a curve with a constant osculating plane) and the tool is constantly aligned in parallel to the binormal vector, i.e. perpendicular to the osculating plane. A simple example is a circular arc in the x-y plane that is machined by a tool aligned in parallel to the z axis. On paths of this type, the tool offset is reduced to a tool length compensation, i.e. the tool is moved so that its tip FS is positioned on the programmed path.

On transition between singular and non-singular curves, linear blocks must be inserted in the same way as for isolated singular points so that the machining point on the tool can move from the tool tip FS to the periphery (on outside corners and convex surfaces) or the paths must be shortened to avoid contour violations (on inside corners and concave surfaces).

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 (inside or outside corner) is determined by the surface normal of the surfaces involved and by the paths defined on them.

The surface normals of the two surfaces forming the edge may point in opposite directions of the overall surface (the front edge of one surface is continued on the rear edge of the second surface), see also 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 inside and outside 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.

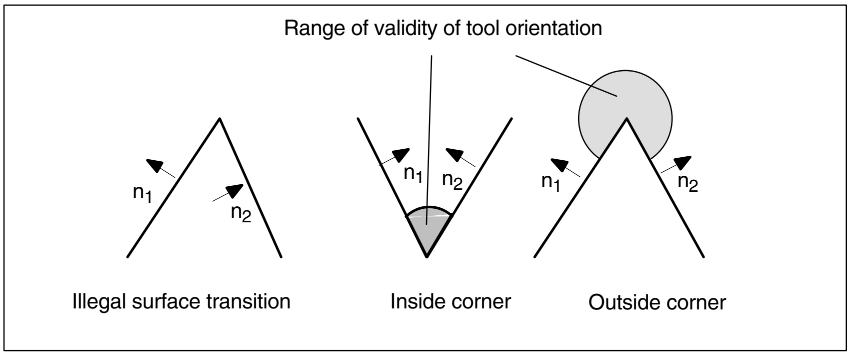


Fig. 2-14 Corners for face milling

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 changes in orientation may not be inserted between the activation block and the first corrected block or between the last corrected block and the deactivation block. Other intermediate blocks are, however, permitted.

2.2.5 Behavior at outside corners

Outside corners are treated as if they were circles with a 0 radius. The tool radius compensation acts on these circles in the same way as on any other programmed path.

The circle plane extends from the final tangent of the first block to the start tangent of the second block.

The orientation can be changed during block transition.

A circle block is always inserted at an outside corner.

A change in orientation between two programmed blocks is executed either before the circle block or in parallel to it.

Programming

- **ORIC** Change in orientation and path motion in parallel (**ORI**entation Change **C**ontinuously)
- **ORID** Change in orientation and path motion in succession (**ORI**entation Change **D**iscontinuously)

The ORIC and ORID program commands are used to determine whether changes in orientation programmed between two blocks are executed before the inserted circle block is processed or at the same time.

When the orientation needs to be changed at outside corners, the change can be implemented in parallel to interpolation or separately from the path motion. When ORID is programmed, the inserted blocks are executed first without a path motion (blocks with changes in orientation, auxiliary function outputs, etc.). The circle block is inserted immediately in front of the second of the two traversing blocks which form the corner.

ORIC

If ORIC is active and there are two or more blocks with changes in orientation (e.g. A2= B2= C2=) programmed between the traversing blocks, then the inserted circle block is distributed among these intermediate blocks according to the absolute changes in angle.

Change in orientation

The method by which the orientation is changed at an outside corner is determined by the program command that is active in the first traversing block of an outside corner.

If the tool orientation at an outside corner is not constant, then the change in orientation is implemented in exactly the same way as described in Subsection 2.1.2 for peripheral milling operations.

2.2.6 Behavior at inside corners

The position of the tool in which it is in contact with the two surfaces forming the corner must be determined at an inside corner. The contact points must be on the paths defined on both surfaces. It is not usually possible to solve this problem exactly since, when the tool is moved along the path on the first surface, it normally touches a point on the second surface which is not on the path.

For this reason, the tool is not moved along the path on the first surface, but deviates from the path in such a way as to ensure that the distance between the contact points and the relevant contours in the position in which the tool contacts both surfaces is minimal, see also Fig. 2-15.

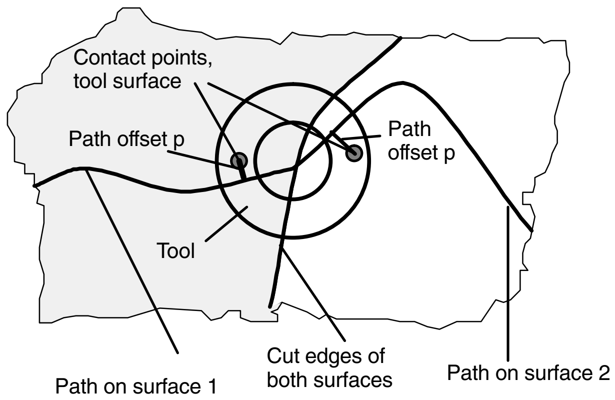


Fig. 2-15 Inside corner with face milling (view in direction of longitudinal axis of tool)

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” the cutter side at an inside corner (the value of the angular difference φ about the tool longitudinal axis between the two contact points on the tool surface is approximately $180°$) is more likely to be the exception (see also Fig. 2-16 on the right). The angle φ will normally stay almost constant so that the distance between the contact points on the tool surface will be relatively small (see also Fig. 2-16 on the left).

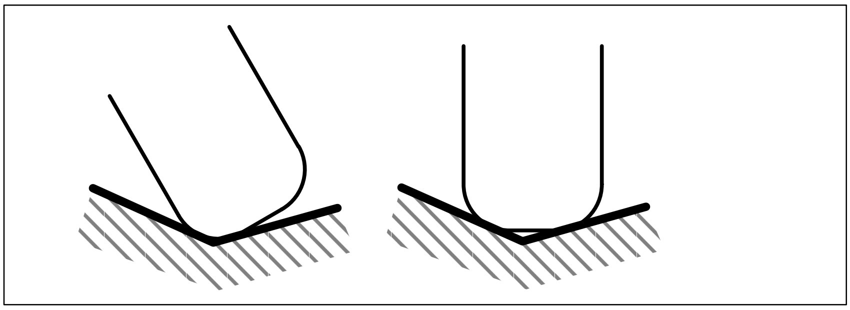


Fig. 2-16 Machining at inside corners

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 inside corners at the block start and block end are overlaid. The current difference in a path point is always perpendicular to the path and in the surface defined by the surface normal vector.

If the tool orientation at an inside corner is not constant, the change in orientation is implemented in the same way as described in Subsection 2.1.3 for 3D peripheral milling, i.e. the tool is moved in the corner so that it contacts the two adjacent surfaces at the block start, block end and at two points $^1/_3$ and $^2/_3$ of the change in orientation. A 3rd-degree polynomial is used to interpolate between these 4 points.

A variable tool orientation in a block that is shortened owing to an inside corner is also treated in the same way as described in Subsection 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 inside corner, the machining operation is aborted with an alarm. No block division takes place at the singular points since the compensatory motions this would involve frequently cause contour violations and the change in machining side on the tool is not generally intended or even foreseen by the user. The alarm is also output during examination of an inside corner if the singularity occurs in the second of the two blocks without the transition to the next block being considered. The system does not therefore detect that a block of this type will form an inside corner in conjunction with the following block and that the singularity would be eliminated again by the second block reduction.

The surface normal vector $\mathbf{n}_F$ 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 inside corner must contain contact points. There is no evaluation of several traversing blocks (i.e. no bottleneck detection), CDON/CDOF are not evaluated. If no contact point can be found, the machining operation is aborted with an alarm (risk of collision).

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 inside corner. Except for such special cases, the user is responsible for ensuring that only tools that can machine along the entire contour without violating it are used.

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 behavior is always NORM. Activation must take place in a linear block.

Example:

```
N10 A0 B0 X0 Y0 Z0 F5000
N20 T1 D1                    ;Radius=5
N30 TRAORI(1)                ;Selection of transformation
N40 CUT3DC                   ;Selection of 3D TRC (peripheral milling)
N50 G42 X10 Y10              ;Selection of TRC
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)               ;Selection of transformation
N40 CUT3DC                  ;Selection of 3D TRC
N50 G42 X10 Y10             ;Selection of TRC
N60 X60
N70 G40 X100 Y0 Z20         ;Deselection of 3D TRC
N80 ...
```

Note

If D0 is programmed when tool radius compensation is active, there is no deselection.
If no geometry axis for the current plane is programmed in the block with the deselection, no deselection takes place.

■

Notes

3 Supplementary Conditions

Availability of the "3D tool radius compensation" function

The function is an option and is available for

- SINUMERIK 840D with NCU 572/573,
  SW 3.1 (peripheral milling) and
  SW 3.2 (face milling) and higher

■

4 Data Descriptions (MD, SD)

4.1 Channelspecific machine data

21080 MD number	**CUTCOM_PARALLEL_ORI_LIMIT** Limit angle between path tangent and tool orientation for 3D tool radius compensation	
Default setting: 3	Minimum input limit: 1.0	Maximum input limit: 89
Changes effective after RESET	Protection level: 2/7	Unit: Degrees
Data type: DOUBLE	Applies as of 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). Generally speaking, the lower the value entered in this machine data, the greater the computing capacity required to check that the above conditions have been met. Linear blocks with constant orientation are an exception.	

21082 MD number	CUTCOM_PLANE_ORI_LIMIT Minimum angle between surface normal and tool orientation with side angle not equal to 0.	
Default setting: 3	Minimum input limit: 1.0	Maximum input limit: 89.0
Changes effective after RESET	Protection level: 2/7	Unit: Degrees
Data type: DOUBLE	Applies as of 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. Otherwise, if this value is undershot, machining is interrupted by an alarm. Generally speaking, the lower the value entered in this machine data, the greater the computing capacity required to check that the above condition has been met. The machine data has no effect on 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 MD number	CUTCOM_PLANE_PATH_LIMIT Minimum angle between surface normal vector and path tangent vector, for 3D face milling	
Default setting: 3	Minimum input limit: 1.0	Maximum input limit: 89.0
Changes effective after RESET	Protection level: 2/7	Unit: Degrees
Data type: DOUBLE	Applies as of 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. Otherwise, if this value is undershot, machining is interrupted by an alarm. Generally speaking, the lower the value entered in this machine data, the greater the computing capacity required to check that the above condition has been met.	

■

5 Signal Descriptions

None

6 Example

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 mill
                                     ; ing, changes in orientation at out
                                     ; side corners constant, insertion
                                     ; depth 10mm N40
ORIWKS A30 B15                       ; Change in orientation at one corner
N50 Y20 A3=1 C3=1                    ; Traversing by specifying axis positions
                                     ; block with change in orientation
                                     ; Specify 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 data
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 cutter)
N30 $TC_DP3[1,1]=20.                 ; Length compensation
N40 $TC_DP6[1,1]=5.                  ; Radius
N50 $TC_DP7[1,1]=3.                  ; Rounding radius
N60
N70
N80 X0 Y0 Z0 A0 B0 C0 G17 T1 D1 F12000  ; Selection of tool
```

```
N90 TRAORI(1)
N100 B4=–1 C4=1                           ; Definition of plane
N110 G41 ORID CUT3DF G64 X10 Y0 Z0        ; Activate tool compensation
N120 X30
N130 Y20 A4=1 C4=1                        ; Outside corner, redefine plane
N140 B3=1 C3=5                            ; Change in orientation with ORID
N150 B3=1 C3=1                            ; Change in orientation with ORID
N160 X–10 A5=1 C5=2 ORIC
N170 A3=–2 C3=1                           ; Change in orientation with ORIC
N180 A3=–1 C3=1                           ; Change in orientation with ORIC
N190 Y–10 A4=–1 C4=3                      ; Redefinition of plane
N200 X–20 Y–20 Z10                        ; Inside corner with previous block
N210 X–30 Y10 A4=1 C4=1                   ; Inside corner, redefinition of plane
N220 A3=1 B3=0.5 C3=1.7                   ; Change in orientation with ORIC
N230 X–20 Y30 A4=1 B4=–2 C4=3 ORID
N240 A3 = 0.5 B3=–0.5 C3=1                ; Change in orientation
N250 X0 Y30 C4=1                          ; Path motion, new level, orientation
                                          ; with relative programming
N260 BSPLINE X20 Z15                      ; Start of spline, relative programming
N270 X30 Y25 Z18                          ; of 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, plane not constant
N330 X–20 G40                             ; Deactivation of tool compensation
N340 M30
```

■

Data Fields, Lists 7

7.1 Machine data

Number	Identifier	Name	Reference
General ($MN_ ...)			
18094	MM_NUM_CC_TDA_PARAM	Number of TDA data	/FBW/ /S7/
18096	MM_NUM_CC_TOA_PARAM	Number of TOA data, which can be set up per tool and evaluated by the CC	/FBW/ /S7/
18100	MM_NUM_CUTTING_EDGES_IN_TOA	Tool offsets per TOA module	S7
18110	MM_NUM_TOA_MODULES	Number of TOA modules	S7
Channelspecific ($MC_ ...)			
20110	RESET_MODE_MASK	Definition of control basic setting after powerup and RESET / part program end	K2
20120	TOOL_RESET_VALUE	Definition of tool for which tool length compensation is selected during powerup or on reset or parts program end as a function of MD 20110	K2
20130	CUTTING_EDGE_RESET_VALUE	Definition of tool cutting edge for which tool length compensation is selected during powerup or on reset or parts program end as a function of MD 20110	K2
20140	TRAFO_RESET_VALUE	Definition of transformation block which is selected during powerup and or RESET or parts program end as a function of MD 20110	K2
20210	CUTCOM_CORNER_LIMIT	Max. angle for intersection calculation with tool radius compensation	W1
20220	CUTCOM_MAX_DISC	Maximum value for tool radius compensation	W1
20230	CUTCOM_CURVE_INSERT_LIMIT	Minimum value for intersection calculation with tool radius compensation	W1
20240	CUTCOM_MAXNUM_CHECK_BLOCKS	Blocks for predictive contour calculation with tool radius compensation	W1
20250	CUTCOM_MAXNUM_DUMMY_BLOCKS	Max. no. of dummy blocks with no traversing movements	W1
20270	CUTTING_EDGE_DEFAULT	Selected cutting edge after tool change	W1
20610	ADD_MOVE_ACCEL_RESERVE	Acceleration reserve for overlaid movements	K1
21080	CUTCOM_PARALLEL_ORI_LIMIT	Limit angle between path tangent and tool orientation with 3D tool radius compensation	

Channelspecific ($MC_ ...)			
21082	CUTCOM_PLANE_ORI_LIMIT	Minimum angle between surface normal and tool orientation with side angle not equal to 0.	
21084	CUTCOM_PLANE_PATH_LIMIT	Minimum angle between surface normal vector and path tangent vector, for 3D face milling	
22550	TOOL_CHANGE_MODE	New tool offsets with M function	W1
22560	TOOL_CHANGE_M_CODE	M function for tool change	W1

7.2 Alarms

Detailed explanations of the alarms, which may occur, appear in

References: /DA/, Diagnostics Guide

or in the Online help.

■

Index

Cross-reference? Indicates the following
Part of Description of Functions / Manual / Section / Subsection / Page

Numbers

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

■

Notes

To
SIEMENS AG
A&D MC BMS
Postfach 3180

91050 ERLANGEN, GERMANY

Phone: +49 (0) 180 / 5050 – 222 [Hotline]
Fax: +49 (0) 9131 / 98 – 63315 [Documentation]
E–mail: mailto:motioncontrol.docu@siemens.de

Suggestions

Corrections

For Publication/Manual:

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

Manufacturer/Service Documentation

Description of Functions

Order No.: 6FC5 397-2BP10-0BA0
Edition: 08.05

From

Name

Company/Department

Street

Zip: Town:

Phone: /

Fax: /

Should you come across any printing errors when reading this publication, please notify us on this sheet.
Suggestions for improvement are also welcome.

Suggestions and/or corrections

Overview of SINUMERIK 840D sl Documentation (08/2005)

General Documentation

- *SINUMERIK* — Brochure
- *SINUMERIK* 840D sl — Catalog NC 61 ***)**
- *SINAMICS* S120 — Catalog D21.2 Servo Control ***)**

User Documentation

- *SINUMERIK* 840D sl — Operator's Guide
  - **HMI Embedded *)**
  - **ShopMill**
  - **ShopTurn**
- *SINUMERIK* 840D sl, 840D, 840Di, 810D — Operator's Guide
  - **HMI Advanced *)**
  - Programming compact
- *SINUMERIK* 840D sl, 840D, 840Di, 810D — **Programming Guide**
  - Fundamentals ***)**
  - Advanced ***)**
  - Programming
  - Lists System Variables
  - ISO Turning/Milling
- *SINUMERIK* 840D sl, 840D, 840Di, 810D — Programming Guide
  - **Cycles**
  - **Measuring Cycles**
- *SINUMERIK* 840D sl, 840D, 840Di, 810D — **Diagnostics Guide *)**

Manufacturer/Service Documentation

- *SINUMERIK* 840D sl — Equipment Manual **NCU *)**
- *SINUMERIK* 840D sl — Equipment Manual **Operator Components *)**
- *SINUMERIK* 840D sl — **Start-up Guide** CNC ***)**
  - Part 1 NCK, PLC, Drive
  - Part 2 HMI
  - Part 3 ShopMill
  - Part 4 ShopTurn
  - Part 5 Basic Software
- *SINUMERIK* 840D sl, 840D, 840Di, 810D — **List Manual ***
  - Part 1
  - Part 2

Manufacturer/Service Documentation

- *SINUMERIK* 840D sl, 840D, 840Di, 810D — **Description of Functions**
  - Basic Machine ***)**
  - Extended Functions
  - Special Functions
- *SINUMERIK* 840D sl, 840D, 840Di, 810D — Description of Functions
  - **Synchronized Actions**
  - **Iso Dialects**
- *SINAMICS* S120 — Description of Functions **Drive Functions**
- *SINUMERIK* 840D sl — Description of Functions **Saftey Integrated**
- *SINUMERIK* 840D sl, 840D, 840Di, 810D — **EMC Guidlines**

Electronic Documentation

- *SINUMERIK SINAMICS Motors* — DOCONCD ***)**, DOCONWEB

***)** These documents are a minimum requirement

Overview of SINUMERIK 840D/840Di/810D Documentation (08/2005)

General Documentation

- *SINUMERIK* 840D/810D — Brochure
- *SINUMERIK* 840D/840Di/810D/ — Catalog Ordering Info. NC 60 ***)**
- *Safety Integrated* Application Manual — **Safety Integrated** Application Manual

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 — **Programming Guide**
  – Short Guide
  – Fundamentals ***)**
  – Advanced ***)**
  – Cycles
  – Measuring Cycles
  – ISO Turning/Milling
  – Lists **System Variables**
- *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/840Di/810D — Description of Functions
  – ManualTurn
  – ShopMill
  – ShopTurn

Manufacturer/Service Documentation

- *SINUMERIK SIMODRIVE* 611D 840D/810D — Description of Functions **Synchronized Actions**
- *SINUMERIK* 840D/840Di/810D — Description of Functions **Drive Functions *)**
- *SINUMERIK* 840D/840Di/810D — **Description of Functions**
  – Basic Machine ***)**
  – Extended Functions
  – Special Functions
- *SINUMERIK* 840D/810D — Description of Functions **Tool Management**
- *SINUMERIK* 840D/810D — **Configuring Kit** HMI Embedded
- *SINUMERIK* 840D/810D — Description of Functions Operator Interface **OP 030**
- *SINUMERIK* 840D/810D — **MCIS**
  – Computer Link
  – Tool Data Inform. Syst.
  – 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
- *MOTION CONTROL SYSTEMS* — **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