SIEMENS

SINUMERIK 840D/840Di
SINUMERIK 810D/FM-NC

Description of Functions **04.2000 Edition**

Extension Functions
(Part 2)

Manufacturer/Service Documentation

Overview of SINUMERIK 840D/840Di/810D/FM-NC Documentation (04.00)

General Documentation

- *SINUMERIK* 840D/810D/FM-NC — Brochure
- *SINUMERIK* 840D/840Di/810D/FM-NC/611 — Catalog Ordering Info NC 60.1 ***)** Technical Info. NC 60.2
- *SINUMERIK SIROTEC SIMODRIVE* Accessories — Catalog Accessories NC-Z

User Documentation

- *SINUMERIK* 840D/810D/FM-NC — **AutoTurn**
  - Short Guide
  - Programming (1)
  - Setup (2)
- *SINUMERIK* 840D/810D — Operator's Guide
  - **Unit Operator Panel**
  - **HPU**
  - **HT 6**
- *SINUMERIK* 840D/840Di/810D/FM-NC — **Diagnostics Guide *)**
- *SINUMERIK* 840D/810D/FM-NC — **Operator's Guide**
  - Short Guide
  - Operator's Guide ***)**

User Documentation

- *SINUMERIK* 840D/840Di/810D/FM-NC — **Program. Guide**
  - Short Guide
  - Fundamentals ***)**
  - Advanced ***)**
  - Cycles
  - Measuring Cycles
- *SINUMERIK* 840D/810D — Operator's Guide
  - **ManualTurn**
  - Short Guide ManualTurn
  - **ShopMill**
  - Short Guide ShopMill
- *SINUMERIK* 840Di — **System Overview**

Manufacturer / Service Documentation

- *SINUMERIK* 840D/810D — Description of Functions
  - **ManualTurn**
  - **ShopMill**
- *SINUMERIK* 840D/840Di/810D — Description of Functions **Synchronized Actions** Wood, Glass, Ceramics
- *SINUMERIK* 840D/810D — Descr. of Functions
  - **Computer Link**
  - **Tool Data Information System**
- *SINUMERIK* — **Configuring** (HW) ***)**
  - FM-NC
  - 810D
  - 840D

Manufacturer / Service Documentation

- *SINUMERIK* 840D/840Di/810D/FM-NC — **Operator Components** (HW) ***)**
- *SINUMERIK SIMODRIVE* 611D 840D/810D — Description of Functions **Drive Functions *)**
- *SINUMERIK* 840D/840Di/810D/FM-NC — **Description of Functions**
  - Basic Machine ***)**
  - Extended Functions
  - Special Functions
- *SINUMERIK* 840D/810D/FM-NC — **Configuring Kit** MMC 100/101
  - Configuring Syntax
  - Development Kit
- *SINUMERIK* 840D/810D/FM-NC — **Screen Kit** MMC 100/101 SW Update and Configuration
- *SINUMERIK* 840D/810D — Description of Functions **Tool Management**
- *SINUMERIK* 840D/810D/FM-NC — Description of Functions Operator Interface **OP 030**

Manufacturer / Service Documentation

- *SINUMERIK SIMODRIVE* — Description of Functions SINUMERIK **Safety Integrated**
- *SINUMERIK* 840D — Description of Functions **Digitizing**
- *SINUMERIK SIMODRIVE* — **Installation & Start-up Guide *)**
  - FM-NC
  - 810D
  - 840D/611D
  - MMC
- *SINUMERIK SIMODRIVE* 840D/840Di/810D FM-NC 611D — **Lists *)**
- *SINUMERIK SIMODRIVE* 840D 611D — Description of Functions **Linear Motor**
- *SINUMERIK SIMODRIVE* 840D 611D — Description of Functions
  - **Hydraulics Module**
  - **Analog Module**
- *SINUMERIK SIMODRIVE SIROTEC* — **EMC** Guidelines

Electronic Documentation

- *SINUMERIK SIMODRIVE* 840D/810D/FM-NC 611, Motors — **DOC ON CD *)** The SINUMERIK System

Manufacturer / Service Documentation

- *SINUMERIK* 840D/810D — Descr. of Functions **ISO Dialects** for SINUMERIK
- *SINUMERIK* — Descr. of Functions **CAM Integration** DNC NT-2000
- *SINUMERIK* 840Di — **Manual** (HW + Installation and Start-up)

***)** These documents are a minimum requirement for the control

SIEMENS

SINUMERIK 840D/840Di/ SINUMERIK 810D/FM-NC

Extension Functions (Part 2)

Description of Functions

Valid for

Control	*Software version*
SINUMERIK 840D	5
SINUMERIK 840Di	5
SINUMERIK 840DE (export version)	5
SINUMERIK 810D	3
SINUMERIK 810DE (export version)	3
SINUMERIK FM-NC	3

04.00 Edition

SINUMERIK® Documentation

Printing history

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

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

Status code in the “Remarks” column:

A . . . . . New documentation.
B . . . . . Unrevised reprint with new Order No.
C . . . . . Revised edition with new status.
If factual changes have been made on the page since the last edition, this is indicated by a new edition coding in the header on that page.

Edtition	Order No.	Remarks
06.94	6FC5 297-0AC30-0BP0	**A**
08.94	6FC5 297-0AC30-0BP1	**C**
02.95	6FC5 297-2AC30-0BP0	**C**
04.95	6FC5 297-2AC30-0BP1	**C**
03.96	6FC5 297-3AC30-0BP0	**C**
08.97	6FC5 297-4AC30-0BP0	**C**
12.97	6FC5 297-4AC30-0BP1	**C**
12.98	6FC5 297-5AC30-0BP0	**C**
08.99	6FC5 297-5AC30-0BP1	**C**
04.00	6FC5 297-5AC30-0BP2	**C**

This manual is included in the documentation on CD-ROM (**DOCONCD**)

Edition	Order No.	Remarks
04.00	6FC5 298-5CA00-0BG2	**C**

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

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

This publication was produced with Interleaf V 7

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.

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

Subject to changes without prior notice.

Order No. 6FC5 297-5AC30-0BP2
Printed in the Federal Republic of Germany

Siemens Aktiengesellschaft

Preface

Reader information

The SINUMERIK documentation is divided into 4 different levels:

- General Documentation
- User Documentation
- Manufacturer/Service Documentation
- OEM Documentation

This documentation is intended for the manufacturers of machine tools. It provides a detailed description of the functionality of the SINUMERIK 840D and SINUMERIK FM-NC control systems.

The Descriptions of Functions only apply to the software versions specified. When a new software version is published, the Description of Functions for that software should be ordered. Old Descriptions of Functions are not necessarily applicable to new software versions.

Please consult your local Siemens office for more detailed information about other SINUMERIK 840D and SINUMERIK FM-NC publications as well as the publications that apply to all SINUMERIK controls (e. g. Universal Interface, Measuring Cycles, ...).

Note

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

Aim

The Description of Functions provides the information required for configuration and installation/start-up.

Target groups

The Description of Functions therefore contains information for:

- The configuring engineer
- The PLC programmer who writes the PLC user program by providing signal lists
- The system start-up engineer after configuration and installation of the machinery and equipment
- The service technician for checking and interpreting the status displays and alarms

Note on using the manual

This manual is structured as follows:

- General list of contents of the manual
- Functional descriptions in alphanumeric order according to the codes used for the Description of Functions
- Appendix with general index

Note

The Description of Functions **Basic Machine** (Part 1) contains both a general index as well as a reference list, a lexicon and an index of abbreviations.

Pages indicated provide the following information:
Part of the Description of Functions / Book / Chapter / Section - Page

If you require information on a certain function, you will find the function as well as the code under which the function is organized on the inside title page of the manual.

If you only require information on a certain term please refer to the index in the Appendix. There you will find the code of the Description of Functions, the Chapter/Section number as well as the page number on which the information about this term is to be found.

Within each of the Description of Functions in Chapters 4 and 5 you will find definitions on effect, data format, input limits etc. for the various signals and data definitions.
These definitions are explained in the Technical Comments section below.

!

Important

This documentation applies to:

- SINUMERIK 840D or SINUMERIK FM-NC control, software version 4
- SINUMERIK 810D control, software version 2

Specification of the software version

The software versions specified in this documentation refer to the SINUMERIK 840D control; the parallel software version for the SINUMERIK 810D control (if the function is released, see /BU/, Catalog NC 60.1) is not specified explicitly. The following applies:

Table 1-1 Equivalent software versions

SINUMERIK 840D software version		SINUMERIK 810D software version
4.3 (12.97)	corresponds to	2.3 (12.97)
3.7 (03.97)	corresponds to	1.7 (03.97)

Explanations of the symbols

Important

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

Order data option

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

Machine manufacturer

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

Danger

This symbol appears whenever death, severe physical injury or substantial material damage **will** occur if the appropriate precautions are not taken.

Caution

This symbol appears whenever minor physical injury or material damage **may** occur if the appropriate precautions are not taken.

Warning

This symbol appears whenever death, severe physical injury or substantial material damage **may** occur if the appropriate precautions are not taken.

Technical Comments

Notation

The following notation and abbreviations are used in this documentation:

- 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 “Feed/spindle speed override” (DB31-48, DBB0), i.e. the signals are stored for specific axes/spindles in data blocks 31 to 48, data block byte 0.
- Machine data –> MD: MD_NAME
- Setting data –> SD: SD_NAME
- The character “≙” means “corresponds to”

Explanation of the abbreviations used in Chapt. 4 and 5

In Chapters 4 and 5 of the Description of Functions you will find a description of the data and signals that bear relevance to the function concerned. Terms and abbreviations used in these tabular descriptions are explained here.

Values in the table

The machine data indicated in the Descriptions of Functions are always values for an NCU572.

For the values of the other NCUs (e.g. NCU570, NCU571, NCU573), please refer to the “Lists” documentation.

References: /LIS/ “Lists”

Default value

The machine/setting data is preset with this value during installation. If these default values are different for each channel, this is marked by an “/”.

Value range (minimum and maximum value)

States the input limits. If no value range is specified, the input limits are dependent on the data type and “∗∗∗” is displayed next to the field.

Activation

When machine data, setting data etc. are altered they are not immediately active. Information about activation of alterations is therefore always stated. The following is a list of the possible activation conditions in order of priority:

- POWER ON (po) “RESET” key on the front panel of the NC module or by switching the power off/on
- NEW_CONF (cf) – “Re-configuring” of PLC interface function – “RESET” key on operator panel, or
- RESET (re) “RESET” key on operator panel, or
- Immediately (im) After input of value

Protection levels

Protection levels 0–7 are available, the protection of levels 0 to 3 (4 to 7) can be removed by setting a password (keyswitch setting). The user only has access to information for a certain protection level or for the lower protection levels. The machine data are assigned different protection levels as standard.

The table lists write protection levels only because read protection levels are derived from the write protection levels:

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

References: /BA/, “Operator’s Guide”
/FB/, A2, “Various Interface Signals”

Unit

The unit refers to the machine data default setting SCALING_FACTOR_USER_DEF_MASK and SCALING_FACTOR_USER_DEF.
If the MD is not based on a physical unit a “–” is entered.

Data type

The following data types are used in the control:

- DOUBLE
  Real or integer values (values with decimal point or integers)
  Inputs limits from $+/-4.19*10^{-307}$ to $+/-1.67*10^{308}$
- DWORD
  Integer values
  Input limits from $-2.147*10^{9}$ to $+2.147*10^{9}$
- BOOLEAN
  Possible input values: true or false or 0 or 1
- BYTE
  Integer values from –128 to +127
- STRING
  Consisting of max. 16 ASCII characters (capital letters, digits and underscore)

Data management

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

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

For the number of components actually implemented for each of the software versions, please consult
References: /BU/, “Ordering Information” or /FB/, K1, “Mode-Groups, Channels, Program Operation Mode”

■

Notes

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

Digital and Analog NCK I/Os (A4)

■

Notes

Brief Description

1

General remarks

Signals can be read and output in the interpolation cycle via the “digital and analog NCK I/Os”. The following functions can be executed with these signals, for example:

- Several feed values in one block
- Several auxiliary functions in a block
- Rapid retraction on finished contour
- Axis-specific delete distance-to-go
- Program branching
- Rapid NC start
- Analog calipers
- Position switching signals
- Punching/nibbling functions
- Analog value control
- etc.

Contents

The specific features and operating characteristics of the digital and analog I/Os are described in this document.
A reference to further relevant documentation (“References”) is given with respect to every function that uses the I/Os.

■

Notes

Detailed Description 2

2.1 General functionality

General

The ability to control or influence time-critical NC functions is dependent on high-speed NCK I/O interfaces or the facility to rapidly address particular PLC I/Os (see Section 2.5).

For this reason, the SINUMERIK 840D, 840Di and FM-NC systems are designed to permit

a) the use of digital and analog NCK inputs and outputs (see Chapter 3).

b) direct addressing of particular PLC I/Os (see Section 2.5).

The hardware inputs and outputs can be read and written via system variables in the part program or synchronized actions.

Via the PLC interface, both the signal states of the digital I/Os and the values of the external analog I/Os can be changed by the PLC user program according to the application.

840D hardware

On the SINUMERIK 840D **on-board NCU** there are 4 digital NCK inputs (inputs 1 to 4) and 4 digital NCK outputs (outputs 1 to 4).

The digital on-board inputs and outputs are stored in the first address byte. With the NCK outputs, the remaining signals of this byte (NCK outputs 5 to 8) can be used via the PLC interface (digital NCK outputs without hardware).

Using the “NCU terminal block” that can be coupled to the drive bus, it is possible to connect further digital NCK inputs/outputs and analog NCK inputs/outputs (hereafter called **external NCK I/Os)**. The “NCU terminal block” is used as a carrier module for up to eight DP compact plug-in modules. Up to two “NCU terminal blocks” can be connected per NCU.

The maximum degree of expansion of the external NCK I/Os is:

- 32 digital NCK inputs (digital inputs 9 to 40)
- 32 digital NCK outputs (digital outputs 9 to 40)
- 8 analog NCK inputs (analog inputs 1 to 8)
- 8 analog NCK outputs (analog outputs 1 to 8)

For further information about the hardware specification see:

References: /PHD/, SINUMERIK 840D NCU Manual

FM-NC hardware

The SINUMERIK FM-NC has no digital on-board inputs or outputs.

The I/O signals of the 1st address byte (NCK inputs 1 to 8 or NCK outputs 1 to 8) can be used via the PLC interface (digital NCK inputs/outputs without hardware).

Connection of external NCK I/Os:

Using the local P bus, it is possible to connect I/O modules of the standard I/Os of the AS300.

If FM servo is used as the 5th axis on the local P bus, it is possible to use the four digital inputs and outputs on the module as external NCK I/Os. The four remaining signals of this byte can be used via the PLC interface (digital NCK inputs/outputs without hardware).

For further information about the hardware specification see:

References: /PHF/, SINUMERIK FM-NC, NCU Manual

Hardware 840 Di digital I/Os

Digital inputs/outputs are provided for the SINUMERIK 840Di via the MCI Board Extension module. The following connections are available:

- 2 handwheels
- 2 sensors
- 4 digital inputs/outputs

Note

The MCI Board Extension module is an option for the SINUMERIK 840Di. The PIN assignment of the cable distribution interface (X121) matches the cable distributor assignment on the SINUMERIK 840D.

SINUMERIK 840Di analog and digital inputs/outputs

Analog and digital inputs/outputs can be operated on the SINUMERIK 840Di by means of SIMATIC S7 bus interface and signal boards linked via the PROFIBUS-DP.

PLC I/Os directly addressable from NCK in SW 5.2

Up to 16 bytes for digital input signals and analog input values plus a total of 16 bytes for digital output signals and analog output values can be addressed directly by the part program. These bytes must be taken into account when the PLC is configured. They must be programmed consecutively. They are processed directly by the PLC operating system. As a result, the signal transfer time between the NC and PLC I/O modules is of a magnitude of 0.5 ms.

Caution

The output bytes specified for the NCK may not be write-accessed by the PLC user program as the access operations between the NCK and PLC would be uncoordinated.

For further details, see 2.5.

Comparator inputs

In addition to the digital and analog NCK inputs, 16 internal comparator inputs (comparator input bytes 1 and 2) are also available.

The signal state of a comparator input is formed by comparing an analog input signal with a threshold value in a setting data.

See Section 2.6 for additional information.

Number

The number of addressable digital NCK input/output bytes and analog inputs/outputs must be programmed by means of general machine data.

Machine data ($MN_ ...)	Number of active ...	Max. number
FASTIO_DIG_NUM_INPUTS	Digital NCK input **bytes**	5
FASTIO_DIG_NUM_OUTPUTS	Digital NCK output **bytes**	5
FASTIO_ANA_NUM_INPUTS	Analog NCK inputs	8
FASTIO_ANA_NUM_OUTPUTS	Analog NCK outputs	8

Note

By definition, the 1st byte is always assigned to the 4 digital I/Os on the MCI Board Extension module on the SINUMERIK 840Di. Even if you have not connected an MCI Board Extension module to the SINUMERIK 840Di, the 1st byte is always assigned to this module.
For this reason, you must enter at least 2 bytes in machine data FASTIO_DIG_NUM... if you want to operate further I/O devices via the PROFIBUS.

Corresponding alarms are generated if the part program addresses inputs/outputs that have not been defined in the above machine data.

These NCK inputs or outputs do not have to actually exist in the hardware. If they do not, the signal states or the binary analog values are set to “zero” in a defined way inside the NCK. The values can be changed by the PLC.

Hardware assignment of external NCK I/Os

The following general machine data ($MN-) are provided for assigning I/O signal modules or I/O modules to external NCK I/Os :

- MD 10366: HW_ASSIGN_DIG_FASTIN[hw] Hardware assignment for external digital inputs
- MD 10368: HW_ASSIGN_DIG_FASTOUT[hw] Hardware assignment for external digital outputs
- MD10362: HW_ASSIGN_ANA_FASTIN[hw] Hardware assignment for external analog inputs
- MD10364: HW_ASSIGN_ANA_FASTOUT[hw] Hardware assignment for external analog outputs

[hw]: Index for addressing the external digital I/O bytes (0 to 3) or the external analog inputs/outputs (0 to 7)

Note

The HW assignment is different on the SINUMERIK 840D, 840Di and FM-NC controls.

The defaults for the **assignment** of I/Os for the SINUMERIK 840Di via machine data MD 10362 to MD 10368 are as follows:

Machine data ($MN_ ...)	Meaning	default
HW_ASSIGN_ANA_FASTIN[0] ...	Assignment for analog input (16-bit access)	050000A0
HW_ASSIGN_ANA_FASTOUT[0] ...	Assignment for analog output (16-bit access)	050000A0
HW_ASSIGN_DIG_FASTIN[0] ...	Assignment for digital input (8-bit access)	05000090
HW_ASSIGN_DIG_FASTOUT[0] ...	Assignment for digital output (8-bit access)	05000090

Modification to MD for PROFIBUS-DP

The machine data $MN_HW_ASSIGN_... have been modified for hardware operation on the PROFIBUS-DP of the SINUMERIK 840Di.

The assignment of bytes 1 to 4 has been redefined. The machine data assignments below apply for PROFIBUS-DP operation:

Byte	New for PROFIBUS-DP	Previous meaning
4th byte	Segment number = 5	Segment number
3rd byte	Not used = 0	Module number
2nd byte	Logical high address	Submodule number
1st byte	Logical low address	Input/output number

Guidelines for machine data $MN_HW_ASSIGN_...:

- Logical address in 1st and 2nd bytes is specified in hexadecimal format. Example: 050001A2 (Hex) equals logical address 418 (Dec).
- Address 0 is reserved for the PLC and cannot be used as an NC I/O.
- The value 05000000 in MD $MN_HW_ASSIGN_... is interpreted as “Slot does not physically exist”. The input is then treated like a simulation input.

System variables

The following table lists the system variables with which NCK I/Os can be read or written directly by the part program.

The number of the NCK input/output is used for addressing.

The following applies to n:
$1 \leq n \leq 8$ * MD 10350: FASTIO_DIG_NUM_INPUTS
$1 \leq n \leq 8$ * MD 10360: FASTIO_DIG_NUM_OUTPUTS

$1 \leq n \leq$ MD 10300: FASTIO_ANA_NUM_INPUTS
$1 \leq n \leq$ MD 10310: FASTIO_ANA_NUM_OUTPUTS

System variable	Meaning	Range of [n]
$A_IN[n]	Read digital NCK input [n]	1 to 3, 9 to 40
$A_INA[n]	Read analog NCK input [n]	1 to 8
$A_INCO[n]	Read comparator input [n]	1 to 16
PBB		
$A_OUT[n]	Read/write digital NCK output [n]	1 to 40
$A_OUTA[n]	Read/write analog NCK output [n]	1 to 8

Note

When this system variable is read by the part program, a preprocess stop (STOPRE command) is initiated inside the control.

Value ranges

$A_PBB_OUT(n) 0<= x <= 255
$A_PBW_OUT(n) –32768 <=x <=32767

Weighting factor

The weighting factors in the general machine data MD 10320: FASTIO_ANA_INPUT_WEIGHT[hw] and MD 10330: FASTIO_ANA_OUTPUT_WEIGHT[hw] allow each individual analog NCK input and output to be adapted to the AD or DA converters of the analog I/O module used.

If the correct weighting factor is set, the value set in system variable $A_OUTA[n] outputs the corresponding voltage value in millivolts at the analog output [n].

Example for 840D

The analog value range is 10V (maximum modulation);
FASTIO_ANA_OUTPUT_WEIGHT[hw] = 10000 (standard value for 840D)
$A_OUTA[1] = 9500 ; 9.5 V is output at analog NCK output 1
$A_OUTA[3] = –4120 ; –4.12 V is output at analog NCK output 3

Application for analog NCK inputs/outputs without hardware:

With weighting factor of 32767, the digitized analog values for part program and PLC accesses are identical. In this way, it is possible to use the associated input or output word for a 1:1 communication between the part program and the PLC.

Assignment to NC functions

Several NC functions are dependent on the functionality of the NCK I/Os. The NCK inputs and/or outputs used for these functions are assigned on a function-specific basis via machine data (e.g. MD 21220: MULTFEED_ASSIGN_FASTIN for “Multiple feedrates in one block”). A byte address must be specified in the machine data for the digital inputs/outputs; the assignment is always made byte by byte.

Byte address	Assignment for the digital NCK inputs/outputs			
0	None			
840D: 1	1 to 4 (on-board I/Os)	and	5 to 8	(NCK-A without hardware)
FM-NC: 1	1	to	8	(NCK-A without hardware)
2	9	to	16	(external NCK I/Os)
3	17	to	24	(external NCK I/Os)
4	25	to	32	(external NCK I/Os)
5	33	to	40	(external NCK I/Os)
128	Inputs 1 to 8 of comparator byte 1 (see Section 2.6)			
129	Inputs 9 to 16 of comparator byte 2 (see Section 2.6)			

Clock-synchronous processing

The I/O modules of the external NCK I/Os on the SINUMERIK 840D can be operated in one of the following two modes:

- **Asynchronously**, i.e. the input and output values are made available in cycles set by the terminal block which are asynchronous to the internal NC processing cycles.
- **Synchronously**, i.e. the input values and the output values are provided synchronously with the settable internal NC processing clock frequency.

The processing mode is selected for individual modules by means of general machine data MD 10384: HW_CLOCKED_MODULE_MASK[tb].

[tb] = Index for terminal block (0 to 1)

In synchronous processing mode, one of the following clock rates can be selected (general MD 10380: HW_UPDATE_RATE_FASTIO[tb]):

- Synchronous inputs/outputs in position control cycles (default setting)
- Synchronous inputs/outputs in interpolation cycles

A lead time set in microseconds can be defined for clocked NCK I/Os in general machine data MD 10382: HW_LEAD_TIME_FASTIO[tb]. This makes it possible to consider the conversion time of the ADC for example, so that the digitized input value is available on the cycle.

The defined cycle frequency or delay time applies to all cycle-synchronous I/O modules of the terminal block addressed with [tb].

On the SINUMERIK FM-NC, the I/O modules of the external NCK I/Os **always operate asychronously**. They are updated in position control cycles.

Monitoring functions

The following functional monitors are provided for external I/Os on the SINUMERIK 840D:

- During power-on:
  - Check whether the I/O modules in the terminal blocks match the MD assignments
- During cyclic operation:
  - Sign-of-life monitoring in interpolation cycles
  - Module monitoring in interpolation cycles
  - Temperature monitoring

In the event of a fault, NC ready is cancelled and an alarm is output.

With the SINUMERIK FM-NC, the following function monitoring features check the external I/Os:

- Sign-of-life monitoring in interpolation cycles
- Module monitoring in interpolation cycles

In the event of a fault, NC ready is cancelled.

For further information about monitoring functions see:

References: /PHD/, SINUMERIK 840D, NCU Manual
/PHF/, SINUMERIK FM-NC, NCU Manual

Response to faults

The digital and analog NCK outputs are switched to “safe” status (i.e. 0V at output) in the event of faults (e.g. NC ready = 0) in the NCU or power failures.

2.2 Digital inputs/outputs of the NCK

2.2.1 Digital inputs of the NCK

Number

With the general MD 10350: FASTIO_DIG_NUM_INPUTS (number of active digital NCK input bytes) the available digital NCK inputs can be defined (in groups of 8).

Function

The digital NCK inputs allow external signals to be injected which can then be used, for example, to control the workpiece machining program sequence. With the system variable **$A_IN[n]**, the signal status of the digital input [n] can be scanned directly in the part program.

The signal state at the hardware input can be changed by the PLC user program (see Fig. 2-1).

Disable input

The PLC user program can disable NCK inputs individually by means of interface signal “Disable digital NCK inputs” (DB10, DBB0 or DB122 ...). In this case, they are set to “0” in a defined manner inside the control.

Set input from PLC

The PLC can also apply interface signal “Setting digital NCK inputs on PLC” (DB10, DBB1 or DBB123 ...) to set each digital input to a defined “1” signal state (see Fig. 2-1). As soon as this interface signal is set to “1”, the signal state at the hardware input or the input disable is inactive.

Read actual value

The signal status of the digital NCK inputs is signaled to the PLC (interface signal “Actual value of digital NCK inputs” (DB10, DBB60, DBB186 ...)). The actual value reflects the real state of the signal at the hardware input; the influence of the PLC is therefore ignored in the “actual value” (see Fig. 2-1).

RESET/POWER ON behavior

After POWER ON and reset, the signal level at the input is passed on. If necessary, the PLC user program can disable or set the inputs to “1” in a defined manner as described above.

Applications

The program sequence can be controlled with conditional go-to statements in the part program as a function of the signal status of an external hardware signal.

For example, digital NCK inputs can be used for the following NC functions:

- Delete distance-to-go with positioning axes
- Fast program branching at the end of block
- Programmed read-in disable
- Multiple feedrates in one block

References: /FB/, S5, “Synchronized Actions”

The NCK inputs are assigned to the NC functions separately for each function and byte in the machine data. Multiple assignments of inputs are not monitored.

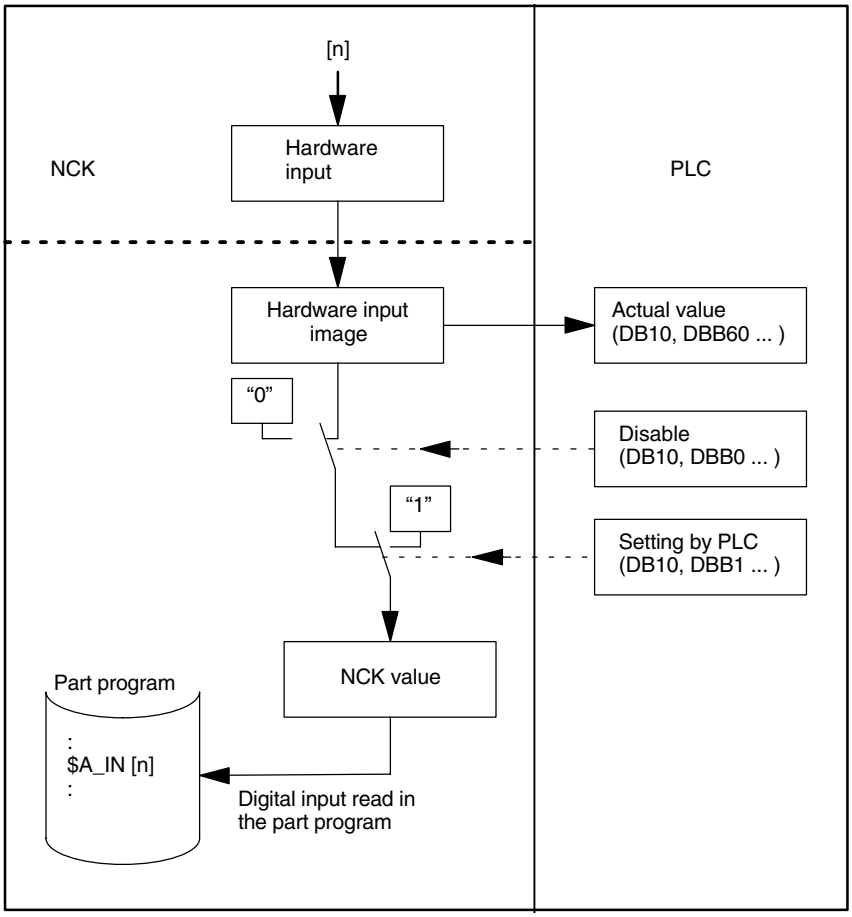


Fig. 2-1 Signal flow for digital NCK inputs

2.2.2 Digital outputs of the NCK

Number

With the general MD 10310: FASTIO_DIG_NUM_OUTPUTS (number of active digital NCK output bytes) the available digital NCK outputs can be defined (in groups of 8).

Function

The digital NCK outputs provide the option of outputting important switching commands at high speed as a function of the program processing status. With the system variable **$A_OUT[n]**, the signal status of the digital output [n] can be set or read again directly in the part program.

There are also several ways of changing this set signal state via the PLC (see Fig. 2-2).

Disable output

The PLC user program is capable of disabling the digital NCK outputs individually with interface signal “Disable digital NCK outputs” (DB10, DBB4, DBB130 ...). In this case, the “0” signal is output at the hardware output (see Fig. 2-2).

Overwrite screen form

Every output that can be set by the NC part program can be **overwritten** from the PLC using the overwrite screen form. Previous NCK values are then lost (see Fig. 2-2).

The following routine has to be carried out to overwrite the NCK value from the PLC:

1. The output in question must be preset with the required signal state at the PLC interface “PLC setting for digital NCK outputs” (DB10, DBB6, DBB132 ...).
2. The setting value becomes the new NCK value for the output (DB10, DBB5, DBB131 ...) when the overwrite screen form is activated (signal transition 0 –> 1). This value remains active until the next time the value is programmed (in the PLC or NC part program).

Setting screen form

Furthermore, a PLC setting for each output can determine whether the instantaneous “NCK value” (e.g. as specified by NC part program) or the PLC value specified via the setting screen form (DB10, DBB7, DBB133 ...) should be sent to the hardware output (see Fig. 2-2).

The following routine has to be carried out to define the PLC value:

1. The output in question must be preset with the required signal state at the PLC interface “PLC setting for digital NCK outputs” (DB10, DBB6).
2. The setting screen form must be set to “1” for the output in question.

Unlike the overwrite screen form, the current NCK value is not lost when a value is set in the setting screen form. As soon as the PLC sets “0” in the setting screen form, the NCK value is again active.

Note

The same setting value (DB10, DBB6) is used at the PLC interface for the overwrite and setting screen forms. Therefore, an identical output signal state is the result if the signal state is changed **simultaneously** in the overwrite **and** setting screen form.

Read setpoint

The instantaneous “NCK value” at the digital outputs can be read by the PLC user program (interface signal “Setpoint of digital NCK outputs” (DB10, DBB64, DBB186 ...)). Please note, however, that this setpoint is unaffected by disabling commands or the setting screen form of the PLC. The setpoint can therefore be different from the actual signal state at the hardware output (see Fig. 2-2).

RESET/end of program

On end of program or RESET, every digital output can be defined as necessary by the PLC user program in the overwrite screen form, setting screen form or disable signal.

POWER ON

After POWER ON, the digital outputs are set to “0” in a defined manner. After POWER ON, this can be overwritten in the PLC user program according to the application using the screen forms described above.

Digital NCK outputs without hardware

If digital NCK outputs defined in MD 10360: FASTIO_DIG_NUM_OUTPUTS are written by the part program but do not exist as hardware, no alarm is output. The NCK value can be read by the PLC (IS “Setpoint ...”)

Applications

This function allows digital hardware outputs to be set instantaneously by bypassing the PLC cycles. Time-critical switching functions can thus be triggered in connection with the machining process and under program control (e.g. on block change).

For example digital NCK outputs are required for the following NC functions:

- Position signals
  References: /FB/, N3, “Software Cams, Position Signals”
- Output of the comparator signals (see Section 2.6)

The NCK outputs are assigned to the NC functions separately for each function in machine data. Multiple assignments of outputs are checked during POWER ON and indicated by an alarm.

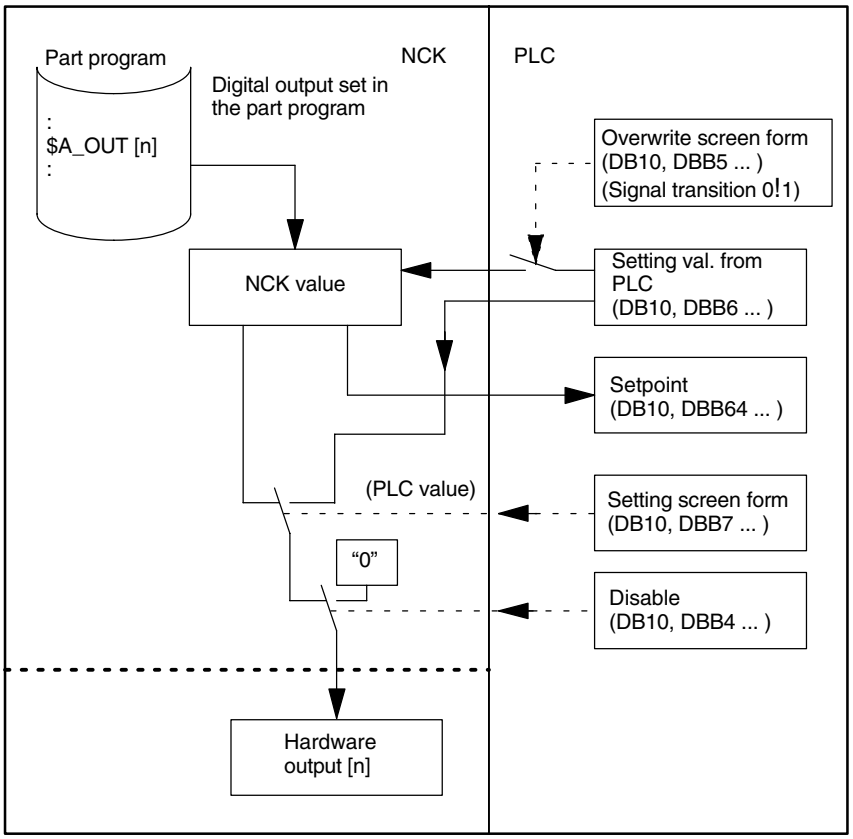


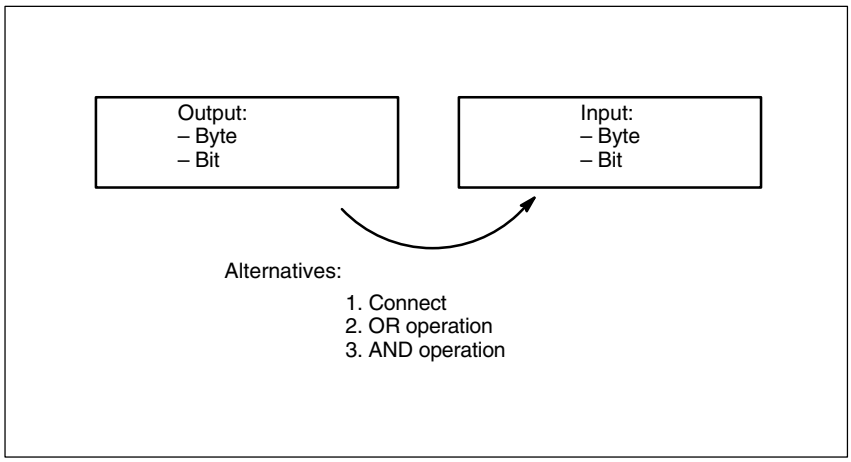
Fig. 2-2 Signal flow for digital NCK outputs

2.3 Connecting and logic operations of fast NCK inputs/outputs

Function

In SW 4 and higher, the fast inputs of the NCK I/Os can be set in the software according to the signal states of the fast outputs.

Overview:



Connect

The fast input of the NCK I/O is set to the signal state of the assigned fast output.

OR operation

The fast input of the NCK I/O takes the signal state which is given by the OR operation of the output signal with the assigned input signal.

AND operation

The fast input of the NCK I/O takes the signal state which is given by ANDing the output signal with the assigned input signal.

Special cases

- If several output bits are assigned to the same input bit, then the one with the highest MD index becomes effective.
- If inputs or outputs are specified which do not exist or are not activated, then the assignment is ignored without alarm. Checking of the active bytes of the NCK I/Os is performed via the entries in the machine data:
  MD 10350: $MN_FASTIO_DIG_NUM_INPUTS and
  MD 10360: $MN_FASTIO_DIG_NUM_OUTPUTS.

Defining assignments

The assignments are specified via machine data:

MD 10361 : $MN_FASTIO_DIG_SHORT_CIRCUIT[n].

Values from 0 to 9 can be specified for n, therefore up to **10** assignments can be specified.

The byte and the bit of the output and input are specified in 2 hexadecimal characters respectively. The **type of logic operation** is specified by entering

0 for connect
A for AND operation
B...for OR operation

in bits 12–15 of the input.

FASTIO_DIG_SHORT_CIRCUIT[n]

	Output		Input		
	Bit	byte	↑ Type of logic op.	Bit	byte
Bit	24–31	16–23	8–15		0–7

Examples

Connect:
$MN_FASTIO_DIG_SHORT_CIRCUIT = ’04010302H’
Output: bit 4, byte 1, connect to
input: bit 3, byte 2

AND operation:
$MN_FASTIO_DIG_SHORT_CIRCUIT = ’0705A201H’
Output: bit 7, byte 5 AND operation with
input: bit 2, byte 1

OR operation:
$MN_FASTIO_DIG_SHORT_CIRCUIT = ’0103B502H’
Output: bit 1, byte 3 OR operation with
input: bit 5, byte 2

2.4 Analog inputs/outputs of the NCK

2.4.1 Analog inputs of the NCK

Number

With the general MD 10300: FASTIO_ANA_NUM_INPUTS (number of active analog NCK inputs) the available analog NCK inputs can be defined.

Function

The system variable **$A_INA[n]** allows the value at the analog NCK input [n] to be directly accessed in the part program.

The analog value at the hardware input can also be influenced by the PLC user program (see Fig. 2-3).

Disable input

The PLC user program is capable of disabling the analog NCK inputs individually with interface signal “Disable analog NCK inputs” (DB10, DBB146). In this case, they are set to “0” in a defined manner inside the control.

Set input from PLC

The PLC can also specify a value for each analog NCK input by applying the interface signal “Setting screen form of analog NCK inputs” (DB10, DBB147) (see Fig. 2-3). As soon as this interface signal is set to “1”, the value set by the PLC (DB10, DBB148 to 163) becomes active for the analog input. The analog value at the hardware input or the input disable is then inactive.

Read actual value

The interface signal “Actual value of analog input of NCK” (DB10, DBB194 to 209) transfers the analog values that are actually present at the hardware inputs to the PLC. The possible influence of the PLC is therefore ignored in the actual value (see Fig. 2-3).

RESET/POWER ON behavior

After POWER ON and RESET, the analog value at the input is passed on. If necessary, the PLC user program can manipulate the NCK inputs as described above in the PLC user program.

Weighting factor

Using the weighting factor in the general MD 10320: FASTIO_ANA_INPUT_WEIGHT[hw] it is possible to adapt each analog NCK input to the various ADCs for reading in the part program (see Fig. 2-3).

In this machine data it is necessary to enter the value x that is to be read in the part program with the system variable $A_INA[n], if the corresponding analog input [n] is set to the maximum value or if the value 32767 is set for this input via the PLC interface. The voltage level at the analog input is then read with the system variable $A_INA[n] as a numerical value with the unit millivolts.

Binary analog value display

See Section 2.5.1

Analog NCK input without hardware

When the part program accesses analog NCK inputs that have been defined in MD 10300: FASTIO_ANA_NUM_INPUTS but that do not exist as hardware inputs, the following values are read:

- The setpoint set by the PLC if the IS “PLC setting for analog NCK inputs” is set to “1” (see Fig. 2-3)
- Otherwise 0 volts

This makes it possible to use the functionality of the analog NCK inputs from the PLC user program without I/O hardware.

Applications

The analog NCK inputs are used particularly for grinding and laser machines (e.g. for the “analog calipers” NC function).

Fast analog NCK inputs

The fast analog inputs must be clock-synchronous. The assignment is specified in MD 10384: HW_CLOCKED_MODULE_MASK.

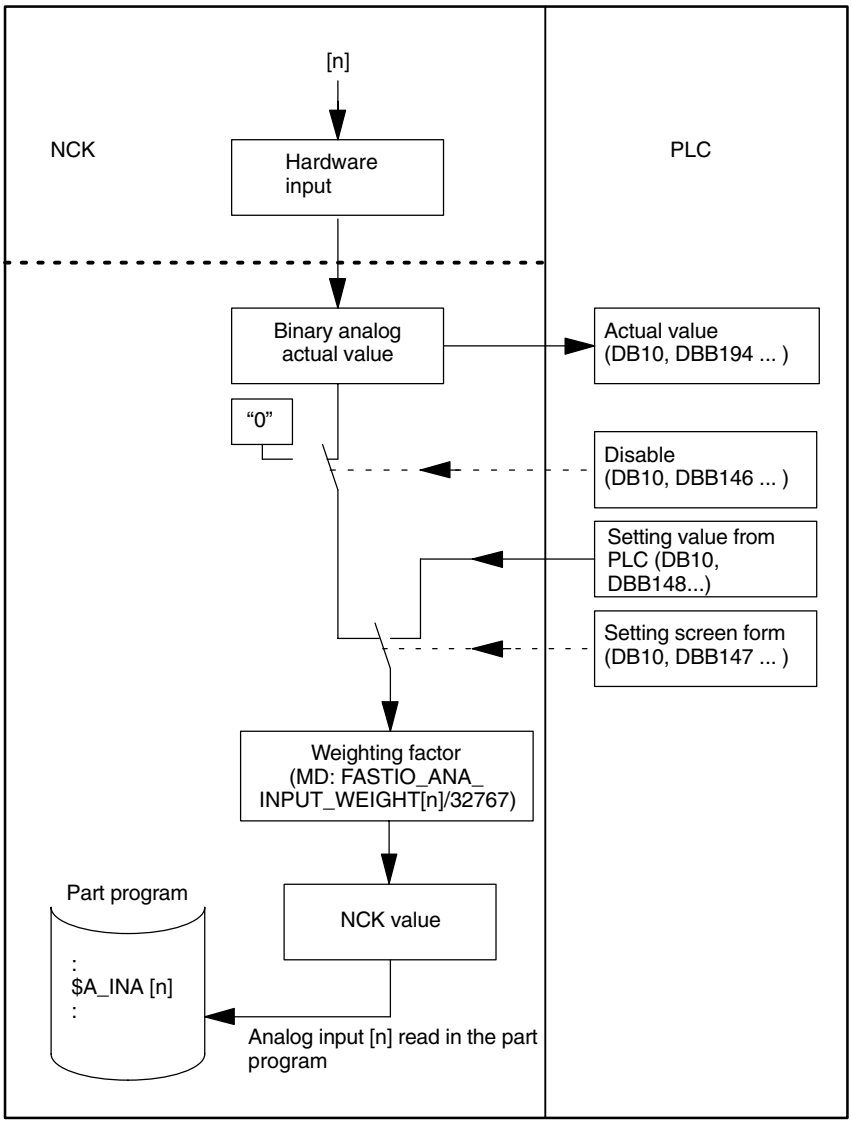


Fig. 2-3 Signal flow for analog NCK inputs

2.4.2 Analog outputs of the NCK

Number

With the general MD 10310: FASTIO_ANA_NUM_OUTPUTS (number of active analog NCK outputs) the available analog NCK outputs can be defined.

Function

The system variable **$A_OUTA[n]** allows the value at the analog output [n] to be specified directly in the part program.

Before output to the hardware output, the analog value set by the NCK can be changed by the PLC (see Fig. 2-4).

Disable output

The PLC user program is capable of disabling the analog NCK outputs individually with interface signal “Disable analog NCK outputs” (DB10, DBB168). In this case, **0 volts** is output at the analog output (see Fig. 2-4).

Overwrite screen form

Every NCK analog value set by the NC part program can be **overwritten** from the PLC using the overwrite screen form. Previous NCK values are then lost (see Fig. 2-4).

The following routine has to be carried out to overwrite the NCK value from the PLC:

1. The output in question must be preset with the required analog value at the PLC interface “PLC setting for analog output n of the NCK” (DB10, DBB170 to 185).
2. The setting value becomes the new NCK value for the analog output (DB10, DBB166) when the overwrite screen form is activated (signal transition 0 –> 1).

This value remains valid until a new value is set for the NCK by the part program, for example.

Setting screen form

Furthermore, a PLC setting for each output can determine whether the instantaneous “NCK value” (e.g. as specified by NC part program) or the PLC value specified via the setting screen form (DB10, DBB167) should be sent to the hardware analog output (see Fig. 2-4).

The following routine has to be carried out to define the PLC value:

1. The output in question must be preset with the required analog value at the PLC interface “PLC setting for analog output n of the NCK” (DB10, DBB170 to 185).
2. The setting screen form (DB10, DBB167) must be set to “1” for the output in question.

Unlike the overwrite screen form, the current NCK value is not lost when a value is set in the setting screen form. As soon as the PLC sets “0” in the setting screen form, the NCK value is again active.

Note

The same setting value (DB10, DBB170 to 185) is used at the PLC interface for the overwrite and the setting screen forms.

Read setpoint

The instantaneous NCK value at the analog outputs can be read by the PLC user program (interface signal “setpoint analog output n of NCK” (DB10, DBB210 to 225)). Please note that this setpoint ignores disabling and the setting screen form of the PLC. The setpoint can therefore differ from the real analog value at the hardware output (see Fig. 2-4).

RESET/end of program

On end of program or reset, every analog output can be defined as necessary by the PLC user program in the overwrite screen form, setting screen form or disable signal.

POWER ON

After POWER ON, the analog outputs are set to “0” in a defined manner. After power-on, this can be overwritten in the PLC user program according to the application using the screen forms described above.

Weighting factor

Using the weighting factor in the general MD 10330: FASTIO_ANA_OUTPUT_WEIGHT[hw] it is possible to adapt each analog NCK output to the various DACs for programming in the part program (see Fig. 2-4).

In this machine data it is necessary to enter the value x that is to cause the analog output [n] to be set to the maximum value or the value 32767 to be set for this output in the PLC interface, if $A_OUTA[n] = x is programmed. The value set with the system variable $A_OUTA[n] then places the corresponding voltage value at the analog output in millivolts.

Binary analog value display

See Section 2.5.1

Special case

Where the part program contains programmed values for NCK analog outputs that have been defined in MD 10310: FASTIO_ANA_NUM_OUTPUTS but do not exist as hardware, no alarm is output. The NCK value can be read by the PLC (IS “Setpoint ...”)

Applications

This function allows analog outputs to be set instantaneously by bypassing the PLC cycles.

The analog NCK outputs are used in particular for grinding and laser machines.

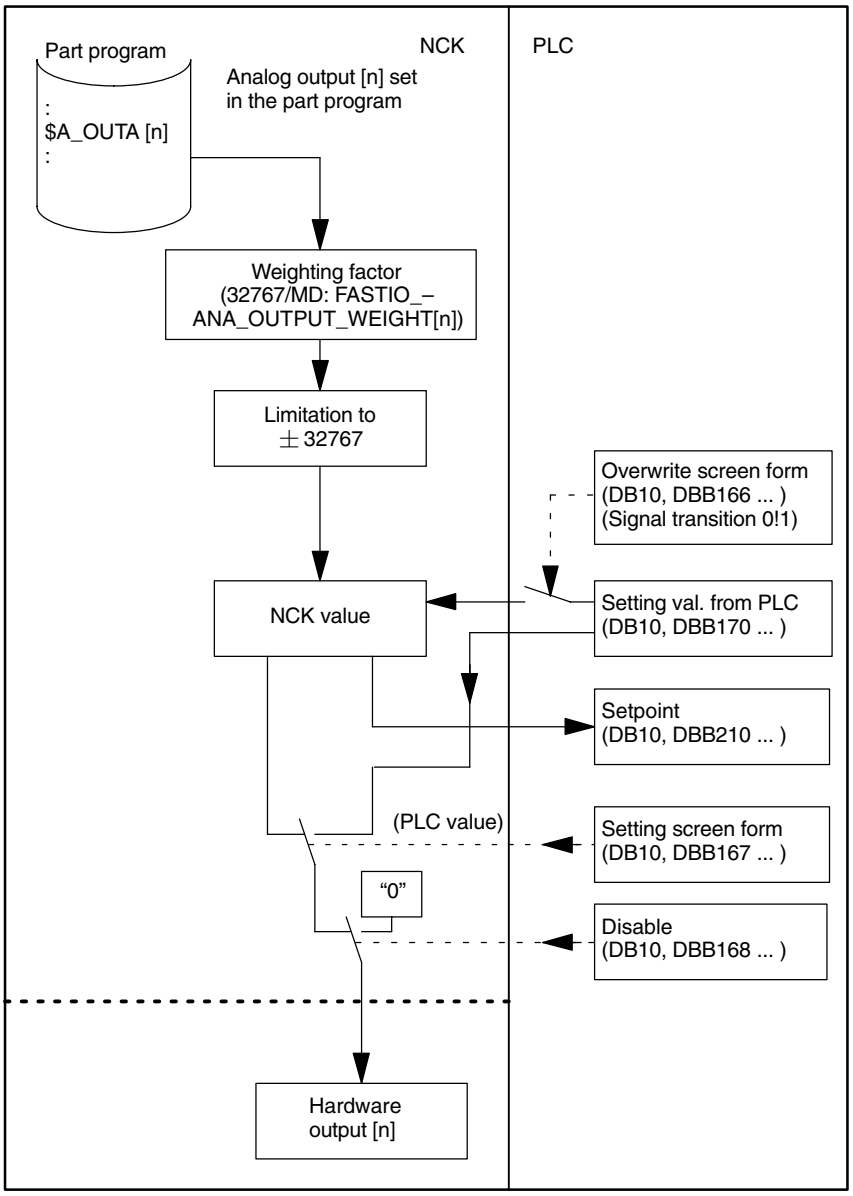


Fig. 2-4 Signal flow for analog NCK outputs

2.5 PLC I/Os directly addressable from NC (SW 5 and later)

Introduction

The class of signals and input/output values described below is processed **directly** and thus significantly faster by the PLC operating system. The signal transfer time between the NC and PLC I/O modules is of a magnitude of 0.5 ms.

	I/O devices	
	central (**ms**)	distributed (**ms**)
Typical read access	0.5	0.5
Typical write access	0.5	0.2

There is no provision for control of signals and analog values (disable, set, overwrite) via the PLC basic and user program. Concurrent access between the NCK and PLC is not meaningful and can lead to malfunctions. Values are transferred in the interpolation cycle.

Accessing

The input/output values are accessed using a special set of system variables from the NC part program and synchronized actions.

Preconditions

Direct PLC I/Os can be addressed with:

Table 2-1 Availability

NCU HW	Version	PLC–SW
840D, NCU 561.2	NCU 572.2 and later	3.10.13 and later
840D, NCU 571.2	NCU 572.2 and later	3.10.13 and later
840D, NCU 572	NCU 572.2 and later	3.10.13 and later
840D, NCU 573	NCU 573.2 and later	3.10.13 and later
810D	CCU2 with PLC315–2 DP	3.10.13 and later

The SINUMERIK SW version must be 5.2 or later.

Configuring

To allow PLC I/Os to be addressed directly, independent input and output areas, each defined by the logical initial address set in the PLC and length in bytes, must be configured by the following machine data.

MD 10395: PLCIO_LOGIC_ADDRESS_IN	Start of input area
MD: 10394: PLCIO_NUM_BYTES_IN	Number of input bytes
MD 10397: PLCIO_LOGIC_ADDRESS_OUT	Start of output area
MD: 10396: PLCIO_NUM_BYTES_OUT	Number of output bytes

The areas defined in the MD must be contiguous, applied **consistently** in the PLC configuration and assigned to the appropriate I/O units.

System variables

$A_PBB_IN[n]	Read input byte 8 (8 bits) directly from PLC I/O, INT
$A_PBW_IN[n]	Read input word (16 bits) directly from PLC I/O, INT
$A_PBD_IN[n]	Read input double word (32 bits) directly from PLC I/O, INT
$A_PBR_IN[n]	Read input real value (32 bits) directly from PLC I/O, REAL

n Byte offset within the PLC input area

Data can be read from the part program and synchronized actions with the system variables for Input. Reading from the part program causes a preprocessing stop.

$A_PBB_OUT[n]	Write output byte (8 bits) directly to PLC I/O, INT
$A_PBW_OUT[n]	Write output word (16 bits) directly to PLC I/O, INT
$A_PBD_OUT[n]	Write output double word (32 bits) directly to PLC I/O, INT
$A_PBR_OUT[n]	Write output real value (32 bits) directly to PLC I/O, REAL

n Byte offset within the PLC output area

Data can be written from the part program and synchronized actions with the system variables for Output. The output data can also be read from the part program and synchronized actions. Reading from the part program causes an automatic preprocessing stop (to achieve synchronization with the real time context).

Memory order

Input double words are stored and must be processed in the memory order of the NCK, i.e. *little endian* (least significant byte in lower memory address).

Note

The smallest addressable unit for a direct PLC I/O is the byte. When signals are processed in write operations, all signal bits in the byte must be set as required. Analogously, the relevant signal bits must be suppressed with NC language tools after a byte has been read.

The system variables mentioned above cannot be used to read or write signals of the machine control panel.

Alignment

The input and output areas for direct PLC I/Os must be assigned in increments according to the data type:

$A_PBB_IN	n: every number < MD 10394: PLCIO_NUM_BYTES_IN
$A_PBB_OUT	n: every number < MD 10396: PLCIO_NUM_BYTES_OUT
$A_PBW_IN	n: every even number < MD 10394: PLCIO_NUM_BYTES_IN
$A_PBW_OUT	n: every even number < MD 10396: PLCIO_NUM_BYTES_OUT
$A_PBD_IN	n: 0, 4, 8, ...< MD 10394: PLCIO_NUM_BYTES_IN
$A_PBD_OUT	n: 0, 4, 8, ...< MD 10396: PLCIO_NUM_BYTES_OUT
$A_PBR_IN	n: 0, 4, 8, ...< MD 10394: PLCIO_NUM_BYTES_IN
$A_PBR_OUT	n: 0, 4, 8, ...< MD 10396: PLCIO_NUM_BYTES_OUT

Evaluation of analog values

In the case of inputs, the part program reads precisely the values coded by the analog/digital converter on the input module. In the case of outputs, the part program must supply exactly those values which the digital/analog converter on the output module can code into the required output value.

2.5.1 Analog value representation of the analog input and output values of the NCK

Conversion of analog values

The analog values are only processed by the NCU in a digital form.

Analog input modules convert the analog process signal into a digital value.

Analog output modules convert the digital output value into an analog value.

Analog value representation

The digitized analog value is identical for input and output values with the same rating range (e.g. voltage range ± 10V DC).

The analog values are coded in the PLC interface as fixed-point numbers (16 bits including sign) in two's complement (see Table 2-2).

Table 2-2 Digital coding of analog values at the PLC interface

Resolution	Binary analog value															
	High byte								Low byte							
Bit number	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Significance of the bits	SG	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

Sign

The sign (SG) of the analog value is always in bit 15.

SG is: "0" → +
"1" → –

Resolution less than 15 bits

The analog value can be finely adjusted depending on the resolution of the digital/analog converter.

If the resolution of the analog module is less than 15 bits, the analog value is entered left-justified. The free less significant places are filled with zeroes.

Table 2-3 shows how the free bit places are filled with zeroes with a 14-bit and a 12-bit analog value.

With a resolution of 14 bits (including sign), the minimum increment is 1.22mV (10V: 8192). In this case, both less significant bits of the analog value (bit0 and bit1) are always 0.

With a resolution of 12 bits (including sign), the increments are 4.8mV (10V: 2048); bits 0 to 3 are always 0.

Table 2-3 Examples of digital analog value coding

Resolution	Binary analog value															
	High byte								Low byte							
Bit number	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Significance of the bits	SG	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
14-bit analog value	0	1	1	1	1	0	0	1	1	0	0	1	1	0	**0**	**0**
12-bit analog value	0	1	1	1	1	0	0	1	1	0	0	1	**0**	**0**	**0**	**0**

The resolution and the nominal range of the analog input/output modules are to be found in:

References: /PHD/, SINUMERIK 840D, NCU Manual
/PHF/, SINUMERIK FM-NC, NCU Manual
/S7H/, SIMATIC S7, Manual

Examples

Here are two examples of digital analog value coding for a nominal range of 10V and 14-bit resolution.

Example 1

Analog value:	9.5V
Absolute value (decimal number):	7782 = |9,5(V)|:10(V) * 8192
Absolute value (binary number):	0111 1001 1001 10
Words (binary number):	0111 1001 1001 10**00**
Words (hexadecimal number):	7998

Example 2

Analog value:	–4.12V
Absolute value (decimal number):	3375 = |–4,12(V)|:10(V) * 8192
Absolute value (binary number):	0011 0100 1011 11
Two's complement:	1100 1011 0100 01
Words (binary number):	1100 1011 0100 01**00**
Absolute value (hexadecimal number)	CB44

2.6 Comparator inputs

Function

2 internal comparator inputs bytes (with 8 comparator inputs each) are available in addition to the digital and analog NCK inputs. The signal status of the comparator inputs is generated on the basis of a comparison between the analog values present at the high-speed analog inputs and high-speed values parameterized in setting data (see Fig. 2-5).

The system variable **$A_INCO[n]** allows the signal status (i.e. the result of the comparison) of comparator input [n] to be scanned directly in the part program.

The following applies to index n: n = 1 to 8 for comparator byte 1
n = 9 to 16 for comparator byte 2

Terms

In this description, the terms “comparator inputs” (with index [n]; range of n: 1 to 8 or 9 to 16) and “comparator input bits” (with index [b]; range of b: 0 to 7) are used.

They are related as follows:

For n = 1 to 8: Comparator input n is equivalent to comparator input bit b = n – 1

For n = 9 to 16: Comparator input n is equivalent to comparator input bit b = n – 9

Example

Comparator input 1 is equivalent to comparator input bit 0.

Assignment of analog inputs

General MD 10530: COMPAR_ASSIGN_ANA_INPUT_1 [b] is set to assign an analog input to input bit [b] of comparator byte 1.

Example

MD 10530: COMPAR_ASSIGN_ANA_INPUT_1[0] = 1
MD 10530: COMPAR_ASSIGN_ANA_INPUT_1[1] = 1
MD 10530: COMPAR_ASSIGN_ANA_INPUT_1[7] = 7
Analog input 1 is assigned to input bits 0 and 1 of comparator byte 1
Analog input 7 is assigned to input bit 7 of comparator byte 1

The assignment for comparator byte 2 must be made analogously in the general MD: 10531 COMPAR_ASSIGN_ANA_INPUT_2 [b].

Comparator parameterization

General MD 10540: COMPAR_TYPE_1 is used to set the following parameters for each bit (0 to 7) of comparator byte 1:

- Comparison type screen form (bits 0 to 7)

  The type of comparison conditions is defined for each comparator input bit.

  Bit = 1: The associated comparator input bit is set to “1” if the analog value is $\geq$ the threshold value.

  Bit = 0: The associated comparator input bit is set to “0” if the analog value is $\leq$ the threshold value.

- Output of the comparator input byte via digital NCK outputs (bits 16 to 23)

  The comparator bits can also be output directly via the digital NCK outputs in whole bytes. This requires specification in this byte (bits 16 to 23) of the digital NCK output byte to be used (see general MD: 10540 COMPAR_TYPE_1).

- Inversion screen form for outputting the comparator input byte (bits 24 to 31).

  For every comparator signal it is also possible to define whether the signal state to be output at the digital NCK output is to be inverted or not.

  Bit = 0: The associated comparator input bit is not inverted.

  Bit = 1: The associated comparator input bit is inverted.

Threshold values

The threshold values used for comparisons on comparator byte 1 or 2 must be stored as setting data. For every comparator input bit [b], you must enter a separate threshold value.

MD 41600: COMPAR_THRESHOLD_1[b], threshold values for input bit [b] of comparator byte 1 (b = 0 to 7)

Comparator signals as digital NCK inputs

All NC functions that are processed as a function of digital NCK inputs can also be controlled by the signal states of the comparators. The byte address for comparator byte 1 (HW byte 128) or 2 (HW byte 129) must be entered in the MD associated with the NC function: “Assignment of hardware byte used”.

Example

NC function “Multiple feedrates in one block”.
Setting in channel-specific MD 21220: MULTFEED_ASSIGN_FASTIN = 129**.**
Now the various feedrates are activated depending on the state of comparator byte 2.

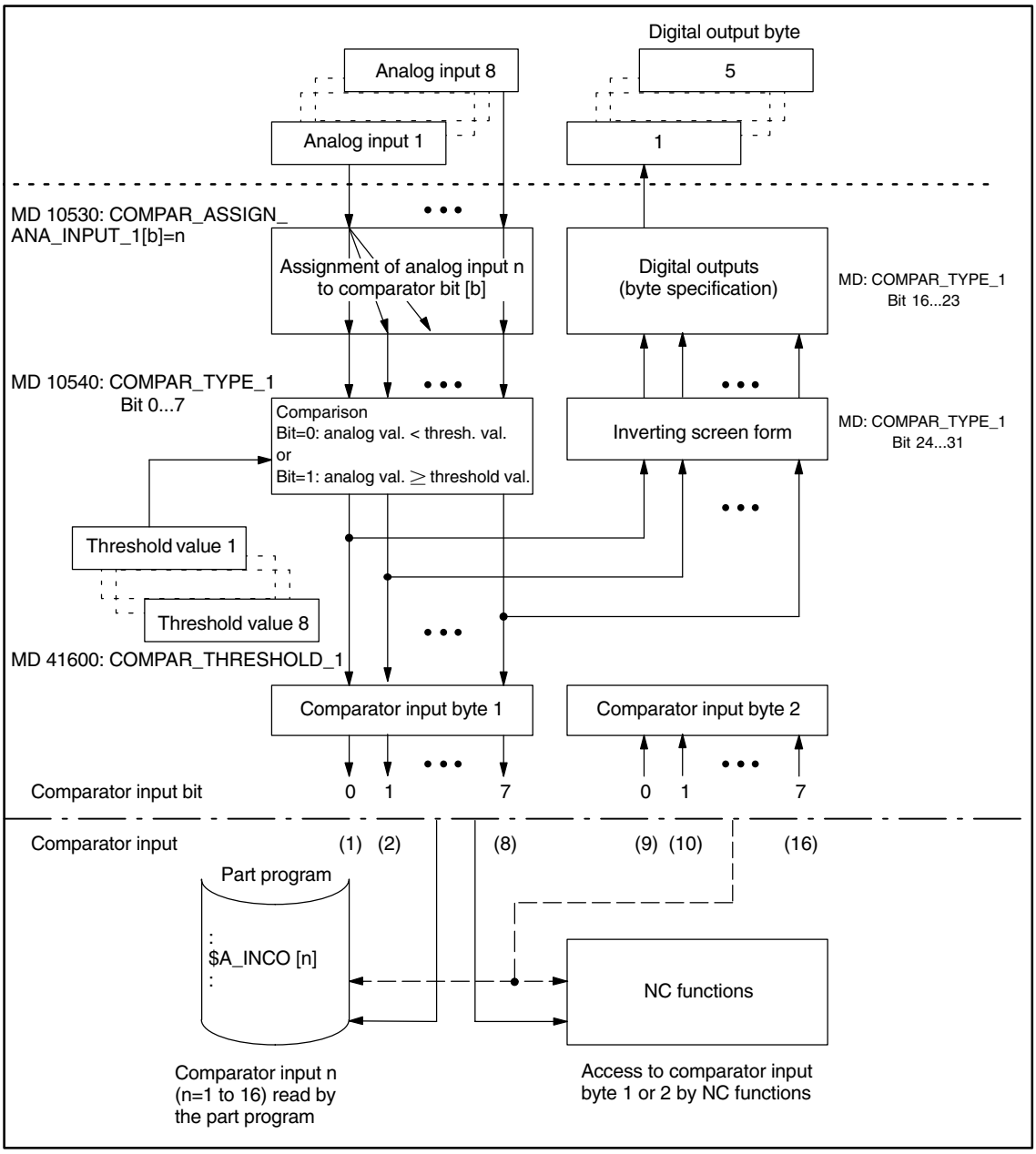


Fig. 2-5 Functional sequence for comparator input byte 1 (or 2)

■

Supplementary Conditions 3

Availability of the function "Digital and analog NC inputs/outputs"

Digital and analog CNC inputs/outputs (DI, DO, AI, AO) are available as follows:

- SINUMERIK FM-NC with NCU 570, SW 2 and higher
  16 DI / 16 DO with extension via S7 I/Os
- SINUMERIK 840D with NCU 571, SW 2 and higher
  4 DI / 4 DO (on-board)
  32 DI / 32 DO with extension via NCU terminal block
- SINUMERIK 840D with NCU 572/573, SW 2 and higher
  4 DI / 4 DO (on-board)
  32 DI / 32 DO and 8 AI / AO with extension via NCU terminal block

Analog I/Os for 840Di

The analog I/Os are connected to the SINUMERK 840Di via the PROFIBUS-DP.

■

Notes

Data Descriptions (MD, SD) 4

4.1 General machine data

10300 MD number	**FASTIO_ANA_NUM_INPUTS** Number of active analog NCK inputs		
Default value: 0	Min. input limit: 0	Max. input limit: 8	
Changes effective after Power On	Protection level: 2 / 4	Unit: –	
Data type: BYTE	Applies from SW version: 2.1		
Significance:	This machine data defines the number of usable analog NCK inputs on the control. Only these analog NCK inputs can be addressed by the NC part program or assigned by NC functions. If more analog NCK inputs are defined in the machine data than are available in the hardware of the control, the binary analog actual value is set to zero in the control for the inputs that do not exist in the hardware. The NCK value can be altered by the PLC (see Section 2.4.1). Note: CPU computing time on the interpolation level is required for processing the digital and analog NCK I/Os. The number of active NCK I/Os should be limited to the demands of the machine so that the interpolation cycle is not overloaded.		

10310 MD number	**FASTIO_ANA_NUM_OUTPUTS** Number of active analog NCK outputs		
Default value: 0	Min. input limit: 0	Max. input limit: 8	
Changes effective after Power On	Protection level: 2 / 4	Unit: –	
Data type: BYTE	Applies from SW version: 2.1		
Significance:	This machine data defines the number of usable analog NCK outputs on the control. Only these analog NCK outputs can be addressed by the NC part program or assigned by NC functions. If more analog NCK outputs are defined in the machine data than are available in the hardware of the control, no alarm is triggered. The analog values specified by the part program can be read by the PLC (see Section 2.4.2). Note: CPU computing time on the interpolation level is required for processing the digital and analog NCK I/Os. The number of active NCK I/Os should be limited to the demands of the machine so that the interpolation cycle is not overloaded.		

10320 MD number	**FASTIO_ANA_INPUT_WEIGHT [hw]** Weighting factor for analog NCK inputs [hw]		
Default value: 840D: 10 000 FM-NC: 11851	Min. input limit: 1	Max. input limit: 10 000 000	
Changes effective after Power On	Protection level: 2 / 4	Unit: –	
Data type: DWORD	Applies from SW version: 2.1		
Significance:	With this MD a weighting factor can be defined for every analog NCK input [n] with which adaptation to the various A/D converters (depending on the I/O module used; different modules can be used on the FM-NC) is possible. [hw] = Index (0–7) for addressing the external analog inputs The value x must be entered in this machine data which is then to be read in the part program with the command x = $A_INA[n] if the corresponding analog input [n] is set to the maximum value or if the value +32767 is set for this input via the PLC interface. The value read from the AD converter or PLC interface must be multiplied by the factor (FASTIO_ANA_INPUT_WEIGHT / 32767) before it can be read by system variable $A_INA[n] in the part program (see Fig. 2-3). An internal value of ± 32767 is generated if the maximum input voltage is applied at the AD converter. Use of the weighting factor for “Analog NCK inputs without hardware”: With a weighting factor of 32767 the values defined by the part program and the PLC are identical (1:1 communication between part program and PLC). This is of advantage when the analog NCK inputs/outputs are used purely as PLC inputs/outputs without analog hardware. Note: The comparator threshold values MD 41600: COMPAR_THRESHOLD_1 or MD 41601: COMPAR_THRESHOLD_2 are also scaled to FASTIO_ANA_INPUT_WEIGHT for comparison purposes according to their analog input assignment.		
Application	Example 1: Measuring range of the analog input module: 0 to 2V (standard range) (FM-NC) Maximum value: 2370mV (corresponds to 32767) FASTIO_ANA_INPUT_WEIGHT[0] = 2370 An analog value of 2V is represented as the digitized value +27648 ($6C00_H$) at IS “Actual value ...” (DB10, DBB199...) and read in the part program with the system variable $A_INA[n] = 2000. Example 2: Measuring range of the analog input module: 0 to 10V (standard range) (FM-NC) Maximum value: 11851mV (corresponds to 32767) FASTIO_ANA_INPUT_WEIGHT[1] = 11851 An analog value of 10V is the digitized value 27648 (= PLC value); 10000 is read with $A_INA[n].		
Related to	IS “Setpoint from the PLC for the analog NCK inputs” (DB10, DBB148–163) IS “Setpoint from the PLC for the analog NCK outputs” (DB10, DBB170–185) IS “Setpoint of analog NCK outputs” (DB10, DBB210–225)		

10330 MD number	**FASTIO_ANA_OUTPUT_WEIGHT [hw]** Weighting factor for analog NCK outputs [hw]		
Default value: 840D: 10 000 FM-NC: 11852	Min. input limit: 1	Max. input limit: 10 000 000	
Changes effective after Power On	Protection level: 2 / 4	Unit: –	
Data type: DWORD	Applies from SW version: 2.1		
Significance:	With this MD a weighting factor can be defined for every analog NCK output [n] with which adaptation to the various DA converters (depending on the I/O module used) is possible. [hw] = Index (0–7) for addressing the external analog outputs The value x must be entered in this machine data which is to cause the analog output [n] to be set to the maximum value or set the value +32767 for this output in the PLC interface if $A_OUTA[n] = x is programmed in the part program. An internal value of ± 32767 therefore represents the maximum output voltage at the DA converter. Use of the weighting factor for “Analog NCK outputs without hardware”: With a weighting factor of 32767 the values defined by the part program and the PLC are identical (1:1 communication between part program and PLC). This is of advantage when the analog NCK outputs are used purely as PLC outputs without analog hardware.		
Application	Example (FM-NC): Output range of the analog output module: 0 to 10V (standard range) Maximum value (overrange): 11852mV (corresponds to 32767) FASTIO_ANA_OUTPUT_WEIGHT[0] = 11852 If $A_OUTA[n] = 10 000 is programmed, +27648 ($6C00_H$) is represented at IS “Setpoint...” (DB10, DBB210...) and output at analog output +10V. Example (840D): Output range of the analog output module: 0 to 10V Maximum value: 10 000mV (corresponds to 32767) FASTIO_ANA_OUTPUT_WEIGHT[0] = 10000 If $A_OUTA[n] = 10 000 is programmed, +32767 (i.e. 7FFF) is represented at IS “Setpoint...” (DB10, DBB210...) and set at analog output +10V.		
Related to	IS “Setpoint from the PLC for the analog NCK inputs” (DB10, DBB148–163) IS “Setpoint from the PLC for the analog NCK outputs” (DB10, DBB170–185) IS “Setpoint of analog NCK outputs” (DB10, DBB210–225)		

10350 MD number	**FASTIO_DIG_NUM_INPUTS** Number of active digital NCK input bytes		
Default value: 1	Min. input limit: 0	Max. input limit: 5	
Changes effective after Power On	Protection level: 2 / 4	Unit: –	
Data type: BYTE	Applies from SW version: 2.1		
Significance:	The number of bytes of the digital NCK inputs that can be used on the control are defined in this machine data. These digital NCK inputs can be read directly by the part program. The signal state at the HW inputs can also be changed by the PLC. If more digital NCK inputs are defined in the machine data than are available in the hardware of the control, a signal status of zero is set in the control for the inputs that do not exist in the hardware. The NCK value can be altered by the PLC. See Section 2.2.1 for a more detailed description.		
Application	Digital NCK inputs 5 to 8 can only be influenced by the PLC (no hardware inputs).		
Related to	IS “Disable the digital NCK inputs” (DB10, DBB0, DBB122 ...) IS “Set the digital NCK inputs from the PLC” (DB10, DBB1, DBB123 ...) IS “Actual value of the digital NCK inputs” (DB10, DBB60, DBB186 ...)		

10360 MD number	**FASTIO_DIG_NUM_OUTPUTS** Number of active digital NCK output bytes		
Default value: 1	Min. input limit: 0	Max. input limit: 5	
Changes effective after Power On	Protection level: 2 / 4	Unit: –	
Data type: BYTE	Applies from SW version: 2.1		
Significance:	The number of bytes of the digital NCK outputs that can be used on the control are defined in this machine data. These digital NCK outputs can be set directly by the part program. The PLC is able to • set the digital outputs with IS “Disable the digital NCK outputs” to “0” in a defined way. • alter the NCK value with IS “Overwrite screen form for digital NCK outputs”. • specify a PLC value with IS “Setting screen form for digital NCK outputs”. If more digital NCK outputs are defined in the machine data than are available in the hardware of the control, no alarm is triggered. The signal states specified by the part program can be read by the PLC. See Section 2.2.2 for a more detailed description.		
Special cases, errors, ...	Digital NCK outputs 5 to 8 can only be processed by the PLC (no hardware outputs).		
Related to	IS “Disable the digital NCK outputs” (DB10, DBB4, DBB130 ...) IS “Overwrite screen form of the digital NCK outputs” (DB10, DBB5, DBB131 ...) IS “Setpoint from the PLC for the digital NCK outputs” (DB10, DBB6, DBB132 ...) IS “Setting screen form of the digital NCK outputs” (DB10, DBB7, DBB133 ...) IS “Setpoint of the digital NCK outputs” (DB10, DBB64, DBB190 ...)		

10361 MD number	**FASTIO_DIG_SHORT_CIRCUIT** Short-circuit of digital inputs/outputs		
Default value: 0	Min. input limit: –	Max. input limit: –	
Changes effective after Power On	Protection level: 2/7	Unit: –	
Data type: DWORD	Applies from SW version: 4.2		
Significance:	The fast NCK input specified in the MD is generated by the software via a logic operation with the assigned digital output. Logic operations: 00: Connect A0: AND operation B0: OR operation Bits 0–7 byte of input, bits 8–15: bit of input + logic operation bits 16–23: byte of output, bits 24–31: bit of output		
Application	AND operation: $MN_FASTIO_DIG_SHORT_CIRCUIT = ’0705A201H’ Output: bit 7, byte 5 AND operation with Input: bit 2, byte 1		
Related to	MD 10350:FASTIO_DIG_NUM_INPUTS, MD 10360:FASTIO_DIG_NUM_OUTPUTS,		
References	FB A4		

10362 MD number	**HW_ASSIGN_ANA_FASTIN [hw]** Hardware assignment of external analog NCK inputs		
Default value: on 840D/810D: 01000000 on 840Di: 050000A0 on FM-NC: 02000000	Min. input limit: on 840D/810D: 01000000 on 840Di: 05000000 on FM-NC: 02000000	Max. input limit: on 840D/810D: 011E0802 on 840Di: 050003FF on FM-NC: 02070004	
Changes effective after Power On	Protection level: 2 / 4	Unit: Hex	
Data type: DWORD	Applies from SW version: 2.1		
Significance:	The following 4 bytes define the assignment between the external analog NCK inputs and the hardware **Applies to 840D/810D and FM-NC**: 1st byte: I/O no. 2nd byte: Submodule no. 3rd byte: Module no. 4th byte: Segment no. **Applies to 840Di**: 1st byte: Logical low address 2nd byte: Logical high address 3rd byte: Not used (00) 4th byte: Segment no. for PROFIBUS-DP (05) Array length = maximum number of analog inputs on NCK is set in MD10300. As soon as value 0 is entered in byte 3 (module no.) external I/Os are no longer processed by the control. A simulated input is defined. The hardware assignment is different on the SINUMERIK 840D/810D, 840Di and FM-NC controls. The individual bytes are explained under MD: 10366: HW_ASSIGN_DIG_FASTIN. [hw] = Index (0–7) for addressing the external analog inputs		
Related to	MD 10366: HW_ASSIGN_DIG_FASTIN MD 10368: HW_ASSIGN_DIG_FASTOUT MD 10364: HW_ASSIGN_ANA_FASTOUT		

10364 MD number	**HW_ASSIGN_ANA_FASTOUT [hw]** Hardware assignment of external analog NCK outputs				
Default value: on 840D/810D: 01000000 on 840Di: 050000A0 on FM-NC: 02000000		Min. input limit: on 840D/810D: 01000000 on 840Di: 05000000 on FM-NC: 02000000		Max. input limit: on 840D/810D: 011E0802 on 840Di: 050003FF on FM-NC: 02070004	
Changes effective after Power On		Protection level: 2 / 4		Unit: Hex	
Data type: DWORD		Applies from SW version: 2.1			
Significance:	The following 4 bytes define the assignment between the external analog NCK outputs and the hardware **Applies to 840D/810D and FM-NC**: 1st byte: I/O no. 2nd byte: Submodule no. 3rd byte: Module no. 4th byte: Segment no. **Applies to 840Di**: 1st byte: Logical low address 2nd byte: Logical high address 3rd byte: Not used (00) 4th byte: Segment no. for PROFIBUS-DP (05) Arry length = maximum number of analog outputs on NCK is set in MD10310. As soon as value 0 is entered in byte 3 (module no.) external I/Os are no longer processed by the control. A simulated input is defined. The hardware assignment is different on the SINUMERIK 840D/810D and FM-NC. The individual bytes are explained under MD: 10366: HW_ASSIGN_DIG_FASTIN. [hw] = Index (0–7) for addressing the external analog outputs				
Related to	MD 10366: HW_ASSIGN_DIG_FASTIN MD 10368: HW_ASSIGN_DIG_FASTOUT MD 10362: HW_ASSIGN_ANA_FASTIN				

10365 MD number	**FASTIO_DIG_SHORT_CIRCUIT** Defined short-circuits between digital I/O signals of the fast NCK I/Os		
Default value: 0	Min. input limit: 0	Max. input limit: –	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: HEX	Applies from SW version: 4.1		
Significance:	Defined short-circuits between the digital input/output signals of the fast NCK I/Os are achieved by a logic operation between the signals read in from the fast NCK I/Os or PLC interface and the defined output signals. Upon the logic operation, the output signals always remain unchanged, the inputs that are to be accounted for internally are achieved from the read inputs and the logic operation. If several output bits are specified for one input bit in overwrite mode, then the result is determined by the last assignment defined in the list. The definition for non-existing or activated inputs/outputs is ignored without alarm. Bit 0–7: Number of the input byte to be described (1–5) Bit 8–15 Bit number within the input byte (1–8) Logic operation: The type of operation is selected by adding a hexadecimal number to the input bit number: 00 Overwrite input as output A0 Input is read input ANDed with the state of the specified output B0 Input is read input ORed with the state of the specified output Bit 16–23: Number of output byte (1–5) Bit 24–31 Bit number within the output byte (1–8)		
Application	$MN_FASTIO_DIG_SHORT_CIRCUIT[0]='0401**0**302H' Input: byte 2, bit 3 Output: byte 1, bit 4 Logic operation: **Overwrite** $MN_FASTIO_DIG_SHORT_CIRCUIT[1]='0705**A**201H' Input: byte 1, bit 2 Output: byte 5, bit 7 Logic operation: **AND** $MN_FASTIO_DIG_SHORT_CIRCUIT[2]='0103**B**502H' Input: byte 5, bit 2 Output: byte 3, bit 1 Logic operation: **OR**		

10366 MD number	**HW_ASSIGN_DIG_FASTIN [hw]** Hardware assignment of external digital NCK inputs				
Default value: on 840D/810D: 01000000 on 840Di: 05000090 on FM-NC: 02000000		Min. input limit: on 840D/810D: 01000000 on 840Di: 05000000 on FM-NC: 02000000		Max. input limit: on 840D/810D: 011E0802 on 840Di: 050003FF on FM-NC: 02070004	
Changes effective after Power On		Protection level: 2 / 4		Unit: Hex	
Data type: DWORD		Applies from SW version: 2.1			
Significance:	The following 4 bytes define the assignment between the external digital NCK I/Os and the hardware **Applies to 840D/810D and FM-NC**: 1st byte: I/O no. 2nd byte: Submodule no. 3rd byte: Module no. 4th byte: Segment no. **Applies to 840Di**: 1st byte: Logical low address 2nd byte: Logical high address 3rd byte: Not used (00) 4th byte: Segment no. for PROFIBUS-DP (05) Arry length = maximum number of digital input bytes on NCK is set in MD10350. As soon as value 0 is entered in byte 3 (module no.), the input byte concerned is not processed by the control. A simulated input is defined. The hardware assignment is different on the SINUMERIK 840D, 840Di and FM-NC. **840D/810D:** I/O no.: Number of the I/O byte on the DP compact module (range: 1 to 2; always 1 for analog inputs/outputs) Submodule no.: Submodule slot on the terminal block into which the DP compact module is slotted (range: 1 to 8) Module no.: Number of the logical slot into which the terminal block with the external I/Os is slotted. The assignment of the logical slot to a physical slot is made with MD 13010: DRIVE_LOGIC_NR (logical drive number). Each module occupies a physical slot. The first 6 slots on the 810D are permanently assigned. Segment no.: for 840/810D always 1 (identifier for 611D bus) Example: HW_ASSIGN_DIGITAL_FASTIN[3] = 01 04 03 02 1st byte: 02 = 2. input byte of a 16 bit input module 2nd byte: 03 = Input module slotted into slot 2 of the terminal block 3rd byte: 04 = Terminal block slotted into logical drive number 4 4th byte: 01 = Identifier for 611D bus				

10366 MD number	**HW_ASSIGN_DIG_FASTIN [hw]** Hardware assignment of external digital NCK inputs
Significance:	**FM-NC:** I/O no.: Number of the input byte on the signal module (range depends on the I/O group: 1 to 4) Submodule no.: Of no relevance for FM-NC (this byte must be preset with 01 in a defined way) Module no.: Number of the physical slot on the local P bus.(Range 1 to 7) Note: The FM-NC is slot 0; the physical slots for the digital and analog I/Os are located to the left of this. Segment no.: Always 2 for FM-NC (identifier for local P bus) Example: HW_ASSIGN_DIGITAL_FASTIN[3] = 02 04 00 02 1st byte: 02 = 2. input byte for a 16 bit digital input module 2nd byte: 00 = Not relevant 3rd byte: 04 = Input module slotted into logical slot 4 4th byte: 02 = Identifier for local P bus [hw] = Index (0 to 3) for addressing the external digital input byte
Related to	MD 10368: HW_ASSIGN_DIG_FASTOUT MD 10362: HW_ASSIGN_ANA_FASTIN MD 10364: HW_ASSIGN_ANA_FASTOUT

10368 MD number	**HW_ASSIGN_DIG_FASTOUT [hw]** Hardware assignment of external digital NCK outputs				
Default value: on 840D/810D: 01000000 on 840Di: 05000090 on FM-NC: 02000000		Min. input limit: on 840D/810D: 01000000 on 840Di: 05000000 on FM-NC: 02000000		Max. input limit: on 840D/810D: 011E0802 on 840Di: 050003FF on FM-NC: 02070004	
Changes effective after Power On		Protection level: 2 / 4		Unit: Hex	
Data type: DWORD		Applies from SW version: 2.1			
Significance:	The following 4 bytes define the assignment between the external digital NCK outputs and the hardware **Applies to 840D/810D and FM-NC**: 1st byte: I/O no. 2nd byte: Submodule no. 3rd byte: Module no. 4th byte: Segment no. **Applies to 840Di**: 1st byte: Logical low address 2nd byte: Logical high address 3rd byte: Not used (00) 4th byte: Segment no. for PROFIBUS-DP (05) Arry length = maximum number of digital output bytes on NCK is set in MD10360. As soon as value 0 is entered in byte 3 (module no.), the input byte concerned is not processed by the control. A simulated input is defined. The hardware assignment is different on the SINUMERIK 840D/810D and FM-NC. The individual bytes are explained under MD: HW_ASSIGN_DIG_FASTIN. [hw] = Index (0–3) for addressing the external analog output bytes				
Related to	MD 10366: HW_ASSIGN_DIG_FASTIN MD 10362: HW_ASSIGN_ANA_FASTIN MD 10364: HW_ASSIGN_ANA_FASTOUT				

10380 MD number	**HW_UPDATE_RATE_FASTIO [tb]** Updating rate of clock-synchronous external NCK I/Os		
Default value: 2	Min. input limit: 2	Max. input limit: 3	
Changes effective after Power On	Protection level: 2 / 4	Unit: –	
Data type: BYTE	Applies from SW version: 2.1		
Significance:	With this machine data, the cycle frequency for the clock-synchronous input and output of the external NCK I/Os is selected (840D only). The cycle time applies to all I/O modules on a terminal block that are operated in synchronism with the clock (MD 10384: HW_CLOCKED_MODULE_MASK[tb]=1). The selection can be made from the following cycle frequencies: Value = 1: Synchronous input/outputs in hardware cycles **(not in SW 2)** (SYSCLOCK_CYCLE_TIME / SYSCLOCK_SAMPL_TIME_RATIO) 2: Synchronous input/outputs in position control cycles (default setting) (MD: POSCTR_SYSCLOCK_TIME_RATIO) 3: Synchronous inputs/outputs in interpolation cycles (MD: IPO_SYSCLOCK_TIME_RATIO) Note on index [tb] (tb = 0 to 1): Index [tb] identifies the connected NCU terminal blocks in ascending order of the defined logical module numbers (parameterization with MD: DRIVE_LOGIC_NR "logical drive number"). Example: An additional 2 terminal blocks which are parameterized with the logical drive number 6 and 7 are connected to the drive bus. The following assignments are made for the terminal blocks in the control: • HW_UPDATE_RATE_FASTIO[0] parameterizes terminal block 1 with no. 6 • HW_UPDATE_RATE_FASTIO[1] parameterizes terminal block 2 with no. 7 This assignment applies analogously to: MD 10380: HW_UPDATE_RATE_FASTIO[tb] and MD 10384: HW_CLOCKED_MODULE_MASK [tb] For more detailed information see **References:** /FB/, G2, "Velocities, Setpoint/Actual Value Systems, Closed-Loop Control" Note: Please consider the hardware response times of the external I/O modules used. **References:** /PHD/, SINUMERIK 840D, NCU Manual		
MD irrelevant for	SINUMERIK FM-NC		
Related to	MD 10382: HW_LEAD_TIME_FASTIO MD 10384: HW_CLOCKED_MODULE_MASK POSCTR_SYSCLOCK_TIME_RATIO IPO_SYSCLOCK_TIME_RATIO SYSCLOCK_SAMPL_TIME_RATIO DRIVE_LOGIC_NR		
References	**References:** /FB/, G2, "Velocities, Setpoint/Actual Value Systems, Closed-Loop Control"		

10382 MD number	**HW_LEAD_TIME_FASTIO [tb]** Lead time for clock-synchronous external NCK I/Os		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective after Power On	Protection level: 2 / 4	Unit: ms	
Data type: DWORD	Applies from SW version: 2.1		
Significance:	A lead time can be defined for digital and analog NCK I/Os (MD 10384: HW_CLOCKED_MODULE_MASK = 1) operated in synchronism with the clock. The input signal is stored this length of time before the defined cycle. The output signal is sent to the hardware this same length of time before the defined cycle. With NCK inputs, for example, this makes it possible to consider the hardware-determined conversion time of the AD converter so that the digitized analog value is available on the cycle. If the value set in this machine data exceeds the fixed cycle time (MD 10380: HW_UPDATE_RATIO_FASTIO), it is limited internally to the largest possible offset (i.e. to the parameterized cycle time). The lead time applies to all NCK inputs/outputs of the terminal block addressed with index [tb] which are operated in synchronism with the clock. **Note** on index [tb] see MD 10380: HW_UPDATE_RATE_FASTIO.		
MD irrelevant for	SINUMERIK FM-NC		
Related to	MD 10380: HW_UPDATE_RATIO_FASTIO MD 10384: HW_CLOCKED_MODULE_MASK		

10384 MD number	**HW_CLOCKED_MODULE_MASK [tb]** Clock-synchronous processing of external NCK I/Os		
Default value: 00	Min. input limit: 00	Max. input limit: FF	
Changes effective after Power On	Protection level: 2 / 4	Unit: Hex	
Data type: BYTE	Applies from SW version: 2.1		
Significance:	With SINUMERIK 840D, the I/O modules of the external NCK I/Os can be operated in the following way: • **Asynchronously**, i.e. the input and output values are made available in cycles set by the terminal block which are asynchronous to the internal NC processing cycles. • **Synchronously**, i.e. the input and output values are made available synchronously to a settable internal NC processing cycle. The mode of operation can be set via a bit screen form (bits 0 to 7) for each individual I/O module of the terminal block address with index [n] (bit 0 for I/O module on slot 1 ... bit 7 for I/O module on slot 8). The bits have the following meaning: Bit n = 0: I/O module on slot n+1 is operated **asynchronously** Bit n = 1: I/O module on slot n+1 is operated **synchronously** The value is of no significance for unassigned slots of the terminal block. Example: HW_CLOCKED_MODULE_MASK[0] = 30 (bit screen form: 0011 0000) The I/O modules of terminal block 1 on slots 6 and 5 are operated in synchronism with the clock. **Note**: Digital NCK inputs/outputs are generally always operated asynchronously. When analog NCK inputs/outputs are used in closed control loops, values often have to be read in and out in synchronism with the clock. **Notes**: – on index [tb] see MD 10380: HW_UPDATE_RATE_FASTIO. – Module 6FC5211-0AA10-0AA0 can only operate synchronously.		
MD irrelevant for	SINUMERIK FM-NC (always operated asynchronously)		
Related to	MD 10382: HW_LEAD_TIME_FASTIO MD 10380: HW_UPDATE_RATIO_FASTIO		

10394 MD number	**PLCIO_NUM_BYTES_IN** Number of direct read inputs bytes of PLC I/Os		
Default value: 0	Min. input limit: 0	Max. input limit: 16	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: BYTE	Applies from SW version: 5.1		
Significance:	Number of PLC I/O input bytes that can be read directly by the NC. These bytes are transferred via the PLC operating system and not influenced by the PLC basic or user program, resulting in a delay time of less than approximately 0.5 ms. They can be read by part programs/synchronized actions via the following system variables: $A_PBB_IN $A_PBW_IN $A_PBD_IN $A_PBR_IN		
Special cases, errors, ...	Machine data MD 10394: PLCIO_NUM_BYTES_IN and MD 10395: PLCIO_LOGIC_ADDRESS_IN must be consistent with the PLC configuring data.		
Related to	MD 10395: PLCIO_LOGIC_ADDRESS_IN		

10395 MD number	**PLCIO_LOGIC_ADDRESS_IN** Start address of direct read input bytes of PLC I/Os		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies from SW version: 5.1		
Significance:	Logical start address of direct PLC input devices From this address onwards, a number of PLCIO_NUM_BYTES_IN bytes for direct use by the NC must be defined by the PLC hardware configuration. These bytes are transferred via the PLC operating system and not influenced by the PLC basic or user program, resulting in a delay time of less than approximately 0.5 ms. They can be read by part programs/synchronized actions via the following system variables: $A_PBB_IN, $A_PBW_IN, $A_PBD_IN, $A_PBR_IN.		
Special cases, errors, ...	Machine data MD 10394: PLCIO_NUM_BYTES_IN and MD 10395: PLCIO_LOGIC_ADDRESS_IN must be consistent with the PLC configuring data.		
Related to	MD 10394: PLCIO_NUM_BYTES_IN		

10396 MD number	**PLCIO_NUM_BYTES_OUT** Number of direct write output bytes of PLC I/Os		
Default value: 0	Min. input limit: 0	Max. input limit: 16	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: BYTE	Applies from SW version: 5.1		
Significance:	Number of PLC I/O output bytes that can be written directly by the NC. These bytes are transferred via the PLC operating system and not influenced by the PLC basic or user program, resulting in a delay time of less than approximately 0.5 ms. They can be written/read by part programs/synchronized actions via the following system variables: $A_PBB_OUT, $A_PBW_OUT, $A_PBD_OUT, $A_PBR_OUT		
Special cases, errors, ...	Machine data MD 10396: PLCIO_NUM_BYTES_OUT and MD 10397: PLCIO_LOGIC_ADDRESS_OUT must be consistent with the PLC configuring data. In the case of an error, other PLC signals would be overwritten.		
Related to	MD 10397: PLCIO_LOGIC_ADDRESS_OUT		

10397 MD number	**PLCIO_LOGIC_ADDRESS_OUT** Start address of direct write output bytes of PLC I/Os		
Default value: 0	Min. input limit: 0	Max. input limit: Plus	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies from SW version: 5.1		
Significance:	Logical start address of direct PLC output devices From this address onwards, a number of PLCIO_NUM_BYTES_OUT bytes for direct use by the NC must be defined by the PLC hardware configuration. These bytes are transferred via the PLC operating system and not influenced by the PLC basic or user program, resulting in a delay time of less than approximately 0.5 ms. They can be written/read by part programs/synchronized actions via the following system variables: $A_PBB_OUT, $A_PBW_OUT, $A_PBD_OUT, $A_PBR_OUT.		
Special cases, errors, ...	Machine data MD 10396: PLCIO_NUM_BYTES_OUT and MD 10397: PLCIO_LOGIC_ADDRESS_OUT must be consistent with the PLC configuring data.In the case of an error, other PLC signals would be overwritten.		
Related to	MD 10396: PLCIO_NUM_BYTES_OUT		

10530 **10531** MD number	**COMPAR_ASSIGN_ANA_INPUT_1[b]** **COMPAR_ASSIGN_ANA_INPUT_2[b]** Hardware assignment of analog inputs for comparator byte 1 (or 2) [bit number]		
Default value: 0	Min. input limit: 0	Max. input limit: 8	
Changes effective after Power On	Protection level: 2 / 4	Unit: –	
Data type: BYTE	Applies from SW version: 2.1		
Significance:	With this MD, the analog inputs 1 to 8 are assigned to a bit number of comparator byte 1 or 2. This input bit of the comparator is set to “1“ if the comparison between the applied analog value and the associated threshold value (MD 41600: COMPAR_THRESHOLD_1 or MD 41601: COMPAR_THRESHOLD_2) verifies that the condition parameterized in MD 10540: COMPAR_TYPE_1 or MD 10541: COMPAR_TYPE_2) is fulfilled. An analog input can be assigned several input bits. The following generally applies for comparator byte 1: **COMPAR_ASSIGN_ANA_INPUT_1 [b] = n** with index: b = number of comparator input bit (0 to 7) with index n = number of analog input (1 to 8) Example: COMPAR_ASSIGN_ANA_INPUT_1[0] = 1 COMPAR_ASSIGN_ANA_INPUT_1[1] = 2 COMPAR_ASSIGN_ANA_INPUT_1[2] = 1 COMPAR_ASSIGN_ANA_INPUT_1[3] = 3 COMPAR_ASSIGN_ANA_INPUT_1[4] = 3 COMPAR_ASSIGN_ANA_INPUT_1[5] = 1 COMPAR_ASSIGN_ANA_INPUT_1[6] = 1 COMPAR_ASSIGN_ANA_INPUT_1[7] = 1 Analog input 1 affects input bit 0, 2 , 5, 6 and 7 of comparator byte 1 Analog input 2 affects input bit 1 of comparator byte 1 Analog input 3 affects input bit 3 and 4 of comparator byte 1 The same also applies to comparator byte 2 with respect to COMPAR_ASSIGN_ANA_INPUT_2[b]. See Section 2.6 for a more detailed description.		
Related to	MD 10540: COMPAR_TYPE_1 MD 10541: COMPAR_TYPE_2		

10540 **10541** MD number	**COMPAR_TYPE_1** **COMPAR_TYPE_2** Parameterization for comparator byte 1 or 2		
Default value: 0	Min. input limit:	Max. input limit:	
Changes effective after Power On	Protection level: 2 / 4	Unit: binary screen form	
Data type: DWORD	Applies from SW version: 2.1		
Significance:	The following parameters can be set with COMPAR_TYPE_1 for the individual input bits (0 to 7) of comparator byte 1: • Bits 0 to 7: Type of comparison screen form (for comparator input bit 0 to 7) Bit = 1: Input bit = 1 when analog value ≥ threshold value Bit = 0: Input bit = 1 when analog value < threshold value (threshold value set with MD 41600: COMPAR_THRESHOLD_1 or MD 41601: COMPAR_THRESHOLD_2) • Bits 8 to 15: Not assigned (to be set to 0 in a defined way) • Bits 16 to 23: Assignment of a hardware output byte for the output of the comparator states (stating byte address) Byte = 0: No output via digital NCK outputs Byte = 1: Output via digital on-board NCK outputs (1 to 4) Byte = 2: Output via external digital NCK outputs 9 to 16 Byte = 3: Output via external digital NCK outputs 17 to 24 Byte = 4: Output via external digital NCK outputs 25 to 32 Byte = 5: Output via external digital NCK outputs 33 to 40 • Bits 24 to 31: Inversion screen form for the output of comparator states (bit 0 to 7) Bit = 0: Input bit is not inverted Bit = 1: Input bit is inverted The same also applies to comparator byte 2 with COMPAR_TYPE_2. See Section 2.6 for additional information.		
Related to	MD 10530: COMPAR_ASSIGN_ANA_INPUT_1 MD 10531: COMPAR_ASSIGN_ANA_INPUT_2 MD 41600: COMPAR_THRESHOLD_1 MD 41601: COMPAR_THRESHOLD_2 MD 10360: FASTIO_DIG_NUM_OUTPUTS		

4.2 General setting data

41600 **41601** MD number	**COMPAR_THRESHOLD_1[b]** **COMPAR_THRESHOLD_2[b]** Threshold values for comparator byte 1 or 2 [bit 0 to 7]		
Default value: 0	Min. input limit: 0	Max. input limit: ± 10 000	
Changes effective immediately	Protection level: 7	Unit: mV	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	COMPAR_THRESHOLD_1[b] defines the threshold values for the individual input bits[b] of comparator byte 1. The same also applies to comparator byte 2 with COMPAR_THRESHOLD_2[b]. Index [b]: Bits 0–7 See Section 2.6 for a more detailed description.		
Related to	MD 10530: COMPAR_ASSIGN_ANA_INPUT_1 MD 10531: COMPAR_ASSIGN_ANA_INPUT_2 MD 10540: COMPAR_TYPE_1 MD 10541: COMPAR_TYPE_2		

■

Notes

Signal Descriptions 5

5.1 NC-specific signals

In Sections 5.1.1 and 5.1.3, only those signals are listed in the overview that are actually described below. A complete list of all signals is to be found in:

References: /LIS/, Lists

5.1.1 Overview of signals from PLC to NC (DB10)

DB10	Signals to NC interface PLC → NC							
DBB	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Disable digital NCK inputs							
	Digital inputs without hardware #)				On-board inputs §)			
	Input 8	Input 7	Input 6	Input 5	Input 4	Input 3	Input 2	Input 1
1	Setting digital NCK inputs on the PLC							
	Digital inputs without hardware #)				On-board inputs §)			
	Input 8	Input 7	Input 6	Input 5	Input 4	Input 3	Input 2	Input 1
4	Disable digital NCK outputs							
	Digital outputs without hardware #)				On-board outputs §)			
	Output 8	Output 7	Output 6	Output 5	Output 4	Output 3	Output 2	Output 1
5	Overwrite screen form for digital NCK outputs							
	Digital outputs without hardware #)				On-board outputs §)			
	Output 8	Output 7	Output 6	Output 5	Output 4	Output 3	Output 2	Output 1
6	Setting value from PLC for the digital NCK outputs							
	Digital outputs without hardware #)				On-board outputs §)			
	Output 8	Output 7	Output 6	Output 5	Output 4	Output 3	Output 2	Output 1
7	Setting screen form for digital NCK outputs							
	Digital outputs without hardware #)				On-board outputs §)			
	Output 8	Output 7	Output 6	Output 5	Output 4	Output 3	Output 2	Output 1

Comments

#) Bits 4 to 7 of the digital NCK outputs can be processed by the PLC even though there are no equivalent hardware I/Os. These bits can therefore be used additionally for data exchange between the NCK and PLC.

§) On the SINUMERIK 840D digital inputs and outputs 1 to 4 of the NCK are on-board inputs and outputs.
No hardware I/Os are available for bits 0 to 3 on the FM-NC. These can be processed by the PLC as described under #).

DB10	Signals to NC interface PLC → NC							
DBB	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
122	Disable digital NCK inputs							
	Input 16	Input 15	Input 14	Input 13	Input 12	Input 11	Input 10	Input 9
123	Setting digital NCK inputs on the PLC							
	Input 16	Input 15	Input 14	Input 13	Input 12	Input 11	Input 10	Input 9
124	Disable digital NCK inputs							
	Input 24	Input 23	Input 22	Input 21	Input 20	Input 19	Input 18	Input 17
125	Setting digital NCK inputs on the PLC							
	Input 24	Input 23	Input 22	Input 21	Input 20	Input 19	Input 18	Input 17
126	Disable digital NCK inputs							
	Input 32	Input 31	Input 30	Input 29	Input 28	Input 27	Input 26	Input 25
127	Setting digital NCK inputs on the PLC							
	Input 32	Input 31	Input 30	Input 29	Input 28	Input 27	Input 26	Input 25
128	Disable digital NCK inputs							
	Input 40	Input 39	Input 38	Input 37	Input 36	Input 35	Input 34	Input 33
129	Setting digital NCK inputs on the PLC							
	Input 40	Input 39	Input 38	Input 37	Input 36	Input 35	Input 34	Input 33
130	Disable digital NCK outputs							
	Output 16	Output 15	Output 14	Output 13	Output 12	Output 11	Output 10	Output 9
131	Overwrite screen form for digital NCK outputs							
	Output 16	Output 15	Output 14	Output 13	Output 12	Output 11	Output 10	Output 9
132	Setting value from PLC for the digital NCK outputs							
	Output 16	Output 15	Output 14	Output 13	Output 12	Output 11	Output 10	Output 9
133	Setting screen form for digital NCK outputs							
	Output 16	Output 15	Output 14	Output 13	Output 12	Output 11	Output 10	Output 9
134	Disable digital NCK outputs							
	Output 24	Output 23	Output 22	Output 21	Output 20	Output 19	Output 18	Output 17
135	Overwrite screen form for digital NCK outputs							
	Output 24	Output 23	Output 22	Output 21	Output 20	Output 19	Output 18	Output 17
136	Setting value from PLC for the digital NCK outputs							
	Output 24	Output 23	Output 22	Output 21	Output 20	Output 19	Output 18	Output 17
137	Setting screen form for digital NCK outputs							
	Output 24	Output 23	Output 22	Output 21	Output 20	Output 19	Output 18	Output 17
138	Disable digital NCK outputs							
	Output 32	Output 31	Output 30	Output 29	Output 28	Output 27	Output 26	Output 25
139	Overwrite screen form for digital NCK outputs							
	Output 32	Output 31	Output 30	Output 29	Output 28	Output 27	Output 26	Output 25
140	Setting value from PLC for the digital NCK outputs							
	Output 32	Output 31	Output 30	Output 29	Output 28	Output 27	Output 26	Output 25
141	Setting screen form for digital NCK outputs							
	Output 32	Output 31	Output 30	Output 29	Output 28	Output 27	Output 26	Output 25
142	Disable digital NCK outputs							
	Output 40	Output 39	Output 38	Output 37	Output 36	Output 35	Output 34	Output 33

DB10	Signals to NC interface PLC → NC							
DBB	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
143	Overwrite screen form for digital NCK outputs							
	Output 40	Output 39	Output 38	Output 37	Output 36	Output 35	Output 34	Output 33
144	Setting value from PLC for the digital NCK outputs							
	Output 40	Output 39	Output 38	Output 37	Output 36	Output 35	Output 34	Output 33
145	Setting screen form for digital NCK outputs							
	Output 40	Output 39	Output 38	Output 37	Output 36	Output 35	Output 34	Output 33
146	Disable analog NCK inputs							
	Input 8	Input 7	Input 6	Input 5	Input 4	Input 3	Input 2	Input 1
147	Setting screen form for analog NCK inputs							
	Input 8	Input 7	Input 6	Input 5	Input 4	Input 3	Input 2	Input 1
148, 149	Setting value from PLC for analog input 1 of the NCK							
150, 151	Setting value from PLC for analog input 2 of the NCK							
152, 153	Setting value from PLC for analog input 3 of the NCK							
154, 155	Setting value from PLC for analog input 4 of the NCK							
156, 157	Setting value from PLC for analog input 5 of the NCK							
158, 159	Setting value from PLC for analog input 6 of the NCK							
160, 161	Setting value from PLC for analog input 7 of the NCK							
162, 163	Setting value from PLC for analog input 8 of the NCK							
166	Overwrite screen form for analog NCK outputs							
	Output 8	Output 7	Output 6	Output 5	Output 4	Output 3	Output 2	Output 1
167	Setting screen form for analog NCK outputs							
	Output 8	Output 7	Output 6	Output 5	Output 4	Output 3	Output 2	Output 1
168	Disable analog NCK outputs							
	Output 8	Output 7	Output 6	Output 5	Output 4	Output 3	Output 2	Output 1
170, 171	Setting value from PLC for analog output 1 of NCK							
172, 173	Setting value from PLC for analog output 2 of NCK							
174, 175	Setting value from PLC for analog output 3 of NCK							
176, 177	Setting value from PLC for analog output 4 of NCK							
178, 179	Setting value from PLC for analog output 5 of NCK							
180, 181	Setting value from PLC for analog output 6 of NCK							
182, 183	Setting value from PLC for analog output 7 of NCK							
184, 185	Setting value from PLC for analog output 8 of NCK							

5.1.2 Description of signals from PLC to NC (DB10)

DB10 **DBB0, 122, 124, 126, 128** Data block	**Disable digital NCK inputs** Signal(s) to NC (PLC → NC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The digital input of the NCK is disabled by the PLC. It is thus set to “0” in a defined way in the control.	
Signal state 0 or signal transition 1 ——> 0	The digital input of the NCK is enabled. The signal state applied at the input can now be read directly in the NC part program. See Section 2.2.1 for more detailed information.	
Related to	IS “Setting by PLC of digital NCK inputs” (DB10, DBB1) IS “Actual value of digital NCK inputs” (DB10, DBB60) MD 10350: FASTIO_DIG_NUM_INPUTS	

DB10 **DB1, 123, 125, 127, 129** Data block	**Setting by PLC of digital NCK inputs** Signal(s) to NC (PLC → NC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The digital NCK input is set to a defined “1” state by the PLC. This means the signal state at the hardware input and disabling of the input (IS “Disable the digital NCK inputs”) have no effect.	
Signal state 0 or signal transition 1 ——> 0	The signal state at the NCK input is enabled for read access by the NC part program. However, the state can be accessed only if the NCK input is not disabled by the PLC (IS “Disable digital NCK inputs” = 0). See Section 2.2.1 for more detailed information.	
Related to	IS “Disable digital NCK inputs” (DB10, DBB0) IS “Actual value for digital NCK inputs” (DB10, DBB60) MD 10350: FASTIO_DIG_NUM_INPUTS	

DB10 **DBB4, 130, 134, 138, 142** Data block	**Disable digital NCK outputs** Signal(s) to NC (PLC → NC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The digital NCK output is disabled. “0V” is output in a defined way at the hardware output.	
Signal state 0 or signal transition 1 ——> 0	The digital output of the NCK is enabled. As a result, the value set by the NC part program or the PLC is output at the hardware output. For further information see Section 2.2.2.	
Related to	IS “Overwrite screen form for the digital NCK outputs” (DB10, DBB5) IS “Setting screen form for the digital NCK outputs” (DB10, DBB7) IS “Setting by PLC for the digital NCK outputs” (DB10, DBB6) MD 10360: FASTIO_DIG_NUM_OUTPUTS	

DB10 **DBB5, 131, 135, 139, 143** Data block	**Overwrite screen form for digital NCK outputs** Signal(s) to NC (PLC → NC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	On signal transition 0 –> 1 the previous NCK value is overwritten by the setting value (IS "Setting value from PLC for digital NCK outputs"). The previous NCK value, which, for example, was directly set by the part program, is lost. The signal status defined by the setting value forms the new NCK value. For more information please see Section 2.2.2	
Signal state 0 or signal transition 1 ——> 0	As the interface signal is only evaluated by the NCK on signal transition 0 –> 1 it must be reset to "0" again by the PLC user program in the next PLC cycle.	
Special cases, errors, ...	Note: The PLC interface for the setting value (DB10, DBB6) is used both by the overwrite screen form (for signal transition 0 –> 1) and the setting screen form (for signal state 1). Simultaneous activation of the two screen forms via the PLC user program must be avoided.	
Related to	IS "Disable digital NCK outputs" (DB10, DBB4) IS "Setting screen form of the digital NCK outputs" (DB10, DBB7) IS "Setting value from PLC for digital NCK outputs" (DB10, DBB6) MD 10360: FASTIO_DIG_NUM_OUTPUTS	

DB10 **DBB6, 132, 136, 140, 144** Data block	**Setting by PLC of digital NCK outputs** Signal(s) to NC (PLC → NC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The signal status for the digital hardware output can be changed by the PLC with the setting value. There are two possibilities: 1. With the 'overwrite screen form': On signal transition 0 –> 1 the PLC overwrites the previous 'NCK value' with the 'setting value'. This is the new 'NCK setting value'. 2. With the 'setting screen form': On signal state 1, the 'PLC value' is activated. The value used is the 'setting value'. On setting value "1", signal level 1 is output at the hardware output; on "0", 0 signal level is output. The corresponding voltage values are given in **References:** /PHF/, SINUMERIK FM-NC NCU Manual **References:** /PHD/, SINUMERIK 840D NCU Manual For more information please see Section 2.2.2	
Signal state 0 or signal transition 1 ——> 0	As the interface signal is only evaluated by the NCK on signal transition 0 –> 1 it must be reset to "0" again by the PLC user program in the next PLC cycle.	
Special cases, errors, ...	Note: The PLC interface for the setting value (DB10, DBB6) is used both by the overwrite screen form (for signal transition 0 –> 1) and the setting screen form (for signal state 1). Simultaneous activation of the two screen forms via the PLC user program must be avoided.	
Related to	IS "Disable digital NCK outputs" (DB10, DBB4) IS "Overwrite screen form of the digital NCK outputs" (DB10, DBB5) IS "Setting screen form of the digital NCK outputs" (DB10, DBB7) MD 10360: FASTIO_DIG_NUM_OUTPUTS	

DB10 **DBB7, 133, 137, 141, 145** Data block	**Setting screen form for digital NCK outputs** Signal(s) to NC (PLC → NC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	Instead of the NCK value, the PLC value is output at the digital hardware output. The PLC value must first be deposited in IS “Setting value from PLC for digital NCK outputs”. The current NCK value is not lost. For more information please see Section 2.2.2	
Signal state 0 or signal transition 1 ——> 0	The NCK value is output at the digital hardware output.	
Special cases, errors, ...	Note: The PLC interface for the setting value (DB10, DBB6) is used both by the overwrite screen form (for signal transition 0 –> 1) and the setting screen form (for signal state 1). Simultaneous activation of the two screen forms via the PLC user program must be avoided.	
Related to	IS “Disable digital NCK outputs” (DB10, DBB4) IS “Overwrite screen form of the digital NCK outputs” (DB10, DBB5) IS “Setting value from PLC for the digital NCK outputs” (DB10, DBB6) MD 10360: FASTIO_DIG_NUM_OUTPUTS	

DB10 **DBB146** Data block	**Disable analog NCK inputs** Signal(s) to NC (PLC → NC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The analog input of the NCK is disabled by the PLC. It is thus set to “0” in a defined way in the control.	
Signal state 0 or signal transition 1 ——> 0	The analog input of the NCK is enabled. This means that the analog value at the input can be read directly in the NC part program if the setting screen form is set to 0 signal by the PLC for this NCK input. See Section 2.4.1 for more detailed information.	
Related to	IS “Setting screen form of the analog NCK inputs” (DB10, DBB147) IS “Setting value from PLC for the analog NCK inputs” (DB10, DBB148) IS “Actual value of the analog NCK inputs” (DB10, DBB199 ...) MD 10300: FASTIO_ANA_NUM_INPUTS	

DB10 **DBB147** Data block	**Setting screen form of analog NCK inputs** Signal(s) to NC (PLC → NC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The setting value from the PLC acts as the enabled analog value (IS “Setting value from PLC for analog NCK inputs”).	
Signal state 0 or signal transition 1 ——> 0	The analog value at the NCK input is enabled for read access by the NC part program on condition that the NCK input is not disabled by the PLC (IS “Disable analog NCK inputs” = 0). See Section 2.4.1 for more detailed information.	
Related to	IS “Disable analog NCK inputs” (DB10, DBB146) IS “Setting value from PLC for the analog NCK inputs” (DB10, DBB148 to 163) IS “Actual value of analog NCK inputs” (DB10, DBB199 to 209) MD 10300: FASTIO_ANA_NUM_INPUTS	

DB10 **DBB148–163** Data block	**Setting value from PLC for analog NCK inputs** Signal(s) to NC (PLC → NC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	With this setting value a defined analog value can be set by the PLC. With IS “Setting screen form of analog NCK inputs”, the PLC selects whether the analog value at the hardware input or the setting value from the PLC is to be used as the enabled analog value. The setting value from the PLC becomes active as soon as IS “Setting screen form” is set to “1”. The setting value from the PLC is specified as a fixed point number (16 bit value including sign) in 2’s complement (see Section 2.5.1). For more information please see Section 2.4.1.	
Related to	IS “Disable analog NCK inputs” (DB10, DBB146) IS “Setting screen form of analog NCK inputs” (DB10, DBB147) IS “Actual value of analog NCK inputs” (DB10, DBB199 to 209) MD 10300: FASTIO_ANA_NUM_INPUTS	

DB10 **DBB166** Data block	**Overwrite screen form of analog NCK outputs** Signal(s) to NC (PLC → NC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	On signal transition 0 –> 1, the previous NCK value is overwritten by the setting value (IS “Setting value from PLC for analog NCK outputs”). The previous NCK value which, for example, was directly set by the part program, is lost. The analog value specified by the PLC setting value forms the new NCK value. For more information please see Section 2.4.2.	
Signal state 0 or signal transition 1 ——> 0	As the interface signal is only evaluated by the NCK on signal transition 0 –> 1 it must be reset to “0” again by the PLC user program in the next PLC cycle.	
Special cases, errors, ...	Note: The PLC interface for the setting value is used both by the overwrite screen form (on signal transition 0 –> 1) and the setting screen form (signal state 1). Simultaneous activation of the two screen forms must be avoided via the PLC user program.	
Related to	IS “Disable analog NCK outputs” (DB10, DBB168) IS “Setting screen form of analog NCK outputs” (DB10, DBB167) IS “Setting value from PLC for the analog NCK outputs” (DB10, DBB170 – 185) MD 10310: FASTIO_ANA_NUM_OUTPUTS	

DB10 **DBB167** Data block	**Setting screen form of analog NCK outputs** Signal(s) to NC (PLC → NC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	Instead of the NCK value, the PLC value is output at the analog hardware output. The PLC value must first be stored in IS “Setting value from PLC for the analog NCK outputs”. The current NCK value is not lost. For more information please see Section 2.4.2.	
Signal state 0 or signal transition 1 ——> 0	The NCK value is output at the analog hardware output.	

DB10 **DBB167** Data block	**Setting screen form of analog NCK outputs** Signal(s) to NC (PLC → NC)
Special cases, errors, ...	Note: The PLC interface for the setting value is used both by the overwrite screen form (on signal transition 0 –> 1) and the setting screen form (signal state 1). Simultaneous activation of the two screen forms must be avoided via the PLC user program.
Related to	IS “Disable analog NCK outputs” (DB10, DBB168) IS “Overwrite screen form of analog NCK outputs” (DB10, DBB166) IS “Setting value from PLC of the analog NCK outputs” (DB10, DBB170–185) MD 10310: FASTIO_ANA_NUM_OUTPUTS

DB10 **DBB168** Data block	**Disable analog NCK outputs** Signal(s) to NC (PLC → NC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The analog output of the NCK is disabled. “0V” is output in a defined way at the hardware output.	
Signal state 0 or signal transition 1 ——> 0	The analog output of the NCK is enabled. As a result, the value set by the NC part program or the PLC is output at the hardware output. For more information please see Section 2.4.2.	
Related to	IS “Overwrite screen form of analog NCK outputs” (DB10, DBB166) IS “Setting screen form of analog NCK outputs” (DB10, DBB167) IS “Setting value from PLC of the analog NCK outputs” (DB10, DBB170–185) MD 10310: FASTIO_ANA_NUM_OUTPUTS	

DB10 **DBB170–185** Data block	**Setting value from PLC for analog NCK outputs** Signal(s) to NC (PLC → NC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	With this setting value, the value for the analog hardware output can be changed by the PLC. There are two possibilities: 1. With the ’overwrite screen form’: On signal transition 0 –> 1 the PLC overwrites the previous ’NCK value’ with the ’setting value’. This is the new ’NCK setting value’. 2. With the ’setting screen form’: On signal state 1, the ’PLC value’ is activated. The value used is the ’setting value’. The setting value from the PLC is specified as a fixed point number (16 bit value including sign) in 2’s complement (see Section 2.5.1). For more information please see Section 2.4.2.	
Signal state 0 or signal transition 1 ——> 0	As the interface signal is only evaluated by the NCK on signal transition 0 –> 1 it must be reset to “0” again by the PLC user program in the next PLC cycle.	
Special cases, errors, ...	Note: The PLC interface for the setting value is used both by the overwrite screen form (on signal transition 0 –> 1) and the setting screen form (signal state 1). Simultaneous activation of the two screen forms must be avoided via the PLC user program.	
Related to	IS “Disable analog NCK outputs” (DB10, DBB168) IS “Overwrite screen form of analog NCK outputs” (DB10, DBB166) IS “Setting screen form of analog NCK outputs” (DB10, DBB167) MD 10310: FASTIO_ANA_NUM_OUTPUTS	

5.1.3 Overview of signals from NC to PLC (DB10)

DB10	Signals from NC interface NC → PLC							
DBB	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
60	Actual value for digital NCK inputs							
					On-board inputs §)			
					Input 4	Input 3	Input 2	Input 1
64	Setpoint for digital NCK outputs							
	Digital inputs without hardware #)				On-board outputs §)			
	Output 8	Output 7	Output 6	Output 5	Output 4	Output 3	Output 2	Output 1
186	Actual value for digital NCK inputs							
	Input 16	Input 15	Input 14	Input 13	Input 12	Input 11	Input 10	Input 9
187	Actual value for digital NCK inputs							
	Input 24	Input 23	Input 22	Input 21	Input 20	Input 19	Input 18	Input 17
188	Actual value for digital NCK inputs							
	Input 32	Input 31	Input 30	Input 29	Input 28	Input 27	Input 26	Input 25
189	Actual value for digital NCK inputs							
	Input 40	Input 39	Input 38	Input 37	Input 36	Input 35	Input 34	Input 33
190	Setpoint for digital NCK outputs							
	Output 16	Output 15	Output 14	Output 13	Output 12	Output 11	Output 10	Output 9
191	Setpoint for digital NCK outputs							
	Output 24	Output 23	Output 22	Output 21	Output 20	Output 19	Output 18	Output 17
192	Setpoint for digital NCK outputs							
	Output 32	Output 31	Output 30	Output 29	Output 28	Output 27	Output 26	Output 25
193	Setpoint for digital NCK outputs							
	Output 40	Output 39	Output 38	Output 37	Output 36	Output 35	Output 34	Output 33

Comments

#) Bits 4 to 7 of the digital inputs and NCK outputs can be processed by the PLC although no equivalent hardware I/Os exist. These bits can therefore be used additionally for data exchange between the NCK and PLC.

§) On the SINUMERIK 840D digital inputs and outputs 1 to 4 of the NCK are on-board inputs and outputs. No hardware I/Os are available for bits 0 to 3 on the FM-NC. These can be processed by the PLC as described under #).

DB10	Signals from NC interface NC → PLC							
DBB	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
194, 195	Actual value for analog input 1 of NCK							
196, 197	Actual value for analog input 2 of NCK							
198, 199	Actual value for analog input 3 of NCK							
200, 201	Actual value for analog input 4 of NCK							
202, 203	Actual value for analog input 5 of NCK							
204, 205	Actual value for analog input 6 of NCK							
206, 207	Actual value for analog input 7 of NCK							
208, 209	Actual value for analog input 8 of NCK							
210, 211	Setpoint for analog output 1 of NCK							
212, 213	Setpoint for analog output 2 of NCK							

214, 215	Setpoint for analog output 3 of NCK
216, 217	Setpoint for analog output 4 of NCK
218, 219	Setpoint for analog output 5 of NCK
220, 221	Setpoint for analog output 6 of NCK
222, 223	Setpoint for analog output 7 of NCK
224, 225	Setpoint for analog output 8 of NCK

5.1.4 Description of signals from NC to PLC (DB10)

DB10 **DBB60,** **186–189** Data block	**Actual value for digital NCK inputs** Signal(s) to PLC (NC → PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	Signal level is 1 active at the digital hardware input of the NCK.	
Signal state 0 or signal transition 1 ——> 0	Signal level 0 is active at the digital hardware input of the NCK. For more information please see Section 2.2.1.	
Special cases, errors, ...	The influence of IS “Disable digital NCK inputs” is ignored for the actual value.	
Related to	IS “Disable digital NCK inputs” (DB10, DBB0) MD 10350: FASTIO_DIG_NUM_INPUTS	

DB10 **DBB64,** **190–193** Data block	**Setpoint for digital NCK outputs** Signal(s) to PLC (NC → PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The NCK value for the digital output currently set (setpoint) is “1”.	
Signal state 0 or signal transition 1 ——> 0	The NCK value for the digital output currently set (setpoint) is “0”. For further information see Section 2.2.2.	
Signal irrelevant for.....	This ’setpoint’ is only output to the hardware output under the following conditions: 1. Output is not disabled (IS “Disable digital NCK outputs”) 2. PLC has switched to the NCK value (IS “Setting screen form for digital NCK inputs”) As soon as these conditions are fulfilled, the setpoint of the digital output corresponds to the ‘actual value’.	
Related to	IS “Disable digital NCK outputs” (DB10, DBB4) IS “Overwrite screen form of the digital NCK outputs” (DB10, DBB5) IS “Setting value from PLC for the digital NCK outputs” (DB10, DBB6) IS “Setting screen form of digital NCK outputs” (DB10, DBB7) MD 10310: FASTIO_DIG_NUM_OUTPUTS	

DB10 **DBB194–209** Data block	**Actual value for analog NCK inputs** Signal(s) to PLC (NC → PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The analog value applied to the analog NCK input is signalled to the PLC. The actual value is set as a fixed point number (16 bit value including sign) in 2's complement by the NCK (see Section 2.5.1). For more information please see Section 2.4.1.	
Signal state 0 or signal transition 1 ——> 0	The effect of the PLC on the analog value (e.g. with IS "Disable analog NCK inputs") is ignored.	
Related to	IS "Disable analog NCK inputs" (DB10, DBB146) IS "Setting screen form of analog NCK inputs (DB10, DBB147) IS "Setting value from PLC for the analog NCK inputs" (DB10, DBB148 to 163) MD 10300: FASTIO_ANA_NUM_INPUTS	

DB10 **DBB210–225** Data block	**Setpoint for analog NCK outputs** Signal(s) to PLC (NC → PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The current set NCK value for the analog output (setpoint) is signalled to the PLC. The setpoint is set as a fixed point number (16 bit value including sign) in 2's complement by the NCK (see Section 2.5.1). For more information please see Section 2.4.2.	
Signal state 0 or signal transition 1 ——> 0	This "setpoint" is only output at the hardware output when the following conditions are fulfilled: 1. Output is not disabled (IS "Disable analog NCK outputs") 2. The PLC has switched to the NCK value (IS "Setting screen form of analog NCK outputs")	
Related to	IS "Disable analog NCK outputs" (DB10, DBB168) IS "Overwrite screen form for analog NCK outputs" (DB10, DBB166) IS "Setting value by PLC of the analog NCK outputs" (DB10, DBB170–185) IS "Setting screen form of analog NCK outputs" (DB10, DBB167) MD 10310: FASTIO_ANA_NUM_OUTPUTS	

■

Notes

6 Example

None

7 Data Fields, Lists

7.1 Interface signals

DB number	Bit, byte	Name	Reference
General		**Signals from PLC to NC**	
10	0, 122, 124, 126, 128	Disable digital NCK inputs	
10	1, 123, 125, 127, 129	Setting digital NCK inputs on the PLC	
10	4, 130, 134, 138, 142	Disable digital NCK outputs	
10	5, 131, 135, 139, 143	Overwrite screen form for digital NCK outputs	
10	6, 132, 136, 140, 144	Setting value from PLC for the digital NCK outputs	
10	7, 133, 137, 141, 145	Setting screen form for digital NCK outputs	
10	146	Disable analog NCK inputs	
10	147	Setting screen form for analog NCK inputs	
10	148–163	Setting value from PLC for the analog NCK inputs	
10	166	Overwrite screen form for analog NCK outputs	
10	167	Setting screen form for analog NCK outputs	
10	168	Disable analog NCK outputs	
10	170–185	Setting value from PLC for the analog NCK outputs	
		Signals from NC to PLC	
10	60, 186–189	Actual value for digital NCK inputs	
10	64, 190–193	Setpoint for digital NCK outputs	
10	194–209	Actual value for analog NCK inputs	
10	210–225	Setpoint for analog NCK outputs	

7.2 Machine data

Number	Identifier	Name	Refer-ence
General ($MN_ ...)			
10300	FASTIO_ANA_NUM_INPUTS	Number of active analog NCK inputs	
10310	FASTIO_ANA_NUM_OUTPUTS	Number of active analog NCK outputs	
10320	FASTIO_ANA_INPUT_WEIGHT	Weighting factor for analog NCK inputs	
10330	FASTIO_ANA_OUTPUT_WEIGHT	Weighting factor for analog NCK outputs	
10350	FASTIO_DIG_NUM_INPUTS	Number of active digital NCK input bytes	
10360	FASTIO_DIG_NUM_OUTPUTS	Number of active digital NCK output bytes	
10362	HW_ASSIGN_ANA_FASTIN	Hardware assignment of external analog NCK inputs	
10364	HW_ASSIGN_ANA_FASTOUT	Hardware assignment of external analog NCK outputs	
10366	HW_ASSIGN_DIG_FASTIN	Hardware assignment of external digital NCK inputs	
10368	HW_ASSIGN_DIG_FASTOUT	Hardware assignment of external digital NCK outputs	
10380	HW_UPDATE_RATE_FASTIO	Update rate of clock-synchronous external NCK I/Os	
10382	HW_LEAD_TIME_FASTIO	Lead time of clock-synchronous external NCK I/Os	
10384	HW_CLOCKED_MODULE_MASK	Processing of external NCK I/Os in synchronism with the clock	
10394	PLCIO_NUM_BYTES_IN	Number of direct read input bytes of PLC I/Os	
10395	PLCIO_LOGIC_ADDRESS_IN	Start address of direct read input bytes of PLC I/Os	
10396	PLCIO_NUM_BYTES_OUT	Number of direct write output bytes of PLC I/Os	
10397	PLCIO_LOGIC_ADDRESS_OUT	Start address of direct write output bytes of PLC I/Os	
10530	COMPAR_ASSIGN_ANA_INPUT_1	Hardware assignment of NCK analog inputs for comparator byte 1	
10531	COMPAR_ASSIGN_ANA_INPUT_2	Hardware assignment of NCK analog inputs for comparator byte 2	
10540	COMPAR_TYPE_1	Parameterization for comparator byte 1	
10541	COMPAR_TYPE_2	Parameterization for comparator byte 2	
Channel-specific ($MC_ ...)			
21220	MULTFEED_ASSIGN_FASTIN	Assignment of input bytes of NCK I/Os for “Multiple feedrates in one block”	V1

7.3 Setting data

Number	Identifier	Name	Refer-ence
General ($MN_ ...)			
41600	COMPAR_THRESHOLD_1	Threshold values for comparator byte 1	
41601	COMPAR_THRESHOLD_2	Threshold values for comparator byte 2	

7.4 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

■

Notes

SINUMERIK 840D/810D/FM-NC Description of Functions, Extension Functions (FB2)

Several Operator Panels and Several NCUs, Distributed Systems (B3)

■

Notes

Brief Description 1

1.1 Topology of distributed system configurations

Properties

Rotary indexing machines, multi-spindle turning machines and complex NC production centers have all or some of the following features:

- More than one NCU owing to large number of axes and channels
- Large dimensions and spatial separation make several control unitss (MMC operator panels and MCP machine control panels) a necessity.
- Modular machine concept, e.g. through distributed switching cabinets

SW 5 offers a flexible general solution for these applications.

Specification of the software version

The software versions specified in this documentation refer to the SINUMERIK 840D control; the parallel software version for the SINUMERIK 810D control (if the function is released, see /BU/, Catalog NC 60.1) is not specified explicitly. The following applies:

Table 1-1 Equivalent software versions

SINUMERIK 840D software version		SINUMERIK 810D software version
4.1 (12.98)	corresponds to	3.1 (12.98)
4.3 (12.97)	corresponds to	2.3 (12.97)
3.7 (03.97)	corresponds to	1.7 (03.97)
5.1 (12.98)	corresponds to	3.1 (12.98)
5.2 (08.99)	corresponds to	3.2 (08.99)
5.3 (04.00)	corresponds to	3.3 (04.00)

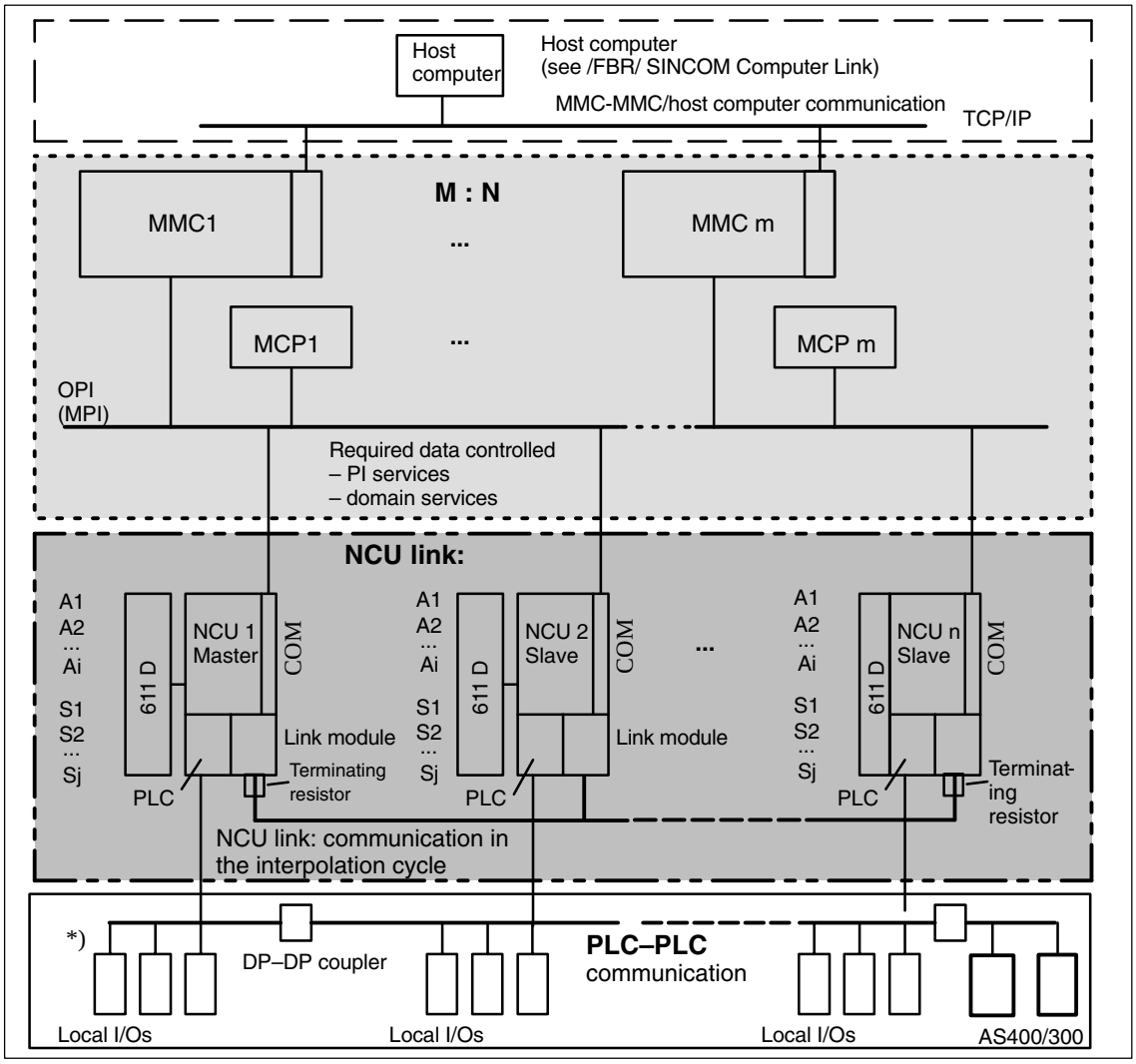


Fig. 1-1 Topology of distributed system configurations

*) The term PLC–PLC communication refers either to
– PLC–PLC cross-communication (Master, slave C.) or
– PLC local I/Os.

The two areas highlighted in the topology display above identify two communications functions to be examined separately in terms of configuration and utilization.

M : N
(m MMC : n NCU)

Assignment of several MMC, MCP control units (M) to several NCUs (N).

- Bus addresses, bus type
- Properties of the MMC
  - Main control panel/secondary control panel
- Dynamic switchover from MMCs/MCPs to other NCUs

For a detailed description of these operations, please refer to Chapter 2.

NCU link

The functions for the NCU link are based on additional communication between the NCUs in the interpolation cycle. The NCU link allows:

- Subordination of a physical axis to several different NCUs
- Interpolation across the NCUs
- An increase in the number of usable axes for an NCU grouping
- An increase in the number of channels for an NCU grouping
- Provision of axis data and signals on the NCU to which a non-local axis is temporarily assigned
- User communication via the NCU grouping by means of link variables

For more information about this topic, please refer to Chapters 1 and 2.

Host computer

Communication between the master computers and control units is described in
References: /FBR/, SINCOM Computer Link

PLC–PLC communication

DP Master, DP Slave, DP–DP coupler, cross-communication via PBK

Bus capacities

The buses illustrated in the above topology diagram are specially designed for their transmission tasks. The following communication specifications are shown in the next figure:

- Number of bus nodes
- Baud rate
- Synchronization

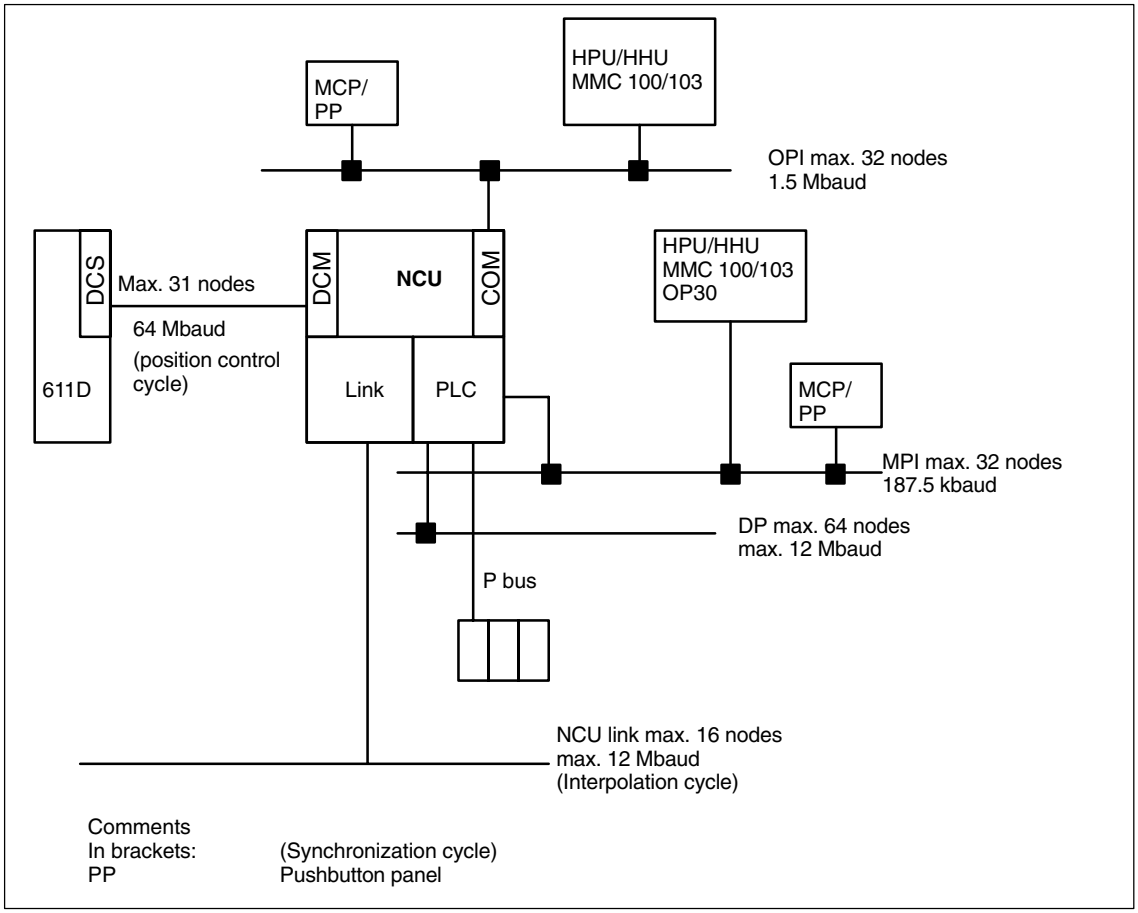


Fig. 1-2 Bus properties

7-layer model structure

Communication takes place on the following protocol layers:

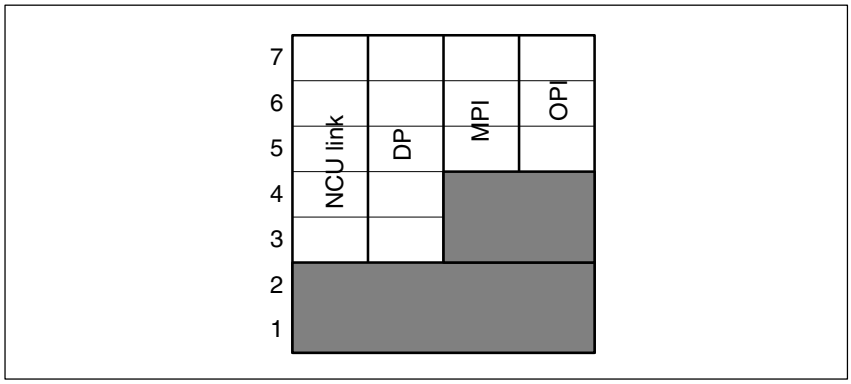


Fig. 1-3 Protocol layers of the 7-layer model

The NCU link and DP can operate faster because they are assigned directly to layer 2.

1.2 Several operator panels and NCUs with control unit management (option)

Introduction

The plant configurations must be highly flexible for complex machines such as rotary indexing machines, multi-spindle machines and complex NC production centers.
Often they need

- Several (m) control units (MMC and MCP) owing to large dimensions of machine and physical separation of operator stations,
- Several (n) NCUs owing to large number of axes and channels.

Limitation

The following subsection describes the additional functionality of Software Version 5. These functions are available only in connection with the **Control Unit Management function**.

The standard functionality applies to all SW versions without the option. However different performance grades depending on SW version must also be taken into account; these are described in Chapter 3.

The standard functionality is described in the following subsection.

While the standard functionality supports only certain restricted combinations of MMCs and NCUs, the control unit management option provides a flexible, universal solution for satisfying the requirements above.

1.2.1 System properties

M:N concept

The M:N concept is represented diagrammatically in Fig. 1-1 “Topology”:

Several control units (MMCs and/or MCPs) are connected to several NCUs via a bus system. As the number of components is variable, they are given the indexes m and n; hence the name M:N concept.

This concept allows the user to connect any control units to any NCUs in the system (within the limits imposed by the hardware) via the bus and to switch them over as and when required.

NCU link

NCU link is an additional direct connection between the NCUs, enabling fast communication (see Sections 1.4 and 2.4).

New features

The features of SW 5 including the control unit management option are as follows:

- Independent connection of MMC and MCP
- Two independent MMC-PLC interfaces on each NCU for two autonomous MMC connections
  - MMC and MCP can be switched over together, or the MMC on its own,
  - MMC states:
    - online/active: operation and monitoring
    - online/passive: A window is displayed with header and alarm line and a message indicating “passive” state
    - offline
- Different bus systems (MPI/OPI) between MMC/MCP and NCUs (changes only take effect after power-up)
- MMC function as server / as Main secondary operator panel
- A combination of both fixed MMCs and switchable MMCs can be connected.
- Suppression mechanism (priority-controlled) if more than two MMCs are competing for an NCU connection
- Up to 32 bus nodes (MMCs, MCPs and/or NCUs, see Fig. 1-2)
- PLC controls the switchover process (control unit switchover to toolbox, directory PSP_PROG\m_zu_n.zip)
- Configuration file NETNAMES.INI with new parameters.

Supplementary conditions

At any one time

- a max. of two MMCs can be online on one NCU,
- only one of them can be in an active state.

1.2.2 Hardware

Operator panel

The OP030, OP031 or OP032 operator panels feature a slimline screen, softkeys, a keyboard, interfaces and a power supply. In addition, an MMC 100.2/102 or 103 computer module is attached to OP031/32.

Machine control panel

The machine control panel (MCP) incorporates a keyboard, rotary button pad and interfaces.

Difference between OP030 and OP031/32

OP030 and OP031/32 differ in their assignment options to an NCU:

- The OP030
  can only have a fixed assignment to one NCU.
  It can be used as a second operator panel for this NCU.
  The addresses of the connected partners can be set for this purpose.
- The OP031/32
  can have an active assignment to another NCU via MMC 100.2/102/103 operation.

References

The operator interfaces are described in the Operator's Guides for the relevant operator panels.

/BA/, Operator's Guide
/FBO/, OP030 Operator Interface
/BH/, Operator Components Manual

Buses

The control units (MMCs and/or MCPs) and NCUs are connected by means of the

- MPI bus (Multi-Point Interface, 187.5 kbaud) or
- OPI bus (Operator Panel Interface, 1.5 Mbaud).

It is possible to combine different bus systems within one installation.

Address assignments

Bus nodes each have a unique address on the bus. The NCU uses

- a common address for the NC and PLC on the **OPI**,
- two separate addresses (for NC and PLC) on the **MPI interface**. The following applies:
  - SW 3.1 and higher:
    PLC address can be re-configured with STEP7.
    The default address on the MPI for the PLC is "2".
  - SW 3.2 and higher:
    As regards the addresses on the MPI interface, the following applies to
    a PLC-CPU 315: NC address = PLC address + 1
  - SW 3.5 and higher:
    As regards the addresses on the MPI interface, the following applies to
    a PLC-CPU 314: NC address = PLC address + 1

Default settings for OPI

Address 0 (MMC) is reserved and 13 (NCK) set as the default in the factory state; these addresses should not be assigned to nodes in M:N systems.

- **Address 0** is reserved for PG diagnostics,
- **Address 13** is preset for service/installation and start-up.
  The address can be reconfigured through operation on the MMC. Reserved the address for NCU replacement if possible.

Defaults for MPI

- **Address 2** for PLC
- **Address 3** for NCU

Number of active MMCs on 1 NCU

A maximum of two MMCs (incl. COROS OPs) can be continually connected actively to an NCU. MMCs on the OPI or MPI count in the same way.

Number of MCPs/ HHUs on 1 NCU

As standard, two MCPs and one HHU can be connected to the OPI or MPI interface. The handheld programming unit (HPU) counts as an MMC and an MCP.

Note

The MPI/MCP network rules outlined in the “SINUMERIK 840D Installation Guide” Section 3.1 must be applied.

In particular, an M:N installation must be connected up by means of cables fitted with terminating resistors (identifiable by switch with which these are switched in and out).

1.2.3 Functions

Defining properties

The flexible M:N concept makes it possible to extensively modify the properties of the control units.

The assignment of the MMC properties can either be

- static or
- dynamic.

Static properties

Static system properties are configured in file NETNAMES.INI (see paragraph below). They become effective at power-up and cannot be changed during the runtime:

- Assignmentof bus nodes – bus system
- Combination of different bus systems (OPI, MPI)
- Assignment of MMCs – NCUs (which MMCs can monitor which NCUs)
- MCP changeover
- Suppression priorities at changeover (see below)
- Utilization properties (see Chapter 2 “Properties of MMCs”):
  - Operator panel is the alarm/ data management **server**
  - The control panel is main or secondary **control panel**

Dynamic properties

The dynamic properties can be changed during runtime.

The states:

Online		Offline
Normal MMC operating mode with communication between the MMC and NCU: operation and/or monitoring are possible.		No communication between the MMC and NCU: operation and monitoring not possible.
active	**passive**	
The operator can operate and monitor.	Operator cannot operate. He sees a window with header and alarm line and a message indicating “passive” state	

Control unit switchover is **enabled**.	Control unit switchover is **disabled**.

Operating the M:N function

The M:N function is operated via the “Channel menu” option.

The channel menu is selected via the “Switch over channel” key.
The horizontal softkeys are used for selecting a channel group (MMC 100.2: max. 8, MMC 102/103: max. 24 channel groups); up to eight connections to channels in different NCUs can be set up in one channel group.

The “Channel menu” screen displays all current connections and the associated symbol names.

Note

In the case of errors during power-up (for example, link setup fails), see Chapter 2 (Power-up).

Suppression strategy

If two MMCs are online on one NCU, and a third MMC would like to go online, then the latter can “suppress” one of the two active MMCs . Communication is then interrupted between this MMC and the NCU.

The algorithm responsible for this suppression is driven by priorities configured in the file NETNAMES.INI (see paragraph 2.1.8 “Suppression”).

1.2.4 Configurability

When the M:N system powers up, it must be aware of the existing control units, NCUs and communications links and their properties.

NETNAMES.INI

All this information is contained in the configuration file NETNAMES.INI which is configured before power-up.

This present description is mainly intended to provide the necessary knowledge for correctly setting up this configuration file for the M:N concept.

This means that

- the hardware configuration is displayed,
- the properties of the components are defined, and
- the desired switchovers/assignments are possible.

1.3 Several operator panels and NCUs, standard functionality

The following applies to all M:N applications in which the control unit management option is not implemented. The level of performance is also dependent on SW version.

1.3.1 System properties

SW 3.1 and higher

Two MCPs and one HHU can be connected to the MPI or OPI.

The handheld programming unit (HPU) counts as an MMC and an MCP.

One of the panels must be an OP030.

SW 3.2 and higher

The configurations “One operator panel and up to three NCUs” are available.

The necessary configuration in the NC for the connection of MCPs/HHUs is made with the basic PLC routine (see Description of Functions, P3: Basic PLC program).

Address must be specified in the case of data exchange between PLCs via Profibus DP (PLC-CPU 315 only) or for global data (double addressing).

The following applies to PLC-CPU 315: NC address = PLC address + 1

SW 3.5 and higher

The configurations “One operator panel and up to four NCUs” and one MMC locally in each case are available.

1.3.2 Hardware

Buses

The connection between MMC 100 or MMC 102/103 (or other operator panel CPUs) and NCUs is made via

- MPI bus (Multi-Point Interface, 187.5 kbaud) and/or
- OPI (Operator Panel Interface, 1.5 Mbaud).

The buses can be used to link **m M**MCs and **n N**CU/PLC units.

Up to SW 3.1

The machine control panel (MCP) must always be connected to connector X101 (OPI) on the NCU.

SW 3.2 and higher

Possibility of installing “Several operator panels and NCUs” on the OPI (X101 on NCU) and the MPI (X122 on NCU).

When setting up the link via MPI, the PLC-CPU315 must be used if there are several NCUs, as it is possible to set the NC address with this PLC.

Up to SW 4 Number of bus nodes: max. 16

SW 4.1 and higher Number of bus nodes: max. 32

1.3.3 Functions

SW 3.1 and higher “Several operator panels and several NCUs” available in the basic version.

SW 3.2 and higher Configuration in the NC for the connection of MCPs/HHUs is made with the basic PLC routine
(see Description of Functions, P3: Basic PLC program).

- Switchover of link to another NCU with the softkey labeled “conn...”:

A menu is overlaid where you can select the connections conn_1, ... conn_n (declared in NETNAMES.INI) via softkeys.
The name (name=...) allocated to the connection in NETNAMES.INI is displayed on the softkeys.

A connection to the new NCU is established by pressing the softkey.

- Changeover behavior on OP030:

It is not possible to change to another bus node online. The connection contained in NETNAMES.INI is permanently configured.

- Changeover behavior on MMC 100:

The “Conn...” softkey is only displayed if more than one link is configured in NETNAMES.INI.
When changing to the new NCU, the existing connection to another NCU is interrupted.

MMC applications, at the instant of link changeover, must no longer need the link to the previous NCU (e.g. for active data backup via V.24). Otherwise the control will issue a message if the connection is required.

With regard to the NCU to which the changeover takes place, the MMC behaves as if the system had been restarted. It is positioned in the operating area which is set as the Start operating area.

- Changeover behavior on MMC 102/103:

The “Conn...” softkey is displayed only if the m to n function is activated on the control.
“m to n” is activated in menu “Start-up/MMC/Operator panel”. Connections remain established with any changeover and the applications which have used these connections remain active. After the changeover, the MMC is in the same operating area with respect to the new NCU as it was previously with respect to another NCU.

- Possible faults

The NCU with which the connection is to be established can refuse the connection setup. Cause: NCU faulty or the NCU cannot operate with an additional MMC at this point.

The number of MMCs that an NCU can set up a connection with at any one time is configured in MD 10134 (MM_NUM_MMC_UNITS = number of MMC communications partners possible at any one time). OP030 requires one unit, MMC 100 or MMC 102/103 requires two units.

- Alarms, messages

MMC 100, OP030	MMC 102/103
It is only possible to output the alarms of the NCU with which a connection is currently established.	The alarms and messages of all connected NCUs can be processed simultaneously.

- Alarm text management

MMC 100, OP030	MMC 102/103
Only one version of the alarm texts can be stored on the operator component. The standard alarm texts are stored once in the same formulation for all NCUs. The possible alarms for all connected NCUs must be stored in the one possible area for user alarms.	It is not possible to set up user alarm texts that apply specifically to the NCU (MMC only manages one alarm text file, SW 3.4 MMC 102/103).

- Connection check (MMC 100, MMC 102/103)

The address of a connected NCU (on OPI bus only) can be altered in the “Connections/Service” menu.
The new NCU address is stored on the NCU.

The softkey labeled “Service” is only displayed if the password for “Protection level Service” has been entered.

When the function for changing the address is started up, a direct connection between the MMC and the relevant NCU must always be established to ensure that the address is not programmed more than once on the bus.

Note

When the NCU (service case) is replaced, or there is a backup battery failure, the address settings are lost.

A general reset on the NCU does not delete the NCU address. The address can only be changed via an MMC.
For display of the current connection, an unambiguous channel name should be assigned in MD 20000: CHAN_NAME (channel name).

SW 3.2 – 3.x

Operation of the M : N function

- MMC 102/103
  Select menu “Start-up/MMC/operator panel and choose between

  M : N (parameters for the MMC are stored in file NETNAMES.INI)
  and

  1 : 1 (addresses can be specified via the MMC).
- MMC 100
  The NETNAMES.INI file must be adapted by means of the application tool. The data are then transferred to the MMC.

SW 3.5 and higher

NC can exchange data with the PLC-CPU 314.

SW 4 and higher

The M:N function is operated via the “channel menu option”.

Precondition: Configuration of the NETNAMES.INI file (see /IAD/, Installation & Start-up Guide 840D, section on MMC).

The channel menu is selected via the “Switch over channel” key.
The horizontal softkeys are used for selecting a channel group
(MMC 100/100.2: max. 8, MMC 102/103: max. 24 channel groups); up to eight connections to channels in different NCUs can be set up in one channel group.

The “Channel menu” screen displays all current connections and the associated symbol names.

Note

In the case of errors during power-up (for example, link setup fails), see Chapter 2 (Power-up).

1.3.4 Configurability

SW 3.1 and higher 2 MMCs : 1 NCU

The option of creating a link between two operator panels and one NCU has been implemented in SW 3.1, as the following diagram illustrates. In this case, there is a fixed assignment between the MCP and the NCU.

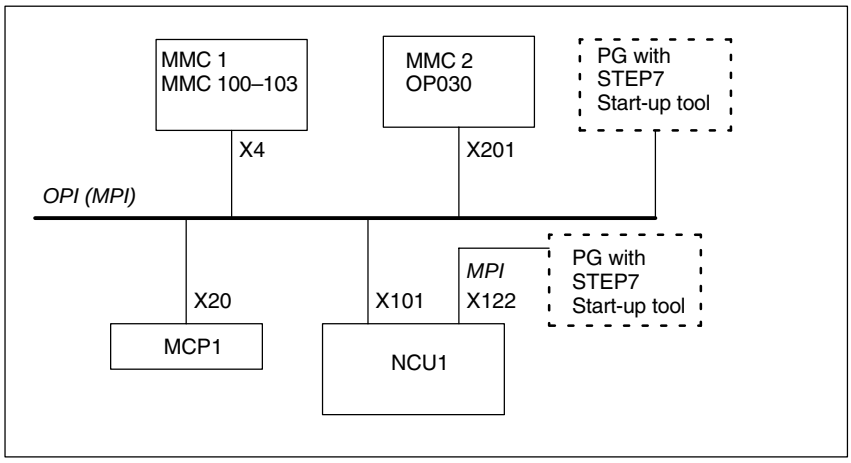


Fig. 1-4 Example configuration for SW 3.1 (m:n corresponds to 2:1)

The operator panels, NCU and machine control panel are all either connected to the OPI bus or the MPI bus. Regarding these components, the network must be **homogeneous**. In SW 3.1, only OP030 may be used as the second operator panel.

The configuration in the diagram would allow, for example, a high-performance operator panel with MMC 100 or MMC 102/103 to be fitted on the front panel of a large machine tool and an OP030 operator panel to be installed in the vicinity of secondary units (or on the rear panel of the machine).

Characteristics

When operating two operator panels in the configuration illustrated above, the user will observe the following system operating characteristics:

1. For the NCU, there is no difference whether the input is from the MMC or OP030 operator panels.
2. Each control unit can visualize selected displays independently of the other unit.
3. Spontaneous events such as alarms are displayed on both control units.
4. The protection level set on one MMC will also apply to the second MMC.
5. The system does not provide for any further coordination between the operator panels.

If the user applies the standard configuration shown in the diagram, then no further special settings are required.

SW 3.2 and higher
1 MMC : 3 NCUs

In SW 3.2 and higher, it is possible to link one operator panel and up to three NCUs (see Fig. 1-5). In this case, the MCP has a fixed assignment to the relevant NCU.

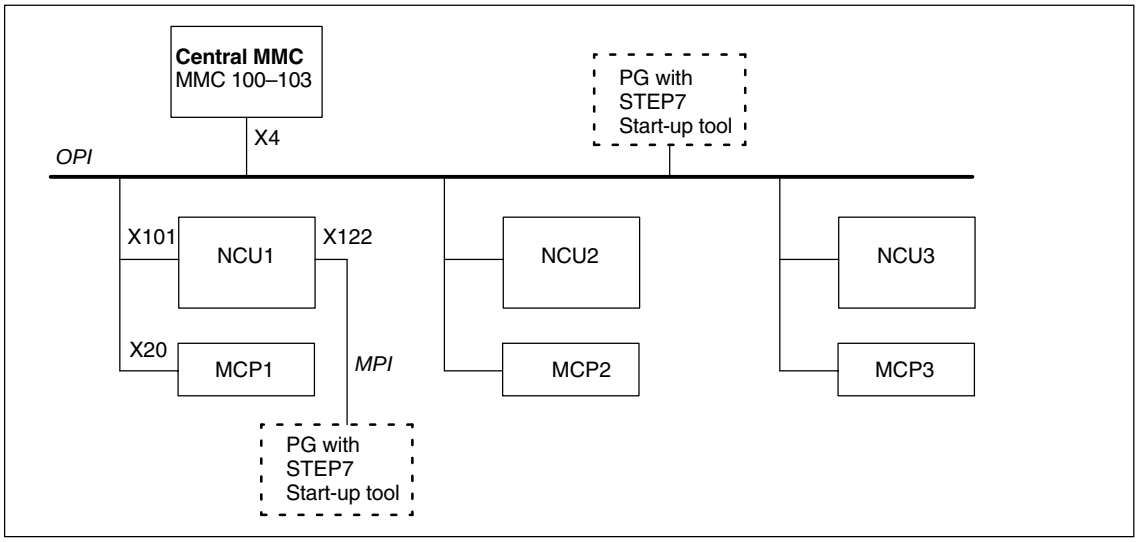


Fig. 1-5 Example configuration for SW 3.2 (m:n corresponds to 1:3)

It is possible to operate several NCUs from one MMC (several autonomous machines or one large machine with several NCUs). At any given time, only one preselected NCU is connected with the MMC for operating sequences. The MMC 100 is provided with one connection only for alarms, whereas the MMC 102/103 is connected to all NCUs for alarms.

Characteristics

The operating characteristics when several NCUs are linked to one MMC are as follows:

1. NCU operation:
   The user must select the NCU to be operated by means of a softkey.
   The operator display in the “Connection” operating area displays the name of the connection and the NCU to which the MMC is currently linked.
2. MMC 100:
   – No application should be active on the connection which is interrupted by the changeover to another NCU (Example: data backup via V.24). System message “V.24 active” is output if an attempt is made to change over the link when an application is active.
   – For the newly established connection, the MMC is positioned in the preset Start operating area (as with MMC restart).
3. MMC 102/103:
   When a link has been set up with another NCU, the last selected operating area (on the previous NCU) is immediately available for the new NCU.
4. PLC-CPU:
   – SW 3.2 and higher
     If the MPI bus is used (connector X122), the PLC must be a PLC-CPU315 to allow variable addressing for the PLC/NC.
   – SW 3.5 and higher
     The PLC-CPU314 can also be used.

OEM solution

As an OEM solution, an MMC 102/103 can be connected via an OPI to up to three NCUs (excluding FM-NC/810D, as it does not have an OPI) as a program and alarm server (m=1, n=3).
In addition, a PG can be connected with a start-up tool.

Note

With SINUMERIK 810D and FM-NC, local MMCs cannot be implemented due to restricted resources.

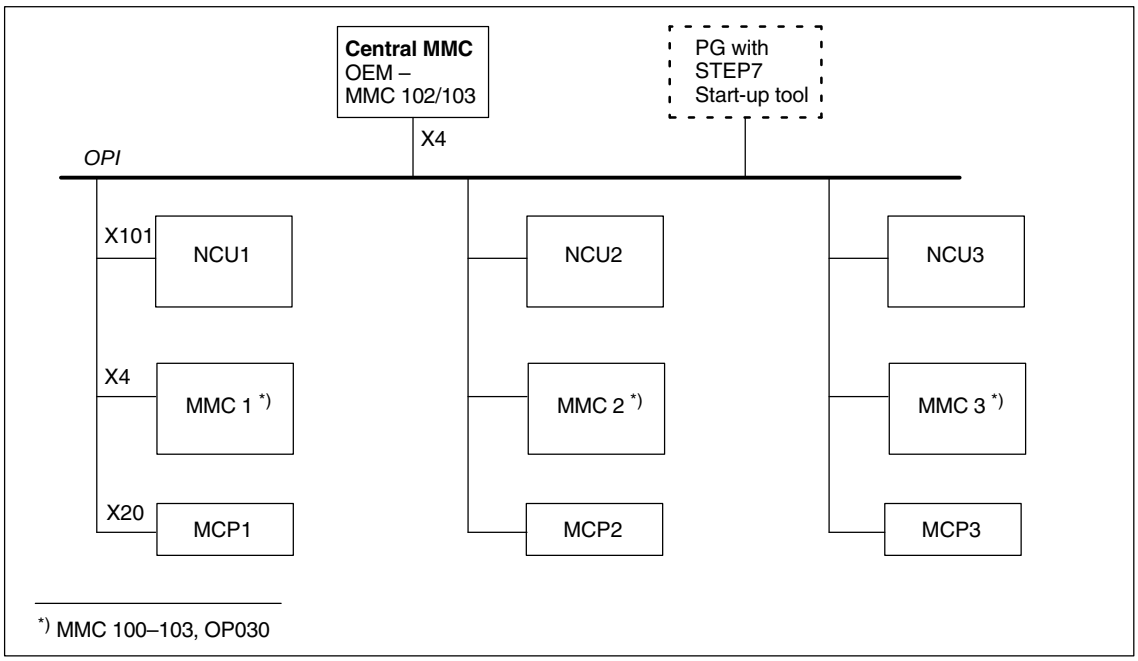


Fig. 1-6 OEM solution as example configuration for SW 3.2: Alarm and program server

Characteristics

The following characteristics are typical of the OEM solution sketched in the above Fig.:

1. NCU operation:
   The user must select the NCU to be operated by means of a softkey.
   The operator screen displays the name of the connection and NCU with which the MMC is currently linked.
2. MMC 100
   can only be connected to a local NCU.
3. MMC 102/103:
   When a link has been set up with another NCU, the last selected operating area (on the previous NCU) is immediately available for the new NCU.

SW 3.5 and higher
1 MMC : 4 NCUs

In addition to the options described above for SW 3.5 and higher, it is also possible to create a link between an operator panel (central MMC) and up to four NCUs, as illustrated in the following diagram. The MCP and the local MMC are permanently assigned to the respective NCU.
A second MMC can be connected to the OPI.

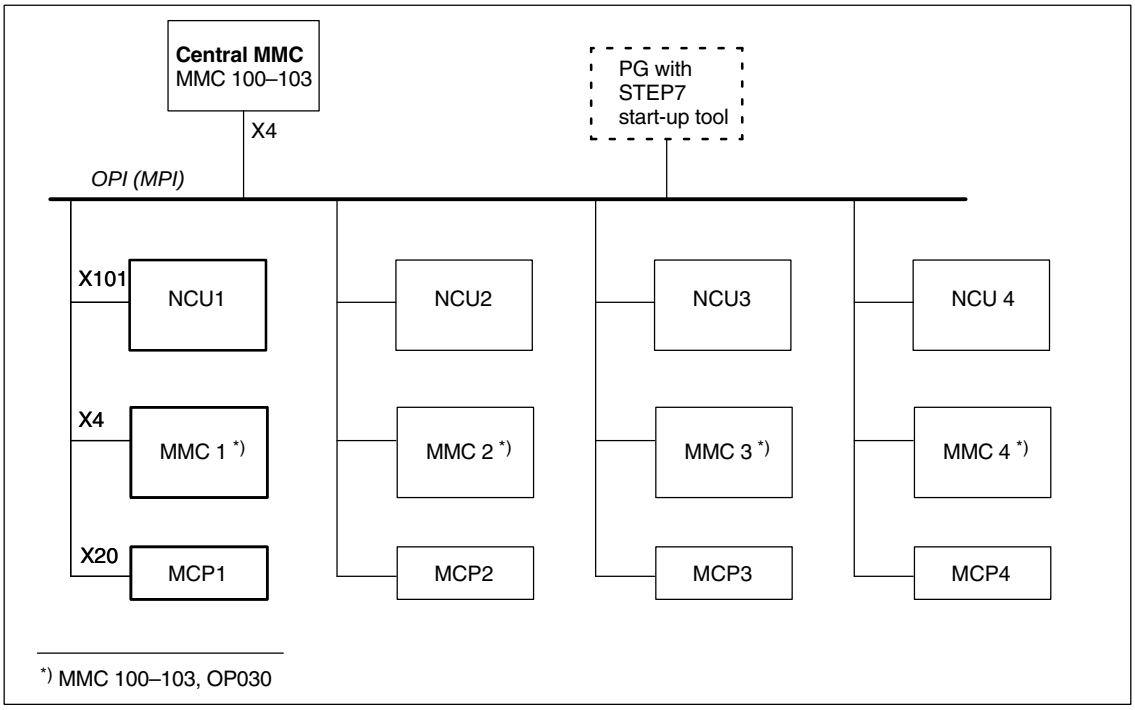


Fig. 1-7 Example configuration for SW 3.5 (m:n corresponds to 1:4)

It is possible to operate several NCUs from one MMC (several autonomous machines or one large machine with several NCUs). At any given time, only one preselected NCU is connected with the MMC for operating sequences. The MMC 100 is provided with one connection only for alarms, whereas the MMC 102/103 is connected to all NCUs for alarms.

Note

With SINUMERIK 810D and FM-NC, local MMCs cannot be implemented due to restricted resources.

Documentation required

References: /BH/, Operator Component Manual
/IAD/, Installation and Start-Up Guide
/FB/ P3, Basic PLC Routine

The following is described in these descriptions:
– MPI/OPI Bus Design, Bus Addresses, /IAD
– Bus Termination, /IAD/, /FB/S7/
– Using basic PLC program to connect MCPs, /FB/P3/
– DIP-FIX settings on MCP, /IAD/

1.4 NCU link

The NCU link, the link between several NCU units of an installation, is used in distributed system configurations.

Introduction NCU link

With high axis/channel requirements, for example, with rotary indexing machines and multi-spindle machines, the computing capacity can exceed the configuration possibilities and storage area offered by one single NCU. Several NCUs interconnected with an NCU link module provide a scalable solution which fully meets the requirements of this type of machine tools. The NCU link module offers fast NCU-NCU communication based on a synchronized 12 MB PROFIBUS interface.

Note

NCU link is available in conjunction with MMC 103.

1.4.1 Types of distributed machines

Machine characteristics

Rotary indexing machines/multi-spindle machines have the following characteristics:

- Global, cross-station units (not assigned to one station):
  - Drum/rotary switching axis and
  - Units that go from station (position) to station (position) such as with rotary indexing machines:
    the rotary axis of the workpiece clamping for multi-face machining operations
    on multi-spindle machines:
    spindle, quill
- Station-related (position-related), fixed-location units:
  - Slides, milling/drilling units used on the part that is changed from station to station for a machining task.

Applications

Rotary indexing machines (**RVM**) and multi-spindle machines (**MS**) are used as highly productive machines in the medium and large batch production. Their main advantage is that many machining steps can be performed on the workpiece in one clamp.

NCU assignments

According to the configuration of the RVM/MS, the many axes of these machines are assigned to different NCUs. (For example, one NCU for each machining unit or group of machining units). The global units are assigned to a separate NCU or distributed accordingly.

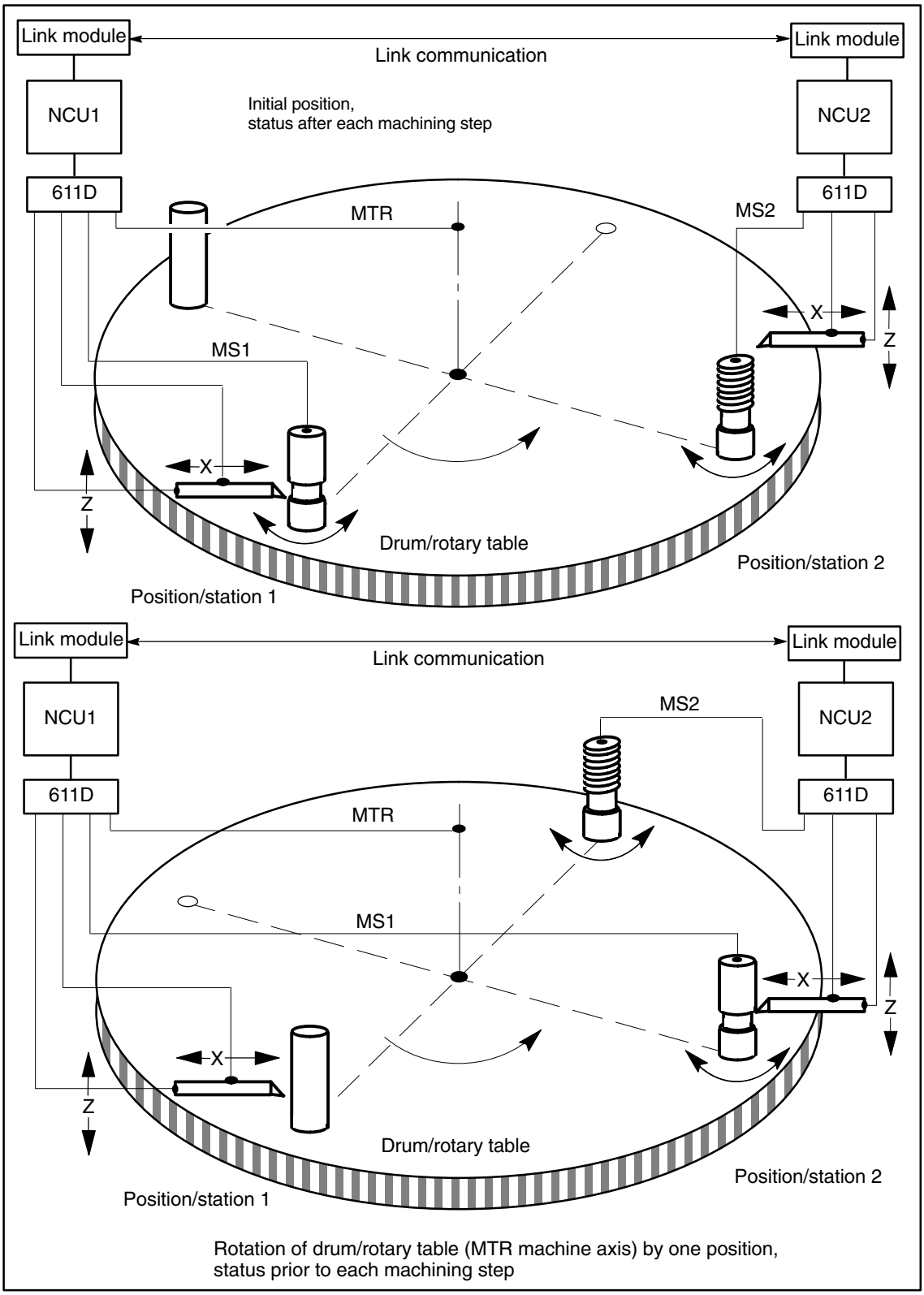


Fig. 1-8 Sectional diagram of a drum changeover

When advancing the rotary table with RVM or the drum with MS, the axis holding the workpiece moves to the next machining unit.

The axis holding the workpiece is now assigned to the machining unit's channel. This is on another channel in the example, but need not be.

As the above diagram "Drum changeover" shows, machine axis MS1, which is physically controlled by NCU1, is brought into position/station2 through rotation of the drum/rotary table. To ensure that a coordinated machining operation between the slide and spindle can now take place in position/station2, the commands for spindle MS1 are transferred in this position by means of link communication. Spindles MS1, MS2, ... are **link axes**.

Physical axes can only be subordinate to the motion control of one NCU channel at any one time. However, the motion control initiative for an axis can be assigned to different NCU channels in succession.

Solutions

To make a physical axis available to several different NCUs, the **Link Axis** property has been introduced. See Sections 1.4.2 and 2.5. In the diagram above, MS1 becomes the link axis from the point of view of NCU2 (bottom diagram) after it has been turned to position/station2.

For variable assignment of channel axes to machine axes according to axis groups, the configuration concept **axis container** has been made available. See Sections 1.4.3 and 2.6.

All link axes which are moved to the next position/station by a particular drum/rotary table must be managed in the same axis container.

1.4.2 Link axes

- Link axes
  - Definition:
    A link axis is an axis
    – whose drive control and position control are subordinate to another NCU or
    – which is the local axis of the NCU concerned, but can be addressed by another NCU.
  - The software option link axis must be installed.
- Coordination
  - The alternate use of a physical axis by several NCUs is dependent on all the relevant NCUs being aware of the status and data of the particular axis and on their ability to coordinate use of the axis.
- Interpolation
  - Local axes and link axes can be interpolated together through motion control by means of one NCU.

- Hardware
  - The NCUs involved in alternate use of axes across NCU limits must be equipped with a **link module**. The NCU link module offers fast NCU-to-NCU communication based on a synchronized 12-Mbaud Profibus interface.

References: /PHD/, Configuring Guide NCU 571–573.2

The following description provides the information required to configure, program and coordinate the distributed machines shown in the drawing.

For details please see Section 2.5.

1.4.3 Flexible configuration

Introduction

On rotary indexing machines/multi-spindle machines, the work-holding axes move from one machining unit to the next. As the machining units are under different NCU channels, it is necessary for the axes holding the workpiece to be dynamically reassigned to the appropriate NCU channel in the event of a station/position change. **Axis containers** are provided for this purpose.

Only one workpiece clamping axis/spindle is active at a time on the local machining unit. The axis container provides the possible connections to all clamping axes/spindles, of which exactly **one** is **activated** for the machining unit.

The following can be assigned via the axis container:

- local axes and/or
- link axes

Switching between the available axes defined in an axis container works by cyclical shifting (rotation) of the entries in the axis container.

The modification can be triggered by the **part program**.

Validity range

Axis containers with link axes are a NCU-cross device (NCU-global) that is coordinated via the control.
It is also possible to have axis containers that are only used for managing local axes.

For details about axis containers, please refer to Section 2.6.

1.4.4 User communication across the NCUs

Definition

- Link variables
  - Every NCU connected by means of a link module can address uniformly accessible **global link variables** for all connected NCUs. Link variables can be programmed in the same was as **system variables**. In general, these variables are defined and documented by the machine manufacturer.
  - Applications for link variables:
    – Global machine states
    – Workpiece clamping open/closed
    – ...
  - Data volume comparatively small
  - Transmission rate very high because information relevant to main run is available for exchange.
  - These system variables can be accessed from the **part program** and from **synchronized actions**. You can configure the size of the memory area for global system variables.
  - All connected NCUs require one interpolation cycle before they can consistently read a new value in a global system variable.

For more information about the global system variables, please refer to 2.7.

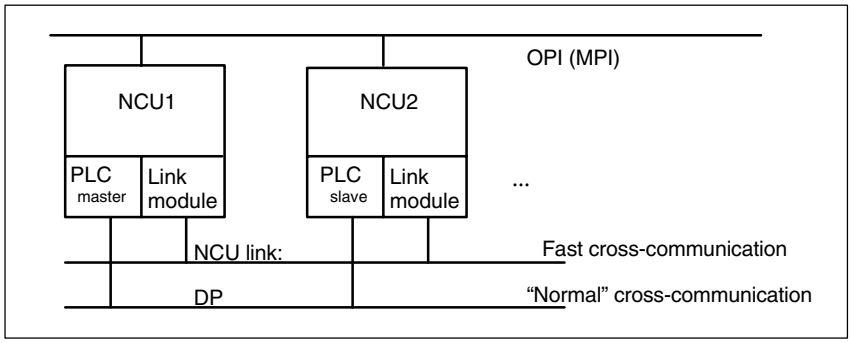


Fig. 1-9 Communication overview

Note

On installations without an NCU link, the link variables can also be used NCU-locally as an additional means of cross-channel communication.
In this case, there is no interpolation cycle between write and read.

References: /FBSY/ Description of Functions Synchronized Actions

■

Notes

Detailed Description 2

2.1 Several operator panels and NCUs with control unit management option

The following chapter provides a detailed description of the preparations and implementation of the operating steps for the M:N concept.

Procedure

1. A number of different configurations are possible with the components of the existing system.
   The user selects one of these options to meet his individual requirements:
   - On the hardware side: by interconnecting components via bus systems
   - On the software side: by configuring static properties using configuration file NETNAMES.INI (see following paragraphs).
     These static properties are made operative during power-up and cannot be altered once the system is running.
   - Control unit switchover function in the PLCs of the relevant NCUs. The PLC control unit switchover function comprises several blocks. These implement the following tasks:
     – Check of switchover conditions
     – Prioritized suppression
     – Switchover

   The PLC SW “Control unit switchover” is supplied as part of the toolbox and can be parameterized if necessary. See Section 2.1.14.

   The option can be used only in PLCs with basic program version 05.03.01 or later.
2. Dynamic properties (such as online/offline states) can be changed when the system is running within the limits specified by the NETNAMES.INI file.

2.1.1 Hardware structure

As described in Chapter 1, a complex system can consist of M (several) control units and n (several) NCUs.
The diagram below illustrates a typical complex system:

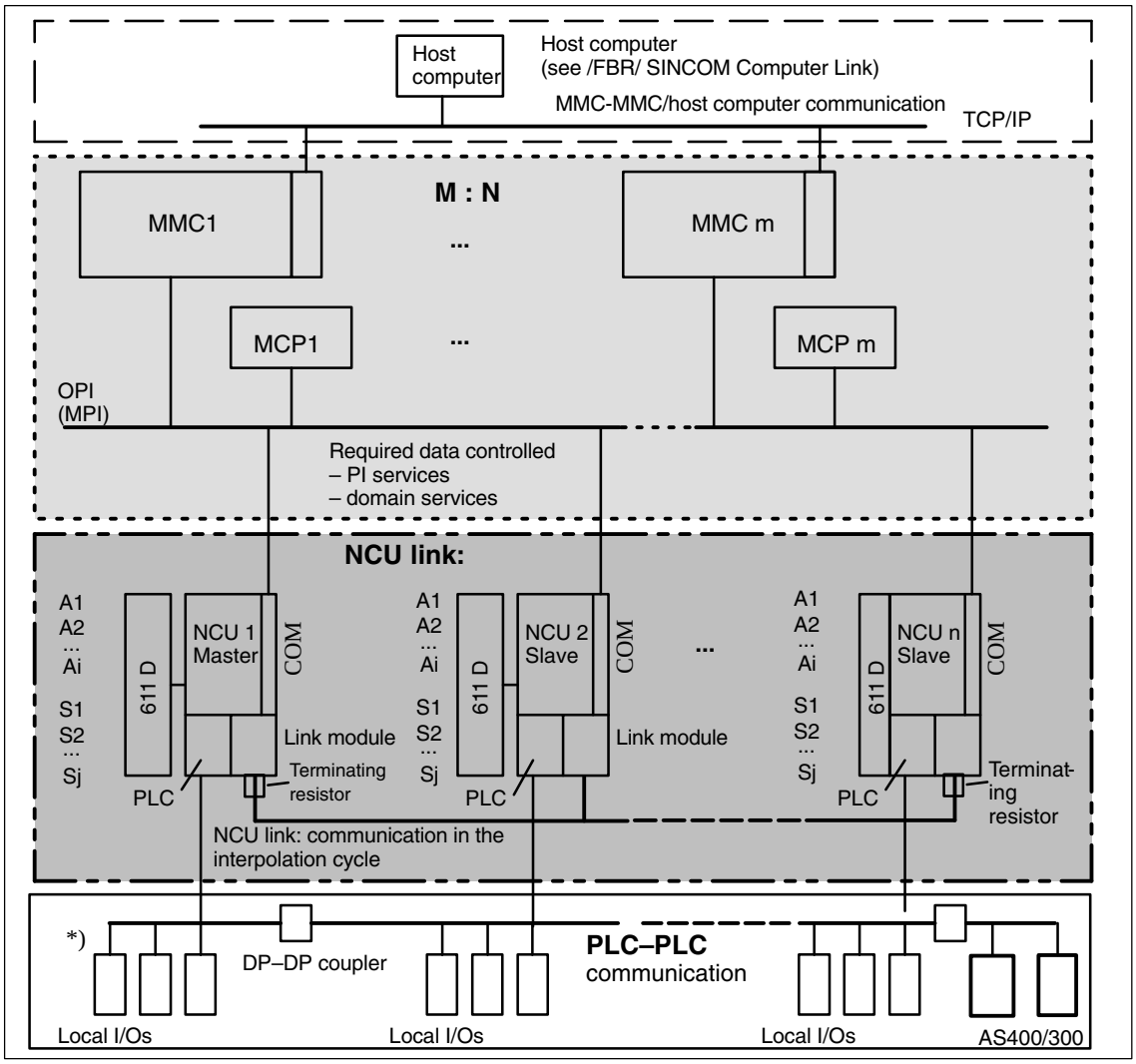


Fig. 2-1 Topology of distributed system configurations

*) PLC–PLC communication refers either to
– PLC–PLC cross-communication (master/slave comm.) or
– PLC-local I/O.

The hardware components are connected to one another via the bus (MPI and/or OPI). The relationships between the bus nodes (identification, properties, assignment and switchover) are software-controlled.

2.1.2 Properties

Client identification

The assignment between bus nodes and the bus system is static and cannot be changed once the system is running. It is configured once in file NETNAMES.INI.

The client identification (CLIENT_IDENT) is composed of bus type and MMC bus address; the MMC uses it when logging on to an NCU to establish an online connection.

Properties

The MMCs in an M:N installation have the following properties:

Server		Control panel	
Maintains a constant 1:N connection		Can be switched to the different NCUs and maintains a constant 1:1 connection (only one at any one time!). Operator can operate and monitor. Connection is set up when the MMC goes online, and is disconnected when it goes offline.	
Alarm server (MMC 103)	**Data management server (MMC 103)**	**Main control panel**	**Secondary control panel**
Receives the alarms from all NCUs in an M:N installation. From its side, a constant 1:N connection is maintained. The process “Receive alarms” is always active and runs in the background.	Establishes all connections configured for it in NETNAMES.INI during power-up and maintains a constant 1:N connection. Can receive, manage and distribute data within the framework of the job list concept.	Example: Main control panel for rotary index machines can be connected to all machining stations.	Example: Secondary control panel for rotary index machines can only be connected to one of two adjacent machining stations.
Cannot be suppressed (see Section 2.1.8)	Cannot be suppressed.	Cannot be suppressed.	Can be suppressed by the main or secondary control panel.

Distribution of properties among the MMC types:

Property	MMC 102/103	MMC 100.2
Server	x	
Main control panel	x	x
Secondary control panel	x	x

MMC is both server and main control panel at the same time

As a server, the MMC maintains constant 1:N connections; as a main control panel it has a switchable 1:1 connection.

If, as a control panel, the MMC is switched to another NCU, it occupies the same connection which it already has as a server. A new connection is not established.

Permissible MMC combinations in an installation

If there is a server (alarm/data management server) in an M:N installation, it is a main control panel at the same time.

In an M:N installation, there can only be one MMC with the following properties:
Windows MMC (MMC 103) : Server and main control panel
or
non-Windows MMC (MMC 100.2) : Main control panel

There can be any number of secondary control panels.

2.1.3 Configuration file NETNAMES.INI

Configuration parameters

As the hardware components can be freely combined (see the previous paragraph “Hardware structure”), it is necessary to provide the system with information about which components are connected, how they are connected to each other and how they interact.

In particular, it is necessary to regulate the competition among the different MMCs for the limited number of available interfaces (suppression, see 2.1.8).

Each MMC has a configuration file NETNAMES.INI for this purpose; this is where the configuration parameters must be stored.

2.1.4 Structure of the configuration file

There is a separate configuration file NETNAMES.INI for **each** MMC.

It is structured as follows:

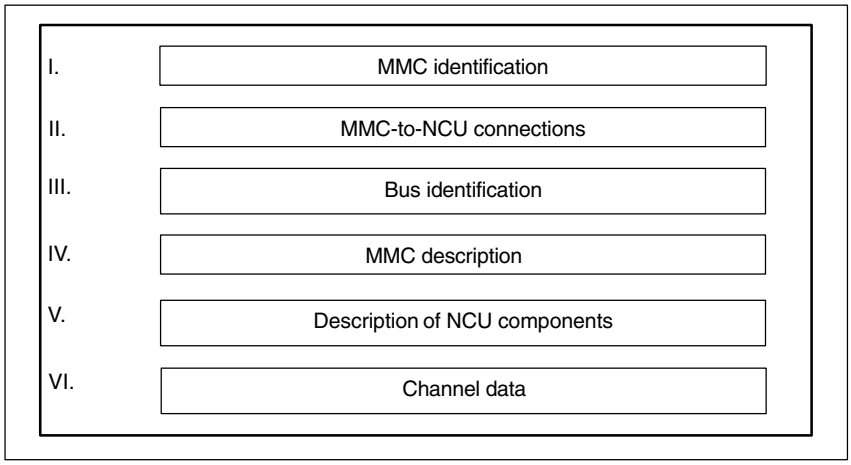


Fig. 2-2 Structure of the configuration file NETNAMES.INI

In the following tables,

- the components which the user may need to adapt or which can be freely named are shown in *italics*,
- Alternative passwords are specified separated by |.

I. MMC identification

Identification of the MMC. The following is valid for NETNAMES.INI:

Element	Explanation	Example
[own]	Header	[own]
owner = *Identifier*	MMC identification	owner = MMC_1

II. MMC-to-NCU connections

Configuring the connections between the MMC and the NCUs:

Element	Explanation	Example
[conn *Identifier*]	Header	[conn MMC_1]
conn_i = *NCU_ID*	Configuring the NCU connection(s) i = 1, ..., 15	conn_1 = NCU_1 conn_2 = NCU_2 ... conn_i = NCU_i

III. Bus identification

Defines which bus the MMC is attached to:

Element	Explanation	Example
[param network]	Header	[param network]
bus = *OPI* | *MPI*	Bus designation	bus = OPI

opi: Operator panel interface with 1.5 Mbaud
mpi: Multi-point interface with 187.5 kbaud

Note

The baud rate is automatically detected on the MMC 100.2.

IV. MMC description

Characterization of the MMC:

Element	Explanation	Example
[param *Identifier*]	Header	[param MMC_1]
mmc_typ = *Type/connection identifier*	MMC characteristics (see below)	mmc_typ = 0x40 MMC is server and main operator panel See below for explanations
mmc_bustyp = *OPI* | *MPI*	Bus the MMC is attached to	mmc_bustyp = OPI
mmc_address = *Address*	MMC_address	mmc_address = 2
mstt_address or mcp_address = *address*	Address of OPI to be switched simultaneously. If not present, there is no OPI to be switched.	mstt_address = 6 or mcp_address = 6

Element	Explanation	Example
name = *Identifier*	Any name allocated by the user (optional, max. 32 characters)	name = MMC_LINKS
start_mode = *ONLINE* | *OFFLINE*	State after power-up. If ONLINE, link is set up via DEFAULT_channel entry to the associated NCU. OFFLINE: No link is set up immediately after power-up.	start_mode = ONLINE (during power-up MMC is connected online to the NCU to which the channel is assigned via channel data (see VI)DE-FAULT_logChanGrp, DE-FAULT_ log_Chan).

Note

Note that the NCU configured via the DEFAULT channel must be the same as the NCU specified under NcddeDefaultMachineName in file **MMC.INI**.

Explanations to **mmc_typ**:
mmc_typ contains type and connection identifiers for the MMC and is transferred to the PLC at switching request. mmc_typ is evaluated as priority for the suppression strategy. See Section 2.1.8.

Bit 7 = ——	(reserved)
Bit 6 = TRUE:	MMC is server (MMC 103) and cannot be suppressed.
Bit 5 = TRUE:	MMC is main control panel.
Bit 4 = TRUE:	MMC is secondary control panel.

The user can specify four additional MMC types which the control unit switchover function of the PLC takes into account in its suppression strategy:

Bit 3 = TRUE:	OEM_MMC_3
Bit 2 = TRUE:	OEM_MMC_2
Bit 1 = TRUE:	OEM_MMC_1
Bit 0 = TRUE:	OEM_MMC_0

If no mmc_typ is entered in file NETNAMES.INI, then the MMC powers up by the method for standard functionality.

V. Description of the NCU component(s)

You must make a separate entry for each NCU component.

Element	Explanation	Example
[param *NCU_ID*]	Header	[param NCU_1]
name= *bel_name*	Any name assigned by the user; is output in the alarm line (optional, max. 32 characters)	name= NCU1
type= *NCU_570* | *NCU_571* | *NCU_572* | *NCU_573*	NCU type	type= NCU_572

Element	Explanation	Example
nck_address = *j*	Address of NCU component on the bus: *j* = 1, 2, ..., 30 *)	nck_address = 14
plc_address = *p*	Address of PLC component on the bus: *p* = 1, 2, ..., 30 *) (only necessary for the MPI bus because j = p for the OPI bus)	plc_address = 14

*) With MPI bus:
Since the associated NCU always occupies the next-higher address than the PLC, the PLC address must not be 31. Address 31 can, for example, be assigned to an MMC.

Note

If the bus node addresses on the MPI bus are configured in conformance with SIMATIC, the configuring engineer can read out the assigned addresses using a SIMATIC programming device and use them to create the NETNAMES.INI file.

VI. Channel data

The control unit switchover option can work only if the control unit knows how channels are assigned to NCUs so that it can set up links between the control unit and NCUs. (Channel menu.)

Strategy

The following operations must be performed:

1. Definition of technological channel groups
2. Assignment of channels to groups
3. Assignment of NCUs to channels
4. Definition of power-up link

NCUs are addressed indirectly on the basis of channel group and channel on the control unit. See 2.1.11 Operator interface.

References: /IAM/, MMC Installation and Start-Up Guide
/BA/, Operator's Guide
/S7HR/, SIMATIC S7–300
/FB/, P3 "Basic PLC Program"

Element	Explanation	Example
[chan *identifier*]	Header (channel menu of MMC_1)	[chan MMC_1]
DEFAULT_logChanGrp = *group*	Channel group of channel during power-up (4.)	DEFAULT_logChanGrp = Mill1
DEFAULT_logChan = *channel*	Selected channel during power-up (4.)	DEFAULT_logChan = channel11
ShowChanMenu = *TRUE* | *FALSE*	TRUE Display channel menu	ShowChanMenu = TRUE

Element	Explanation	Example
logChanSetList = *group list*	List of channel groups (1.)	logChanSet = mill1, mill2
[*group*]	Head (2.)	[mill1]
logChanList = *channel1, channel2,...*	Groups channels separated by comma (2.)	logChanList = channel11, channel12, channel13
[*channel*]	Head (3.)	[channel11]
logNCName = *identifier*	Log. identifier of an NCU (3.)	logNCName = NCU_1
ChanNum = i (i = 1, 2, 3,...)	Number of channel configured for associated NCU (3.)	ChanNum = 1
And so on for all channels in group		
Continue with next group and its channels		

A complete example of how to configure the channel menu can be found in 6.1.

2.1.5 Creating and using a configuration file

Syntax

The configuration file must be generated as an ASCII file. The syntax is the same as that used in Windows *.ini files.

In particular, the following is applicable:

- Passwords must be typed in small letters.
- Comments can be inserted in the parameter file (limited on the left by “;” and on the right by end of line).
- Blanks may be used as separators at any position except in identifiers and passwords.

MMC 100.2, OP030

The NETNAMES.INI file generated on the PC/PG is loaded via the V.24 interface and permanently stored in the FLASH memory of the control units as described in:

References: /IK/, Installation Kit.

MMC 102/103

The NETNAMES.INI file can be processed directly with an editor (in menu “Start-up/MMC/Editor or DOS_SHELL) on the hard disk of the operator component. The NETNAMES.INI file is stored in the installation directory C:\USER\ .

Sample

For a sample configuration file, please refer to Chapter 6.

2.1.6 Power up

Defaults Standard functionality

The following defaults are applied if no NETNAMES.INI configuring file is loaded into the MMC 100.2/OP030 or if the latter cannot be interpreted (standard M:N = 1:1):

- The bus type used is automatically determined
- MMC has address 1
- OP030 has address 10
- NCU and PLC both have address 13 for an OPI bus
- NCU has address 3 and PLC address 2 for an MPI bus

With option

If, however, a special NETNAMES.INI file is created, then it must correspond exactly to the actual network on account of the special features described below.

If an M:N-capable MMC fails to set up a link to the NCU during power-up or in the case of a configuring error, the MMC switches over to OFFLINE operating mode. In this MMC mode the operator can switch over to the area application via the Recall key and then to the start-up area.

Compatibility

The use of the above defaults establishes compatibility with earlier software versions for operation of the panel.

Power-up with MMC 100.2

An MMC 100.2 control unit can only set up an active link to the NCU if the configuration in NETNAMES.INI is correct as described in Section “Structure of the configuration file”. MMC 100 and OP030 can power up parallel on one NCU, because as bus nodes they have different addresses.
The OP030 can be used as a second operator panel that has a fixed assignment to an NCU.

If the configured addresses do not match the real addresses (NC/PLC address), the start-up engineer can use the following key sequence to power-up an NCU that is not configured.

Sequence of operations

1. MMC boots on the NCU with bus address 13, if the NETNAMES.INI was not changed (original works settings).
2. File NETNAMES.INI has been altered, message
   “MMC 100 version xx.xx.xx: waiting for connection ...” is output
   - Press the “1” key, message:
     “choice: ’1’=set new start-address, ’^’ =boot” is displayed
   - Press the “1” key, the bus addresses of all nodes connected to the bus are displayed. The message :
     “Please try one of the shown addresses or press ’^’ to reboot
     ’1’,_,_,_,’6’,_,..._,’D’,_,...” is output.

- Press “D” key and INPUT
- MMC boots on the NCU with bus address 13 (if an NCU is configured under the address found).

3. Enter new NC address in the Start-up/NC/NC address operating area and confirm with “Yes”.
4. NC reset (new address is only valid after NC reset)
5. Configure connection/channel menu in the NETNAMES.INI file and transfer to the MMC
6. After the NCU addresses have been assigned, the bus can be wired for m:n operation.

Note

You can operate an OP030 and an MMC 100.2 on an interface without assigning parameters (various bus addresses are available in the delivery state).

Power-up with MMC 102/103, standard functionality

The MMC 102/103 can power up even if the link to the NCU cannot be made due to errors in the configuring parameters.
An NCU address can be specified explicitly through the entry of a “1 : 1” connection in the “Start-up/MMC operator panel” menu. When the MMC has powered up again, the communications link between the MMC and NCU/PLC will work properly.

Sequence of operations

1. MMC boots on the NCU with bus address 13, if NETNAMES.INI was not changed (original works settings).
2. NCU bus address was changed, the following alarm is output “120201 name: communication failed”
   - Set the connection to 1:1 in the Start-up/MMC/Operator panel operating area and enter “13” as the NC address
   - Confirm with OK and boot the MMC
3. – 6. As on MMC 100.2

Note

In the event of an error, check
– the active bus nodes in the menu
– start-up/NC/NCK addresses (MMC 100.2 and 103),
– start-up/MMC/operator panel (MMC 103).

Power-up with MMC102/103, option

If the control unit switchover option is installed, a configuring problem can be corrected as follows:

1. Select the channel menu with the input key
2. Go to the area switchover screen by pressing Recall
3. Select Start-Up.

Documentation required

References: /BH/, Operator Components Manual
/IAD/, Installation & Start-up Guide
/FB/ P3, Basic PLC Program

The following is described in these descriptions:
– Creation of MPI/OPI bus link, bus addresses, /IAD
– Bus terminator, /IAD/, /FB/S7/
– Using basic PLC program to connect MCPs, /FB/P3/
– DIP-FIX settings on MCP, /IAD/

2.1.7 MMC switchover

With the M:N concept, you can change the control unit properties and states configured in the NETNAMES.INI file during operation.

For example, the user can

- change MMCs (see Section 2.1.9),
- change MCPs (see Section 2.1.13).

Up to two MMCs can be online at the same time on one NCU. A suppression strategy (see Section 2.1.8) is provided to avoid conflicts when more than two MMCs want to go online on one NCU.

The MMC properties are configured for each MMC in the NETNAMES.INI file. If an MMC wants to go online on an NCU via the switchover protocol, its parameters are passed on to the PLC of the respective NCU. The PLC program **Control Unit Switchover** evaluates the parameters:

– Check suppression conditions
– Switchover if necessary

2.1.8 Suppression

Up to two MMCs can be online on each NCU. If this is the case, and another MMC would like to go online, it is necessary to ensure that there are no conflicts. This is achieved by means of the suppression algorithm described below.

Procedure

- The PLC sends an offline request to the MMC to be suppressed.
- It returns a positive or negative acknowledgement to the PLC:
  – If it is positive, the MMC is suppressed (see below). It terminates the communication with the NCU and goes into offline mode.

    Any MCP assigned to the MMC is deactivated by the PLC.
  – A negative acknowledgement is output if processes run on the MMC that cannot be interrupted, e.g. operation via V.24 or data transfer between the NCU and MMC.

    In this case the MMC is not suppressed; it remains online on this NCU.

Suppression strategy

The PLC program “Control Unit Switchover” operates according to the

– priorities of the control units and

– the active processes

The priority depends on parameter **mmc_typ** in configuring file NETNAMES.INI (see paragraph above “Structure of the configuration file”).

If an MMC wants to go online to an NCU, it stores mmc_typ (priority) in its MMC_PLC interface. The Control Unit Switchover program evaluates this according to the following table:

MMC property	Priority
Server	6
Main control panel	5
Secondary control panel	4
OEM-MMC 3	3
OEM-MMC 2	2
OEM-MMC 1	1
OEM-MMC 0	0

Suppression rules

The following rules apply for the MMC suppression:

- High priority suppresses lower or equal priority subject to the following supplementary conditions:
  – Servers cannot be suppressed, as they require a permanent connection to each NCU.
  – MMCs on which the following processes are active cannot be suppressed:
    – Data transfer, e.g. from/to NCU
    – MMC is in the process of switching to the relevant NCU
    – MMC is just changing operating mode.
    –OEM disables switchover
- Equal priority of nodes between active MMC and competitor MMC:
  – The active MMC is suppressed

2.1.9 Connection and switchover conditions

In order to

– allow an MMC which is currently working offline to go online on a particular NCU or

– switch an MMC which is working online over to another NCU,

1. Call the channel menu on this MMC by pressing the channel switchover key
2. Select the channel group via a horizontal softkey
3. Select the appropriate vertical softkey for the channel. See 2.1.10.

The MMC is then switched to online operation or to another NCU, provided that its change in status is not blocked by one of the following conditions (displayed in message line).

Table 2-1 Messages associated with MMC switchover (MMC 103 without message number)

MMC100	Message text
109001	No switchover: Switchover disable set in this PLC
109002	No switchover: Target PLC occupied, try again
109003	No switchover: Switchover disable set in target PLC
109004	No switchover: PLC occupied by higher-priority MMCs
109005	No switchover: No MMC on target PLC can be suppressed
109006	No switchover: Select channel invalid
109007	Channel switchover in progress
109009	Switchover: Error in internal state
109010	Suppression: Error in internal state
109012	Control unit switchover, PLC timeout: 002
109013	Activation rejected

Note

Corresponding messages are output without a message number on the MMC 103.

Additional messages can be generated in the MMC 100.2 and MMC 103 indicating the current status or errors in the configuration or the operating sequence.

For details see
References: /DA/, Diagnostics Guide, Chapter 1

2.1.10 Implementation of control unit switchover

Control unit switchover is an extension of channel switchover.

Channel switchover

“Channel switchover” is a configuring means by which channels of any chosen NCUs can be grouped and named individually. MMC switchover to another NCU is implemented as part of channel switchover functionality.

Configuring of channels is based on file NETNAMES.INI. See 2.1.4.

2.1.11 User interface

Function

You can establish a connection between the MMC unit and one of the connected NCU/PLC units in every operating area.

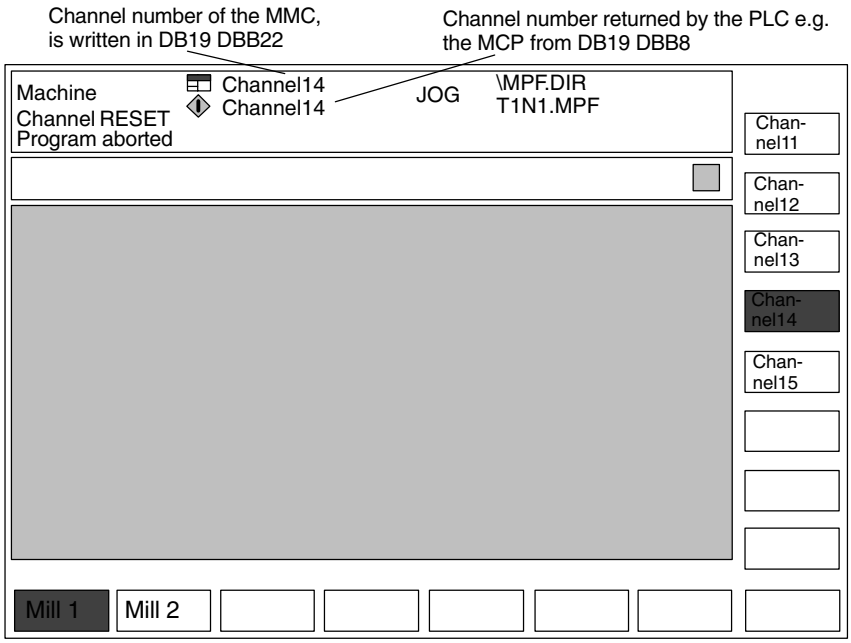


Fig. 2-3 Channel menu (the comments refer to the 1st MMC interface)

Only the channels of the respective group are displayed.

Activate the channel changeover key. The currently existing connection is displayed by means of the highlighted softkeys (horizontal, vertical) if the channel menu is active.

Channel switchover

You can change to other channels by means of the vertically arranged softkeys.

Group switchover

You can switch to another group by means of the softkeys on the horizontal menu (see previous Section); the channels of the currently selected group are now displayed on the vertical softkeys. Switchover to another channel (and if necessary to another NC) only takes place upon activation of a vertical softkey.

NC switchover

You can change to another NC via the vertical softkeys if the channel is not on the current NC.
Procedure: If required, configure a channel area NCs (horizontal softkeys 1–8) and put one channel from each NCU on vertical softkeys.

Note

The softkeys only offer the connections that are really assigned and whose channels are active in the respective NC.

2.1.12 Operator mode switchover

Two MMCs can be online at the same time on one NCU. In order to avoid both gaining write access to the same data or file simultaneously, there are two operating modes, i.e.

- the active and
- the passive operating mode.

Only one of the two MMCs can be active; the other is passive.

The interaction takes place according to the following rules:

Active operating mode

- The user requests active operating mode by pressing a key on the operator panel.
  Active mode has the following characteristics:
  - All operations and operating areas are activated.
  - Operator can operate and monitor.
  - The MCP assigned to the MMC is activated.
  - If data transfer processes (e.g. operation via V.24, reloading part programs, executing the job list, alarms) are running between the other MMC and the joint NCU, it cannot become active immediately.

Passive operating mode

- Passive mode is effective when the other MMC has requested active mode.
  Active mode has the following characteristics:
  - The connection to the NCU remains established.
  - All operations are deactivated.
  - Operator cannot operate. A window is displayed with header and alarm line and a message indicating “passive” state
  - The global menu is activated.
  - Any services initiated before (in active mode) remain active (e.g. operation via V.24, reloading part programs, executing the job list, alarms).
  - The MCP assigned to the MMC is deactivated.

The active operating mode can be selected by 2 different methods:

- Input key or
- Channel switchover key and channel selection

Rules for operating mode changeover

The following rules apply for changing the operating modes (see also Section 2.1.8 “Suppression strategy”):

- An MMC that goes online on an NCU is given active mode on it.
  If another MMC was already active on this NCU, it switches to passive mode if permitted by the PLC.
- If two MMCs are online, the operating mode is changed by pressing the key (“Input”, ENTER, RETURN) used to select the active operating mode.
- Changeover from the active to the passive operating mode might be rejected by the MMC if the current MMC application cannot be aborted or is still in progress. Likewise, active mode cannot be selected on an MMC if the other MMC currently linked to the NCU cannot be switched to passive mode.
- If an online request is issued by an MMC
  - and up to now no MMC is online:
    The MMC issuing the request goes online and switches to active mode.
    Any assigned MCP is activated by the PLC.
  - and an MMC is already online:
    This MMC switches to passive mode.
    The MMC issuing the request goes online and is given active mode.
  - and two MMCs are already online (both of secondary control panel type):
    The MMC active up to now switches to passive mode and is suppressed.
    The MMC issuing the request goes online and is given active mode.
  - and two MMCs are already online (one of main control panel type and in active mode, the other of secondary control panel type and in passive mode):
    The MMC active up to now switches to passive mode. The MMC that was passive up to now is suppressed.
    The MMC issuing the request goes online and is given active mode.
  - and two MMCs are already online (one of secondary control panel type and in active mode, the other of main control panel type and in passive mode):
    The MMC active up to now switches to passive mode and is suppressed.
    The MMC issuing the request goes online and is given active mode.
- If two MMCs are online on one NCU and the active MMC goes offline, it first switches to passive mode. Then the second MMC switches to active mode and the first one disconnects the link to the NCU.

Note

The MMC type is assessed as priority for the suppression strategy. See 2.1.8.

If the active MMC cannot be switched to passive mode, then the competing MMC is switched to passive mode.

2.1.13 MCP changeover

An MCP cannot be switched over independently of the MMC it is assigned to. It can be switched over only if

- the MMC switches over and
- the MCP address is stored in the MMC parameter block or the MMC–PLC interface (see paragraph “Structure of the configuration file” in this section).
- MCP_enable is set in the control unit switchover function on the PLC.

Activating/ deactivating the MCP

If an MCP is assigned to the MMC in the NETNAMES.INI file, it is activated/deactivated as part of the operating mode change. The MCP switchover in the PLC is called by the operating mode change as a subfunction. The parameters for the MCP switchover are stored in the MMC-PLC interface.

MMC is changing operating mode	MCP is
active –> passive	deactivated
passive –> active	activated

2.1.14 PLC program “Control Unit Switchover”

Introduction

“Control unit switchover” is an important controlling function in the overall M:N strategy:

- MMC makes requests regarding the dynamic assignment of MMCs to NCUs according to the configured options in NETNAMES.INI and displays information about existing links.
- The PLC control unit switchover checks the priorities of the requests and the states of the components involved and switches over if necessary.
- The NCU sets signals and evaluates signals required in connection with the control unit switchover function.

The control unit switchover function is a SW package in the toolbox. It is immediately available with its **standard functionality**, but can be modified for special applications according to individual requirements.

Provision is made for two categories of modification:

1. Simple parameterization of standard functionality
2. More fundamental re-configuring of the control unit switchover function

Reasons for more fundamental, user-specific re-configuring (2.) can be as follows:

- Suppression strategy which differs from standard functionality
- Operating mode switchover which differs from standard functionality
- Independent handling of override switch when switching over control unit
- Existence of a 2nd machine control panel on an MMC

Note

The logic in the MMCs (automatic control unit switchover) is fixed. It exists in two variants for the MMC 103.2 and 100.2. The flexibility of the solution in SW 5 includes the following features:

1. Configuring: NETNAMES.INI
2. See below for paramterization of standard functionality of PLC program “Control unit switchover”.
3. See Chapter 6 for more fundamental user-specific re-configuring

(1) Standard functionality

This is implemented as an optional **PLC program**.

Program structure

The control unit switchover program consists of:

1. FB101/DB101: Online/offline operating mode switchover
2. FB102/DB102: Active/passive operating mode switchover
3. FC103: Machine control panel switchover

Every program section is implemented in a separate function block (FB) or function (FC). The variables are stored in a separate instance data block (DB) for each FB. The control unit switchover **main program** is stored in **FB101.** The latter block must be called in an organization block (OB) to activate the functions. FB102 and FC103 are called repeatedly in MB101.

MCP switchover

The MCP switchover is not mandatory. It can be enabled or disabled via FB101/DB101 variable:
MSTT_enable
Assignment in the declaration table:

TRUE	MCP switchover is active
FALSE	MCP switchover is not active, no FC103 call

The MCP assigned to the active MMC is always activated. If the operating mode switchover function is disabled, then the MCP assigned to the last MMC to go online is activated.

Power-up condition:

To prevent the last selected MCP from being activated when the NCU is restarted, input parameter **MCP1BusAdr** should be set to 255 (address of 1st MCP) and **MCP1Stop** to TRUE (disable 1st MCP) when FB1 is called in OB100.

Enabling commands:

When one MCP is switched over to another, any active feed or axis enabling commands will remain operative.

!

Important

After an MCP switchover, the override switch on the new MCP is effective immediately.
The keys actuated at the instant of switchover remain operative. If no MCP is installed on the newly selected MMC, then it will not be possible to cancel the key functions from this MMC. Measures for situations of this type must be implemented in the PLC user program.

Operating mode switchover

The operating mode switchover is not mandatory. It can be enabled or disabled via FB101/DB101 variable:
aktiv_enable
Assignment in the declaration table:

TRUE	Operating mode switchover is active
FALSE	Operating mode switchover is not active, no FB102 call

Error messages

If disturbances occur (e.g. interface signal failure) while the program is running, corresponding alarms/error messages are transferred to data block DB2. 6 **alarms** are implemented:

1. Error in MMC bus address, MMC bus type
2. No confirmation MMC1 offline
3. MMC1 is not going offline
4. No confirmation MMC2 offline
5. MMC2 is not going offline
6. Online-request MMC is not going online (calling MMC)

and an **error message**:

Sign-of-life monitoring error

When the defaults in FB101 are left unchanged, the alarms begin at
DB2.DBX188.0 (1st alarm) and end at
DB2.DBX188.5. (6th alarm)
With variable
DBX_Byte_alarm, the byte value for the 6 alarms can be changed from the default setting of 188. With variable

DBX_Byte_report, the byte value of the operational message can be changed from the default setting of 192.

Mixed mode

Definition:
The term “mixed mode” refers to a state in which a conventional OP without control unit switchover function is connected to the first MMC interface on the NCU. The control unit switchover then operates exclusively on the 2nd MMC interface.

Parameter “MMC_mixed_mode” (variable in FB101/DB101) can be set to switch the operating mode from pure MMC operation to mixed mode.

To ensure that the control unit switchover function operates correctly in mixed mode, the following settings relating to mixed mode must be made:

FB101/DB101 variable:
MMC_mixed_mode
The following assignments must be made for an FB101 call:

TRUE	Mixed mode is active. Control unit switchover operates on the 2nd interface
FALSE	Mixed mode is not active. Control unit switchover operates on both interfaces

Supplementary conditions:

- A machine control panel (MCP) must not be configured for the first online interface. The first online interface is always assigned **active** mode status in mixed operation.
- To allow the second online interface to be assigned active mode status, it is possible to assign active mode status to both online interfaces in mixed operation. However, certain supplementary conditions apply.

Warning

When data are input from both control units simultaneously, there is a risk that inconsistent data will be transferred to the control.

Server mode

Once a server has occupied the online interface of an NCU, it cannot be displaced (suppressed) by any other device.

Processing operating authorization for servers

Three server-related program branches for handling MMC requests are implemented in the control unit switchover program:

1. Request for relinquishing operating focus
2. Request for setting operating focus
3. Relinquish operating focus

Each branch checks/processes the first and second online interfaces.

The requests are positively acknowledged if no switchover disabling commands are active. The “Relinquish operating focus” includes deactivation of the relevant machine control panel.

See also Figures 6-10 to 6-12.

Wait times for acknowledgement signals

To render the program independent of timers, two wait times based on repeated reading of the system time are implemented via SFC64 in the control unit switchover program. The wait times for acknowledgements can be changed if necessary by means of:

FB101/DB101 variables:

waiting_period_1	Wait for activation/online MMC
waiting_period_2	Wait for deactivation/offline MMC
	Wait for MMC sign of life

Values of between 0–32 (seconds) can be assigned to the FB101 variables. These values are entered in ms.

Program integration

If the control unit switchover program is to be called as a function in a higher-level PLC program, then it must be ensured that FB101, FB102 and FC103 and associated instance data blocks DB101, DB102 have not already been used elsewhere.

Initialization

When the NCU is restarted, all signals relating to control unit switchover on the PLC interface in DB19 are set to zero.

Note

Before an NCU is initialized, it must be ensured that it is **not currently linked online** to any MMC.
It may be necessary to perform an MMC restart.

Resetting of interface by PLC

The interface signals relating to control unit switchover can be reset selectively as follows (without RESET on the NCU):

FB101/DB101 variable:
initialization
TRUE Reset signals in DB19 once.
After signal reset the Initialization parameter is automatically reset to FALSE.

Sign-of-life monitoring

As soon as an MMC goes online, it sends a sign of life signal in DB10 DBB108, separately for both MMCs. If an MMC in online mode does not send a sign of life signal for longer than the time set in waiting_period_2, then the PLC program generates message: “Sign-of-life monitoring error”. This message is not cancelled until one of the MMCs is switched from offline to online mode again.

Identifier for MMC “Control unit switchover exists”

In certain operating states, MMCs must be able to detect whether the control unit switchover function exists. The “online request” interface signal in DB19.DBW110 **m_to_n_alive** is provided for this purpose. As soon as block FB101 is called in the PLC, it also sends a sign of life signal, consisting of the cyclic incrementation of **m_to_n_alive** (ring counter).

Generation after adaptations

After static parameters have been modified in FB101, DB101 must be
– deleted,
– generated again,
– called and
stored.

Blocks and functions used

Function blocks	FB101, FB102
Instance data blocks	DB101, DB102
Functions	FC103
DB of interface	DB19
Global data block for error messages	DB2, (DB3)
Timer auxiliary function	SFC64

2.2 Several operator panels and NCUs, standard functionality

The M:N system for all software versions **without** the **Control Unit Management** option is described below. The different performance levels of SW versions from 3.1 onwards are specified in connection with individual functions and as an overview in Chapter 3.

2.2.1 Configurations

Configuration parameters

As it is possible to freely combine hardware components, it is necessary to inform the system which components are combined and in what manner. On the MMC 102/103, this is done by means of an operator dialog in the Start-up area. In the case of the MMC 100/OP030, the configuration parameters are entered through the creation of a configuration file which is loaded for start-up. The file must be structured as described below.

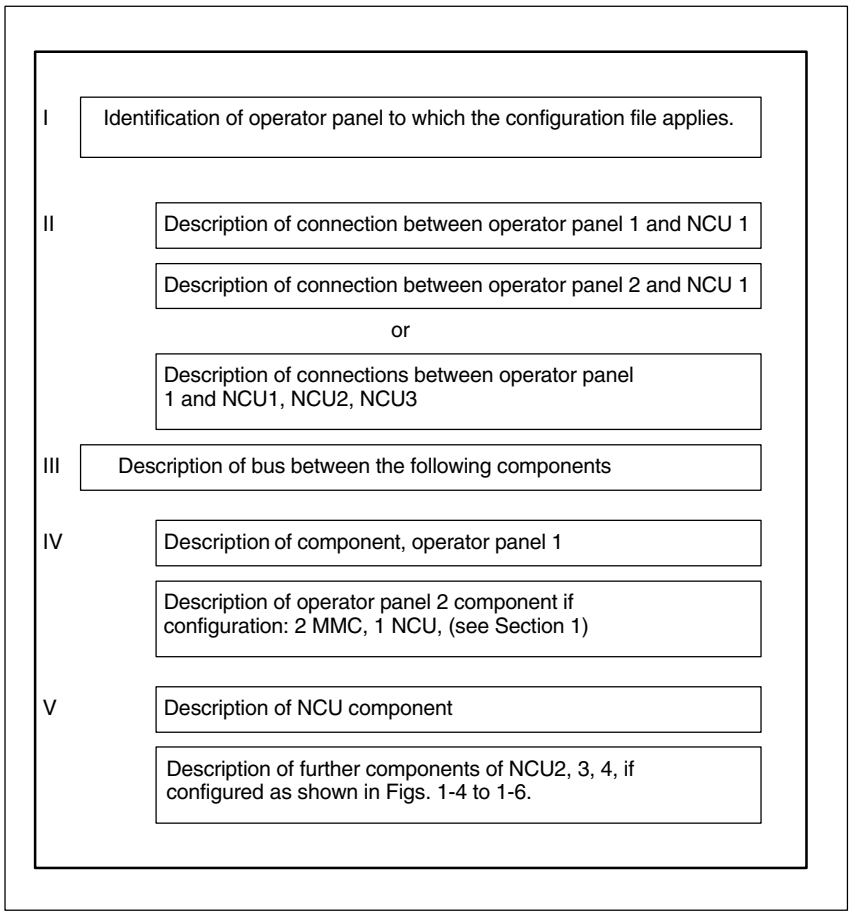


Fig. 2-4 Structure of configuration file NETNAMES.INI

Examples

For complete examples of configuration files, please refer to Chapter 6 of this Description.

Syntactic declarations

The configuration file must be generated as an ASCII file. The syntax is the same as that used in Windows *.ini" files.

In the following tables, the components which the user may need to adapt or which he can name freely are typed in *italics*. Alternative passwords are specified separated by an |. Passwords must be typed in small letters. Comments can be inserted in the parameter file. They must start with ";" and are limited on the right by the end of line. Blanks may be used as separators at any position except for in identifiers and passwords.

Number of configuration files

A configuration file is required for every connected operator panel. The configuration files of different operator panels included in one configuration differ from one another only in the first entry which contains the assignment of the file to a specific panel ([own] see below). For practical purposes, the core of the file is generated just once and then copied for the other panels. The identifier of the operator panel to which the file applies is then inserted in each copy.

I. Identification of operator panel

Identification of operator panel to which the configuration file applies.

Table 2-2 Identification of operator panel

Description division	Formal	Example
Header	[own]	[own]
Next line	owner = *Identifier*	owner = MMC_2

Identifier A description division of an operator panel must be generated with the selected identifier according to IV.

Keywords:
own Introduction identification division
owner Owner

II. Connections

Description of connections from the operator panel components to the NCU to be addressed. An entry of the following type is required for each operator panel.

Table 2-3 Description of connections OP – NCU

Description division	Formal	Example
Header	[conn *Identifier*]	[conn MMC_1]
Next line(s)	conn_i = *NCU_ID*	conn_1 = NCU_1

Identifier A description division of an operator panel must be generated with the selected identifier according to IV.
NCU_ID A description division of the NCU must be generated with the selected NCU identifier according to V.

Keywords:
conn Introduction connection division
conn_1 Password for connection (in SW 3.1 only), otherwise i = 1, 2, ..., 8.

III. Description of bus

The hardware allows links to be implemented via different buses which are differentiated mainly by their baud rates. The bus type used must be specified.

Table 2-4 Description of bus

Description division	Formal	Example
Header	[param network]	[param network]
Next line	bus = opi | mpi	bus = opi

Keywords:

param network	Introduction of network description division
bus	Bus
btss	Operator panel interface with 1.5 Mbaud
mpi	Multi-point interface with 187.5 Kbaud

Note

The baud rate is automatically detected on the MMC 100/100.2.

IV. Description of operator panel component(s)

A separate entry must be generated for every individual panel component connected to the bus. A maximum of two entries in SW 3.x.

Table 2-5 Description of operator panel component

Description division	Formal	Example
Header	[param *Identifier*]	[param MMC_1]
Next lines (optional)	name= *bel_name*	name = MMC_A
(optional)	type= *mmc_100* | *mmc_102* | *op_030*	type = mmc_100
	mmc_address = j	mmc_address = 1

Identifier	Entry for first or second operator component.
bel_name	Arbitrary name of max. 32 characters
mmc_100 | *mmc_102* | *op_030*	Type of operator component
j	Address of operator components on the bus: *j* = 1, 2, ... 15 from SW 4.x:= 1, 2, ... 31

Keywords:

param	Introduction parameters for an (MMC) component
name	Arbitrary name of operator component to be described
type	Type of operator component
mmc_address	Bus address of operator component

V. Description of NCU component(s)

A separate entry must be generated for every single NCU component connected to the bus.

Table 2-6 Description of NCU component

Description division	Formal	Example
Header	[param *NCU_ID*]	[param NCU_1]
Next lines (optional)	name= *bel_name*	name= NCU1
(optional)	type= *ncu_570* | *ncu_571* | *ncu_572* | *ncu_573*	type= ncu_572
*)	nck_address = *j*	nck_address = 13
*)	plc_address = *p*	plc_address = 13

NCU_ID Entry for NCU component. *One* for SW 3.1
bel_name any name of max. 32 characters; with MMC 102/103 the name entered here (e.g. NCU1) is output in the alarm line
ncu_570| NCU type, (ncu_570 not applicable to configuration 1 MMC, 3 NCUs)
ncu_571 | *ncu_572*| *ncu_573*
j Address of NCU component on the bus: *j* = 1, 2, ... 15 from SW 4.x: = 1, 2, ... 31 *)
p Address of PLC component on the bus: *p* = 1, 2, ... 15 from SW 4.x: = 1, 2, ... 31 *)
When bus = mcp *j* and *p* must be set identically.
***)** The following applies when bus = mpi:
As the associated NCU is always assigned the next-higher address than the PLC, the PLC address must not be 31. Address 31 can, for example, be assigned to an MMC.

Keywords:
param Introduction parameters for an (NCU) component
name Arbitrary name of operator component to be described
type Type of operator component
nck_address Bus address of NCU
plc_address Bus address of PLC.

Note

If the bus node addresses on the MPI bus are configured in conformance with SIMATIC, the configuring engineer can read out the assigned addresses using a SIMATIC programming device and use them to create the NETNAMES.INI file.

Defaults

The following defaults are applied if no NETNAMES.INI configuring file has been copied into the MMC 100/OP030 or if the file cannot be interpreted:

- The bus type used is automatically determined
- MMC has address 1
- OP030 has address 10
- NCU and PLC both have address 13 for an OPI bus
- NCU has address 13 (from SW 3.5: **3**) and the PLC has address 2 for MPI bus.

If the network configuration actually corresponds to these default settings, then it is not necessary to explicitly generate and load a NETNAMES.INI file. If, however, a special file is generated, then it must correspond exactly to the actual network on account of the special features described below.

Compatibility

The use of the above defaults establishes compatibility with earlier software versions for operation of the panel.

2.2.2 Switchover of connection to another NCU (SW 3.2 to 3.x)

Operator interface

The data area menu has been extended by the softkey “Connections”. This goes to a submenu in which the connections (conn_1, ... conn_n) declared in NETNAMES.INI are displayed for selection via individual softkeys. The name (name=...) allocated in NETNAMES.INI to the connection is displayed on the softkeys. A connection to the new NCU is established by actuating the corresponding softkey.

In detail, the behavior depends on the type of MMC.

Changeover behavior on OP030

ONLINE change to another bus node is not possible on the OP030. The NETNAMES.INI file contains a permanently configured connection.

Changeover behavior MMC100

The softkey “Connections” is only displayed if more than one connection is implemented in NETNAMES.INI. When changing to the new NCU, the existing connection to another NCU is interrupted. When the changeover takes place, MMC applications must no longer have any need for a connection to the existing NCU (e.g. active data storing via V.24). If this rule is contravened, the control system outputs a corresponding message.
Concerning the NCU to which the changeover takes place, the MMC behaves as with a restart. It is positioned in the operating range preset as start operating range.

Changeover behavior MMC 102/103

The “Connections” softkey is only displayed if the M:N function is activated on the control. The “M:N” function is activated in the “Start-up/MMC/Operator panel” menu. All communications connections remain established with any changeover and the applications which have used these connections remain active. Concerning the new NCU, the MMC is after the changeover in the same operating range as before with another NCU.

Possible defects

With change of the connection to another NCU, it is possible that the NCU with which the connection is to be established rejects this. There may be a defect in the NCU or no other MMC unit can be operated at that time from the NCU any more.
MD 10134: MM_NUM_MMC_UNITS (number of possible simultaneous MMC communications partners) contains the setting which defines how many MMCs can be processed by an NCU at one time. OP030 requires one unit, MMC 100 or MMC 102/103 requires two units.

Alarms/ messages

MMC 100, OP030
Only the alarms of the NCU with which a link is currently active can be output. Acceptance of configuration acc. configuration diagram in Chapter 1, subsection “Configurability”.

MMC 102/103
The alarms and messages of all connected NCUs can be processed simultaneously.

Alarm text storage

MMC 100, OP030
Only one version of the alarm texts can be stored on the operator component. The standard alarm texts are identical for all NCUs and exist in one version. The possible alarms of all connected NCUs must be stored in the single possible range for user alarms.

MMC 102/103
It is not possible to set up user alarm texts that apply specifically to the NCU (MMC only manages one alarm text file, SW 3.4 MMC 102/103).

Link check

MMC 100, MMC 102/103

The address of a connected NCU (on OPI bus only) can be altered in the “Connections/Service” menu.
The new NCU address is stored on the NCU.

The softkey labeled “Service” is only displayed if the password for “Protection level service” has been entered.

When the function is started up, a direct connection between the MMC and the relevant NCU must be established before the address is altered to ensure that the address is not programmed more than once on the bus.
(See paragraph “Power-up” below for instructions on modifying the address.)

Note

With replacement of the NCU (service case) or with failure of the backup battery, the address is no longer stored.
A general reset on the NCU does not delete the NCU address. The address can only be changed via an MMC.
To ensure that the current connection is shown in the basic display, the channel name must be assigned unambiguously in MD 20000: CHAN_NAME (channel name).

2.2.3 Switchover of connection to another NCU (SW 4 and higher)

Note

The channel menu function is an option and must be configured in the “NETNAMES.INI” file.

You can change to the channel menu in all operating areas by activating the channel switchover key. The only change is to the horizontal and vertical softkeys.
The horizontal softkeys are for selecting a channel group (max. 24), up to 8 connections to channels in different NCUs can be set up in one channel group.

The “Channel menu” screen displays all current communication connections and the associated symbol names.

2.2.4 Creating and using the configuration file

MMC 100, OP030

The NETNAMES.INI ASCII file generated on the PC/PG is loaded via the V.24 interface and permanently stored in the FLASH memory of the control units as described in:

References: /IK/, Installation Kit.

MMC 102/103

The NETNAMES.INI file can be processed directly with an editor (in menu “Start-up/MMC/Editor or DOS_SHELL) on the hard disk of the operator component. The NETNAMES.INI file is stored in the installation directory:

C:\MMC2
from Software Version 4 C:\USER\NETNAMES.INI.

2.2.5 Power up

Differences between MMC 100 and MMC 102/103

Owing to the differences in operating and power-up characteristics, different start-up procedures are required for the two MMC types:

- MMC 100 always runs in “M:N” mode, when “M:N” is configured in the NETNAMES.INI file.
- The mode can be set in the “Start-up/MMC/Operator panel” menu on the MMC 102/103. The MMC 102/103 powers up as standard in a “1 : 1” link with an NCU, the NCU address can be specified directly in this case. If the “M:N” mode is set on the MMC 102/103, then the MMC searches the NETNAMES.INI file for the names of the partners specified for this function. The addresses are freely assignable.
  Recommendation: Keep address 0 free (for PG)
  Keep address 13 free (for service case: NCU replacement)
- The OP030 is not functionally capable of “M:N”. It can be used as a second operator panel that is permanently assigned to an NCU (“1 : 1” link). The addresses of the connected partners can be set for this purpose.

Note

It is advisable to make a written record of the procedure (address assignments, etc.) beforehand.

Start-up

The NCUs are assigned bus address 13 in the delivery state. Every NCU on the bus must be allocated its own, unique bus address.

Addresses are assigned in:

- MMC: NETNAMES.INI file
- NCK: “Start-up/NC/NCK address” menu
- MCP: Switches... (address and baud rate if applicable, see also /Start-Up Guide/
  OB100 parameters: ...(see also FB1/P3/).

Note

An NCK address is not deleted with “Delete SRAM” (switch S3= position “1” on NCU).

Power-up with MMC 100

The power-up process is the same as described in Section 2.1.6 for the MMC 100.2.

Power-up with MMC 102/103

The power-up process is the same as described in Section 2.1.6 for the MMC 102/103.

2.2.6 NCU replacement

In the case of NCU replacement or an additional NCU, the procedure is analogous to start-up (see 2.2.5).

Variant 1

1. Establish 1:1 connection between MMC and NCU
2. Power-up MMC on NCU with bus address “13” (see above)
3. Enter new NC address via the Start-up/NC/NC address operating area and boot NCU.
4. Wire bus again for M:N operation

Variant 2

1. The NCU, which is the “power-up NCU” for an MMC connected to the bus, is disabled. (The MMC powers up at the first connection configured in the NETNAMES.INI file)
2. Power-up MMC on NCU with bus address 13 (see above)
3. Enter new NC address via the Start-up/NC/NC address operating area and boot NCU.
4. Activating “Power-up NCU” again

Note

- Bus address 13 must be reserved for servicing purposes (i.e. must not be assigned to a bus node).
- MMC 100/100.2:
  The name length in file NETNAMES.INI (configuring in channel menu) is limited to 5 characters.
- MMC 102/103:
  The data “mst_address” is not evaluated, but used for the purpose of bus node documentation.
  If the channels are on different NCUs, “m:n” must be set in the operating area Start-up/MMC/Operator panel.

Data exchange between NC<–>PLC

In configurations consisting of 1 x MMC and n x NCU, it is often necessary to synchronize the NCs.

The following synchronization options are available:

- NCK I/Os on drive bus (digital, analog, writing of NC and PLC).
- Normal PLC I/Os (I/O link).
- Link via PROFIBUS-DP (PLC-CPU315 required).
- Link via the global data function of SIMATIC S7 (PLC-CPU315 required). This option is also available on the PLC-CPU 314 with SW 3.5 and higher.

2.3 Restrictions in relation to equipment

Rejection of link

On switchover to another NCU, the NCU selected for the new link may reject the connection. The cause may be a defect in the NCU or no additional MMC unit can be accepted. In this case, the MMC 100 automatically switches over to connection 1 after approx. five seconds. MMC 102/103 displays “#” for the variables.

Alarms, messages

Handling of alarms/messages is dependent on the MMC type:

1. MMC 100 / OP030
   Due to the equipment restrictions on driver level and the limited working memory, alarms/messages of only one NCU can be processed simultaneously.
2. MMC 102/103
   The MMC manages only one alarm text file. The NCU name assigned in the NETNAMES.INI file is displayed as the NCU identifier in front of every alarm or message. To obtain user texts specific to the NCU, it is possible to define user areas in the PLC for certain NCUs. The alarms/messages of all connected components can be processed and displayed simultaneously.

Operator interface

The operator interface characteristics depend on the MMC type.

1. MMC 100
   Fields and variables of one NCU can be displayed simultaneously in a window. Alarms and messages are displayed only by the NCU which is connected with the MMC.
   Up to four connections (one active connection (alarms, messages), three other connections) can be displayed simultaneously via user configuration (OEM), whereby all variables of a connection must be contained in one window (window-specific connections).
2. MMC 102/103
   Generally, fields and variables of different NCUs can be displayed in the same window (as OEM application).
   Alarms and messages can be displayed on all NCUs (to which the MMC has a connection).
3. OP030
   OP030 can only be configured as a “1 : 1” connection for an NCU.

When the MMC 100 and MMC 102/103 are used in the standard configuration (Chapter 1, subsection “Configurability”), it is not necessary to configure the operator interface. If variables of different NCUs must be output simultaneously in a display, configuration is necessary.

References: /PK/, SINUMERIK MMC100/EBF Configuration kit

With MMC 100, all variables of a window must proceed from one NCU. With MMC 102/103, a suitable mixing of the variables of different NCUs is permissible.

2.4 NCU link

2.4.1 Introduction

Owing to the limitation on the memory and computing capacity elements, the number of channels or axes per NCU is restricted. A single NCU is not sufficient to fulfill the requirements made by complex and distributed machines, such as multi-spindle and rotary indexing machines. For this reason, the control system and closed-loop axis controls are distributed among several NCUs.

In order to ensure, however, that channels and axes can continue to operate on an interrelated, cross-NCU basis, the system provides so-called

- NCU Link functionality.

This includes:

Functional expansions

The following applications are possible in SW 5 and higher:

- Cross-NCU interpolation (coupling of setpoints, actual values and VDI signals)
- Real exchange of axes
- Cross-NCU access to axis values and axial system variables
- NCU-user communication supported by NCU link variables
- Generation of alarms on the NCU affected by an irregularity, even if the cause of the problem is on another NCU.

2.4.2 Technological description

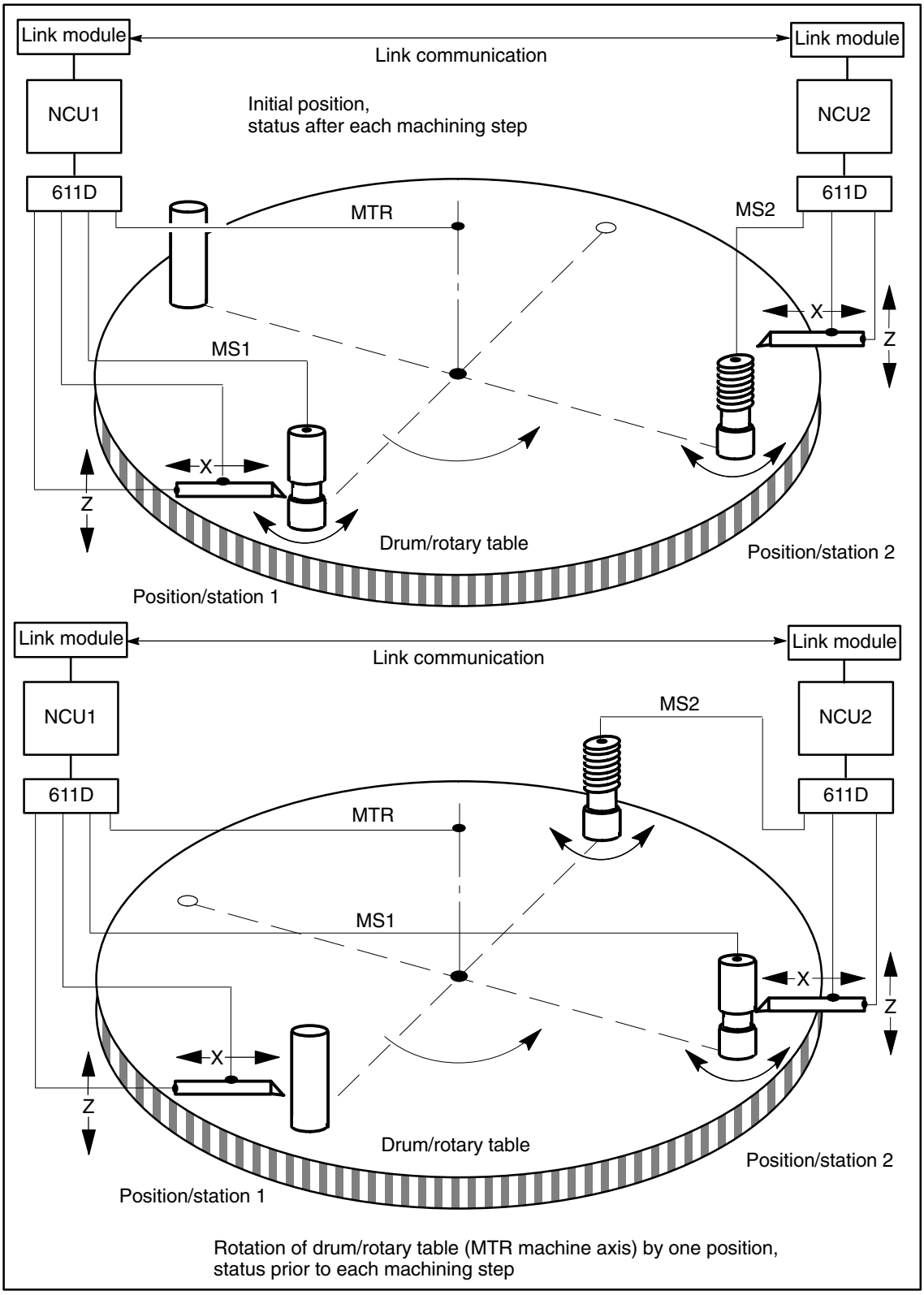


Fig. 2-5 Sectional diagram of a drum changeover

Fig. 2-5 shows the main components of a simple multi-spindle plant. Several spindles are mounted mechanically on the drum, each of which can used to perform a different machining operation. Together with the slide (X and Z axes), they form a machining station which is assigned to one channel. A workpiece is rotated by one spindle.

The workpiece to be machined is loaded and unloaded only once. The tool is mounted on the slide (e.g. X, Z axes). Various different tools can be loaded for each machining operation.

The tool is continuously assigned to the machining station. The workpiece-holding spindles are moved from one machining station to the next.

The spindle can only be checked for the current machining process. The channel must be able to address the slide axes and the current spindle at any given time. Every time a spindle moves on to the next machining unit, however, the spindle addressed by the channel must be a different machine axis. The "**Axis container**" concept solves the variable imaging of channel axes on machine axes. The machine axes might belong to another NCU connected by means of the NCU link. An accessible machine axis belonging to another NCU is referred to as a **link axis** (see 2.5).

The following subjects are closely related to the NCU link function and dealt with in separate subsections.

- Link axes
- Axis container
- User communication across the NCUs
- Configuration of the link grouping

2.5 Link axes

Note

NCU link is available in conjunction with MMC 103.
The corresponding option is required to be able to define the number of available link axes.

Introduction

This subsection describes how an axis (for example, B1 in Fig. 2-6), which is physically connected to the drive control system of NCU2, can be addressed not only by NCU2, but also by NCU1.

Requirements

- The NCUs involved, NCU1 and NCU2, must be connected with fast link communication via the link module.

References: /PHD/, Configuring Guide NCU 571–573.2, link module

- The axis must be configured appropriately through machine data.
- The option link axis must be installed.
- The link communication must be activated with MD 18780: $MN_MM_NCU_LINK_MASK. The link grouping must be configured as described in 2.5.1.

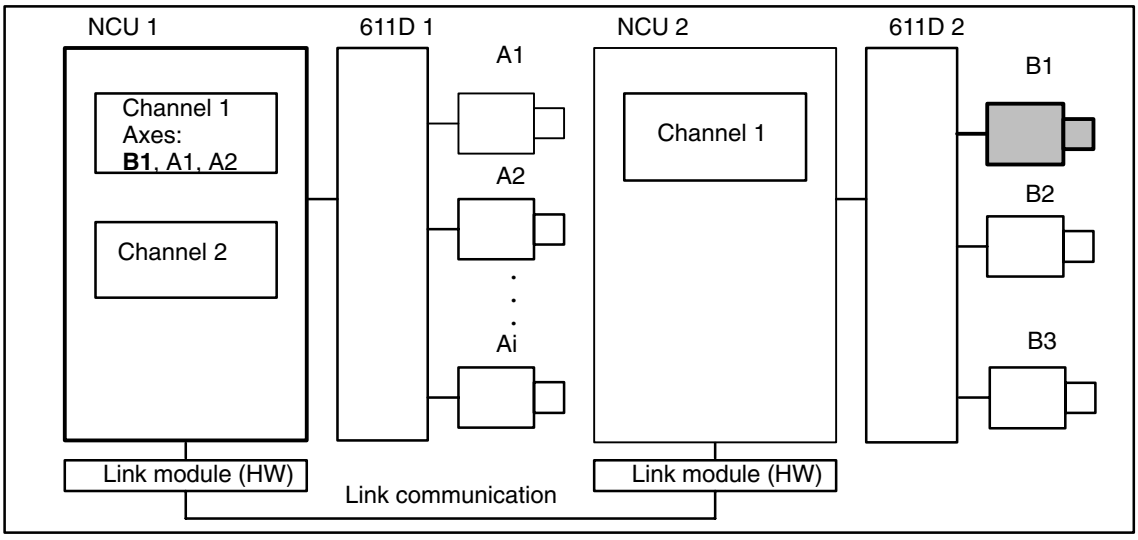


Fig. 2-6 Overview of link axes

Terms

The following terms are important for understanding the subsequent description:

- **Link axis**
  Link axes are axes that are physically connected to another NCU and controlled by their servo loop. Link axes can be dynamically assigned to channels of **another** NCU. From the standpoint of a particular NCU, they are not → local axes. The **axis container** concept described in 2.6 is for the dynamic change of assignment to a channel (special case: channel on another NCU).
  Axis exchange with GET and RELEASE from the part program is only available for link axes within an NCU. In order to cross the NCU link, the axis must first be placed in the NCU or a channel using the axis container function so that it can then be exchanged optionally in the same way as any other axis.
- **Local axis**
  A local axis is only addressed by the NCU to whose drive bus it is connected.
- **Link communication**
  The link communication is implemented on the NCUs involved by means of the link module. The link communication consists of setpoints, actual values, alarm handling, global variables (data) and signals (axis signals, PLC signals).
- **Home NCU**
  The NCU which establishes the drive bus connection for a ”→ link axis and implements the position control is called the home NCU of the link axis.
  In Fig. 2-6 NCU 2 is the home NCU for → link axis B1.
- **Interpolation**
  The **link axis** option makes interpolation between → local axes and axes on other NCUs possible for NCUs with → link communication.
  If the interpolation is not only local, cyclical data exchange (setpoints, actual values, ...) takes place within an interpolation cycle. In particular, this causes dead time when waiting for external events.
- **Axis change**
  Use of a → link axis by a specific NCU can change dynamically. An **axis container** mechanism is provided for this purpose as described in 2.6. The part program command GET is not available for link axes; the part program command GETD is only available within an NCU.
  Up to SW 4, it was only possible to exchange axes between different channels of an NCU.
- **Configuration of link axes**
  NCUs that want to use the → link axes must configure the **NCU identifiers** for the home NCU of the link axis in addition to the usual channel and axis machine data.
- **Home channel**
  Channel in which the setpoint-generating part program for the axis is executed after the installation has powered up.

Note

Link axes expand the limit set by the number of possible connections on the drive bus: With a maximum of 16 NCUs on the NCU link there are theoretical limits of max. 160 channels and 4960 axes/spindles. The following maximum values apply for one NCU: 31 simultaneous axes from 31 local and 32 link axes.

2.5.1 Configuration of link axes

SW 4

Up to SW 4, channel axes are directly mapped on the machine axes of the same NCU via
MD 20070: AXCONF_MACHAX_USED (see Fig. 2-7 left).

SW 5

From SW 5, the channels operate with one of 31 logical axes from the logical machine axis image. This image points to

- local axes
- link axes
- container slots.

Container slots in turn point to

- local axes or
- link axes.

The following diagram illustrates the interrelationships:

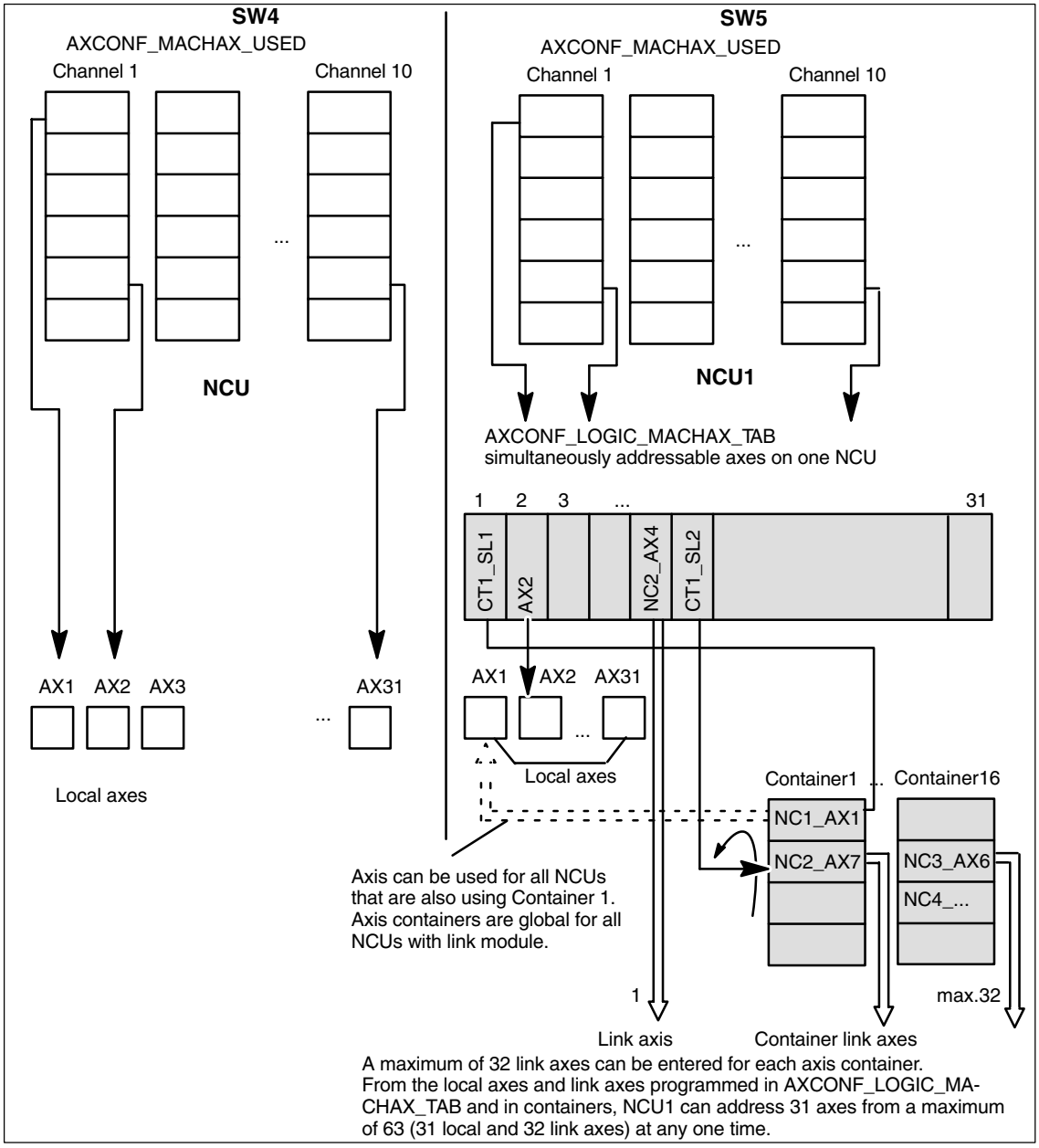


Fig. 2-7 A schematic comparison of the SW4 and SW5 configuration

With link axes

To enable link axes to be addressed throughout the system, the configuration must contain information about the axis NCUs. There are two types of NCU axis, i.e. local axes and link axes.

Differentiation local/link axes

The table that must be created via MD 10002: AXCONF_LOGIC_MACHAX_TAB is used to differentiate between local and link axes. See Fig. 2-7 center right and Fig. 2-8.

Note

The **axis container** functions are described in subsection 2.6.

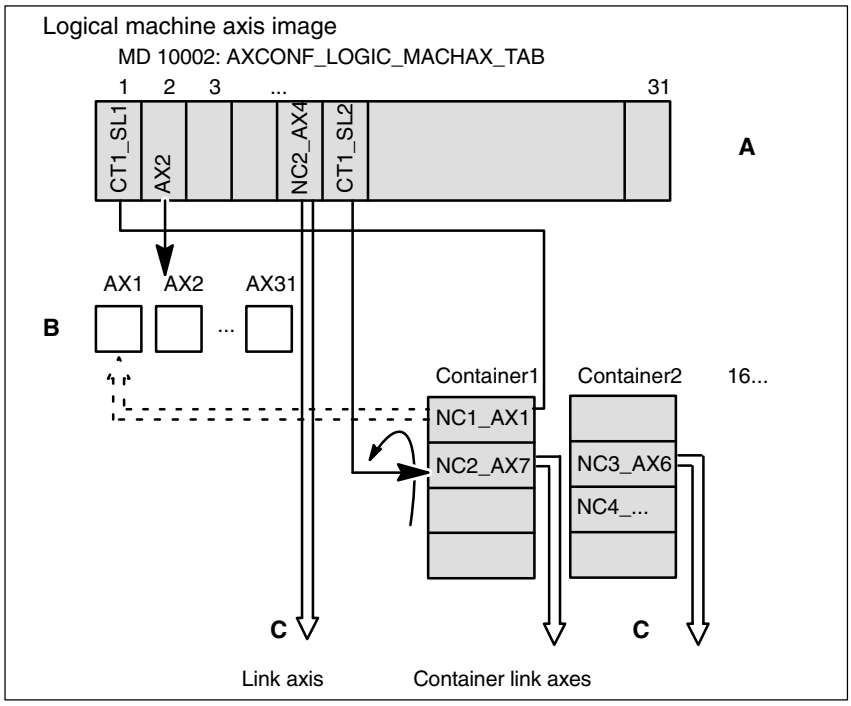


Fig. 2-8 Assignment of channel axes to local machine axes and link axes

Explanation

The **logical machine axis image** A addresses local machine axes B and link axes C.

The number of local machine axes in B is limited. The maximum permissible number for a specific system can be found in Catalog NC60.1.

All axes that can address the NCU are contained in B and C together.

Entries in A have the following format:

$MN_AXCONF_LOGIC_MACHAX_TAB[n] = NCj_AXi
with

n	index in Table A
NC	stands for **NCU** with
j	**NCU number**, 1 <= j <= 16
i	axis number, 1 <= i <= 31

Channel axes are no longer directly assigned to machine axes in MD 20070: AXCONF_MACHAX_USED as they were in SW version 4 and earlier, but are now assigned to **logical machine axis image A.**

Viewed from the part program, the only accessible machine axes are those which can be addressed by the channel (possibly via axis container, see below) via the logical machine axis image at a given point in time.

Default settings

By default, the settings of logical machine axis image A are local axis name AX1 for entry 1, and local axis name AX2 for entry 2, ...
With these, MD blocks that were generated for Software Versions lower than 4 can still be used with Software Version 5, if only local axes are addressed.

Examples

For example, the logical machine image can contain the following expressions:

NC2_AX7	machine axis 7 of NCU 2
AX2	local machine axis 2

If only expressions of the latter format AXi are entered in the logical machine axis image, this corresponds to a configuration up to Software Version 4, where only local axes are addressed.
Caution: The default settings are as follows:
MD 10002: AXCONF_LOGIC_MACHAX_TAB[0] = AX1
MD 10002: AXCONF_LOGIC_MACHAX_TAB[1] = AX2
...

Note

Another valid format for entries in the logical machine axis image A is:
MD 10002: AXCONF_LOGIC_MACHAX_TAB[n] = CTx_SLy with

CT	stands for **container**
x	container number, 1 <= x <= 16
SL	stands for **slot**
y	Slot number, 1 <= y <= 32

Axis containers represent a grouping of axes which can be altered dynamically. Axis containers are described in Section 2.6.

2.5.2 Axis data and signals

Introduction

Axis data and signals for a link axis are produced on its home NCU. The NCU that has caused the movement of a link axis, is provided with axis data and signals from the system:

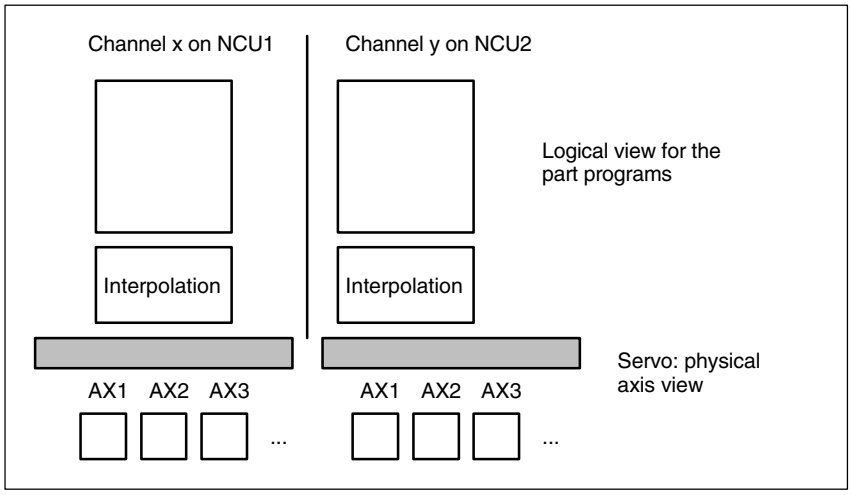


Fig. 2-9 Views of axes

Implicitly active link communication

During interpolation, data are made available for axes which are physically subordinate to a non-local servo (identifiable from entries in MD10002: AXCONF_LOGIC_MACHAX_TAB or axis container) via the link communication in the same manner as they are provided for local axes from the logical viewpoint of part programs. The procedure remains concealed from the applications.

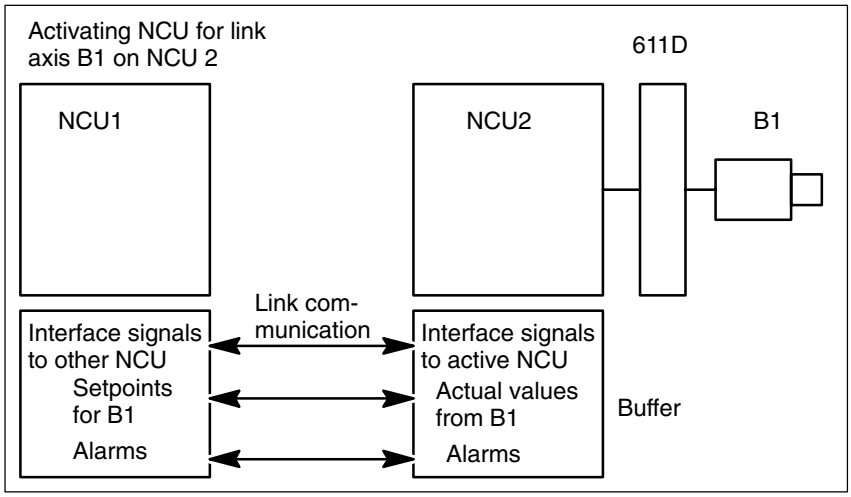


Fig. 2-10 Exchange of operating data and signals of a link axis

Servo loop

The servo loop is implemented on the NCU on which the axis is physically connected to the drive. This NCU also contains the associated axis interface. The position setpoints for link axes are generated on the active NCU and transferred via the NCU link.

Communication types

There are two types of link communication:

- Cyclical communication
- Non-cyclical communication

Cyclical communication

– Setpoint for the link axis,
– Actual values from the link axis
– Status signals of the link axis
– Status signals of the NCUs
are transmitted cyclically. Actual values and status signals of a link axis are updated and made available to the NCU that is interpolating this axis.

Non-cyclical communication

– Exchange of link variables
– Warm start requirements
– Activation of axis container rotation
– Modifications to NCU-global machine and setting data.
– Activation of axial machine data for link axes
– Alarms

Transfer time

Delays incur for transferring **setpoints** to the home NCU of a link axis and returning its **actual values**. With an interpolation group of local axes and link axes, the control delays the setpoints for the local axes of the interpolating NCU by one interpolation cycle, such that consistent values are taken into account for the interpolation.

If a channel needs the actual values of an axis of another NCU, e.g. a spindle with thread cutting, two interpolation cycles will lapse before they are available. The setpoints then generated are sent one interpolation cycle later to the position control for the above reason.

Response of the AXIS-VAR server to errors

If the server cannot supply any values for an axis (e.g. because the axis concerned is a link axis), then it returns a default value (generally 0).

For the purpose of testing, machine data 11398:
AXIS_VAR_SERVER_SENSITIVE can be used to set the axis data server sensitively so that it returns an error message rather than default values.
0: Default value
1: Error message

2.5.3 Supplementary Conditions

Output of alarms from position controller or drive

Axis alarms are always output on the NCU which is producing the interpolation value. If an alarm is generated for a link axis by the position controller, then the alarm is transferred to the NCU which is currently processing the interpolation.

On the assumption that axis alarms which cause the NCK-Ready relay to drop out (Nck-NoReady) are attributable to faults on the drive bus, the alarm is also output on the NCU to which the axis or the drive bus is physically connected. The reaction “Ready relay dropout” is only activated on this NCU.

Output of alarms following EMERGENCY STOP

If an EMERGENCY STOP request is activated by the PLC on an NCU, then all axes physically connected to drives on this NCU are switched to follow-up mode. This means that: even axes which are being interpolated by a different NCU are also switched to follow-up. Since this status prevents any further constructive machining operations on the other NCUs, an additional alarm is generated which is designed to stop all axis motions instantaneously.

This additional alarm must be acknowledged by an operator panel reset. If the original (EMERGENCY STOP) alarm is still active at this time, then the additional alarm can be successfully reset, but another alarm (self-clearing) is then produced which prevents axis motion or a new program start until the original alarm has been acknowledged.

Output of alarms with alarm reaction NCK-NoReady

If a serious alarm resulting in dropout of the NCK-Ready relay is activated on an NCU, then the effects of the alarm will apply to all other NCUs which are addressing an axis via link communication on the first NCU. An additional alarm which causes all other axes to stop instantaneously is activated on each of the other NCUs.

For alarm acknowledgement, see EMERGENCY STOP.

Compensations

The compensation functions

- CEC
- EEC
- QEC

are **not** available for link axes.

Switching off grouped NCUs

If an NCU assigned to an NCU grouping is switched off or restarted by NCK RESET, then the other NCUs in the grouping are also affected (see also 2.8). An alarm is generated on the NCUs which are still running to prevent them continuing with the machining operation.

Powering up an NCU grouping

If one NCU in the grouping is restarted, e.g. due to changes to machine data, then the other NCUs in the grouping also execute a warm restart.

Nibbling and punching

To execute nibbling and punching operations, high-speed inputs and outputs must be connected and parameterized on the “interpolation” NCU (on which the part program is being executed). Commands “High-speed nibbling and punching”, e.g. PONS and SONS are not available for link axes.

Travel to fixed stop

If an axis container axis is being held against a fixed stop, the axis container cannot rotate. Axes can travel to fixed stops on different NCUs and be subsequently clamped without restriction.

Frames

Link axes can only be included in the program commands for frames if they are also designated as geometry axes. The command changes the geometry only for the channel to which the axis is currently assigned. A frame command for an axis which is not defined as a geometry axis is rejected with alarm 14092.

Revolutional feedrate

Although setting data 43300: ASSIGN_FEED_PER_REV_SOURCE referred directly to a machine axis in SW 4 and earlier, the MD refers in SW 5 and higher to the logical machine image and, via this, to a machine axis (local or link axis).

2.5.4 Programming with channel and machine axis identifiers

Channel axis identifiers

Example:

```
WHENEVER $AA_IW[Z] < 10 DO ...        ;Current position of the Z axis
```

Machine axis identifiers

Example:

```
WHENEVER $AA_IW[AX3] < 10 DO ...      ;Scan current position of machine
                                       axis AX3
```

This method of programming is permitted only if machine axis AX3 is known in the channel at the time of scanning.

Note

In SW 5.2 and higher, system variables which can be used in conjunction with channel axis identifiers are specially marked in the “Advanced” Programming Guide (Appendix).

2.5.5 Flexible configuration

Introduction

Rotary indexing machines and multi-spindle machines have special requirements as regards the flexible assignment of channel axes to machine axes.

Requirement profile

When advancing the table of the rotary indexing machine or the drum of the multi-spindle machines the axes/spindles are brought to a new station or position. The NCU which controls the axes of a station as local axes must be able to address the newly changed axes/spindles. The hitherto addressable axes/spindles can now be discarded for this purpose.

Solution

A configuration of the relevant axes in an axis container specified in machine data enables different machine axes to be located in succession behind a channel axis that remains constant. Advancing the rotary table or drum is performed synchronously with the advancing of the axes entered in the axis container.

Axes in an axis container can also be configured as geometry axes.

Note

The axis container has no mode group reference, i.e. the workpiece-holding, travelling axis can change from one mode group to another at different machining stations.

2.6 Axis container

Definition

An axis container can be imagined as a circular buffer in which

- local axes and/or
- link axes

are assigned to channels. Axes in an axis container are also referred to as **container axes**. Assignments can be “shifted” (“rotation” of the circular buffer) by means of program commands. The term “axis” in this case refers to both axes and spindles. All machine axes in the axis container must be assigned to exactly **one** channel axis at any given point in time.

Note

Rotation of the drum or rotary table is analogous to the rotation of the circular buffer with the assigned axis entries.

Description

The link axis configuration described in Section 2.5.1 allows reference to be made to axis containers in the logical machine axis image, in addition to direct reference to local axes or link axes. This type of reference consists of:

- container number and
- slot (circular buffer slot within the corresponding container)

An entry in a circular buffer is either:

- a local axis **or**
- a link axis

(either axis or spindle)

Axis container names in SW 5.2 and higher

Axis container names can be freely defined with machine data
MD 12750 : AXCT_NAME_TAB
in SW 5.2 and higher. The names assigned can then be used:

- in axis container rotation commands AXCTSWE( ) and AXCTSWED( ) to address the container to be rotated and
- when scanning the states of axis containers using system variables:
  – $AC_AXCTSWA[ ]
  – $AN_AXCTSWA[ ]
  – $AN_AXCTAS[ ]

SW 5.1 Parameter CTi

In this software version, **channel axis identifiers**, which refer to the container to be rotated via the logical machine axis image, must be used instead of axis container identifiers.

Definition of container contents

Machine data MD 12701 ... 12716:
$MN_AXCT_AXCONF_ASSIGN_TAB1 ...n defines the default assignment between an axis container slot and a machine axis within an NCU grouping for axis container 1...n. The assignment between an axis container slot and the selected channel is programmed in MD 20070:
$MC_AXCONF_MACHAX_USED and MD 10002:
$MN_AXCONF_LOGIC_MACHAX_TAB.

Example

In the example illustrated in Fig. 2-11, the 3rd channel axis (3rd entry in $MC_AXCONF_MACHAX_USED) is a container axis. The 3rd entry in $MC_AXCONF_MACHAX_USED refers to the 8th entry in $MN_AXCONF_LOGIC_MACHAX_TAB and this (CT3_SL2) in turn to the 3rd axis container and its container slot 2. This 2nd entry in $MN_AXCT_AXCONF_ASSIGN_TAB3 (NC3_AX1) defines the 1st machine axis of NCU3 as a container axis of axis container 3, i.e. in the initial state, the 4th channel axis is the 1st machine axis of NCU3.

The 5th channel axis is also a container axis: The 5th enter in $MC_AXCONF_MACHAX_USED refers to the 7th entry in $MN_AXCONF_LOGIC_MACHAX_TAB and this (CT1_SL1) in turn to the 1st axis container and its container slot 1. This 1st entry in $MN_AXCT_AXCONF_ASSIGN_TAB1 (NC1_AX1) assigns the 1st machine axis of NCU1 to the 1st slot of axis container 1, i.e. in the initial state, the 1st machine axis of NCU1 is assigned to the 5th channel axis.

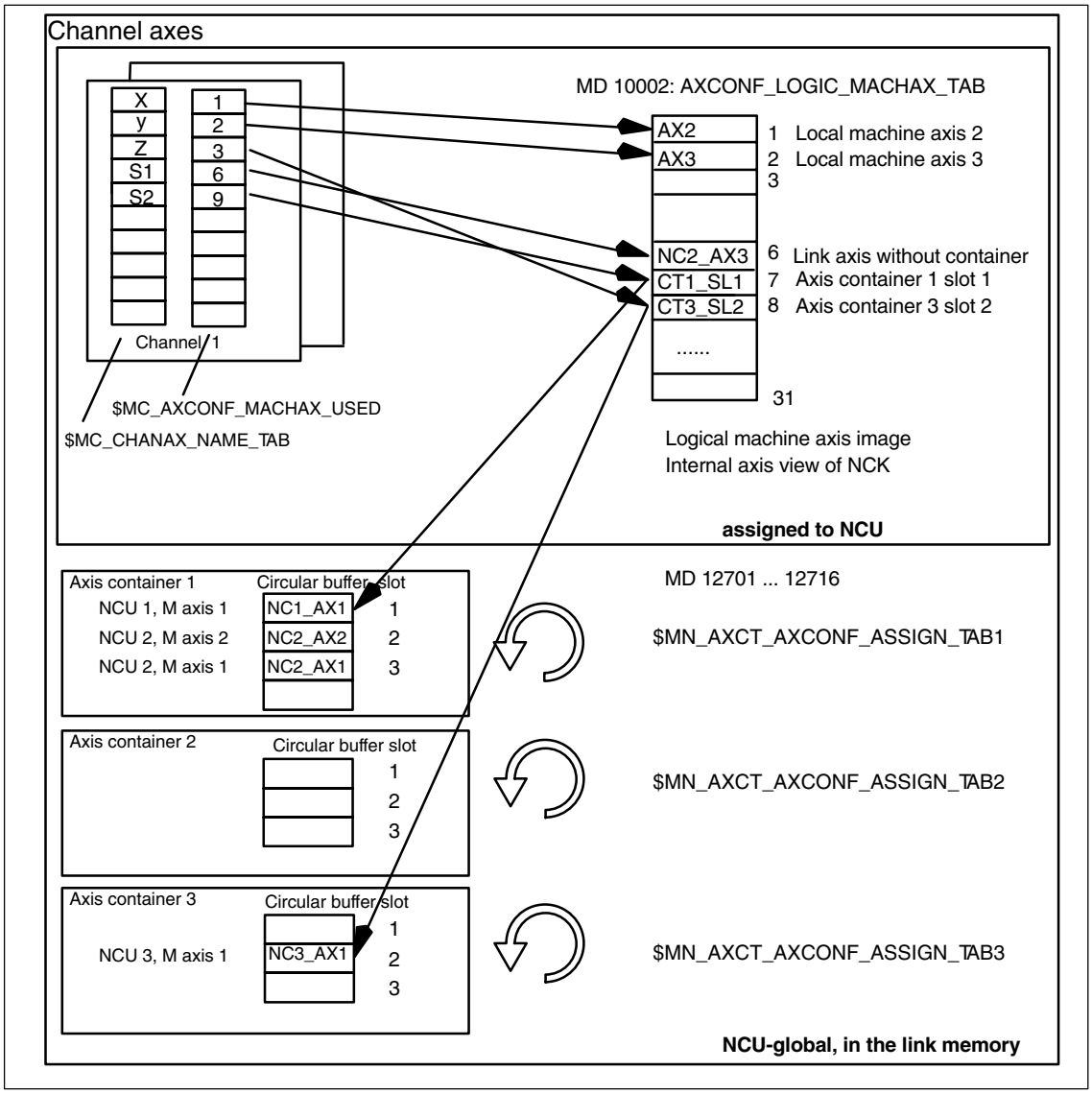


Fig. 2-11 Mapping of channel axes via the logical machine axis image onto axis container

Axis container entries contain local machine axes or link axes from the view of an individual NCU. The entries in the logical machine axis image MD10002: AXCONF_LOGIC_MACHAX_TAB for a single NCU are permanent.

Container rotation

The contents of the axis container slots are variable inasmuch as the contents of the circular buffer (axis container) can be shifted together by ± n increments. The number of increments n is defined for each axis container in

SD 41700: $SN_AXCT_SWWIDTH.

The number of increments n is evaluated modulo in relation to the number of actually occupied container slots. In doing so, new contents are created for all slots of an axis container (exception: 0 and slot number = increment number).

System variables provide information about the current status of an axis container; these system variables can be read addressed from the part program and synchronized actions. See 2.6.1.

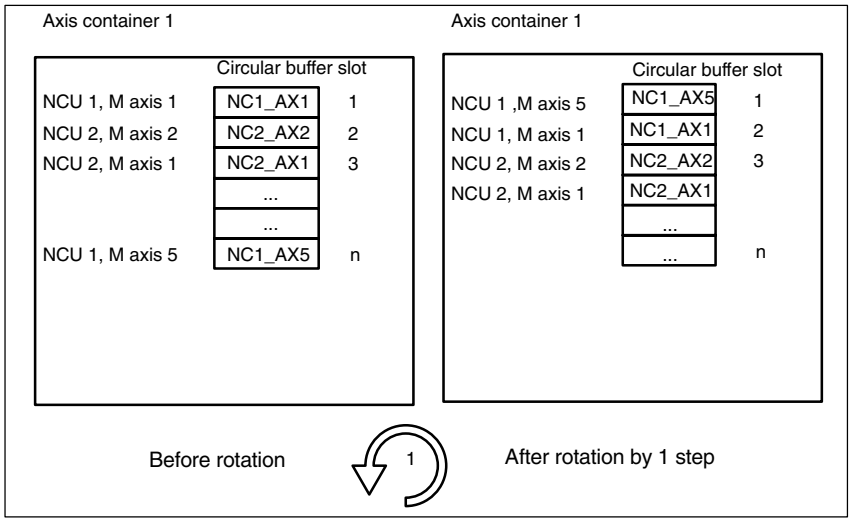


Fig. 2-12 Shifting the entries to the axis container slots

The axis container model has the following characteristics:

- A channel always sees a fixed number of axes with defined channel axis names (logical machine axis image)
- The “rotation” of the axis container sets new machine axes for **all** channels that have axes in the same axis container.

Frames and axis container rotations

The assignment between channel axes and machine axes can change when the axis container rotates. The current frames remain unchanged after a rotation. The user himself is responsible for ensuring that the correct frames are selected after a rotation by, for example, programming basic frame masks.

Activation of axis container rotation

The application must ensure that the desired local or link axes are addressed by issuing commands in the part program for rotating the axis container to a specific position.

For example, when rotating the drum of a multi-spindle machine into a new position, it must be ensured that each position addresses the newly changed spindle by rotation of the axis container.

Note

Axis containers can be used jointly by different channels of an NCU and by channels of other NCUs.
If axes of different channels display reference to the same axis container via the logical machine axis image, then **all** channels concerned see **different axes** after a rotation. Consequently: The time for a rotation must be coordinated between the channels. This is performed by means of the available language commands.

Each entry in the axis container must be assigned to the correct channel at all times. The system variables in 2.6.1 offer the possibility for the part program or synchronized action to gain information about the current axis container state.

Commands for the axis container rotation

The requirement outlined above for coordinating channels that jointly use an axis container is contained in the effects of the command **AXCTSWE**.

Notation:

AXCTSWE(CT1, CT2, ... CTi) ;The function name stands for:
;AXis ConTainer SWitch Enable

CT1, CT2, ... CTi are the identifiers of the axis containers that must be advanced. The increment must be stored in setting data

SD 41700: AXCT_SWWIDTH[container number]

(container-specific). The SD 41700: AXCT_SWWIDTH (AXis ConTainer SWitch WIDTH) is available to all NCUs via the link module (i.e. all NCUs connected via a link module “see” the same values).

Function:
Each channel whose axes are entered in the specified container issues an **enable for a container rotation** if it has finished machining the position/station. If the enables for **all** channels for the axes of the container have been received, container rotation takes place with the increments set in SD 41700: AXCT_SWWIDTH[container number] (the direction of rotation is also assessed if there is a leading sign).

The following variant is provided to simplify start-up:

AXCTSWED(CT1) ;The function name stands for:
;AXis ConTainer SWitch Enable Direct

The axis container rotates according to the settings in setting data SD 41700: AXCT_SWWIDTH[container number]. The call may be used only if the other channels containing axes in the container are in the **RESET** state.

Note

In **SW 5.2** and higher, the axis container names assigned in
MD 12750: $MN_AXCT_NAME_TAB
can be used for commands AXCTSWE and AXCTSWED.

In the earlier software version **SW 5.1**, **channel axis identifiers** must be specified which refer to the container to be rotated via the logical machine axis image. The rotation must be specified in a separate command AXCTSWE(channel axis) for each container.

Implicit wait

There is an implicit wait for the completion of a requested axis container rotation if one of the following events has occurred:

- Part program language commands which will cause a container axis assigned to this axis container in this channel to move
- GET(channel axis name) for an appropriate container axis
- The next AXCTSWE(CTi) for this axis container

Note

Even an IC(0) will result in a wait including synchronization where necessary (block-by-block change in addressing according to increment even though absolute dimension is set globally).

Synchronization with axis position

If the new container axis assigned to the channel after a container rotation does not have the same absolute machine position as the previous axis, then the container is synchronized with the new position (internal REORG).

Note

The SD 41700: AXCT_SWWIDTH[container number] is only updated for new configurations. If after the incremental rotations of the RVM/MS the position has reached a switching position before the original position, the container can continue to be rotated **forwards**, in order to reach the original position of the container again. The drum or rotary table must however be turned **back** to the original position, so that measuring and supply cables are not interrupted.

Home channel of a container axis

If more than one channel has access authorization (i.e. a “reference”) to the axis due to the setting in MD 20070: AXCONF_MACHAX_USED, the write access to the axis (setpoint input) can be passed on. Machine data MD 30550: AXCONF_ASSIGN_MASTER_CHAN creates a default assignment between an axis and a channel. MD 10002: AXCONF_LOGIC_MACHAX_TAB is set to define which NCU “possesses” the axis after power-up or is producing the interpolation value. Since the axial machine data for link axes are identical on all NCUs, MD 30550: AXCONF_ASSIGN_MASTER_CHAN is evaluated only if the NCU has write authorization to the axis (see logical machine axis view in MD 10002: AXCONF_LOGIC_MACHAX_TAB).

Axis replacement

Passing the write authorization to an axis (setpoint input) by means of Get, Release,..., works for a container axis in the same way as for a normal axis. Write authorization can only be replaced between the channels of **one** NCU. Write authorization cannot be passed beyond the boundaries of an NCU.

2.6.1 System variables for axis containers

States of an axis container

The following system variables allow part programs and synchronized actions to access information about the current state of an axis container.

Legend:

r	Read
PP	Part program
SA	Synchronized action
SW	Software version
n	Axis container identifier with SW 5.2 and channel axis identifier with SW 5.1

Name	Type /SW	Description/values	Index	Access PP	Access SA
$AC_AXCTSWA[n] (AXis ConTainer SWitch Active)	BOOLEAN /5	**Channel** status of axis container rotation/ 1: The channel has enabled axis container rotation for axis container n and this rotation is not yet complete. 0: The axis container rotation has ended Examples: See Chapter 6 “Axis container coordination”	Identifier	r	r
$AN_AXCTSWA[n]	BOOLEAN /5	**Axis container rotation**/ 1: An axis container rotation is executed immediately by the axis container n 0: No axis container rotation is active Examples: See Chapter 6 “Axis container coordination”	Identifier	r	r
$AN_AXCTAS[n] (AXis ConTainer Actual State)	INT /5	**Current rotation of axis container** The number of slots by which the axis container has just been rotated is specified for axis container n./ 0 to (max. number of occupied slots in axis container –1) The default setting is entered after power on. This is the value 0.	Identifier	r	r

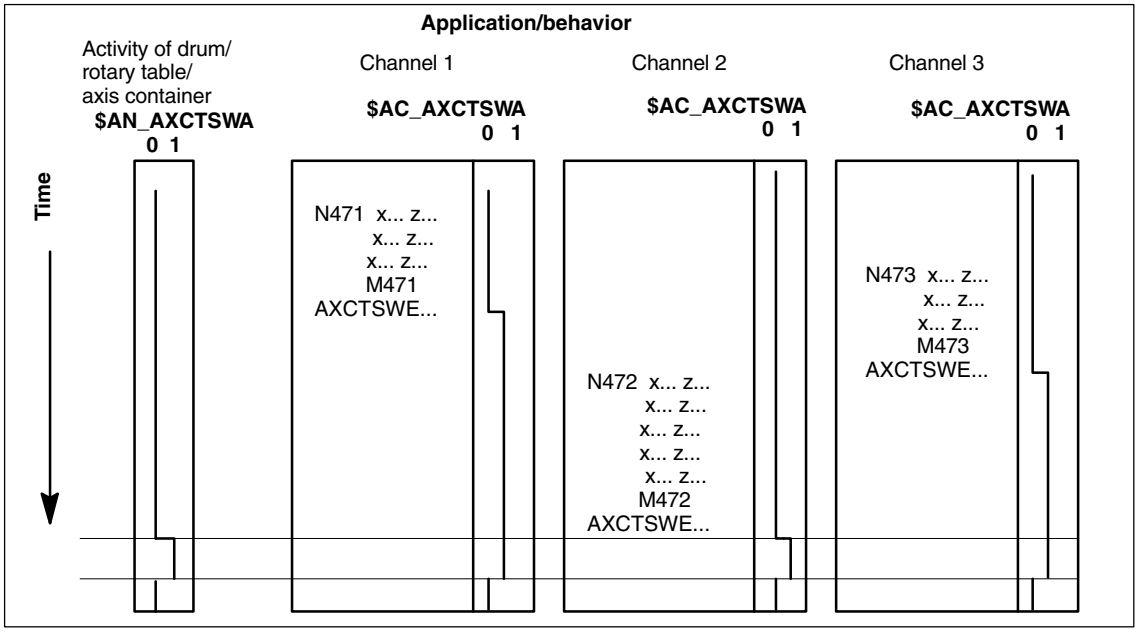


Fig. 2-13 Axis container rotation dependent on enable by channels concerned

2.6.2 Machining with axis container (schematic)

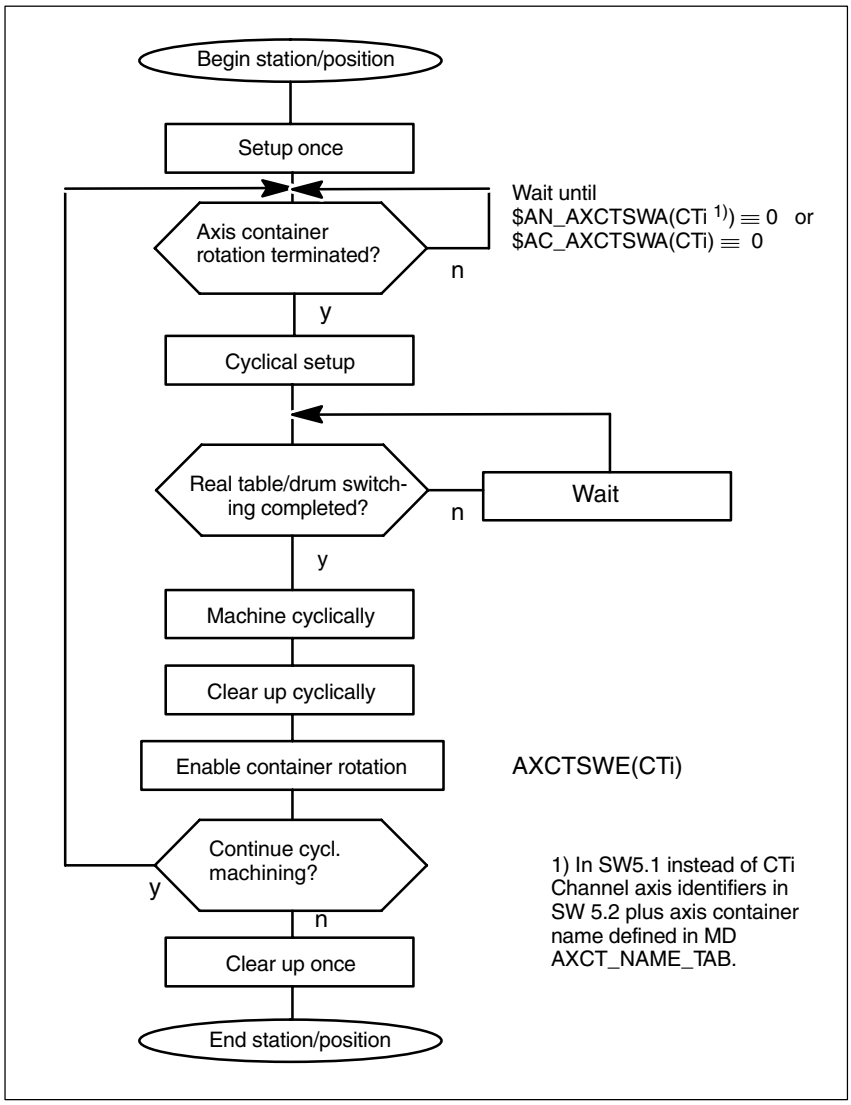


Fig. 2-14 Schematic machining of a station/position

Note:
An NCU machining cycle which is in charge of the rotation of the rotary table or the drum for multi-spindle machines contains the query of enables for container rotation of all NCUs concerned. If all enables are present, switching to the next position/station takes place. The axis containers are rotated accordingly.

2.6.3 Axis container behavior after power ON

The container always assumes the state defined in the machine data when the power is switched on, irrespective of its status as the power supply was switched off, i.e. the user must distinguish between the actual status of the machine and the default setting and compensate accordingly by specifying appropriate axis container rotations. He can do this, for example, by means of an ASUB containing AXCTSWED in one channel while the other channels are still in the RESET state.

2.6.4 Axis container response to mode switchover

A container axis in an axis container which has been enabled for rotation cannot be traversed in JOG mode. An axis container can only be rotated in JOG mode by means of an ASUB.

2.6.5 Axis container behavior in relation to ASUBs

An enabling command for axis container rotation cannot be cancelled, i.e. if an axis container rotation has been enabled in an ASUB, the enabling command remains effective even when the ASUB has ended.

2.6.6 Axis container response to RESET

A reset cancels the enabling command for axis container rotation. The reset channel is then no longer involved in the axis container rotation. The enabling commands in the other active channels can effect a rotation. If all channels except one have been reset, the one remaining active channel can set the rotary position directly with AXCTSWED.

2.6.7 Axis container response to block searches

An axis container rotation (AXCTSWE) cannot be enabled and activated in one block, but the enabling and activation commands must be programmed in separate action blocks. In other words, the axis container status changes in response to each separate rotation command as a function of the status of other channels.

2.6.8 Supplementary conditions for axis container rotations

Note

Through appropriate programming measures, the user must ensure that
– the right zero offsets are effective after the container switch and
– that no transformations are active during the container switch.

Axial machine data

If an axis is assigned to an axis container, then certain axial machine data must be identical for all axes in the axis container as the data are activated. This can be ensured by making a change to this type of machine data effective all container axes and all NCUs which “see” the axis concerned. The message: “Caution: This MD will be set for all container axes” is output at the same time.

During power-up, all axial machine data of this type are synchronized with the values of the machine axis in **slot 1 of the axis container**. In other words, the relevant machine data are transferred from the machine axis in slot 1 of the axis container to all other container axes. If machine data with other values are overwritten by this process, the message: “The axial MD of the axes in axis container <n> have been adapted” is output.

If a slot in the axis container is re-assigned (through writing of machine data MD12701–12716: AXCT_AXCONF_ASSIGN_TAB<n>), then the following message is output: “The MD of the axes in axis container <n> will be adapted on next power-up”.

Axial machine data of the type discussed above are identified by attribute **containerEqual** (equal for all axes in the axis container). With an NCU link, the axis container is defined **on the master NCU** (see Section 2.4).

Axis states

If a container axis is active in axis mode or as a positioning spindle (POSA, SPOSA) and its axis container needs to be rotated, then the rotation cannot be executed until the container axis has reached its end position.

A container axis which is active as a **spindle continues to turn** as the axis container rotates.

SPCON (switchover to position control) is “attached” to the physical spindle, i.e. this status is passed on with the spindle when an axis container rotates. SETMS (master spindle), on the other hand, refers to the channel and remains active in the channel when an axis container rotates.

Continuous path mode G64

An axis container rotation interrupts G64 mode in a channel in which a container axis in the rotating container is also a channel axis, even if it does not belong to the path grouping. This interruption does not occur, however, until an axis in the rotated axis container is programmed again.

PLC axes

If a container axis in a container which is enabled for rotation must become a PLC axis, then this status change request is stored, and the changeover to PLC axis status does not take place until **after** completion of the axis container **rotation**.

Command axes

A container axis in a container enabled for rotation cannot be declared a command axis. The traverse request is stored in the channel and executed on completion of the axis container rotation.

Exceptions to this rule are synchronized actions M3, M4, M5 and a motion-changing S function: If an axis container rotation is active and the spindle is transferred to the control of another NCU, alarm 20142 (channel %1 command axis %2: Invalid axis type) is output. These synchronized actions do not change a channel axis into a command axis, but leave it in its original state. Synchronized actions of this type cannot be stored.

References: /FBSY/ Description of Functions Synchronized Actions

Reciprocating axes

A container axis in a container enabled for rotation cannot become a reciprocating axis, i.e. this change in status does not take place until the axis container has finished rotating. The status change command remains active.

Axis couplings

An axis container cannot rotate while an axis coupling, in which one of its container axes is involved, is still active. The coupling must be deselected (COUPOF) prior to rotation and selected again (COUPON) afterwards. A new COUPDEF command is not necessary.

Compile cycles

In SW 5.2, a compile cycle axis cannot be a container axis.

Main run offset values

The main run offset values (DRF offset, online tool offset, synchronized action offset, compile cycle offset) for a channel axis assigned to a container slot remain valid after the relevant axis container has rotated. External zero offsets cannot remain valid after an axis container rotation as these refer to specific machine axes. If an external zero offset is active, the axis container rotation is rejected with alarm 4022.

Axial frame

The axial frame of a channel axis, which is also a container axis, is no longer valid after an axis container rotation. Since the axis container rotation assigns a new machine axis to the channel axis, but the axial frame is referred to a machine axis, **the rotation thus also changes the axial frame**. If the two frames do not coincide, a synchronization process (internal REORG) is performed.

The assignment between a channel axis and a machine axis is altered by the axis container rotation. The current frames remain unchanged after a rotation. The user himself is responsible for ensuring that the correct frames are selected after a rotation by, for example, programming basic frame masks.

Transformations

If the container axis is a spindle which is involved in a transformation, then the **transformation** must be deselected before the axis container rotation **is enabled**. Otherwise alarm 17605 is activated.

Gantry grouping

Gantry axes cannot be axes in an axis container.

Drive alarms

When a drive alarm is active for a container axis, then the associated axis container cannot rotate until the alarm cause has been eliminated.

2.7 Cross-NCU user communication, link variables

Introduction

For large machine tools, rotary indexing machines and multi-spindle machines, whose movement sequences are controlled by more than one NCU, the applications on a single NCU must be able to exchange information rapidly with the other NCUs connected via link module.

For this purpose there are:

- Link variables

2.7.1 Link variables

Definition

Link variables are **system-global data** that can be addressed by the connected NCUs as **system variables** if link communication is configured. The
– **contents** of these variables,
– their **data type**,
– their **use,**
– their position (**access index**) in the link memory
are defined by the user (in this case this is usually the machine manufacturer).

Precondition

– To active NCU link communication, MD 18780: MM_NCU_LINK_MASK must be set.

– The link grouping must be installed and configured according to 2.8.

Application

As link variables are formally system variables, they can be read/write accessed in

part programs and in
synchronized actions

(as a rule).
Access possibilities for the individual link variables are specified under 2.7.2.

Note

On installations without an NCU link, the link variables can also be used NCU-locally as an additional means of cross-channel communication.

Structure

Each NCU connected to an entire system with link module sees a **link memory** in which the link variables are stored uniformly. Data exchange takes place after changes in the following interpolation cycle.

Size of link memory

The size of the link memory can be configured within the limits set by machine data

MD 18700: MM_SIZEOF_LINKVAR_DATA.

It is necessary to define the same size for all connected NCUs. If there are deviations, the system adapts the link memory size of all NCUs according to the largest size specified. If the memory area of the link memory is exceeded during an access attempt, alarm 17020 is output.

Initialization of link memory

After power-up, the link memory is initialized with 0.

Data types of link variables

The link memory can contain link variables with the following data types:

- INT $A_DL**B**[i] ; Data **B**yte (8 bits)
- INT $A_DL**W**[i] ; Data **W**ord (16 bits)
- INT $A_DL**D**[i] ; Data **D**ouble word (32 bits)
- REAL $A_DL**R**[i] ; **R**eal data (64 bits)

According to the data type, 1, 2, 4, 8 bytes are addressed when reading/writing the link variables.

The position offset in bytes in the data area for global data is determined directly by the programmed field index. This is thus independent of the data type and specifies the offset in bytes.

Range of values

The data types have the following value ranges:

BYTE:	–128 to 255
WORD:	–32768 to 65535
DWORD:	–2147483648 to 2147483647
REAL:	–4.19e–308 to 4.19e–307

Alarm 17080 is generated when the upper value range limit is violated and alarm 17090 with violation of the lower value range limit and alarm 14096 in the case of an illegal type conversion.

Addressing with access to global variables

Index i always represents the distance in bytes from the beginning of the link memory. The index is counted from 0. This means that:

Type	Interpretation of (i) (counting starts at 0 each time)
$A_DLB[i]	After byte i there is a data of the byte type. $A_DLB[7] addresses the byte 8 from the beginning of the link memory.
$A_DLW[i]	After byte i there is a data of the word type. $A_DLW[4] addresses the word which is located on byte 5 from the beginning of the link memory.

Type	Interpretation of (i) (counting starts at 0 each time)
$A_DLD[i]	After byte i there is a data of the double word type. $A_DLD[12] addresses the double word which is located on byte 13 from the beginning of the link memory.
$A_DLR[i]	After byte i there is a data of the real type. $A_DLR[24] addresses the real value which is located on byte 25 from the beginning of the link memory.

Link memory use

The link memory can have different assignments for processes that are completely separated in time. The various NCU applications that access the link memory jointly **at any one time** must use the link memory **in a uniform way**.

Access from synchronized actions

If an impermissible index is used for access to the link memory from a synchronized action or a part program, alarm 20149 is issued.

Write access to link variables

When writing to link variables of the link memory, for example as follows

$A_DLB[5] = 21,

a write element is required. The write element serves for communication with further NCUs which must see the modified contents in the link memory. Each write process to a link variable requires a write element. It is busy with the write process until the main run executing data exchange with the other NCUs is completed.

Since global data can be written by all channels and NCUs, the user must ensure proper coordination of write and read access operations. Variables are written immediately if an NCU link connection is active. Writing and immediate readback of a variable produces the same result. Variables are written only in synchronism with the main run. Writing and immediate readback in the same part program block produces a different result.

Number of write elements

The write elements available for writing to link variables are limited. Their number is defined in:

MD 28160: MM_NUM_LINKVAR_ELEMENTS.

If no more write elements are available for an intended write process, alarm 14763 is issued. The set number of write elements only limits the number of write processes that can be written in one block.

Dynamic response during write

Writing the link variables is immediately completed for the **local** NCU in the current interpolation cycle (in the sequence of commands). If the user does not exceed the number of possible write processes (can be checked in system variable $A_LINK_TRANS_RATE) in the current interpolation cycle, **all other** NCUs will have access to the written information 2 interpolation cycles later. If link variables are used exclusively to coordinate the channels of a multi-channel NCU, they can be written in the same interpolation cycle.

Note

The user (machine manufacturer) must ensure that the time is consistent for larger data blocks that are logically associated with one another. Transfer takes place word by word. The data quantity which can be transferred in the same interpolation cycle is specified in system variable $A_LINK_TRANS_RATE (see below). A transmission can be protected by marking variables of the link memory as semaphores.

2.7.2 System variables of the link memory

The following system variables are available for accessing the link memory:

Legend:

r	Read
w	Write
R	Read with implicit preprocessing stop
W	Write with implicit preprocessing stop
PP	Part program
SA	Synchronized action
SW	Software Version

Name	Type /SW	Description/values	Index	Access PP	Access SA
$A_DLB[i]	INT/5	Addresses a byte in the link memory / 0 to $MN_MM_SIZEOF_LINKVAR_DATA –1	Counter	r/w	r/w
$A_DLW[i]	INT/5	Addresses a data word in the link memory / 0 to $MN_MM_SIZEOF_LINKVAR_DATA–2	Counter	r/w	r/w
$A_DLD[i]	INT/5	Addresses a data double word in the link memory / 0 to $MN_MM_SIZEOF_LINKVAR_DATA–4	Counter	r/w	r/w
$A_DLR[i]	REAL/ 5	Addresses a REAL value in the link memory / 0 to $MN_MM_SIZEOF_LINKVAR_DATA–8	Counter	r/w	r/w
$A_LINK_TRANS_RATE	INT/5	For synchronized actions: Number of bytes that can still be transferred in the current interpolation cycle via link communication. / –2147483648 to 2147483647			r

Note

Use of index i is described in detail in 2.7.1.

2.8 Configuration of a link grouping

Introduction

The preceding chapters described how to configure link axes and axis containers. Both require a link communication to be established between the NCUs concerned. Setting up the link communication takes place by means of:

- The link module hardware
  References: /PHD/, Configuring Guide NCU 571–573.2
- Machine data

The following section describes how to use the required machine data.

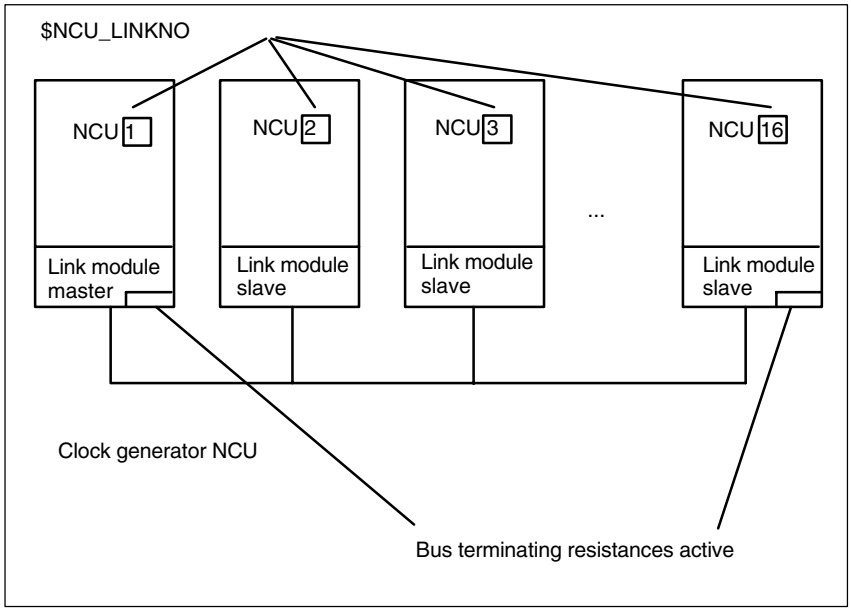


Fig. 2-15 Link grouping

Link grouping

A link grouping consists of a minimum of 2 and maximum of 16 NCUs interconnected by link modules.

– The link module master (MD 12510: NCU_LINKNO = 1) plays a leading role in this process. It synchronizes the interpolation cycle and sets up slave communication in power-up. It is advisable to assign the NCU numbers in continuous ascending order for the slave modules.

– The first and last module in the physical chain must activate the bus terminating resistances.

– The software version must be identical on all NCUs in a link grouping.

Machine data

Machine data
MD 18780: MM_NCU_LINK_MASK
ensures that link communication is established. It provides the dynamic memory space that is required for communication in the NCUs equipped with link modules.

Machine data
MD 12540: LINK_BAUDRATE_SWITCH
specifies the data transfer rate of the link communication with the following assignment:

Set value	Rate		
0	9,600	Kbaud	
1	19,200	Kbaud	
2	45,450	Kbaud	
3	93,750	Kbaud	
4	187,500	Kbaud	
5	500,000	Kbaud	
6	1,500	Mbaud	
7	3,000	Mbaud	
8	6,000	Mbaud	
9	12,000	Mbaud	Default setting

Machine data
MD 12550: LINK_RETRY_CTR
specifies the maximum number of times the link communication is repeated when an error occurred during frame transfer.

Machine data
MD 12530: LINK_NUM_OF_MODULS
specifies the number of link modules taking part in the link communication

Machine data
MD 12510: NCU_LINKNO
assigns a logical link number to an NCU; this number is used for link identification in conjunction with link axes and link communication. Identifications can be assigned independently of the physical sequence of the modules in the link string. The module with NCU_LINKNO = 1 is **master**.

Warning

Assignment of NCU_LINKNO must be unambiguous. An alarm is issued if there is an error.

Machine data
MD 12520: LINK_TERMINATION
specifies for the software which NCUs correspond to the bus terminating resistances. The numbers set refer to the entries defined in MD 12510: NCU_LINKNO. 0 corresponds to the first definition, 1 to the second, etc. from MD 12510: NCU_LINKNO.

Note

MD 12520: LINK_TERMINATION need only be set for the prototype hardware of the link module. It is only meaningful for the software.

The NCUs that are **physically** connected at the **beginning** and **end** of the bus must activate the terminating resistors. This measure is necessary for the link communication to work.

Machine data

MD 30554: AXCONF_ASSIGN_MASTER_NCU

defines for the purposes of power-up which NCU in an NCU grouping will be responsible for generating the axis setpoint (master NCU).

Machine data

MD 30560: IS_LOCAL_LINK_AXIS

specifies that the axis drive needs to be started during power-up, even if the axis is operating under the control of another NCU. It is evaluated only if machine data required to create a link grouping have been set, but link communication has failed due to an error.

Note

It may be necessary to increase the interpolation cycle due to the number of link axes and write elements.

■

Supplementary Conditions 3

3.1 Several operator panels and NCUs with control unit management option

Configuration

The number of configurable control units is only limited by the availability of bus addresses on the individual bus segments of the different bus types.

The restrictions relating to the current SW version specified in the catalog or release notes apply.

Availability

The control unit management option is available in **SW 5.3** and later.

3.2 Several operator panels and NCUs, standard functionality

Availability from SW 3.1

Configuration of "several operator panels and several NCUs":
available in the basic version. The number of NCUs that can be connected is limited to 1 and the operator panels to 2. One of the panels must be an OP030.
Programming language Step 7 can be used.

Expansions from SW 3.2

Configuration "1 operator panel and up to 3 NCUs"
available according to Figs. 1-2 and 1-3.

When the link is created via the MPI line (187.5 kbps), a PLC-CPU315 must be used if the configuration includes more than one NCU since this PLC allows variable setting of the NC address.
Addresses must likewise be specified for exchange of data between PLCs via PROFIBUS-DP (PLC-CPU315 only) or for global data (duplicate address assignments).

Expansions from SW 3.5

Configuration "1 operator panel and up to 4 NCUs and locally 1 MMC each"
available according to Fig. 1-4.
Data exchange between NC and PLC is now also available with the PLC-CPU 314.

Expansions from SW 4

Operation of the m:n connection
via the channel menu (see section 2.1.11), which can be selected via the "Switchover channel" key.
Prerequisite for the channel menu is a configuration via the NETNAMES.INI file (see /IAD/, Installation and Start-up Guide 840D, section MMC).
The channel menu function is an option.

Bus connection:
The address space (previously 0, ..., 15) has been extended to (0, ..., 31).

Note

If an address higher than 15 is used, all components connected to the bus must be capable of processing addresses between 0 and 31.

3.3 Link axes

Availability

1. Precondition is that the NCUs are networked with link modules.
2. The **link axis** function is an option that is available from SW 5 for NCU 573.2 in variants for 12 axes and 31 axes; it is required for each link axis (max. 32). It is determined by number.
3. The **axis container** function is an option that is available from SW 5 for NCU 573 12 axes/31 axes; it is required for each container. If a link module is employed, this function can be used without the additional option.

References: /PHD/, Configuring Guide NCU 571–573.2

3.4 Axis container

Availability

Axis container is an option that is available for the NCU 573.2 with SW 5 and higher. In cases where axis containers are configured for link axes, the supplementary conditions for such containers as defined in Section 3.3 “Link axes” also apply.

■

Notes

Data Descriptions (MD, SD) 4

4.1 Machine data for several operator panels

The following machine data is provided for functions with SW 3.2 and higher:

10134 MD number	**MM_NUM_MMC_UNITS** Resource units for MMC communications partners that are possible at the same time		
Default value: 6 (FM-NC: 4, NCU571: 3, 810D: 3, MMC 100–103: 2, OP030: 1)	Min. input limit: 1	Max. input limit: 10	
Changes effective after Power On	Protection level: 2 / 2	Unit: –	
Data type: DWORD	Applies from SW version: 3.2		
Significance:	Number of simultaneously possible MMC communication partners with which the NCU can exchange data. The value influences how many communications jobs can be managed by the NC. The higher the value, the more MMCs can be simultaneously connected to the NC. Depending on the value entered in the machine data, DRAM is made available in the NCU for this function. Instructions regarding the modification of memory areas must be observed (see FB2/S7/). The unit of MD 10134 is a resource unit. A standard OP030 requires 1 resource unit and an MMC 100/103 2. OEM variants may require more or fewer resources. • Setting the value lower (than would normally be required by the number of connected MMCs) does not necessarily cause problems. Occasionally operations may not work if several communication-intensive operator actions (e.g. loading program) are being conducted at the same time: Alarm 5000 is displayed. The operator action must be repeated. • If the value is set higher, more dynamic memory than necessary will be used up. If the memory is required for other purposes, the value should be reduced accordingly.		

4.2 Machine data for link communication

4.2.1 General machine data

10002 MD number	**AXCONF_LOGIC_MACHAX_TAB[n]** List of machine axes available on an NCU (logical NCU machine axis image)		
Default value: AX1, AX2, ...	Min. input limit: –	Max. input limit: –	
Changes effective after Power On	Protection level: 1 / 1	Unit: –	
Data type: STRING	Applies from SW version: 5		
Significance:	MD AXCONF_LOGIC_MACHAX_TAB maps channel axes on: 1. Local axes (default: AX1, AX2 ... AX31) The entry $MN_AXCONF_LOGIC_MACHAX_TAB[n] = AX3 assigns local axis AX3 to axis index n. (For n = 3 the default setting AX3 is available. Therefore MD blocks for Software Version 4 and lower are compatible in Version 5). 2. Link axes (axes that are physically connected to another NCU) The entry $MN_AXCONF_LOGIC_MACHAX_TAB[n] = NCj_AXi assigns axis AXi on NCU j to axis index n (link axis). Limits: n Machine axis address (of the local NCU) 1 ... 31 j NCU number 1 ... 16 i Machine axis address (of the local/remote NCU) 1 ... 31 3. Axis containers in which local or link axes are contained again. The entry $MN_AXCONF_LOGIC_MACHAX_TAB[n] = CTr_SLs assigns container r and slot s to axis index n. Limits: n Machine axis address (of the local NCU) 1 ... 31 r Container number 1 ... 16 s Slot number (location) in container 1 ... 32		
MD irrelevant for	Systems without link modules		
Application example(s)	Initial.ini (extract) on NCU3: $MN_AXCONF_LOGIC_MACHAX_TAB[4] = NC5_AX7 CHANDATA(2) $MC_AXCONF_MACHAX_USED[1] = 5 $MC_AXCONF_CHANAX_NAME_TAB[1] = MyAx_Y Part program block “G0 MyAx_Y = 100” running on NCU3/channel 2 traverses the 7th axis of NCU 5.		
Related to	AXCT_AXCONF_ASSIGN_TABi (create entries in containers i)		

10087 MD number	**SERVO_FIFO_SIZE** Size of data buffer between interpolator and position control		
Default value: 2	Min. input limit: 2	Max. input limit: 3	
Changes effective after Power On	Protection level: 3/2	Unit: –	
Data type: DWORD	Applies from SW version: 5.2		
Significance:	The MD specifies the size of the data buffer between the interpolator and position control. If several NCUs for rotary indexing machines/multi-spindle turning machines are connected via the NCU link, the value for all connected NCUs must be set to 3. This setting compensates for the difference in transmission time between local setpoints and setpoints from another NCU via the NCU link.		

11398 MD number	**AXIS_VAR_SERVER_SENSITIVE** Response of AXIS-VAR server to error condition		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after Power On	Protection level: 2/7	Unit: –	
Data type: BYTE	Applies from SW version: 5		
Significance:	If the server cannot supply any values for an axis (e.g. because the axis concerned is a link axis), then it returns a default value (generally 0). For the purpose of testing, this machine data can be used to set the axis data server sensitively so that it returns an error message rather than default values. 0: Default value 1: Error message		
MD irrelevant for	Systems without link modules		
Related to	MM_NCU_LINK_MASK		

12510 MD number	**NCU_LINKNO** NCU number in an NCU group		
Default value: 1	Min. input limit: 1	Max. input limit: 16	
Changes effective after Power On	Protection level: 1 / 1	Unit: –	
Data type: DWORD	Applies from SW version: 5		
Significance:	Number of name for identifying an NCU within an NCU grouping. In an NCU grouping (NCU cluster), the NCUs are interconnected via a link bus.		
MD irrelevant for	Systems without link modules		
Application	See Section 2 “Configuration of link axes”		
Related to	MM_NCU_LINK_MASK		

12520 MD number	**LINK_TERMINATION** NCU numbers for which bus terminating resistors are active		
Default value: 0	Min. input limit: 0	Max. input limit: 15	
Changes effective after Power On	Protection level: 1 / 1	Unit: –	
Data type: BYTE	Applies from SW version: 5		
Significance:	LINK_TERMINATION defines for which NCUs the bus terminating resistors must be activated by the link module for the timing circuit.		
Related to	MM_NCU_LINK_MASK		

12530 MD number	**LINK_NUM_OF_MODULES** Number of NCU link modules		
Default value: 2	Min. input limit: 2	Max. input limit: 16	
Changes effective after Power On	Protection level: 1 / 1	Unit: –	
Data type: DWORD	Applies from SW version: 5		
Significance:	The machine data specifies how many link modules are taking part in the link communication.		
MD irrelevant for	Systems without link modules		
Related to	MM_NCU_LINK_MASK		

12540 MD number	**LINK_BAUDRATE_SWITCH** Link bus baud rate		
Default value: 9	Min. input limit: 0	Max. input limit: 9	
Changes effective after Power On	Protection level: 1 / 1	Unit: –	
Data type: DWORD	Applies from SW version: 5		
Significance:	The values entered here define the baud rate assigned for the link communication: **Set value** **Rate** 0 9,600 Kbaud 1 19,200 Kbaud 2 45,450 Kbaud 3 93,750 Kbaud 4 187,500 Kbaud 5 500,000 Kbaud 6 1,500 Mbaud 7 3,000 Mbaud 8 6,000 Mbaud 9 12,000 Mbaud		
MD irrelevant for	Systems without link modules		
Related to	MM_NCU_LINK_MASK		

12550 MD number	**LINK_RETRY_CTR** Maximum number of message frame repeats in event of error		
Default value: 4	Min. input limit: 1	Max. input limit: 15	
Changes effective after Power On	Protection level: 1 / 1	Unit: –	
Data type: DWORD	Applies from SW version: 5		
Significance:	Maximum number of message frame repeats in event of error. If an error is detected during data transfer, the transmission can be repeated as often as specified in the MD. Baud rateRecommended repetition value 187,500 Kbaud 1 500,000 Kbaud 1 1,500 Mbaud 1 3,000 Mbaud 2 6,000 Mbaud 3 12,000 Mbaud 4		
MD irrelevant for	Systems without link modules		
Related to	MM_NCU_LINK_MASK		

12701 ... 12716 MD number	**AXCT_AXCONF_ASSIGN_TAB1[s] ... AXCT_AXCONF_ASSIGN_TAB16[s]** List of axes in axis container 1, ..., 16		
Default value: “ ” (empty string)	Min. input limit: –	Max. input limit: –	
Changes effective after Power On	Protection level: 7/2	Unit: –	
Data type: STRING	Applies from SW version: 5		
Significance:	Assignment of an axis container slot (slot s) to a machine axis or link axis. Up to 32 slots can be assigned to axes in an axis container. Notation for entries: NCm_AXn with NCU number m: 1...16 and machine axis address n: 1 ... 31 Example: NC2_AX1 ; The axis is on NCU2 where it is the 1st machine axis. AX5 ; Local axis 5 with only one NCU; the axis container mechanism is used ; only by more than one channel on an NCU. The reference to an axis container slot of a channel is defined by machine data MD $MC_AXCONF_MACHAX_USED and MD $MN_AXCONF_AXCONF_LOGIC_MACHAX_TAB. The axis that is actually assigned at a specific time depends on the container rotation state. All **channels** that access an axis container use the entries stored here. If channels from different NCUs access this container, pay attention to consistency across the NCUs! Example: CHANDATA(1) $MC_AXCONF_MACHAX_USED[4]=9 $MN_AXCONF_LOGIC_MACHAX_TAB[8]=CT1_SL1 $MN_AXCT_AXCONF_ASSIGN_TAB1[0]=“NC1_AX1” $MN_AXCT_AXCONF_ASSIGN_TAB1[1]=“NC2_AX1” This machine data is distributed via the NCU link. If $MA_AXCONF_ASSIGN_MASTER_CHAN is set for a container axis, then $MC_AXCONF_MACHAX_USED in the assigned channel must refer to the entry in $MN_AXCONF_LOGIC_MACHAC_TAB. This contains a reference to the axis container slot to which this axis is assigned in the default setting. A power-up alarm is otherwise generated.		
Related to	AXCONF_LOGIC_MACHAX_TAB		

12750 MD number	**AXCT_NAME_TAB** List of axis container names		
Default setting: “CT1”, “CT2”, ... “CT16”	Min. input limit: –	Max. input limit: –	
Changes effective after Power On	Protection level: 1 / 1	Unit: –	
Data type: STRING	Applies from SW version: 5.2		
Significance:	A name can be assigned to each axis container. The axis container rotation commands AXCTSWE and AXCTSWED can be used in conjunction with these names. If no container names are defined, channel axis names must be used (SW 5.1) which address the desired axis container via the logical machine axis image. This also applies to the addressing of system variables for axis containers. The default axis container names are CT1, CT2, ..., CT16		
Application	The name of the 1st axis container is AXCT1. Definition in MD: $MN_AXCT_NAME_TAB[0] = “AXCT1”		

18700 MD number	**MM_SIZEOF_LINKVAR_DATA** Size of the NCU link variable memory		
Default value: 0	Min. input limit: 0	Max. input limit: 100	
Changes effective after Power On	Protection level: 7/2	Unit: –	
Data type: DWORD	Applies from SW version: 5		
Significance:	Defines the number of bytes of link memory for NCU-global data.		

18780 MD number	**MM_NCU_LINK_MASK** Activation of NCU link communication		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after Power On	Protection level: 1 / 1	Unit: –	
Data type: DWORD	Applies from SW version: 5		
Significance:	Activation of NCU link communication. Bit-coded activation data (only 1 variant currently defined). A value of 1 must be set to activate link communication. (can be used for start-up of local axes [1:1], before link connections are powered up)		
Related to	IS_LOCAL_LINK_AXIS, NCU_LINK_NO, LINK_TERMINATION, LINK_NUM_OF_MODULES, LINK_BAUDRATE_SWITCH, LINK_RETRY_CTR		

18790 MD number	**MM_MAX_TRACE_LINK_PIONTS** Size of trace data buffer for NCU link		
Default value: 8	Min. input limit: 0	Max. input limit: 500	
Changes effective after Power On	Protection level: 1 / 1	Unit: –	
Data type: DWORD	Applies from SW version: 5		
Significance:	MM_MAX_TRACE_LINK_DATAPIONTS defines the size of an internal data buffer containing the trace recordings for the NCU link functionality. The MD is evaluated only if bit 0 is set in MM_TRACE_LINK_DATA_FUNCTION BIT 0.		
Related to	TRACE_SCOPE_MASK, MM_TRACE_DATA_FUNCTION, MM_MAX_TRACE_DATAPOINTS, TRACE_STARTTRACE_EVENT, TRACE_STARTTRACE_STEP, TRACE_STOPTRACE_EVENT, TRACE_STOPTRACE_STEP, TRACE_VARIABLE_NAME, TRACE_VARIABLE_INDEX, MM_TRACE_LINK_DATA_FUNCTION		

4.2.2 Channel-specific machine data

28160 MD number	**MM_NUM_LINKVAR_ELEMENTS** Number of write elements for the NCU link variables		
Default value: 0	Min. input limit: 0	Max. input limit: –	
Changes effective after Power ON	Protection level: 7/2	Unit: –	

28160 MD number	**MM_NUM_LINKVAR_ELEMENTS** Number of write elements for the NCU link variables	
Data type: DWORD		Applies from SW version: 5
Significance:	Defines the number of elements available to the user for writing link variables ($A_DLx) from the part program. Each write process to a link variable requires a write element. Enabling the element is synchronous with the next traversing block. This way the MD defines the number of global link variables that can be written from a part program block. Enabled write elements can be reused for new write processes. The memory requirements per element are approx. 24 bytes. One element is required to write one link variable. The element is enabled in synchronism with the next motion block. The elements are also used for block search with NCU link variables.	

4.2.3 Axis-specific machine data

30554 MD number	**AXCONF_ASSIGN_MASTER_NCU** Initial setting defining which NCU generates setpoints for the axis		
Default value: 0	Min. input limit: 0	Max. input limit: 16	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: BYTE		Applies from SW version: 5.3	
Significance:	The MD is evaluated only if NCU link communication is configured. If a machine axis is activated via $MC_AXCONF_LOGIC_MACHAX_TAB in several NCUs of a grouping, then a master NCU must be defined. This master NCU will be assigned the task of generating setpoints for the axis during power-up. In the case of axes activated in only one NCU, the number of the relevant NCU or 0 must be entered here. Any other entry will generate an alarm during power-up. The MD must be set identically for all the axes in an axis container!		
Related to	MD 10002: AXCONF_LOGIC_MACHAX_TAB MD 30550: AXCONF_ASSIGN_MASTER_CHAN		

30560 MD number	**IS_LOCAL_LINK_AXIS** Axis is a local link axis		
Default value: 0	Min. input limit: 0	Max. input limit: 0	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: BOOLEAN		Applies from SW version: 5	
Significance:	The MD is evaluated only if the NCU link function “$MN_MM_NCU_LINK_MASK = 1” has been activated, but NCU link communication is not yet functional, e.g. owing to the fact that not all NCUs in the NCU link grouping have yet been started up. Setting IS_LOCAL_LINK_AXIS = 0 determines that the axis will not be linked to the local NCU during power-up, because this would require an external partner (another NCU). Drive data can, however, be supplied to the drive for this axis.		
MD irrelevant for	Systems without link modules		
Related to	MM_NCU_LINK_MASK		

4.3 Setting data for link communication

41700 SD number	**AXCT_SWWIDTH** Axis container rotation setting		
Default value: 1	Min. input limit: 1	Max. input limit: –	
Changes effective after NewConfig	Protection level: 3 / 3	Unit: –	
Data type: BYTE	Applies from SW version: 5		
Significance:	Setting specifying how many increments an axis container must rotate. The SD is NCU-global. As an axis container in which not only local axes are administered is NCU-global, in this case SD 41700 is also NCU-global. All **NCUs** whose axes appear in the axis container use the increment value stored here. All **channels** that access an axis container use the increment value stored here. If a value which is higher than the number of slots occupied in the relevant axis container is entered, the value is calculated modulo in relation to the number of occupied slots. Unlike other setting data, a NEWCONF must be performed to refresh this setting data.		

■

Signal Descriptions 5

5.1 Defined logical functions/Defines

Note

The specified values must be entered in the interface areas indicated in the tables below. The values must be checked to distinguish between the logical functions and results.

Table 5-1 BUSTYP

Name	Value	Interface DB19	Meaning
MPI	1	DBW 100,102,104, 120, 130 bits 8–15	MMC to MPI, 187.5 Kbaud
OPI	2	“	MMC to MPI, 1.5 Kbaud

STATUS

Table 5-2 Functions

Name: Status	Value	Interface DB19	Meaning
OFFL_REQ_PLC	1	Online interface 1. : DBB 124 2. : DBB 134	PLC to MMC: PLC would like to suppress MMC, sends offline request to MMC
OFFL_CONF_PLC	2	Online interface 1. : DBB 124 2. : DBB 134	MMC to PLC: Acknowledgement of OFFL_REQ_PLC The meaning of the signal is dependent on Z_INFO DBB 125 or DBB 135
OFFL_REQ_OP	3	Online interface 1. : DBB 124 2. : DBB 134	MMC to PLC: MMC would like to go offline from this NCU and outputs an offline request
OFFL_CONF_OP	4	Online interface 1. : DBB 124 2. : DBB 134	PLC to MMC: Acknowledgement of OFFL_REQ_OP The meaning of the signal is dependent on Z_INFO DBB 125 or DBB 135
ONL_PERM	5	Online request interface DBB 108	PLC to MMC: PLC notifies MMC as to whether it can go online or not. The meaning of the signal is dependent on Z_INFO: DBB109
S_ACT	6	Online interface 1. : DBB 124 2. : DBB 134	MMC to PLC: MMC goes online or changes operating focus. The meaning of the signal is dependent on Z_INFO DBB 125 or DBB 135

Table 5-2 Functions

Name: Status	Value	Interface DB19	Meaning
OFFL_REQ_FOC	7	Online interface 1. : DBB 124 2. : DBB 134	MMC to PLC: MMC would like to take operating focus away from this NCU
OFFL_CONF_FOC	8	Online interface 1. : DBB 124 2. : DBB 134	PLC to MMC: Acknowledgement of OFFL_REQ_FOC The meaning of the signal is dependent on Z_INFO DBB 125 or DBB 135
ONL_REQ_FOC	9	Online interface 1. : DBB 124 2. : DBB 134	MMC to PLC: MMC would like to set operating focus to this NCU
ONL_PERM_FOC	10	Online interface 1. : DBB 124 2. : DBB 134	PLC to MMC: Acknowledgement of ONL_REQ_FOC The meaning of the signal is dependent on Z_INFO DBB 125 or DBB 135

Table 5-3 Z_INFO

Name	Value	Interface DB 19	Meaning
DISC_FOC	9	DBB125 DBB135	MMC switches operating focus to another NCU.
OK	10	DBB 109 bits 0–3 DBB125 DBB135	Positive acknowledgement
CONNECT	11	DBB125 DBB135	MMC has gone online on this NCU.
MMC_LOCKED	13	DBB 109 bits 0–3 DBB125 DBB135	MMC has set switchover disable. Processes which must not be interrupted by a switchover operation are currently in progress on this MMC.
PLC_LOCKED	14	DBB 109 bits 0–3 DBB125 DBB135	The MMC switchover disable is set in the MMC–PLC interface. MMC cannot go offline from this NCU or change operating focus.
PRIO_H	15	DBB 109 bits 0–3 DBB125 DBB135	MMCs with a higher priority are operating on this NCU. MMC cannot go online to this NCU.

Table 5-4 STATUS and Z_INFO can be combined as follows

Name: Status	Z_INFO	Meaning
OFFL_REQ_PLC	OK	PLC would like to suppress online MMC and sends the offline request
OFFL_CONF_PLC	OK	MMC positively acknowledges the offline request from the PLC. MMC then goes offline.
OFFL_CONF_PLC	MMC_LOCKED	MMC negatively acknowledges the offline request. MMC cannot go offline because uninterruptible processes are currently in progress.
OFFL_REQ_OP	OK	MMC would like to go offline from the online NCU and outputs an offline request
OFFL_CONF_OP	OK	PLC positively acknowledges the offline request. MMC then goes offline from this NCU.
OFFL_CONF_OP	PLC_LOCKED	PLC negatively acknowledges the offline request from the MMC. User has set the MMC switchover disable, MMC cannot go offline, MMCx_SHIFT_LOCK = TRUE, x=1 or 2, 1st or 2nd MMC–PLC interface.
ONL_PERM	No. of MMC–PLC online interface, OK	PLC issues the online enabling command to the requesting MMC. MMC can then go online to this NCU. Contents of Z_INFO: Bit 0 ..3: OK Bit 4 .. 7: No. of the MMC–PLC online interface to which MMC must be connected: 1 First MMC–PLC online interface 2 Second MMC–PLC online interface
ONL_PERM	MMC_LOCKED	The requesting MMC cannot go online. Two MMCs on which uninterruptible processes are in progress are connected online to this NCU. The PLC cannot suppress either of the two MMCs.

Table 5-4 STATUS and Z_INFO can be combined as follows

Name: Status	Z_INFO	Meaning
ONL_PERM	PLC_LOCKED	The requesting MMC cannot go online. User has set the MMC switchover disable, MMCx_SHIFT_LOCK = TRUE, x=1 or 2, 1st or 2nd MMC online interface.
ONL_PERM	PRIO_H	The requesting MMC cannot go online. Two MMCs that are both higher priority than the requesting MMC are connected online to the NCU. The PLC cannot suppress either of the two MMCs.
S_ACT	CONNECT	The requesting MMC has gone online. The PLC now switches on the MMC sign-of-life monitoring function.
S_ACT	DISC_FOCUS	Server MMC has disconnected the operating focus from this NCU.
OFFL_REQ_FOC	OK	Server MMC would like to disconnect the operating focus from this NCU and outputs an offline focus request.
OFFL_CONF_FOC	OK	PLC positively acknowledges the offline focus request. Server MMC can disconnect the operating focus.
OFFL_CONF_FOC	PLC_LOCKED	PLC negatively acknowledges the offline focus request. User has set the MMC switchover disable, server MMC cannot disconnect the operating focus, MMCx_SHIFT_LOCK = TRUE, x=1 or 2, 1st or 2nd MMC–PLC interface.
ONL_REQ_FOC	OK	Server MMC would like to set the operating focus on this NCU and outputs an online focus request.
ONL_PERM_FOC	OK	PLC positively acknowledges the online focus request. Server MMC then switches the operating focus to this NCU.
ONL_PERM_FOC	PLC_LOCKED	PLC negatively acknowledges the online focus request. User has set the MMC switchover disable, server MMC cannot set the operating focus,, MMCx_SHIFT_LOCK = TRUE, x=1 or 2, 1st or 2nd MMC–PLC interface.

5.2 Interfaces in DB 19 for M:N

The MMC–PLC interface in DB19 is divided into 3 areas

Online request interface

The online request sequence is executed on this interface if an MMC wants to go online.

MMC writes its client ID to ONL_REQUEST and waits for the return of the client ID in

ONL_CONFIRM.

After the positive acknowledgement from the PLC, the MMC sends its parameters and waits for online permission (in PAR_STATUS, PAR_Z_INFO).

MMC parameter transfer:

Client identification	–> PAR_CLIENT_IDENT
MMC type	–> PAR_MMC_TYP
MCP address	–> PAR_MSTT_ADR

With the positive online permission, the PLC also sends the number of the MMC–PLC online interface DBB109.4–7 to be used by the MMC.

The MMC then goes online and occupies the online interface assigned by the PLC.

Online interfaces

Two MMCs can be connected online to one NCU at the same time.

The online interface is available for each of the two online MMCs separately.

After a successful online request sequence, the MMC receives the number of its online interface from the PLC.

The MMC parameters are then transferred to the corresponding online interface by the PLC.

The MMC goes online and occupies its own online interface via which data are then exchanged between the MMC and PLC.

MMC data interfaces

User data from/to the MMC are defined on these:

- DBB 0–49 MMC1 interface
- DBB 50–99 MMC2 interface

These data and signals are always needed to operate MMCs.

M:N sign-of-life monitoring

This is an additional monitoring function which must not be confused with the MMC sign-of-life monitor. For further information, please refer to the relevant signals.

In certain operating states, MMCs with activated M:N switchover (parameterizable in NETNAMES.INI) must be capable of determining from a PLC data whether they need to wait or not before linking up with an NCU.

Example:
MMC with an activated control unit switchover function must be capable of starting up an NCU without issuing an online request first.

MMC must go online for service-related reasons.

The operation is coordinated in the online request interface via data DBW110:
M_TO_N_ALIVE

The M:N sign of life is a ring counter which is incremented cyclically by the PLC or set to a value of 1 when it overflows.

Before an MMC issues an online request, it must check the sign of life to establish whether the M:N switchover is activated in the PLC.

Procedure:

MMC reads the sign of life at instants T0 and T0 + 1.

Case 1: Negative acknowledgement for read operation, DB19 does not exist. MMC goes online without prior online request

Case 2: m_to_n_alive = 0
Control unit switchover not activated.
MMC goes online without prior online request

Case 3: m_to_n_alive (T0) = m_to_n_alive (T0+1)
Control unit switchover not activated
MMC goes online without prior online request.

Case 4: m_to_n_alive (T0) <> m_to_n_alive (T0+1)
Control unit switchover activated

Cases 1 to 3 apply only under special conditions and not in normal operation.

Online request interface

No.	Name
Possible value	Meaning

DB19 DBW100	ONL_REQUEST
Client_Ident	MMC would like to go online and use the online request interface. It first writes its Client_Ident as a request. Bit 8 .. 15: Bus type: MPI 1 or MCP 2 Bit 0 .. 7: MMC bus address

DB19 DBW102	ONL_CONFIRM
Client_Ident	If the online request interface is not being used by another MMC, the PLC returns the Client identification as positive acknowledgement. Bit 8 .. 15: Bus type: MPI 1 or MCP 2 Bit 0 .. 7: MMC bus address

DB19 DBW104	PAR_CLIENT_IDENT MMC parameter transfer to PLC
Client_Ident	Bit 8 .. 15: Bus type: MPI 1 or MCP 2 Bit 0 .. 7: MMC bus address

DB19 DBB106	PAR_MMC_TYP MMC parameter transfer to PLC
MMC type from NETNAMES.INI	Type properties of the MMC configured in file NETNAMES.INI. Evaluated by the PLC when MMC is suppressed (server, main/secondary operator panel, ...), see description of file NETNAMES.INI

DB19 DBB107	PAR_MSTT_ADR MMC parameter transfer to PLC
MCP address from NETNAMES.INI	Address of MCP to be switched over or activated/deactivated with the MMC. Parameter from NETNAMES.INI
255	No MCP is assigned to MMC, no MCP will be activated/deactivated

DB19 DB108	PAR_STATUS PLC sends MMC pos./neg. online permission
ONL_PERM (5)	PLC notifies MMC as to whether it can go online or not. The meaning of the signal is dependent on PAR_Z_INFO:

DB19 DBB109	PAR_Z_INFO PLC sends MMC pos./neg. online permission
No. of MMC–PLC online interface, OK (10)	PLC issues the online enabling command to the requesting MMC. MMC can then go online to this NCU. Bit 0 ..3: OK Bit 4 .. 7: No. of the MMC–PLC online interface to which MMC must be connected: 1 First MMC–PLC online interface 2 Second MMC–PLC online interface
MMC_LOCKED (13)	The requesting MMC cannot go online. Two MMCs on which uninterruptible processes are in progress are connected online to this NCU. The PLC cannot suppress either of the two MMCs.
PLC_LOCKED (14)	The MMC switchover disable is set in the MMC–PLC interface.
PRIO_H (15)	The requesting MMC cannot go online. Two MMCs that are both higher priority than the requesting MMC are connected online to the NCU. The PLC cannot suppress either of the two MMCs.

Sign of life of M:N switchover

DB19 DBW110	M_TO_N_ALIVE
1 .. 65535	Ring counter that is cyclically incremented by the PLC. Indicator for the MMCs that the M:N switchover is active and ready.

1. MMC–PLC online interface

DB19 DBW120	MMC1_CLIENT_IDENT
	See PAR_CLIENT_IDENT After issuing positive online permission, the PLC transfers the MMC parameters to the online interface PAR_CLIENT_IDENT –> MMC1_CLIENT_IDENT

DB19 DBB122	MMC1_TYP
	See PAR_MMC_TYP After issuing positive online permission, the PLC transfers the MMC parameters to the online interface PAR_MMC_TYP –> MMC1_TYP

DB19 DBB123	MMC1_MSTT_ADR
	See PAR_ MSTT_ADR After issuing positive online permission, the PLC transfers the MMC parameters to the online interface PAR_ MSTT_ADR –> MMC1_MSTT_ADR

DB19 DBB124	MMC1_STATUS Requests from online MMC to PLC or vice versa The meaning of the signal is dependent on MMC1_Z_INFO, see also: DEFINEs possible combinations of STATUS and Z_INFO for control unit switchover
OFFL_REQ_PLC (1)	PLC to MMC: PLC would like to suppress MMC, sends offline request to MMC
OFFL_CONF_PLC (2)	MMC to PLC: Acknowledgement of OFFL_REQ_PLC
OFFL_REQ_OP (3)	MMC to PLC: MMC would like to go offline from this NCU and outputs an offline request
OFFL_CONF_OP (4)	PLC to MMC: Acknowledgement of OFFL_REQ_OP
S_ACT (6)	MMC to PLC: MMC goes online or changes operating focus
OFFL_REQ_FOC (7)	MMC to PLC: MMC would like to take operating focus away from this NCU
OFFL_CONF_FOC (8)	PLC to MMC: Acknowledgement of OFFL_REQ_FOC
ONL_REQ_FOC (9)	MMC to PLC: MMC would like to set operating focus to this NCU
ONL_PERM_FOC (10)	PLC to MMC: Acknowledgement of ONL_REQ_FOC

DB19 DBB125	MMC1_Z_INFO Requests from online MMC to PLC or vice versa The meaning of the signal is dependent on MMC1_STATUS, see also: DEFINEs possible combinations of STATUS and Z_INFO for control unit switchover
DISC_FOC (9)	MMC switches operating focus to another NCU
OK (10)	Positive acknowledgement
CONNECT (11)	MMC has gone online to this NCU
PLC_LOCKED (14)	The MMC switchover disable is set in the MMC–PLC interface. MMC cannot go offline from this NCU or change operating focus.
PRIO_H (15)	MMCs with higher priority are online to this NCU. MMC cannot go online to this NCU.

Bit signals

DB 19 **DBX 126.0** Data block	**MMC1_SHIFT_LOCK** Disable/enable MMC switchover	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 5
Signal state 1 or signal transition 0 ——> 1	MMC switchover or change in operating focus is disabled. The current MMC–NCU connection status remains unchanged.	
Signal state 0 or signal transition 1 ——> 0	MMC switchover or change in operating focus is enabled.	

DB 19 **DBX** Data block	**MMC1_MSTT_SHIFT_LOCK** Disable/enable MCP switchover	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 5
Signal state 1 or signal transition 0 ——> 1	MCP switchover is disabled. The current MCP–NCU constellation remains unchanged.	
Signal state 0 or signal transition 1 ——> 0	MCP switchover is enabled	

DB 19 **DBX 126.2** Data block	**MMC1_ACTIVE_REQ** MMC1 requests active operating mode	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 5
Signal state 1 or signal transition 0 ——> 1	MMC to PLC: Passive MMC1 requests active operating mode	
Signal state 0 or signal transition 1 ——> 0	PLC to MMC: Request received	

DB 19 **DBX 126.3** Data block	**MMC1_ACTIVE_PERM** Active/passive operating mode	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 5
Signal state 1 or signal transition 0 ——> 1	PLC to MMC: Passive MMC can change to active operating mode	
Signal state 0 or signal transition 1 ——> 0	PLC to MMC: Active MMC must change to passive operating mode	

DB 19 **DBX 126.4** Data block	**MMC1_ACTIVE_CHANGED** Active/passive operating mode of MMC	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 5
Signal state 1 or signal transition 0 ——> 1	MMC to PLC: MMC has completed changeover from passive to active mode	
Signal state 0 or signal transition 1 ——> 0	MMC to PLC: MMC has completed changeover from active to passive mode	

DB 19 **DBX126.5** Data block	**MMC1_CHANGE_DENIED** Operating mode changeover rejected	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 5
Signal state 1 or signal transition 0 ——> 1	MMC to PLC or PLC to MMC depending on status of interface: Operating mode cannot be changed owing to uninterruptible processes on active MMC	
Signal state 0 or signal transition 1 ——> 0	MMC to PLC or PLC to MMC depending on status of interface: Acknowledgement of MMC1_CHANGE_DENIED (FALSE —>TRUE)	

2. MMC–PLC online interface

The signals of the 2nd MMC–PLC online interface are analogous in meaning to the signals of the 1st MMC–PLC online interface. MMC2_ ... replaces MMC1_... in the explanatory texts.

DB19 DBW130	MMC2_CLIENT_IDENT
	See DB19 DBW120

DB19 DBB132	MMC2_TYP
	See DB19 DBW122

DB19 DBB133	MMC2_MSTT_ADR
	See DB19 DBB 123

DB19 DBB134	MMC2_STATUS
	See DB19 DBB124

DB19 DBB135	MMC2_Z_INFO
	See DB19 DBB 125

DB19 DBX136.0	MMC2_SHIFT_LOCK
	See DB19 DBX126.0

DB19 DBX136.1	MMC2_MSTT_SHIFT_LOCK
	See DB19 DBX126.1

DB19 DBX136.2	MMC2_ACTIVE_REQ
	See DB19 DBX126.2

DB19 DBX136.3	MMC2_ACTIVE_PERM
	See DB19 DBX126.3

DB19 DBX136.4	MMC2_ACTIVE_CHANGED
	See DB19 DBX 126.4

DB19 DBW136.5	MMC2_CHANGE_DENIED
	See DB19 DBX126.5

MMC sign-of-life monitor

After an MMC has gone online to an NCU, the MMC sign of life is set in the interface. (E_BTSSReady, E_MMCMPI_Ready, E_MMC2Ready)

The signals are automatically set by the MMC when it goes online and stay set for as long as it remains online.

They are provided separately for each MMC–PLC interface and used by the PLC to monitor the MMC sign of life.

1. MMC–PLC online interface

A distinction between an MMC link via the OPI (1.5 Mbaud) or the MPI (187.5 kbaud) is made on this interface.

The signal corresponding to the bus type is set while the MMC is online.

DB**10** DBX104.0	MCP1 ready
FALSE	MCP1 is not ready
TRUE	MCP1 is ready

DB**10** DBX104.1	MCP2 ready
FALSE	MCP2 is not ready
TRUE	MCP2 is ready

DB**10** DBX104.2	HHU ready
FALSE	HHU is not ready
TRUE	HHU is ready

DB**10** DBX108.3	E_MMCBTSSReady
FALSE	No MMC online to OPI
TRUE	MMC online to MPI

DB**10** DBX108.2	E_MMCMPIReady
FALSE	No MMC online to MPI
TRUE	MMC online to MPI

2nd MMC–PLC online interface

This interface utilizes a group signal for both bus types. No distinction is made between OPI and MPI.

DB**10** DBX108.1	E_MMC2Ready
FALSE	No MMC online to OPI or MPI
TRUE	MMC online to OPI or MPI

The sign-of-life monitor is switched on by the PLC as soon as an MMC has gone online to its interface and switched off again when it goes offline.

Sign-of-life monitor is **switched on:**

– As soon as an MMC logs on online to its MMC–PLC interface with S_ACT/ CONNECT.

Sign-of-life monitor is **switched off:**

– As soon as MMC goes offline

1. MMC wants to switch over and logs off from the PLC with
   OFFL_REQ_OP/ OK
   PLC acknowledges the MMC with OFFL_CONF_OP/ OK
2. MMC is suppressed by the PLC with OFFL_REQ_PLC/ OK
   MMC acknowledges the PLC with OFFL_CONF_PLC/ OK

In both instances the PLC detects that an MMC is going offline and waits for the TRUE–FALSE edge of its sign-of-life signal.

The PLC then ceases to monitor the sign-of-life signal.

5.3 Signals for NCU link and axis container

DB10 **DBX 107.6** Data block	**NCU link active** Signal from NC channel –> PLC	
Edge evaluation:	Signal(s) updated:	Signal(s) valid from SW vers.: 5
Signal state 1 or signal transition 0 ——> 1	NCU link communication is active	
Signal state 0 or signal transition 1 ——> 0	No NCU link communication is active	
Signal irrelevant for.....	Systems without NCU modules	
References	PHD	

DB 31–61 **DBX 60.1** Data block	**NCU link axis active** Signal from NC axis –> PLC	
Edge evaluation:	Signal(s) updated:	Signal(s) valid from SW vers.: 5
Signal state 1 or signal transition 0 ——> 1	Axis is active as NCU link axis	
Signal state 0 or signal transition 1 ——> 0	Axis is used as a local axis	
Signal irrelevant for.....	Systems without NCU modules	
References	PHD	

DB 31–61 **DBX 61.1** Data block	**Axis container rotation active** Signal from NC axis –> PLC	
Edge evaluation:	Signal(s) updated:	Signal(s) valid from SW vers.: 5
Signal state 1 or signal transition 0 ——> 1	An axis container rotation is active for the axis	
Signal state 0 or signal transition 1 ——> 0	An axis container rotation is not active for the axis	

DB 31–61 **DBX 61.2** Data block	**Axis ready** Signal from NC axis –> PLC	
Edge evaluation:	Signal(s) updated:	Signal(s) valid from SW vers.: 5
Meaning	The signal is processed on the home NCU in the NCU link grouping. The home NCU is the NCU to which the axis is physically connected.	
Signal state 1 or signal transition 0 ——> 1	Axis is ready	
Signal state 0 or signal transition 1 ——> 0	Axis is not ready This status is set if – the channel or – the operating mode group or – the NCK has generated the “not ready” alarm.	

■

Examples

6

6.1 Configuration file NETNAMES.INI with control unit management option

A sample configuration file NETNAMES.INI for the MMC 1 control unit for a system with four NCUs on the OPI is outlined below.
See Section 2.1.4 for explanations.

Note

The marginal notes (bold print) on the left of the page serve to structure the information and are not part of the file.

```
; NETNAMES.INI Example 1 start
```

MMC identification

```
; Identification division
[own]
owner           = MMC_1
```

MMC-NCU connections

```
; Connection division
[conn MMC_1]

conn_1          = NCU_1         ; NCU 1
conn_2          = NCU_2         ; NCU 2
conn_3          = NCU_3         ; NCU 3
conn_4          = NCU_4         ; NCU 4
```

Bus identification

```
; Description division
[param network]

bus             = OPI           ; OPI bus (1.5 Mbaud)
```

MMC description

```
[param MMC_1]
mmc_typ        = 40          ; = 0100 0000: MMC is server and
                             ; main control panel
mmc_bustyp     = OPI         ; bus the MMC is attached to
mmc_address    = 10          ; MMC address
mstt_address   = 6           ; address of OPI to be switched simult-
                             ; aneously
name           = MMC_LINKS   ; name of MMC
start_mode     = ONLINE      ; MMC switches during power-up to
                             ; online mode on the DEFAULT NCU, joint
                             ; channel data see below
```

Description of NCU components

```
[param NCU_1]
type           = NCU_572     ; NCU type
nck_address    = 20          ; address j of NCU component on bus
plc_address    = 20          ; address p of PLC component on bus
name           = NCU1        ; name of NCU

[param NCU_2]
type           = NCU_572     ; NCU type
nck_address    = 21          ; address j of NCU component on bus
plc_address    = 21          ; address p of PLC component on bus
name           = NCU2        ; name of NCU

[param NCU_3]
type           = NCU_572     ; NCU type
nck_address    = 22          ; address j of NCU component on bus
plc_address    = 22          ; address p of PLC component on bus
name           = NCU3        ; name of NCU

[param NCU_4]
type           = NCU_572     ; NCU type
nck_address    = 23          ; address j of NCU component on bus
plc_address    = 23          ; address p of PLC component on bus
name           = NCU4        ; name of NCU

; End of description division
```

Channel data

```
; Sample of a channel menu configuration
; with M:N assignment option

[chan MMC_1]
DEFAULT_logChanSet = G_1        ; Group setting during power-up
DEFAULT_logChan = K_1_1         ; Channel setting during power-up
ShowChanMenu  =TRUE             ; Display channel menu

                  ; List of channel groups
logChanSetList    = G_1, G_2, G_3, G_4

[G_1]
logChanList       = K_1_1, K_1_2  ; Group G_1 channels

[G_2]
logChanList       = K_2_1, K_2_2  ; Group G_2 channels

[G_3]
logChanList       = K_3_1, K_3_2  ; Group G_3 channels

[G_4]
logChanList       = K_4_1, K_4_2  ; Group G_4 channels

[K_1_1]
logNCName         = NCU_1         ; 1st Channel of 1st group
ChanNum           = 1

[K_1_2]
logNCName         = NCU_1         ; 2nd Channel of 1st group
ChanNum           = 2

[K_2_1]
logNCName         = NCU_2         ; 1st Channel of 2nd group
ChanNum           = 1

[K_2_2]
logNCName         = NCU_2         ; 2nd Channel of 2nd group
ChanNum           = 2

[K_3_1]
logNCName         = NCU_3         ; 1st Channel of 3rd group
ChanNum           = 1

[K_3_2]
logNCName         = NCU_3         ; 2nd Channel of 3rd group
ChanNum           = 2

[K_4_1]
logNCName         = NCU_4         ; 1st Channel of 4th group
ChanNum           = 1

[K_4_2]
logNCName         = NCU_4         ; 2nd Channel of 4th group
ChanNum           = 2


; NETNAMES:INI example 1 end
```

6.2 User-specific re-configuring of PLC program “Control unit switchover”

Introduction

The solution outlined roughly below should be selected only if at least one of the following configuring requirements is applicable:

- Displacement strategy which differs from standard functionality
- Operating mode switchover which differs from standard functionality
- Independent handling of override switch for switchover of control unit
- Existence of a 2nd machine control panel on an MMC

Method of description:

1. Description of operational sequences
2. Description of available functionality (Defines)
3. Graphic representation of sequences in diagrammatic form

Implementation details can also be obtained from the standard configuration which is included in the toolbox.

6.2.1 Description of operational sequences (overview)

Overview:

MMC sends online request	An MMC would like to link up with an NCU and sends this request to the PLC of the relevant NCU.
MMC arrives	An MMC goes online to an NCU, i.e. it links up to the NCU.
MMC goes	An MMC breaks off the link to an NCU.
Displacement	An MMC must abort the link with an NCU because another MMC wants to go online to the same NCU.
Operating focus changeover in server mode	A server maintains a permanent link to the NCUs to which it is assigned. The operator can switch the operating focus from one NCU to another without interrupting the existing link.
Active/passive operating mode	An online MMC can operate in two different modes: Active mode: Operator can operate and monitor Passive mode: Operators sees header information and the “passive” identifier.
MCP switchover	As an option, an MCP assigned to the MMC can be switched over at the same time as the MMC.

6.2.2 Description of operational sequences (details)

Introduction

The operational sequences are described using identifiers for defined, logical functions (example: `OFFL_REQ_OP/ OK`) whose programming application has been described earlier in this section. The functions are coded according to Section 5.1. The functions store values in the interface which can be addressed from the PLC and the MMC. An MMC utilizes the online-request interface while it competing for the use of an online interface. MMCs which are already linked to an NCU utilize one of the two available online interfaces. Details of these interfaces can be found in Chapter 5 and in

References: Lists

In order to illustrate complete operating sequences, the description covers MMC activities which cannot be influenced as well as **modifiable PLC activities**.

MMC sends online request

If the MMC is already linked online to an NCU (online NCU) and would like to communicate with another NCU (target NCU), it must first notify the PLC of the online NCU that it wishes to switch over to the target NCU.

It sends the offline request `OFFL_REQ_OP/ OK` to the online PLC.

`OFFL_CONF_OP/ OK`:

Online PLC has received the offline request. MMC can now send an online request to the target PLC.

`OFFL_CONF_OP/ PLC_LOCKED`

Online PLC has received the offline request. The MMC switchover is disabled in the MMC–PLC interface. The MMC cannot link up with another NCU and must remain online.

On receipt of the positive acknowledgement `OFFL_CONF_OP/ OK`, the MMC sends its online request to the target PLC of the relevant NCU by transmitting its *client identification*.

Client identification: Unique MMC identifier comprising bus type and MMC bus address. (ONL_REQUEST DB19, DBW100)

The target PLC sends the MMC a positive or negative acknowledgement:

Pos. acknowledgement: Target PLC returns the client identification to the MMC. (ONL_CONFIRM, DB19, DBW102)
MMC sets its parameters on the online-request interface. (Client ident, MMC type, MCP address).
MMC can go online once it has received online permission from the target PLC.

Neg. acknowledgement: Target PLC does not return the client identification to the MMC. (ONL_CONFIRM, DB19, DBW102 not identical to client identification of requesting MMC).
MMC cannot go online.

Example:
Another MMC is currently switching over to the same NCU. This switchover operation must not be interrupted. The MMC remains online to the online NCU.

Once the MMC has received positive acknowledgement from the PLC, it may need to displace another online MMC. It will then receive positive/negative online permission from the PLC.

Positive:

`ONL_PERM/ OK`

On receipt of positive online permission (DB 19, DBB 108, 109), the MMC can go online. An MMC–PLC interface is allocated to the MMC at the same time as online permission. (1 or 2, details can be found in the interface description in Chapter 5).

Negative:

`ONL_PERM/ MMC_LOCKED`

The requesting MMC cannot go online. Two MMCs currently executing uninterruptible processes are linked online to this NCU. The PLC cannot displace either of the two MMCs. The MMC remains online to the online NCU.

`ONL_PERM/ PLC_LOCKED`

The requesting MMC cannot go online. The MMC switchover is disabled in the MMC–PLC interface. The MMC remains online to the online NCU.

`ONL_PERM/ PRIO_H`

The requesting MMC cannot go online. Two MMCs that are both higher priority than the requesting MMC are connected online to the NCU. The PLC cannot displace either of the two MMCs. The MMC remains online to the online NCU.

MMC arrives

Once the MMC has sent an online request to the target PLC and received online permission from it, it can set up a link to the target NCU.

It goes online and notifies the PLC with (station active) `S_ACT/ CONNECT` that it has linked up with the NCU.

The MMC sets up its sign of life signal in accordance with the allocated interface.

The MMC then requests
in the case of operator panel: Active operating mode on the target NCU or
in the case of a server: Operating focus on the target NCU.

The PLC then activates MMC sign-of-life monitoring for the new MMC.

See: Active/passive operating mode

See: Operating focus changeover in server mode

MMC goes

An MMC aborts communication with an NCU.

Communication can be aborted for two different reasons:

1. The operator wishes to switch the MMC to another NCU. The MMC has send an online request to the target PLC and received online permission (`ONL_PERM/ OK`). It has notified the online PLC of its intention to switch over with `OFFL_REQ_OP/ OK` and received a positive acknowledgement (`OFFL_CONF_OP/ OK`). Due to the switchover to the target NCU, the MMC sign of life in the online PLC is changed from TRUE to FALSE. The falling edge combined with the sequence described above signals to the online PLC that the MMC has broken off the link to the online NCU. If an MCP is assigned to the MMC and activated, it is now deactivated by the PLC. Passive operating mode is set in the PLC for the MMC which has gone offline.
   See: Active/passive operating mode
2. The MMC is excluded (displaced) from the PLC by the online request from another MMC. See displacement.

Displacement

Two MMCs are linked online to an NCU, each is occupying an MMC–PLC interface. A third MMC would like to go online.

The PLC must displace one of the two MMCs according to a predefined strategy.

It requests the MMC to be displaced to abort communication with the NCU by sending it the offline request (`OFFL_REQ_OP/ OK`).

The MMC returns a positive or negative acknowledgement to the PLC:

Positive:

`OFFL_CONF_PLC/ OK`

MMC breaks off the link to the NCU and switches to the offline state.

The MMC sign of life in the PLC changes from TRUE to FALSE.

The falling edge combined with the sequence described above signals to the online PLC that the MMC has broken off the link to the online NCU.

If an MCP is assigned to the MMC and activated, it must now be deactivated by the PLC.

The PLC also ceases to monitor the MMC sign of life signal.

Passive operating mode is set in the PLC for the MMC which has been displaced.

See “Active/passive operating mode” further below.

Negative:

`OFFL_CONF_PLC/ MMC_LOCKED`

The MMC is executing processes which cannot be interrupted (e.g. operation via V.24 interface, data exchange between NCU and MMC).

The MMC remains online to the current NCU.

Operating focus changeover in server mode

A server maintains a permanent link to the NCUs to which it is assigned. The operator can switch the operating focus from one NCU to another without interrupting the existing link.

If the operator wishes to switch the operating focus to another NCU, the focus PLC and target PLC must first be interrogated to determine whether they will permit a focus switchover.

The MMC first sends the focus offline request signal (`OFFL_REQ_FOC/ OK`) to the focus PLC.

The focus PLC returns either a positive or negative acknowledgement to the MMC:

Positive:

`OFFL_CONF_FOC/ OK`

PLC positively acknowledges the focus offline request. MMC can disconnect the operating focus.

Negative:

`OFFL_CONF_FOC/ PLC_LOCKED`

PLC negatively acknowledges the online focus request. The operating focus changeover is disabled in the MMC–PLC interface (same signal as for MMC switchover disable). The operating focus remains on the current NCU.

After a positive acknowledgement (`OFFL_CONF_FOC/ OK`) from the focus PLC, the MMC sends query signal `ONL_REQ_FOC/ OK` regarding focus changeover to the target PLC.

The target PLC returns a positive or negative acknowledgement to the MMC:

Positive:

`ONL_PERM_FOC / OK`

PLC positively acknowledges the focus offline request. MMC can disconnect the operating focus.

Negative:

`ONL_PERM_FOC / PLC_LOCKED`

PLC negatively acknowledges the focus offline request. The operating focus changeover is disabled in the MMC–PLC interface (same signal as for MMC switchover disable). The operating focus remains on the current NCU.

After the MMC has received permission from the target PLC to change the operating focus (`ONL_PERM_FOC, OK`), the MMC logs off from the focus PLC with `S_ACT/ DISC_FOCUS` and changes the focus to the target PLC.

The MMC must finally request active operating mode in the target NCU. The previous focus PLC must set active operating mode for this MMC–PLC interface after receiving `S_ACT/ DISC_FOCUS` and deactivate any active MCP assigned to the MMC which has gone offline.

See: Active/passive operating mode

Active/passive operating mode

After an MMC has gone online to an NCU, it can assume one of two different operating states:

Active mode: Operator can operate and monitor

Passive mode: Operators sees header information and the “passive” status identifier.

After switching to an NCU, it first requests active operating mode in the online PLC.

If two MMCs are linked online simultaneously to an NCU, one of the two is always in active mode and the other in passive mode.

The operator can request active mode on the passive MMC by pressing a key.

If an MCP has been configured for the online MMCs, the MCP of the active MMC is switched on.

The MCP of the passive MMC is deactivated, i.e. only one MCP is active at a time on an NCU.

Four signals are provided in the MMC–PLC interface for each of the two online MMCs. These signals are used by the PLC to control operating mode changeovers.

Table 6-1 Signals (x = 1, 2: 1st or 2nd MMC–PLC interface)

MMC–PLC interface	Value	Meaning
MMCx_ACTIVE_REQ	FALSE–>TRUE	MMC to PLC: Passive MMC requests active operating mode
	TRUE–>FALSE	PLC to MMC: Request received
MMCx_ACTIVE_PERM	FALSE–>TRUE	PLC to MMC: Passive MMC can change to active operating mode
	TRUE–>FALSE	PLC to MMC: Active MMD must change to passive operating mode
MMCx_ACTIVE_CHANGED	FALSE–>TRUE	MMC to PLC: MMC has completed changeover from passive to active mode
	TRUE–>FALSE	MMC to PLC: MMC has completed changeover from active to passive mode
MMCx_CHANGE_DENIED	FALSE–>TRUE	MMC to PLC or PLC to MMC depending on interface: Operating mode cannot be changed owing to uninterruptible processes on active MMC
	TRUE–>FALSE	MMC to PLC or PLC to MMC depending on interface: Acknowledgement of MMCx_CHANGE_DENIED(FALSE–>TRUE)

An example of how operating modes can be switched over is described in the following sequence.

Two MMCs online to one NCU, MMC_1 in active operating mode, MMC_2 in passive operating mode, operator requests active operating mode on MMC_2.

These sequence applies equally to the following cases:

– An MMC goes online to an NCU to which another MMC is linked online and in active mode. It requests active operating mode.

– An MMC goes online to an NCU to which no other MMC is linked online. It requests active operating mode. (The sequence “PLC requests active MMC to switch to passive operating mode” is not included here).

Signal state for this case:

Table 6-2

MMC_1	VALUE	MMC_2	Value
MMC1_ACTIVE_REQ	FALSE	MMC2_ACTIVE_REQ	FALSE
MMC1_ACTIVE_PERM	TRUE	MMC2_ACTIVE_PERM	FALSE
MMC1_ACTIVE_CHANGED	TRUE	MMC2_ACTIVE_CHANGED	FALSE
MMC1_CHANGE_DENIED	FALSE	MMC2_CHANGE_DENIED	FALSE

MMC_2 requests active operating mode and sets MMC_2_AKTIVE_REQ = TRUE.

The PLC acknowledges the request from MMC_2 with MMC_2_ACTIVE_REQ = FALSE.

The PLC then requests MMC_1 to change to passive operating mode with MMC1_ACTIVE_PERM = FALSE.

MMC_1 can respond to this request in two different ways:

1. MMC_1 can change to passive operating mode:
   MMC_1 switches from active to passive operating mode and acknowledges the changeover with
   MMC1_ACTIVE_CHANGED = FALSE.
   If an MCP is assigned to the MMC and activated, it is now deactivated by the PLC.
   The PLC notifies MMC_2 that it can change to active operating mode by sending MMC2_ACTIVE_PERM = TRUE.
   MMC_2 changes state and acknowledges the change with MMC2_ACTIVE_CHANGED = TRUE. If an MCP is assigned to MMC_2, it is now activated by the PLC.
2. MMC_1 cannot change to passive mode (processes which cannot be interrupted are in progress on MMC_1):
   MMC_1 sets MMC1_CHANGE_DENIED = TRUE, operating status cannot be changed.
   The PLC acknowledges with MMC1_CHANGE_DENIED = FALSE and grants MMC_1 permission to remain in active mode with MMC1_ACTIVE_PERM = TRUE. By sending MMC2_CHANGE_DENIED = TRUE, it notifies MMC_2 that MMC_1 cannot switch over to passive mode.
   MMC_2 then acknowledges with MMC2_CHANGE_DENIED = FALSE and remains in passive operating mode.

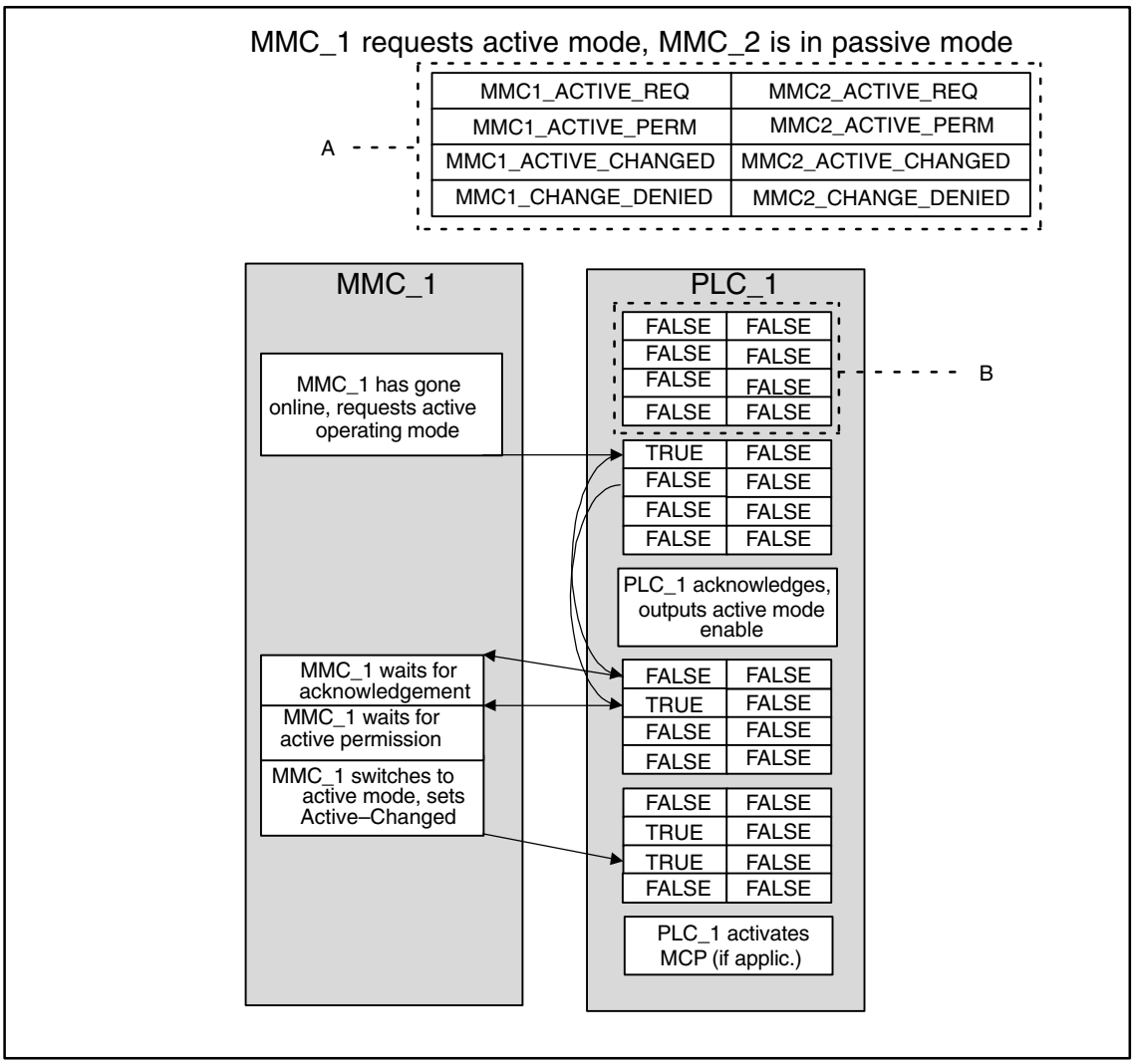


Fig. 6-1 MMC_1 requests active mode, MMC_2 is in passive mode

Please note: The arrangement of the signals of a block in box PLC_x (marked as B) corresponds to the arrangement of signal names in the header section (marked as A). Blocks B repeat in box PLC_x from top to bottom as a function of time.

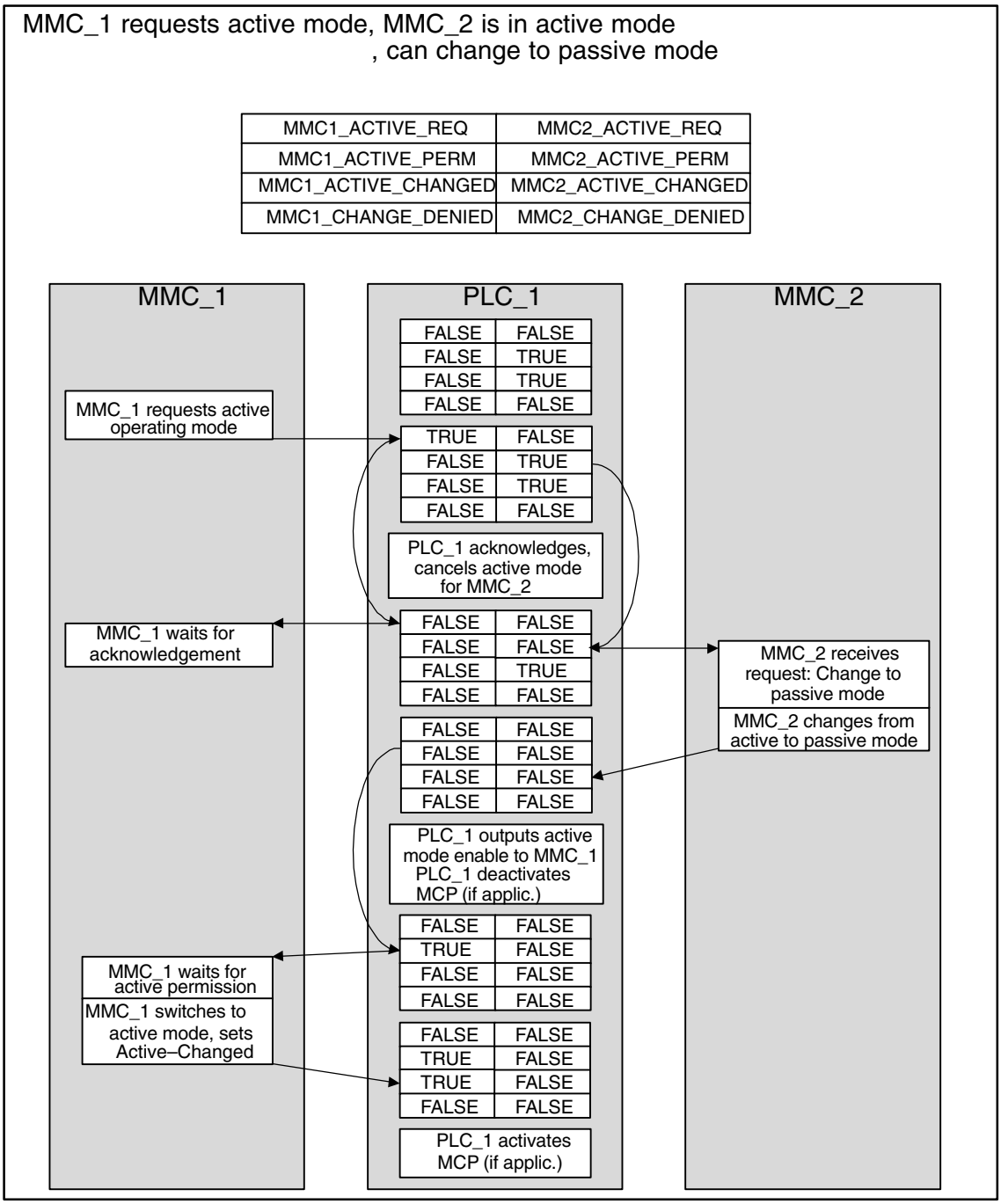


Fig. 6-2 MMC_1 requests active mode, MMC_2 is in active mode, can change to passive mode

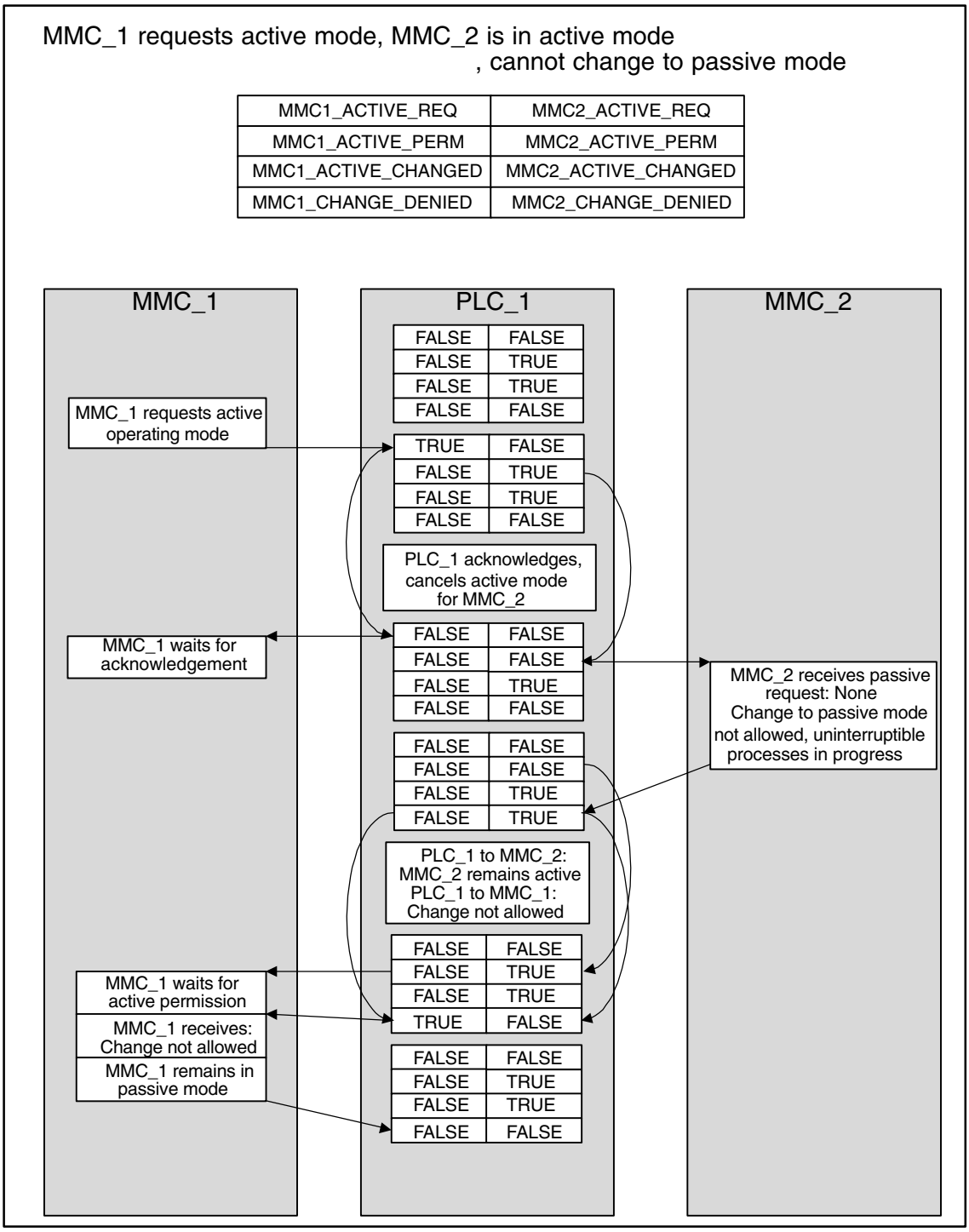


Fig. 6-3 MMC_1 requests active mode, MMC_2 is in active mode, but cannot change to passive mode

MCP_SWITCH-OVER

A control unit consists of an MMC and an MCP; these can both be switched over as a unit.

If an MCP has been configured for the MMC in configuring file NETNAMES.INI, it will be activated and deactivated with the MMC.

The MCP of whichever MMC is currently in active operating mode is activated.

In other words, only **one** MCP is ever active at any time on an NCU.

The MCP is activated by the PLC:

– MMC changes to active operating mode. (signal MMCx_ACTIVE_CHANGED: FALSE –> TRUE, x = 1,2 first or second MMC–PLC interface)

The MCP is deactivated by the PLC

– MMC changes to passive operating mode (signal MMCx_ACTIVE_CHANGED: TRUE –> FALSE, x = 1,2 first or second MMC–PLC interface)

– MMC goes offline as a result of switchover or displacement
The MMC sign-of-life signal changes from TRUE to FALSE when an MMC goes offline. After the edge change, the PLC deactivates the allocated MCP.

– Server MMC disconnects operating focus from the current NCU and switches it over to another. The server transmits S_ACT/ DIS_FOCUS as the last signal on its own MMC–PLC interface. The PLC then deactivates the corresponding MCP.

6.2.3 Defined logical functions/Defines

Note

Please refer to Section 5.1 for the legal values for bus type, functions/status and additional information plus permissible combinations of status and additional information. The logical identifiers of functions are used in the following diagrams.

6.2.4 Graphic representation of function sequences

Overview

Figs. 6-4 to 6-9 describe the switchover operation for an operator station and Figs. 6-10 to 6-12 the switchover operation for a server.

The diagrams describe how an operator station is switched over (switchover from NCU_1 to NCU_2).

If an MMC in the offline status wishes to go online to an NCU (e.g. during power-up), sequence OFFL_REQ_OP (...) and OFFL_CONF_OP(...) is omitted.

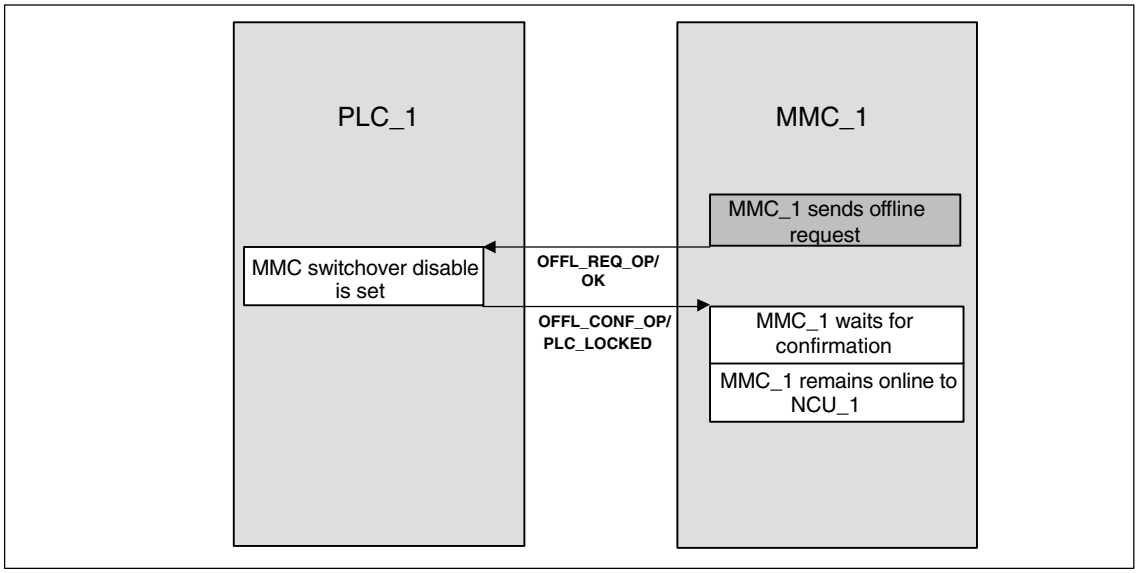


Fig. 6-4 MMC_1 is linked online to NCU_1 and wants to switch over to NCU_2, switchover disable is set in PLC_1

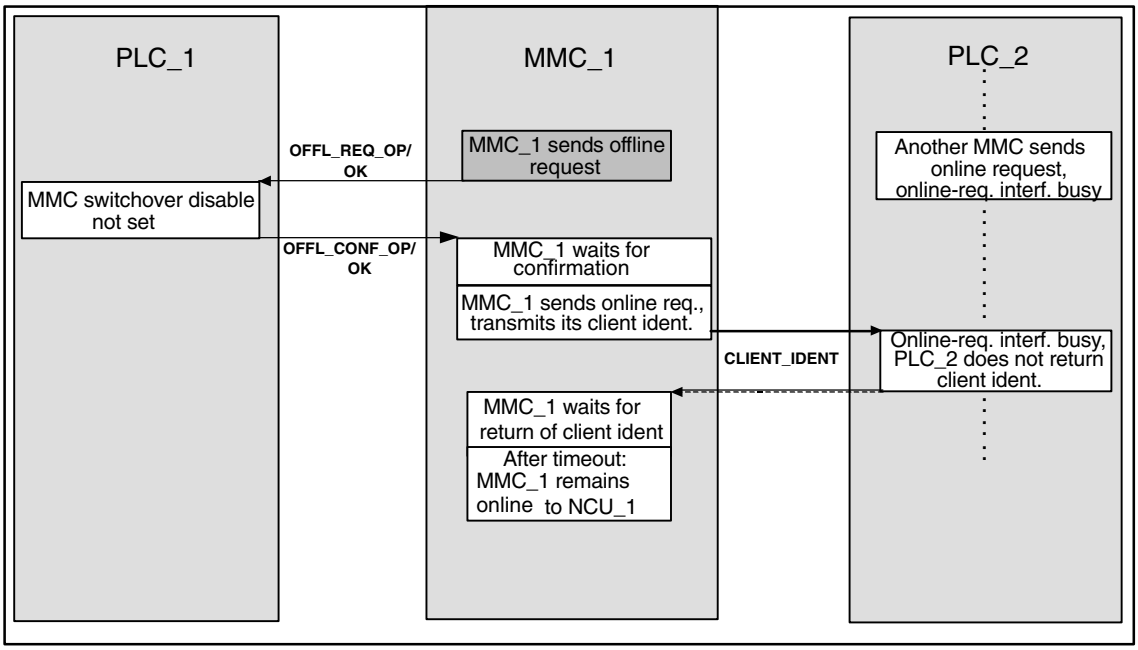


Fig. 6-5 MMC_1 online to NCU_1, MMC_1 wants to switch over to NCU_2, online-request interface in PLC_2 occupied by another MMC

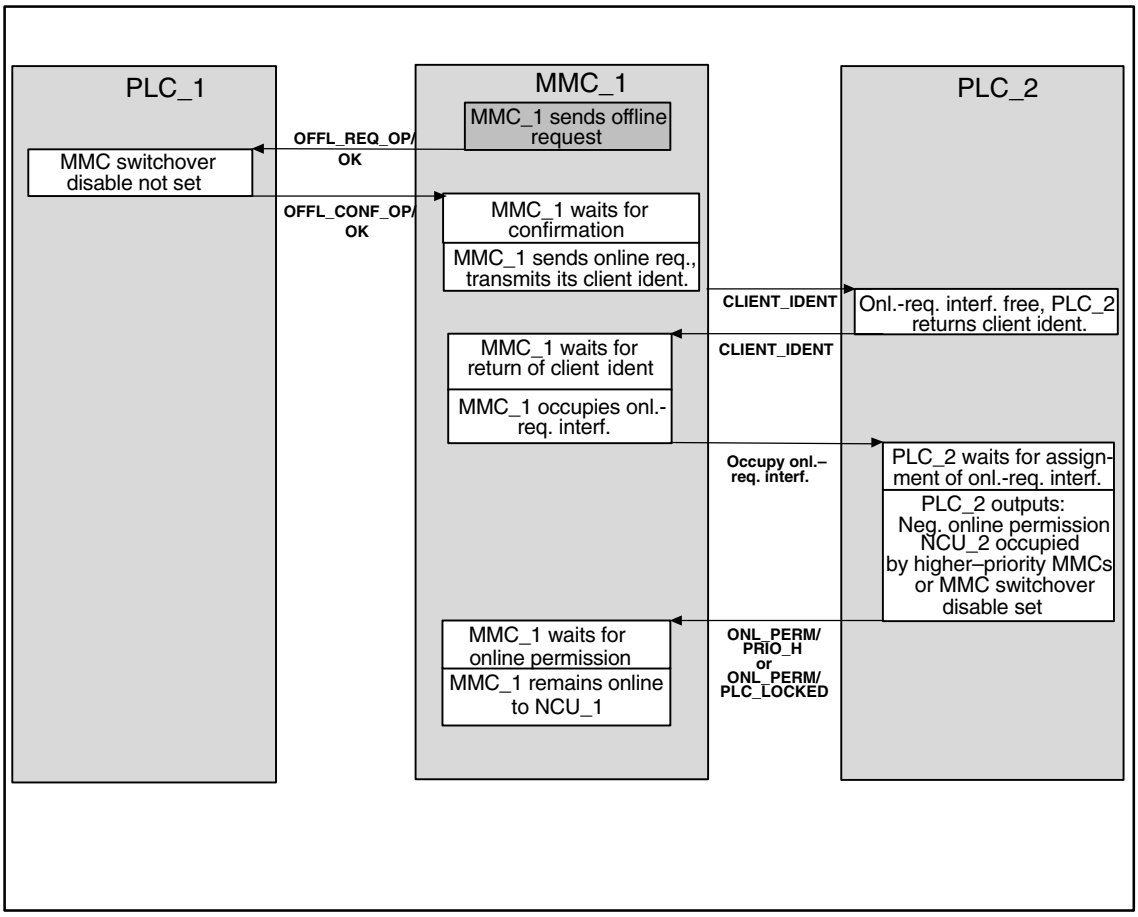


Fig. 6-6 MMC_1 online to NCU_1, MMC_1 wants to switch over to NCU_2, but does not receive permission from PLC_2

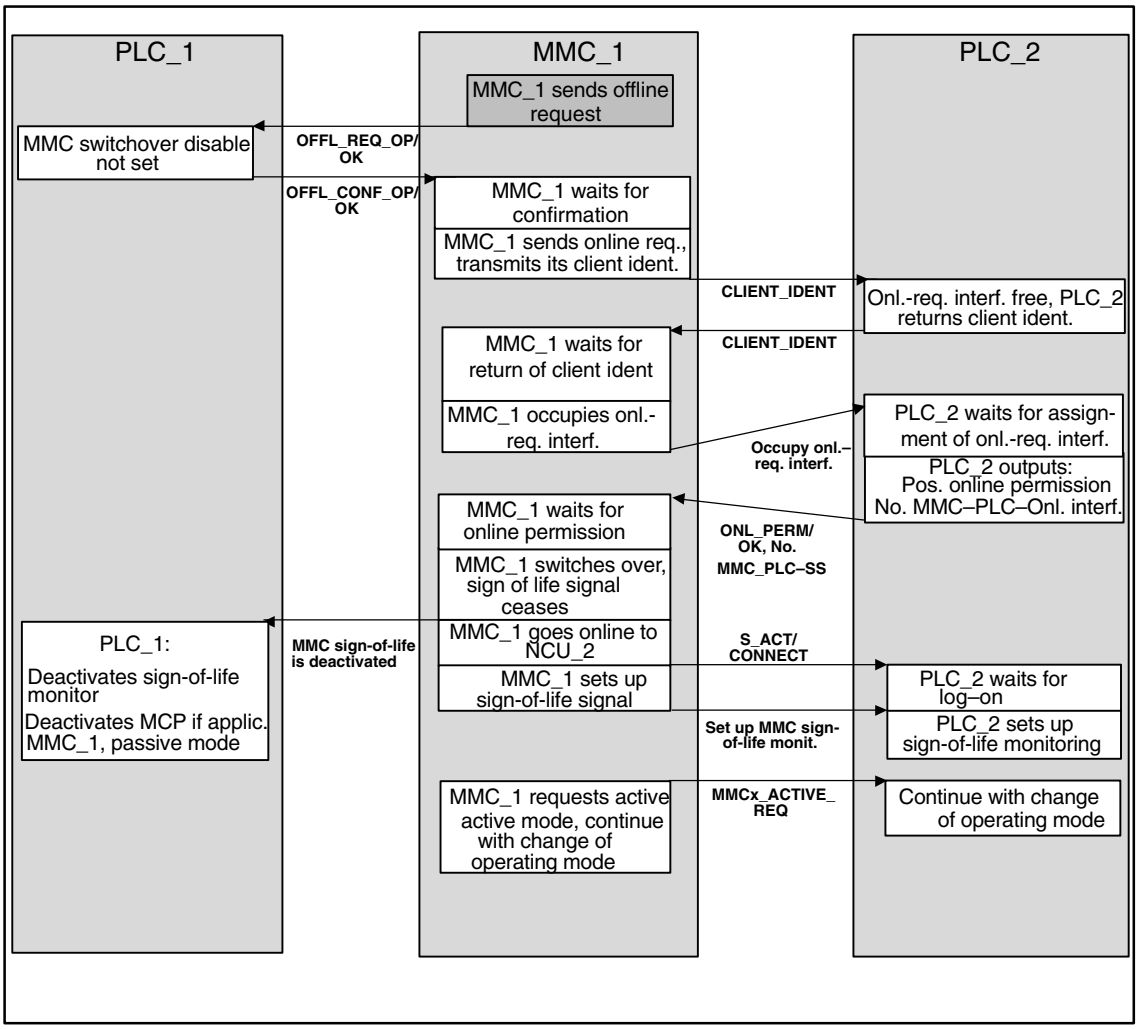


Fig. 6-7 MMC_1 online to NCU_1, MMC_1 switches over to NCU_2 (no displacement)

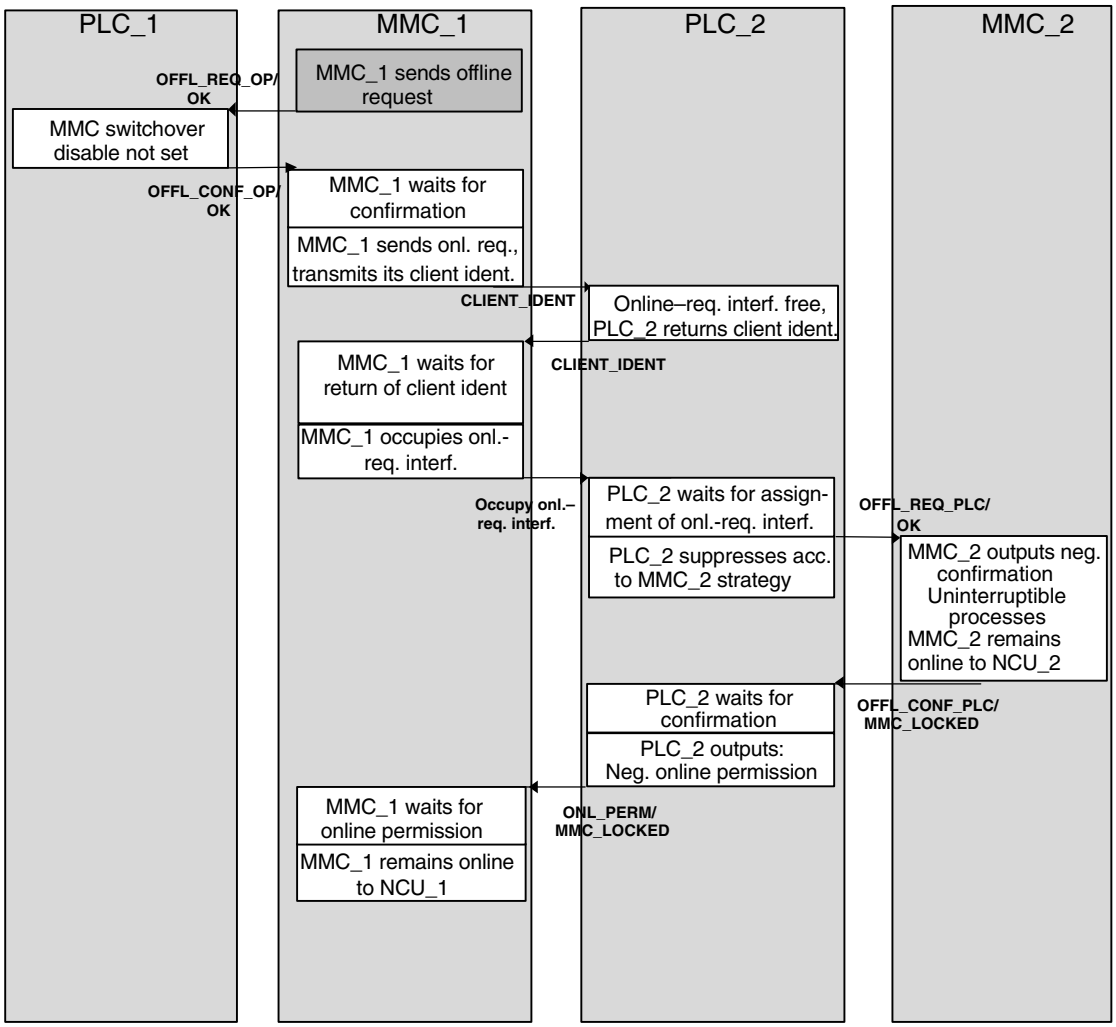


Fig. 6-8 MMC_1 online to NCU_1, MMC_2 online to NCU_2, MMC_1 wants to switch over to NCU_2, but MMCs executing uninterruptible processes are online to NCU_2

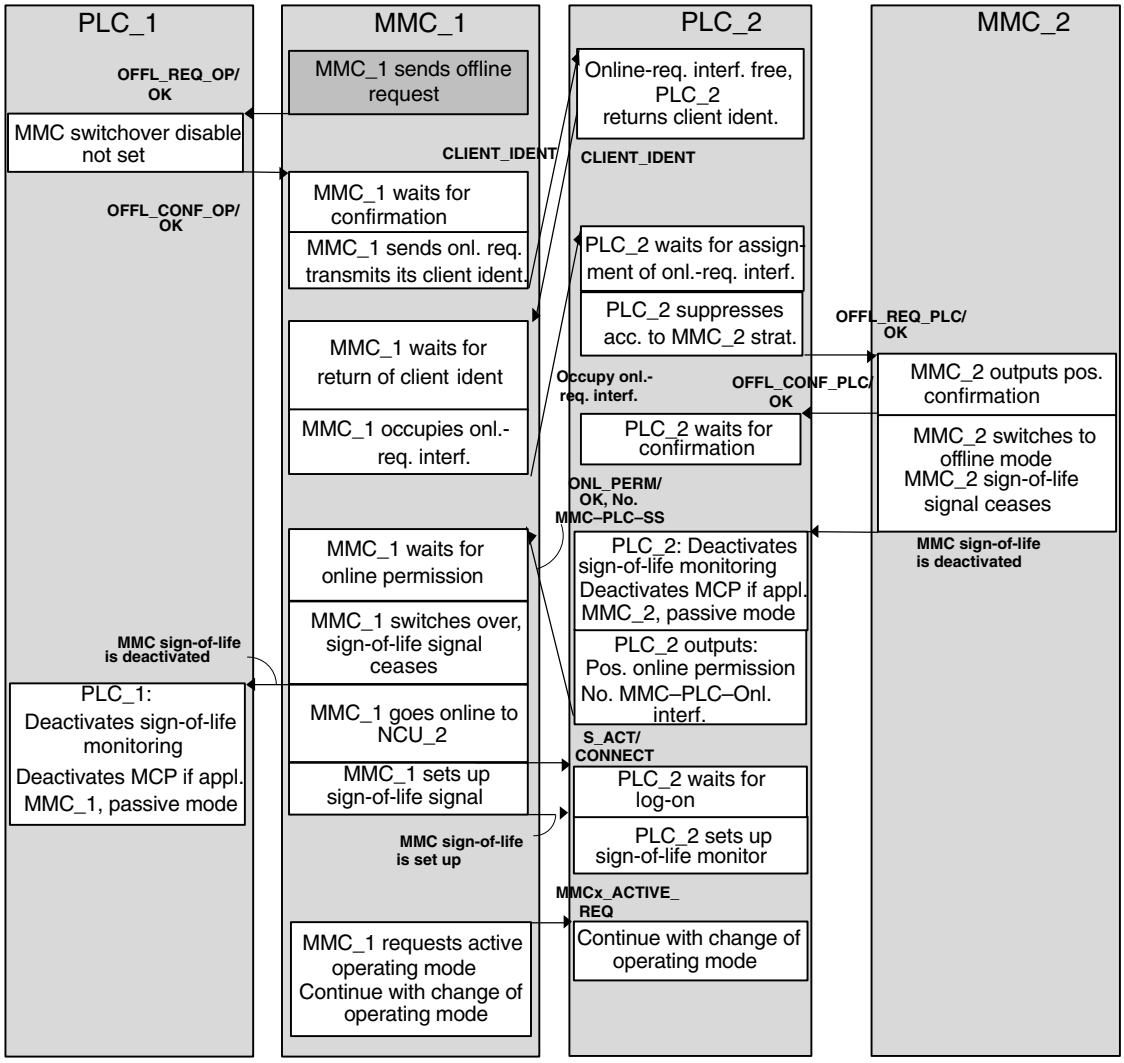


Fig. 6-9 MMC_1 online to NCU_1, MMC_2 online to NCU_2, MMC_1 switches from NCU_1 to NCU_2, MMC_2 is displaced

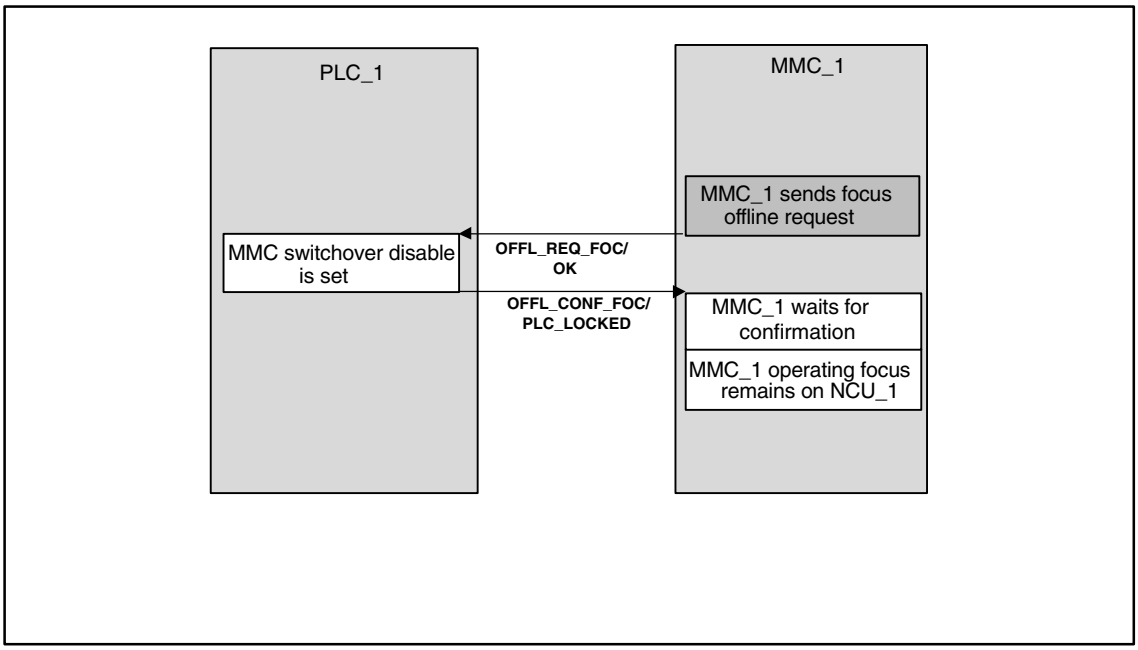


Fig. 6-10 MMC_1 server, wishes to switch operating focus from NCU_1 to NCU_2, switchover disabled in PLC_1

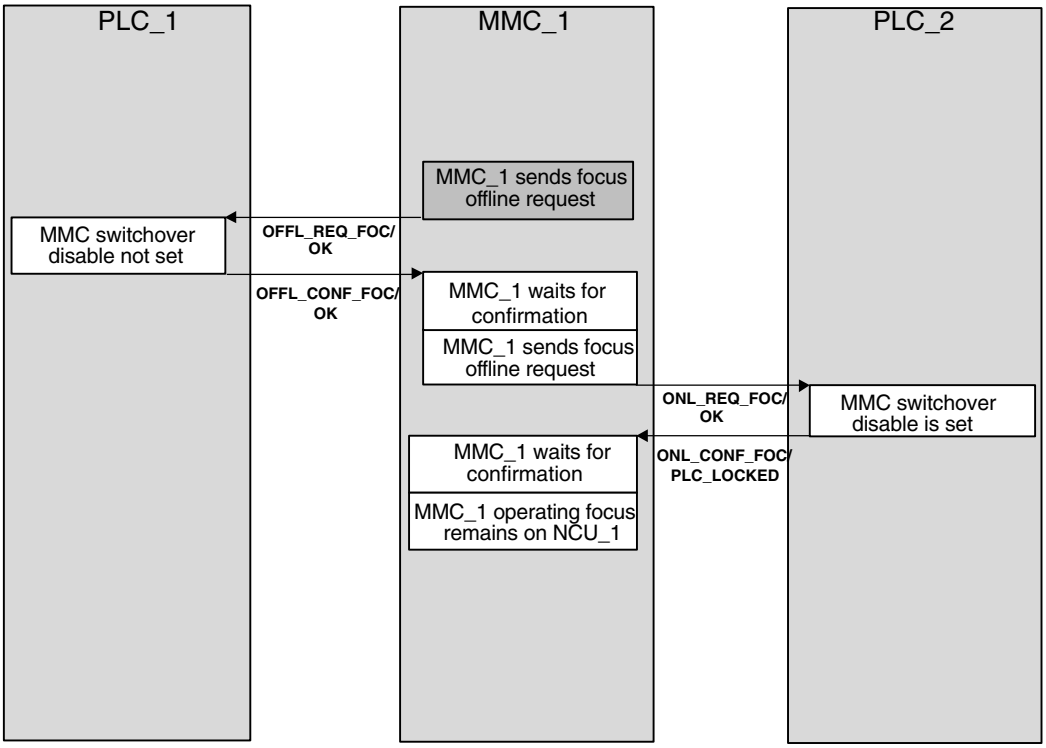


Fig. 6-11 MMC_1 is server, wishes to switch operating focus from NCU_1 over to NCU_2, switchover is disabled in PLC_2

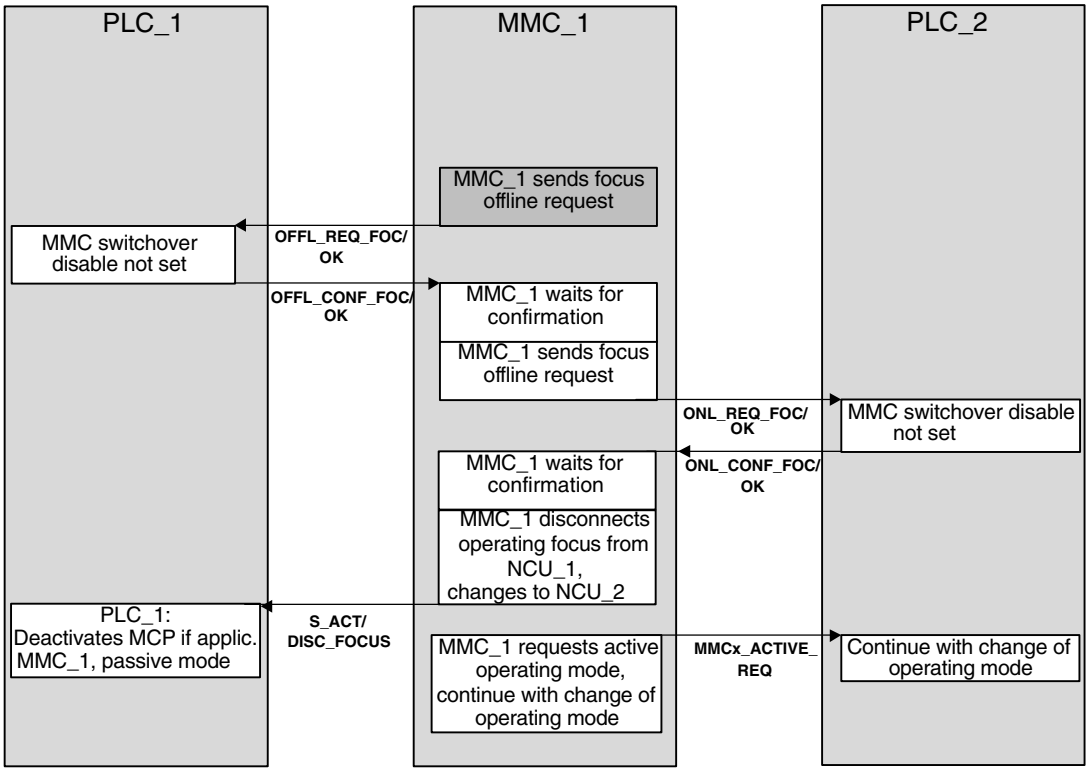


Fig. 6-12 MMC_1 is server, wishes to switch operating focus from NCU_1 over to NCU_2, switchover not disabled in PLCs, MMC_1 can change operating focus

6.3 Configuration file NETNAMES.INI, standard functionality

6.3.1 Two operator panels and one NCU

For a system according to SW 3.1 (consisting of two control units and one NCU on the OPI, see Chapter 1), a sample configuration file for the second control unit is detailed below.
For explanations, see Chapter 1 “Configurability”.

```
; NETNAMES.INI example 2 start

;Identification division
[own]
owner =          MMC_2

;Connection division
[conn MMC_1]
conn_1           NCU_1

[conn MMC_2]
conn_1           NCU_1

;Description division
[param network]
bus =            opi

[param MMC_1]
mmc_address =    1

[param MMC_2]
mmc_address =    3

[param NCU_1]
nck_address =    13
plc_address =    13

; NETNAMES.INI example 2 end
```

6.3.2 One operator panel and three NCUs

For a system according to SW 3.2 (consisting of one control unit and three NCUs on the OPI, see Chapter 1, paragraph “Configurability”), a sample configuration file is detailed below.
For explanations, see Chapter 1, paragraph “Configurability”.

Any adaptations which may need to be made are described in Chapter 2, paragraph “Configurations”.

```
; NETNAMES.INI, example 3 start
; Identification division:
[own]
owner=          MMC_1

;Connection division: For a maximum of 3 connections intended
[conn MMC_1]
conn_1=         NCU_1
conn_2=         NCU_2
conn_3=         NCU_3

;Description division: The network is defined unambiguously
[param network]
bus=            opi

[param MMC_1]
name=           arbitrary_name
type=           MMC_100
mmc_address=    1

[param NCU_1]
name=           arbitrary_name1
type=           ncu_572
nck_address=    12
plc_address =   12

[param NCU_2]
name=           arbitrary_name2
type=           ncu_573
nck_address=    14
plc_address=    14

[param NCU_3]
name=           arbitrary_name3
type=           ncu_573
nck_address=    15
plc_address=    15

; NETNAMES.INI, example 3 end
```

6.4 Link axis

Assumption

NCU1 and NCU2 have one link axis each,
machine data e.g.:

```
; Machine data of NCU1:
$MN_NCU_LINKNO = 1                      ; Set NCU number to 1
                                        ; (master NCU)
$MN_MM_NCU_LINK_MASK = 1                ; Activate link function
$MN_SERVO_FIFO_SIZE = 3                 ; Size of data buffer
                                        ; between interpolation
                                        ; and servo loop
$MN_MM_LINK_NUM_OF_MODULES = 2          ; Number of link modules

$MN_AXCONF_LOGIC_MACHAX_TAB[0] = “AX1”
$MN_AXCONF_LOGIC_MACHAX_TAB[1] = “AX2”
$MN_AXCONF_LOGIC_MACHAX_TAB[2] = “NC2_AX3” ; Link axis

; Unique NCU axis names
$MN_AXCONF_MACHAX_NAME_TAB[0] = “NC1_A1”
$MN_AXCONF_MACHAX_NAME_TAB[1] = “NC1_A2”
$MN_AXCONF_MACHAX_NAME_TAB[2] = “NC1_A3”

CHANDATA(1)
$MC_AXCONF_MACHAX_USED[0] = 1
$MC_AXCONF_MACHAX_USED[1] = 2
$MC_AXCONF_MACHAX_USED[2] = 3

...

; Machine data of NCU2:
$MN_NCU_LINKNO = 2              ; Set NCU number to 2 (slave NCU)
$MN_MM_NCU_LINK_MASK = 1
$MN_SERVO_FIFO_SIZE = 3
$MN_MM_LINK_NUM_OF_MODULES = 2

$MN_AXCONF_LOGIC_MACHAX_TAB[0] = “AX1”
$MN_AXCONF_LOGIC_MACHAX_TAB[1] = “AX2”
$MN_AXCONF_LOGIC_MACHAX_TAB[2] = “NC1_AX3” ; Link axis

; Unique NCU axis names
$MN_AXCONF_MACHAX_NAME_TAB[0] = “NC2_A1”
$MN_AXCONF_MACHAX_NAME_TAB[1] = “NC2_A2”
$MN_AXCONF_MACHAX_NAME_TAB[2] = “NC2_A3”

CHANDATA(1)
$MC_AXCONF_MACHAX_USED[0] = 1
$MC_AXCONF_MACHAX_USED[1] = 2
$MC_AXCONF_MACHAX_USED[2] = 3
```

6.5 Axis container coordination

The characteristic as a function of time is displayed from top to bottom in the following tables. The data are valid on condition that only two channels have axes in the container.

6.5.1 Axis container rotation without a part program wait

Channel 1	Channel 2	Comment
AXCTWE(C1)	Part program ...	Channel 1 enables the axis container for rotation
Part program **without** movement of a container axis	Part program ...	
	AXCTSWE(C1)	Channel 2 enables the axis container for rotation, container rotates because both channels have enabled rotation
Part program **with** movement of a container axis	Part program **with** movement of a container axis	**without wait**

6.5.2 Axis container rotation with an implicit part program wait

Channel 1	Channel 2	Comment
AXCTWE(C1)	Part program ...	Channel 1 enables the axis container for rotation
Part program **with** movement of a container axis	Part program ...	Channel 1 **waits implicitly** for axis container rotation
	AXCTSWE(C1)	Channel 2 enables the axis container for rotation, container rotates. Channel 1 continues.

6.5.3 Axis container rotation by one channel only (e.g. during power-up)

Channel 1	Channel 2	Comment
AXCTWE**D**(C1)	In the RESET state	Instantaneous rotation

6.6 Evaluating axis container system variables

6.6.1 Conditional branch

Channel 1	Comment
AXCTWE(CT1)	Channel 1 enables the axis container for rotation
MARKER1: Part program **without** movement of a container axis	
IF $AC_AXCTSWA[CT1] == 1 GOTOB MARKE1	**Conditional branch** dependent on completion of axis container rotation.
Part program **with** movement of a container axis	

6.6.2 Static synchronized action with $AN_AXCTSWA

Channel 1	Comment
IDS =1 EVERY $AN_AXCTSWA[CT1] == 1 DO M99	Static synchronized action: Always output auxiliary function M99 at the beginning of an axis container rotation.
	References: /FPSY/, FB Synchronized Actions

6.7 Configuration of a multi-spindle turning machine

Introduction

The following example describes the use of:

- Several NCUs in the NCU link group
- Flexible configuration with axis containers

Machine description

- Distributed on the circumference of a drum A (front-plane machining) the machine has:
  - 4 main spindles HS1 to HS4
    Each main spindle has the possibility of material feed (bars, hydraulic bar feed, axes: STN1–STN4).
  - 4 cross slides
  - Each slide has two axes.
  - Optionally a powered tool S1–S4 can operate on each slide.
- Distributed on the circumference of a drum B (rear-plane machining) the machine has:
  - 4 counterspindles GS1 to GS4
  - 4 cross slides
  - Each slide has two axes.
  - Optionally a powered tool S5–S8 can operate on each slide.
  - The position of each counterspindle can be offset through a linear axis for example for transferring parts from the main spindle for rear-plane machining in drum B. (Transfer axes. Axes ZG1–ZG4).
- Links:
  - If drum A rotates, **all** main spindles of this drum are subordinate to another group of slides.
  - If drum B rotates, **all** main counterspindles and all transfer axes of this drum are subordinate to another group of slides.
  - Rotations of drums A and B are autonomous.
  - Rotations of drums A and B are limited to 270°.
    (range and torsion of supply lines)

Term: Position

Main spindle HS_i and counterspindle GS_i together with their slides characterize a position.

NCU assignment

The axes and spindles of a position (for this example) are each assigned to an NCU. One of the NCUs, the master NCU, controls the axes for the rotations of drums A and B additionally. There are 4 NCUs with a maximum of the following axes:

Axis number

Per NCU_i the following axes/spindles must be configured:

Slides	1: X_i1, Z_i1
	2: X_i2, Z_i2
Spindles:	HS_i, GS_i, powered tools: S1, S2
Transfer axis:	ZG_i
Bar feed:	STN_i.

For the master NCU, in addition to the above-mentioned axes there are the two axes for rotating drums A and B. The list shows that it would not be possible to configure the axis number for a total of 4 positions via an NCU. (Limit 31 axes, required are 4 + 10 + 2 axes).

Axis container

With rotation of drums A/B, HS_i, GS_i, ZG_i and STN_i must be assigned to another NCU and must therefore be configured as link axes in axis containers.

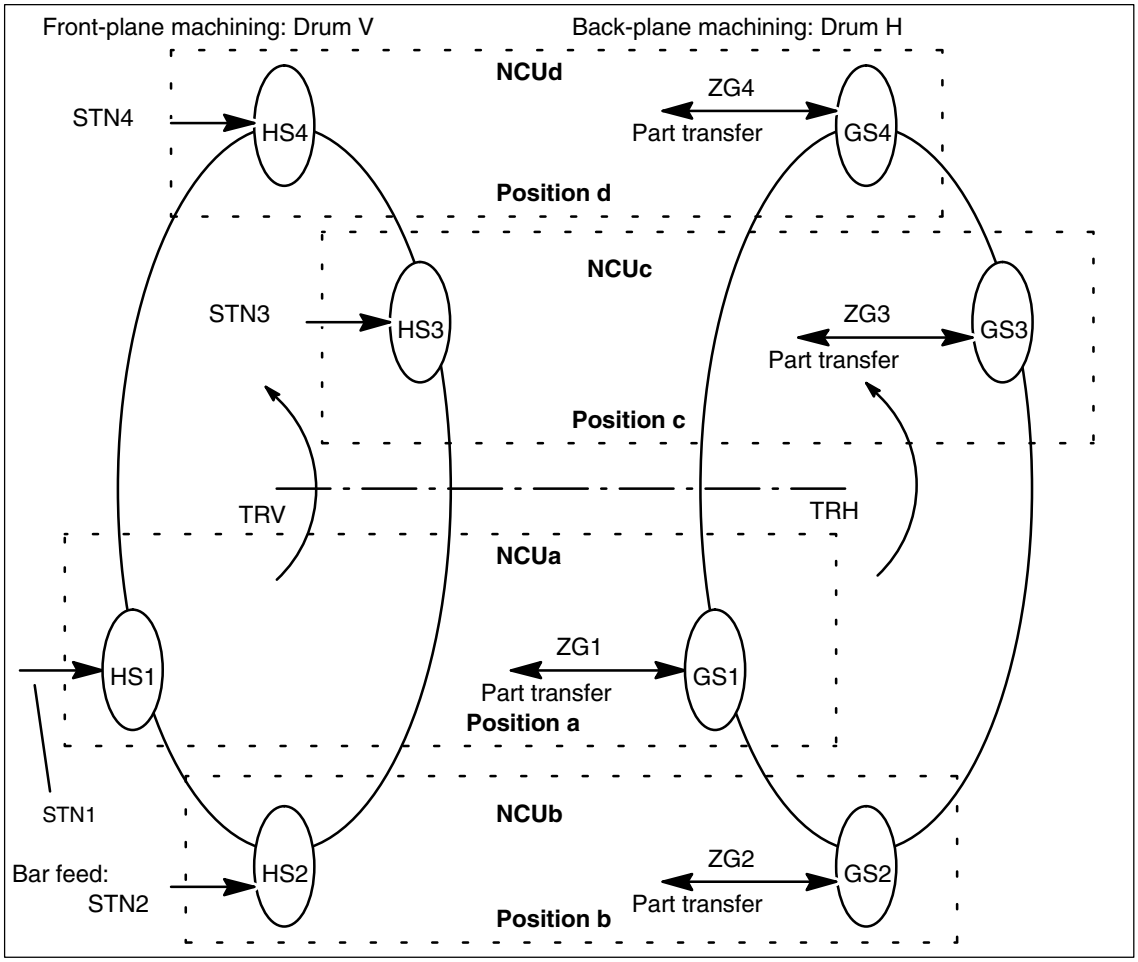


Fig. 6-13 Main spindles HS_i, countersp. GS_i, bar infeed axis STN_i and transfer axes ZG_i diagrammatic

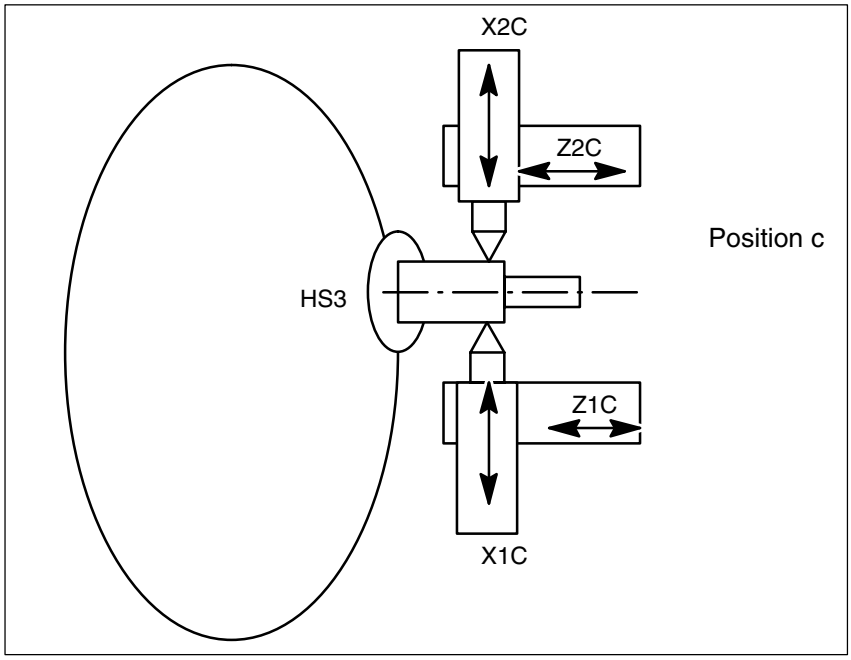


Fig. 6-14 Two slides per position can also operate together on one spindle.

Note

For clarifying the assignment of axes to slides and positions, the axes are named as follows:
Xij with i slide (1, 2), j position (A–D)
Zij with i slide (1, 2), j position (A–D)
Positions and their slides remain in a fixed position, whereas main spindles, counterspindles, bar feed axes STN and transfer axes ZG move to new positions by rotation of drums V or H.

For example, the axes to be managed per NC when the slide is taken into account are as follows for the configurations shown in the foregoing illustrations:

Axes of the master NCU

Table 6-3 Axes of master NCU: NCUa

Common axes	local axes	Remark
	TRV (drum V)	Master NCU only
	TRH (drum H)	Master NCU only
	X1A	Slide 1
	Z1A	Slide 1
	X2A	Slide 2
	Z2A	Slide 2
	S1	Slide 1
	S2	Slide 2
HS1		Axis container necessary
GS1		Axis container necessary
ZG1		Axis container necessary
STN1		Axis container necessary
4	8	

Axes of NCUb to NCUd

The NCUs that are not master NCUs have the same axes with the exception of the axes for the drive for drums TRV and TRH. The letter characterizing the position must be replaced accordingly for NCU and axis name (a, A → b, B to d, D).

Configuration rules

The following rules were applied for the configuration described below:

- Main spindle, counterspindles and axes that are assigned to different NCUs through drum rotation while they are operating as illustrated in the above Fig. “Main spindle ...” must be configured in an axis container. (HS_i, GS_i, ZG_i, STN_i).
- All main spindles for drum A are in the same container (No. 1).
- All bar feed axes for drum A are in the same container (No. 2).
- All counterspindles for drum B are in the same container (No. 3).
- All transfer axes for drum B are in the same container (No. 4).
- Main spindles HS_i and their counterspindle GS_i as well as the transfer axes for counterspindle ZG_i and the bar feed axes STN_i of the main spindle are assigned as follows for uniform load distribution purposes:
  NCUa HS1 – STN1,
  NCUb HS2 – STN2, ... etc.
- Slide axes Xij, Zij are solely local axes with a fixed NCU assignment.

- Slides are all assigned to a separate channel on an NCU.
  Thus slides can be moved autonomously.

Configuration possibilities

- Main or counterspindles are flexibly assigned to the slide.
- In each position the main spindle and counterspindle spindle speed can be determined independently.

  Exceptions:

  During part change from front-plane machining in drum V to rear-plane machining in drum H, the main spindle and counterspindle must be brought to the same spindle speed (synchronous spindle coupling).

  If slide 2 is also active in front-plane machining to “support” slide 1, in this case the main spindle speed is also valid for slide 2. Accordingly, if slide 1 is active in rear-plane machining, the counter spindle speed is also applicable for slide 1.

Small changes in speed

Owing to the unavoidable time delays incurred in the processing of actual values, abrupt changes in speed should be avoided during cross-NCU machining operations. Compare axis data and signals.

Configuration for NCU1

Uniform use of channel axis names in the part programs:

S4	main spindle
S3	counterspindle
X1	infeed axis
Z1	longitudinal axis
S1	powered tool
Z3	transfer axis
TRV	drum V for main spindle
TRH	drum H for counterspindle
STN	hydraulic bar feed

Axes highlighted in **bold** characterize the current channel as home channel for the axis in conjunction with axis exchange.

Table 6-4 NCUa, position: a, channel: 1, slide: 1

Channel axis name	..._MA-CHAX_ USED	$MN_ AXCONF_LOGIC_MA-CHAX_TAB	Container, slot entry (string)	Machine axis name
S4	1	AX1: CT1_SL1	1 1 NC1_AX1	HS1
S3	2	AX2: CT3_SL1	3 1 NC1_AX2	GS1
X1	3	AX3:		X1A
Z1	4	AX4:		Z1A
Z3	5	AX5: CT4_SL1	4 1 NC1_AX5	ZG1
S1	*6*	*AX6:*		*WZ1A*
STN	7	AX7: CT2_SL1	2 1 NC1_AX7	STN1
TRV	11	AX11:		TRV
TRH	12	AX12:		TRH
x2 *				
z2 *				

Table 6-5 NCUa, position: a, channel: 2, slide: 2

Channel axis name	..._MA-CHAX_ USED	$MN_ AXCONF_LOGIC_MA-CHAX_TAB	Container, slot entry (string)	Machine axis name
S4	1	AX1: CT1_SL1	1 1 NC1_AX1	HS1
S3	2	AX2: CT3_SL1	3 1 NC1_AX2	GS1
Z3	5	AX5: CT4_SL1	4 1 NC1_AX5	ZG1
STN	7	AX7: CT2_SL1	2 1 NC1_AX7	STN1
X2	8	AX8:		X2A
Z2	9	AX9:		Z2A
S1	*10*	*AX10:*		*WZ2A*
x1 *				
z1 *				

Note

- * due to program coordination via axis positions and 4-axis machining in one position
- Entries in the axis container locations should have the following format: “NC1_AX..” required with the meaning NC1 = NCU 1. In the above tables, NCUa is imaged on NC1_..., NCUb on NC2_... etc.

Further NCUs

The above listed configuration data must be specified accordingly for NCUb to NCUd. Please bear the following in mind:

- Axes TRA and TRB only exist for NCUa, channel 1.
- The container numbers are maintained for the other NCUs as they were specified for the individual axes
- The slot numbers are:
  NCUb → 2
  NCUc → 3
  NCUd → 4.
- The machine axis names are:
  NCUb → HS2, GS2, ZG2, STN2
  NCUc → HS3, GS3, ZG3, STN3
  NCUd → HS4, GS4, ZG4, STN4.

Axis containers

The information relating to containers given in Table 6-4 and the container entries of the similarly configured NCUs, NCUb to NCUd, are specified in the following tables, sorted according to containers and slots, as they have to be set in machine data

MD 12701: $MN_AXCT_AXCONF_ASSIGN_TAB1[slot]
. . .
MD 12716: $MN_AXCT_AXCONF_ASSIGN_TAB16[slot]

with slots: 1 – 4 for the 4 positions of a multi-spindle turning machine.

Note

For the MD entry $MN_AXC_AXCONF_ASSIGN_TAB$_i$[slot], the values (without decimal point and machine axis name) that are entered under initial position in the above tables must be set.

Table 6-6 Axis container and their position-dependent contents for drum A

Container	Slot	Initial position (TRA o°)	Switch 1 (TRA 90°)	Switch 2 (TRA 180 °)	Switch 3 (TRA 270 °)	Switch 4 = (TRA 0 °)
1	1	NC1_AX1, HS1	NC2_AX1, HS2	NC3_AX1, HS3	NC4_AX1, HS4	NC1_AX1, HS1
	2	NC2_AX1, HS2	NC3_AX1, HS3	N4C_AX1, HS4	NC1_AX1, HS1	NC2_AX1, HS2
	3	NC3_AX1, HS3	NC4_AX1, HS4	NC1_AX1, HS1	NC2_AX1, HS2	NC3_AX1, HS3
	4	NC4_AX1, HS4	NC1_AX1, HS1	NC2_AX1, HS2	NC3_AX1, HS3	NC4_AX1, HS4
2	1	NC1_AX7, STN1	NC2_AX7, STN2	NC3_AX7, STN3	NC4_AX7 STN4	NC1_AX7, STN1
	2	NC2_AX7, STN2	NC3_AX7, STN3	NC4_AX7, STN4	NC1_AX7, STN1	NC2_AX7, STN2
	3	NC3_AX7, STN3	NC4_AX7, STN4	NC1_AX7, STN1	NC2_AX7, STN2	NC3_AX7, STN3
	4	NC4_AX7, STN4	NC1_AX7, STN1	NC2_AX7, STN2	NC3_AX7, STN3	NC4_AX7, STN4

Drum movement	0 °	+ 90 °	+ 90 °	+ 90 °	– 270°

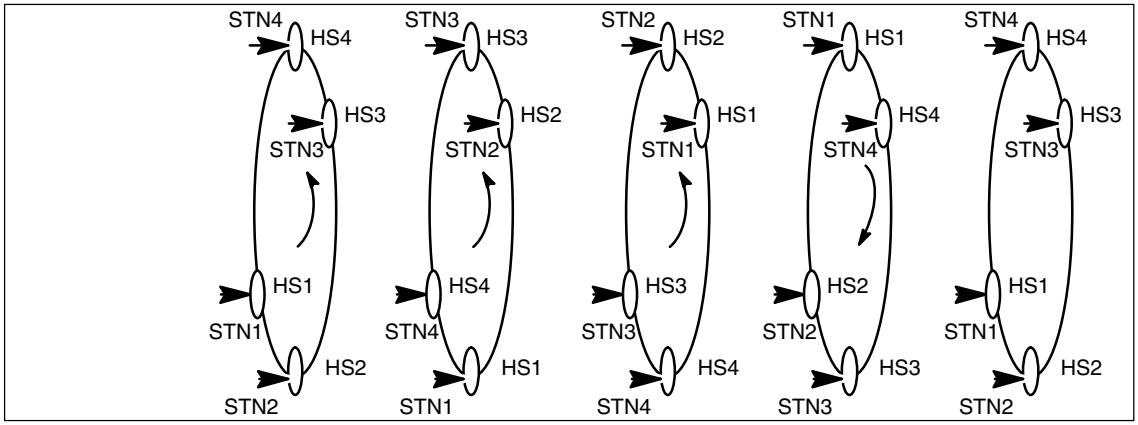


Fig. 6-15 Positions of drum A

Table 6-7 Axis container and their position-dependent contents for drum B

Container	Slot	Initial position (TRB o°)	Switch 1 (TRB 90°)	Switch 2 (TRB 180 °)	Switch 3 (TRB 270 °)	Switch 4 = (TRB 0 °)
3	1	NC1_AX2, GS1	NC2_AX2, GS2	NC3_AX2, GS3	NC4_AX2, GS4	NC1_AX2, GS1
	2	NC2_AX2, GS2	NC3_AX2, GS3	NC4_AX2, GS4	NC1_AX2, GS1	NC2_AX2, GS2
	3	NC3_AX2, GS3	NC4_AX2, GS4	NC1_AX2, GS1	NC2_AX2, GS2	NC3_AX2, GS3
	4	NC4_AX2, GS4	NC1_AX2, GS1	NC2_AX2, GS2	NC3_AX2, GS3	NC4_AX2, GS4
4	1	NC1_AX5, ZG1	NC2_AX5, ZG2	NC3_AX5, ZG3	NC4_AX5 ZG4	NC1_AX5, ZG1
	2	NC2_AX5, ZG2	NC3_AX5, ZG3	NC4_AX5, ZG4	NC1_AX5, ZG1	NC2_AX5, ZG2
	3	NC3_AX5, ZG3	NC4_AX5, ZG4	NC1_AX5, ZG1	NC2_AX5, ZG2	NC3_AX5, ZG3
	4	NC4_AX5, ZG4	NC1_AX5, ZG1	NC2_AX5, ZG2	NC3_AX5, ZG3	NC4_AX5, ZG4

■

Notes

Data Fields, Lists

7

7.1 Interface signals

DB number	Bit, byte	Name		Reference
General		**Signals from NC to PLC**		
10	104.0	MCP1 ready		
10	104.1	MCP2 ready		
10	104.2	HHU ready		
10	107.6	NCU link active		
10	108.1	MMC2–CPU ready (MMC to OPI or MPI)		
10	108.2	MMC-CPU1-Ready (MMC to MPI)		A2
10	108.3	MMC-CPU1-Ready (MMC to OPI, standard connection)		A2

DB number	Bit, byte	Name		Reference
General		**Connection request indication interface**		
19	DBW100	ONL_REQUEST	Online request from MMC	
19	DBW102	ONL_CONFIRM	Acknowledgement to MMC	
19	DBW104	PAR_CLIENT_IDENT	MMC bus address, bus type	
19	DBB106	PAR_MMC_TYP	Main / secondary control panel / alarm server	
19	DBB107	PAR_MSTT_ADR	Address of MCP to be activated	
19	DBB108	PAR_STATUS	Connection status	
19	DBB109	PAR_Z_INFO	Additional information connection status / No. of the MMC-PLC interface	
19	DBW110	M_TO_N_ALIVE	Ring counter, M:N switchover act.	

DB number	Bit, byte	Name		Reference
General		**Online interface**		
19	DBW120	MMC1_CLIENT_IDENT	MMC bus address, bus type	
19	DBB122	MMC1_TYP	Main / secondary control panel / alarm server	

DB number	Bit, byte	Name		Reference
19	DBB123	MMC1_MSTT_ADR	Address of MCP to be (de)activated	
19	DBB124	MMC1_STATUS	Connection status	
19	DBB125	MMC1_Z_INFO	Additional information connection status	
19	DBX126.0	MMC1_SHIFT_LOCK	MMC switchover disable	
19	DBX126.1	MMC1_MSTT_SHIFT_LOCK	MCP switchover disable	
19	DBX126.2	MMC1_ACTIVE_REQ	MMC requests active operating mode	
19	DBX126.3	MMC1_ACTIVE_PERM	Enable from PLC to change the operating mode	
19	DBX126.4	MMC1_ACTIVE_CHANGED	MMC has changed operating mode	
19	DBX126.5	MMC1_CHANGE_DENIED	MMC active/passive switchover rejected	
19	DBW130	MMC2_CLIENT_IDENT	MMC bus address, bus type	
19	DBB132	MMC2_TYP	Main / secondary control panel / alarm server	
19	DBB133	MMC2_MSTT_ADR	Address of MCP to be (de)activated	
19	DBB134	MMC2_STATUS	Connection status	
19	DBB135	MMC2_Z_INFO	Additional information connection status	
19	DBX136.0	MMC2_SHIFT_LOCK	MMC switchover disable	
19	DBX136.1	MMC2_MSTT_SHIFT_LOCK	MCP switchover disable	
19	DBX136.2	MMC2_ACTIVE_REQ	MMC requests active operating mode	
19	DBX136.3	MMC2_ACTIVE_PERM	Enable from PLC to change the operating mode	
19	DBX136.4	MMC2_ACTIVE_CHANGED	MMC has changed operating mode	
19	DBX136.5	MMC2_CHANGE_DENIED	MMC active/passive switchover rejected	

DB number	Bit, byte	Name	Reference
General		**Signals from NC to PLC**	
31–61	60.1	NCU link axis active	
31–61	61.1	Axis container rotation active	
31–61	61.2	Axis ready	

7.2 Machine / setting data

Number	Identifier	Name	Reference
General ($MN_ ...)			
10002	AXCONF_LOGIC_MACHAX_TAB[n]	Logical NCU machine axis image	
10087	SERVO_FIFO_SIZE	Size of data buffer between interpolation and position controller tasks	
10134	MM_NUM_MMC_UNITS	Number of simultaneous MMC communication partners	
11398	AXIS_VAR_SERVER_SENSITIVE	Response of AXIS-VAR server to error condition	
12510	NCU_LINKNO	NCU number in an NCU group	
12520	LINK_TERMINATION	NCU numbers for which bus terminating resistors are active	
12530	LINK_NUM_OF_MODULES	Number of NCU link modules	
12540	LINK_BAUDRATE_SWITCH	Link bus baud rate	
12550	LINK_RETRY_CTR	Maximum number of message frame repeats in event of error	
12701 ... 12716	AXCT_AXCONF_ASSIGN_TAB1[s] ... AXCT_AXCONF_ASSIGN_TAB16[s]	List of axes in the axis container	
12750	AXCT_NAME_TAB[n]	List of axis container names	
18700	MM_SIZEOF_LINKVAR_DATA	Size of the NCU link variable memory	
18780	MM_NCU_LINK_MASK	Activation of NCU link communication	

Channel ($MC_ ...)			
20000	CHAN_NAME	Channel name	K1
20070	AXCONF_MACHAX_USED	Machine axis number valid in channel	K2
28160	MM_NUM_LINKVAR_ELEMENTS	Number of write elements for the NCU link variables	

Axis ($MA_ ...)			
30550	AXCONF_ASSIGN_MASTER_CHAN	Default assignment between an axis and a channel	K5
30554	AXCONF_ASSIGN_MASTER_NCU	Initial setting defining which NCU generates setpoints for the axis	
30560	IS_LOCAL_LINK_AXIS	Axis is a local link axis	

Setting data ($SN_ ...)			
41700	AXCT_SWWIDTH[container number]	Axis container rotation setting	
43300	ASSIGN_FEED_PER_REV_SOURCE	Revolutional feedrate for positioning axes/spindles	V1

7.3 Alarms

A more detailed description of the alarms which may occur is given in

References: /DA/, Diagnostic Guide

or in the online help in systems with MMC 102/103.

■

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

Operation via PC/PG (B4)

■

Notes

Brief Description 1

Applications

The “Operation via PG/PC” functionality

- must be utilized if no operator panel is installed.
- can be utilized as a handling support for OP030 panels

Hardware

The following HW requirements must be fulfilled:

- PG/PC with at least 486DX33 processor and 8 MB main memory
- ENHANCED mode (386 mode) operation of MS WINDOWS must be available
- PG/PC with MPI/OPI interface (provided with PG 720/720C/740/760). An MPI card (6ES7793-2AA00-0AA0) is available for PCs with free ISA slot.
- VGA monitor with a resolution of 640x480 or higher.

Implementation with SW 3.1 and higher

The option of creating a link between two operator panels and one NCU has been implemented in SW 3.1. The machine control panel MCP is permanently allocated to the NCU.

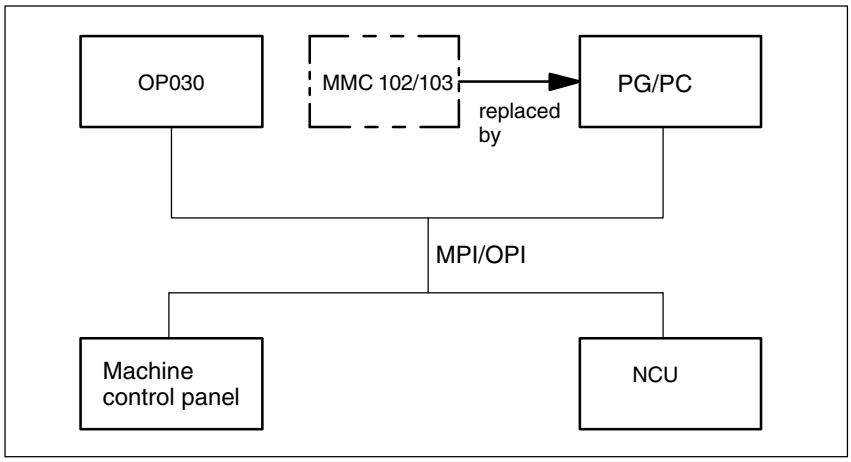


Fig. 1-1 Configuration with OP030 and PG/PC or MMC 102/103

All operator panels and the NCU are either connected to the OPI bus or all to the MPI bus. A **homogeneous** network must be provided with respect to these components.

Implementation with SW 3.2 and higher

From SW 3.2, an additional option of linking one operator panel and up to three NCUs is implemented. The machine control panel is permanently allocated to the relevant NCU.

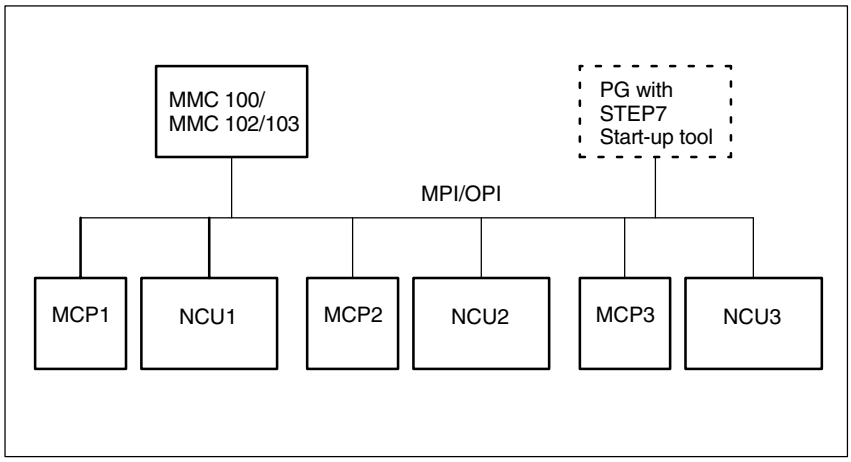


Fig. 1-2 Configuration in SW 3.2 m:n corresponds to 1 : 3

References: /FB/ 2, B3, “Several Operator Panels and NCUs”

Software installation

see Chapter 2

User interfaces

The user interfaces are described in the Operator’s Guides of the relevant operator panels.

References: /BA/, Operator’s Guide
/FBO/, OP 030 Operator’s Guide

Restrictions

If the “Operation via PG/PC” functionality is used in addition to an OP030 operator panel, the conditions relating to configuration and coordination described in /FB2/, B3, “Several operator panels and NCUs” must be observed.

References: /FB/ 2, B3, “Several Operator Panels and NCUs”

■

Detailed Description 2

2.1 Software installation

2.1.1 System requirements

Hardware requirements

The following hardware requirements must be fulfilled to allow operation via PG/PC

- IBM® AT-compatible PG/PC with 486DX33 microprocessor
- At least 8 MB of main memory
- Diskette drive (3 1/2 inch)
- Hard disk drive for data management
- Monochrome or color monitor
- Keyboard
- PG/PC with MPI interface (available for PG 720/720C/740/760)
  Limited operation without MPI card is possible (e.g. interactive programming)
  Note: RS232-MPI adapter is not supported.
- Mouse
- Connecting cable for link between PG/PC and NCU module

Note

All operator panels and the NCU are either
– connected to the OPI bus or
– connected to the MPI bus.
A **homogeneous** network must be provided with respect to these components.

Software requirements

Software configuration for operation via a PG/PC:

- MS-DOS® operating system, version 6.x or higher
- WINDOWS™ user interface, version 3.1 or higher
- MPI interface driver (contained in the supplied software)
- WINDOWS™ 32s, version 1.30.166.0 or higher
  (You will find the current version in “windows\system\win32s.ini”.)
  If WINDOWS™ 32s is not installed, it can be installed from 2 supplied diskettes (call setup.exe).

2.1.2 Installation

Storage area of MPI card

The storage area of the MPI card must be excluded from use by the memory manager (files: CONFIG.SYS, SYSTEM.INI).

Example of entry in SYSTEM.INI:
[386enh]
EmmExclude=..<card area>
(see HW description of card)

Scope of delivery

System software:

– Approx.10 diskettes with compressed MMC 102/103 software and installation tools

– 2 diskettes WINDOWS 32s subsystem (= Microsoft setup)

Please proceed as follows to install the software:

Call

1. **Start SETUP.EXE**
   Insert the first installation diskette and use the WINDOWS™ file manager to start the SETUP.EXE file.

   The installation program requests all further inputs or diskette changes required in the user dialog.

2. **Enter installation path**
   Select the directory plus the installation path (see screenshot) to which you wish to copy the Software.

   With “**Continue**”, you continue the installation, with “**Exit Setup**” you interrupt the installation procedure.
   This also applies to further operations.

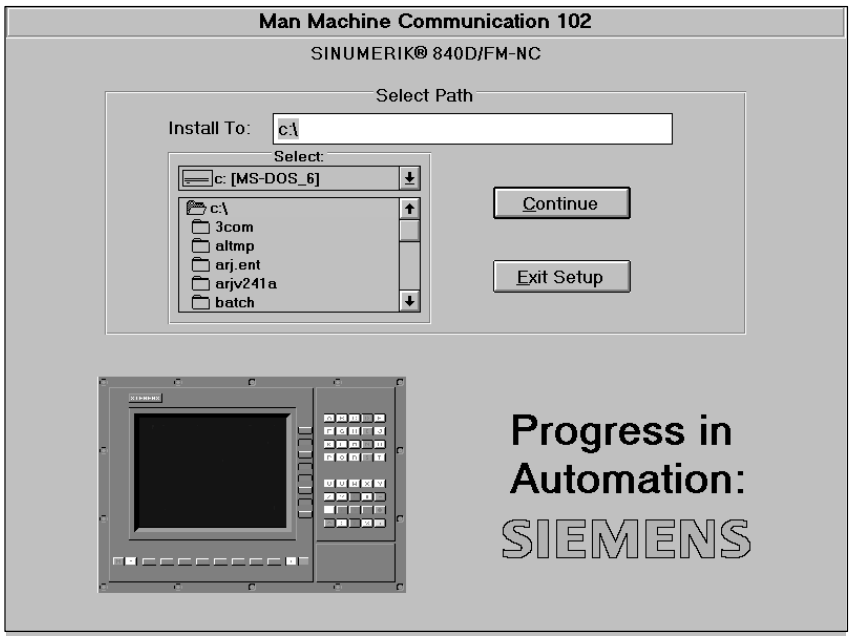


Fig. 2-1 Enter installation path

3. Select operation with MPI or without MPI

Fig. 2-2 Operation with/without MPI

4. Select turning or **milling**

Fig. 2-3 Select turning/milling

Note

If you want to change the selection mode later, select the directory “mmc2” and copy “dpturn.exe” (turning) or “dpmill.exe” (milling) into the directory “dp.exe”.

5. Select drive
only if several local disk drives are available
Select the drive for the tmp directory (see Fig.)

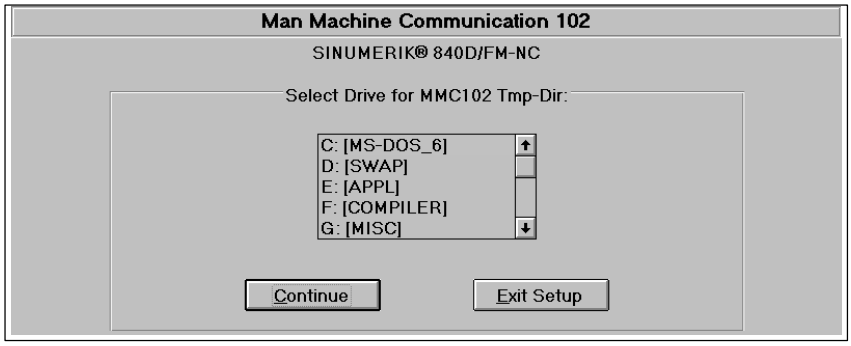


Fig. 2-4 Select drive

If this does not apply, select drive C:\.

Note

The contents of the directory “tmp” are deleted on the installation drive with each restart of MMC102.

Following the selection, a status display with the inputs made is shown.

Fig. 2-5 Status display of the installation mode

6. Press **Continue** to request the installation diskettes.

Note

Please observe the requests made on the screen.

The program group “SINUMERIK 840D MMC V3.2” is generated.
With successful installation, the following message is displayed:
“MMC 102 Installation is complete”

If you want to change the installation path, press **Go back**.

7. Make settings

7 a **OPI** interface (1.5 Mbaud), configuration: **1 MMC** to **1 NCU** (on delivery)

Additional settings are not required.

7 b **MPI** interface (187.5 Kbaud), configuration: **1 MMC** to **1 NCU** (on delivery)

1. Determination of the NCK/PLC bus address
   - if PLC < SW 3.2, then NC address = 13, PLC address = 2
   - if PLC $\geq$ SW 3.2 and module PLC 314, then NC address = 13, PLC address = 2
   - if PLC $\geq$ SW 3.2 and module PLC 315, then NC address = 3, PLC address = 2
2. Entering the addresses in files
   - File “S7CFGPGX.DAT”
     In the file “S7CFGPGX.DAT” on the MPI driver directory (<installation path>\MMC2\DRV.ID) the following entries must be adapted to the existing hardware configuration by means of an ASCII editor:

Interrupt setting
“hwint_vector”: Setting the interrupt for the MPI card. This interrupt may not be used by another card (e.g. network adapter).
Default setting: 10.

Settings for baud rate
“baud rate”, “tslot” and “tgap”: Settings for the baud rate. These 3 settings must always be activated/deactivated together by removing/inserting the leading “#” (comment).
When the baud rate is changed, the setting “ADDRESS1=\PLC, 10000d01” for 1.5 Mbaud or “ADDRESS1=\PLC, 10000201” for 187.5 Kbaud must also be adapted in file <installation path>\MMC2\MMC.ini, section [840D].
Default setting: 1.5 Mbaud.

– File “netnames.ini”
The following lines in the file must be changed:

#	bus = opi	must be replaced by	= **mpi**
#	nck_address = 13	must be replaced by	= **3** (if PLC $\geq$ SW3.2) = 13 (if PLC < SW3.2)
#	plc_address = 13	must be replaced by	= **2**

Parallel Step7/AS300 application

Installation in parallel with the Step7/AS300 SW can give rise to problems. It may be necessary to reconfigure the drives and restart the system.

2.1.3 Software supplementary conditions

- **Function keys**
The function keys may not be actuated in any of the displays until the display has fully built up.
- **Monochrome screen**
When a monochrome screen is used, the colors used by the MMC must be adapted accordingly. For this purpose, select the color scheme “Monochrome” or “Mono positive” in display “Start-up\MMC\Color setting”.
- **Easy parameterization**
The display “Start-up\MMC\OPI parameters” can now be called even if there is no link to the NC kernel. This means that the OPI parameters for baud rate and network address can be set easily.

2.1.4 Start program

Program call

The MMC 102/103 software is started on a PG/PC either

- from the program manager through selection of the “SINUMERIK 840D MMC V2.3” program group followed by a **double click** on the **“MMC Startup” symbol** or

Fig. 2-6 SINUMERIK 840D MMC program group

- from the file manager by a **double click** on file **REG_CMD.EXE**.

Communication

If no communication link can be established to the NCK or 611D, then the message “No communication to NCK” is displayed. If the data exchange is interrupted, e.g. by an NCK reset, then the MMC 102/103 software tries to re-establish the communication link itself.

2.1.5 End program

Deselect program

The following steps must be taken to deselect the MMC 102/103 software:

1. Press function key **F10**
   A horizontal softkey bar is displayed.
2. Press function key **Shift** + **F9**
3. End the program by selecting the softkey **Exit**.

2.2 Operation via PG/PC

2.2.1 General operation

Operating philosophy

The special function keys of the operator keyboard can be used with the full keyboard. Operator inputs can be made using the mouse or via the keyboard.

Key assignments

The following table shows the assignments between the function keys and the softkeys/special keys:

Note

The editor displays only the characters which can be input via the operator panel keyboard.

Table 2-1 Key assignments between operator keyboard and full keyboard

Full keyboard	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12
with SHIFT	**vertic** soft. **1**	**vertic** soft. **2**	**vertic** soft. **3**	**vertic** soft. **4**	**vertic** soft. **5**	**vertic** soft. **6**	**vertic** soft. **7**	**vertic** soft. **8**	>	M		
without SHIFT	**horiz** soft. **1**	**horiz** soft. **2**	**horiz** soft. **3**	**horiz** soft. **4**	**horiz** soft. **5**	**horiz** soft. **6**	**horiz** soft. **7**	**horiz** soft. **8**	^		W	Y
Full keyboard:	5	Esc	Insert	Home	Page Up	Page Down	Enter					
without SHIFT	:	X		?	‘	%						

Alarm or message line

3000 EMERGENCY STOP

Alarm or message line for displaying information for the operator.

i-R

Selection fields **i** and **R** which appear in every display have the following meaning:

- The **i** field is selected with the **Help key** or by a **mouse click**. No help information is available in Software Version 1.
- The **R** field is selected with the key **F9** or by a **mouse click**. Selection of this field activates the Recall function, i.e. returns the user to the preceding level.

Input fields

Traversing range upper limit	0	mm
Traversing range lower limit	0	mm

To allow the input of data, the input cursor is positioned in the appropriate input field by means of the **TAB** or **SHIFT + TAB** keys or by a **mouse click**. The editing mode is always preset to **Overwrite**. It is possible to switch back and forth between overwrite mode and insert mode by means of the **Insert** key.

List fields

Measurement	Freq. response
Measured quantity	Following error

The functions offered are selected with the cursor keys **UP (↑)** and **DOWN (↓)** or **by a mouse click**. The displayed function is valid.
The list fields are selected by means of the **TAB** or **SHIFT + TAB** keys or a **mouse click**.

Single/ multiple selector button

Enable

◉ intern. ○ extern.

The desired function is activated with the cursor keys **LEFT (←)** and **RIGHT (→)** or by a **mouse click**.
The function fields are selected by means of the **TAB** or **SHIFT + TAB** keys or by a **mouse click**.

Multiple selector button			Single selector button	
☒	= active	or:	◉	= active
☐	= not active		○	= not active

Activation of fields

To be able to alter values and functions, the window with the input field must be activated by means of keys **CTRL + TAB** or with the key **HOME** (yellow frame = focus).

2.2.2 Additional information

Axis selection

The “Select axis/Select next axis” inputs in axis-specific displays are always made via the uniformly positioned vertical softkeys **AXIS+** and **AXIS–**.

Function selection/ deselection

All functions are activated by means of softkey **START** and deactivated by means of softkey **STOP**.

Password

When the softkey **Set password** is selected, a dialog box is displayed into which the password can be entered. Passwords are input as described in:

References: /BA/, Operator’s Guide
/FB/, A2, “Various Interface Signals“

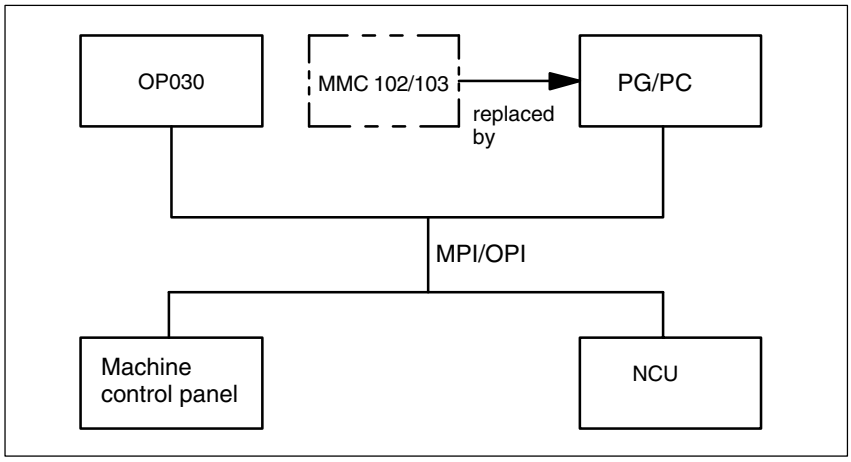
Key assignments

Apart from the assignments of keys **F1** to **F12** and **SHIFT + F1** to **F10**, the conditions and key assignments are the same as those under WINDOWS ™ 3.1.

The key combination **ALT + TAB** can be selected at any time to switch from “Operation via PG/PC” to other WINDOWS ™ applications.

2.2.3 Operation of operator panels

The system responds as follows, for example, when two panels are operated in the configuration illustrated below:



1. The inputs from the MMC or OP030 operator panel have the same priority as regards the NCU.
2. The operator panels are mutually independent in terms of data display, i.e. the display selected on one panel is not affected by the display on the other.
3. Spontaneous events such as alarms are displayed on both control units.
4. The protection level with the highest authorization in accordance with the lowest activated protection level number applies to both operator panels.
5. The system does not provide for any further co-ordination between the operator panels.

For further information, please refer to
References: /FB/,B3, Several Operator Panels and NCUs
/BA/ Operator's Guide

2.3 Simulation of part programs

A Windows 32s, version 1.30.166.0 or higher, must be installed in order to operate the part program simulation.

For operating instructions, please refer to
References: /BA/ Operator's Guide

■

Notes

3 Supplementary Conditions

Availability of function

The “Operation via PG/PC” function is available in the basic version with SW 3.1 and higher. With SW 3.1, the number of NCUs which may be connected is limited to one and the number of operator panels to two. One of them must be an OP030.
With SW 3.2 and higher, an operator panel with MMC 100 or MMC 102/103 can also be connected with up to three NCUs.

4 Data Descriptions (MD, SD)

No special machine data exist for this function.

Notes

Signal Descriptions 5

No signals are required at the NCK-PLC interface for this function.

Example 6

None

Data Fields, Lists 7

No signals or machine data are required for this function.

7.1 Alarms

A more detailed description of the alarms which may occur is given in

References: /DA/, Diagnostic Guide

or in the online help.

Notes

SINUMERIK 840D/FM-NC Description of Functions, Extended Functions (FB2)

Remote Diagnosis (F3)

■

Notes

Brief Description 1

1.1 Remote diagnosis for SINUMERIK (SW 4 and higher)

Scope of delivery and installation

The remote diagnosis software SW 5 for SINUMERIK is supplied on a CD-ROM. The CD-ROM contains several directories for creating diskettes. The appropriate directories are divided into different types of installation. Beneath these directories, the installations are divided into diskettes. All of the diskettes each in a subdirectory provide a version of remote diagnosis for different operating systems.

- Windows 95/98/NT Win32\disks 0–4
- Win 3.x: Win16\disks 1–3
- DOS: DOS\disks 1–3

To create one of these diskette sets, the contents of the corresponding directories of a version are copied to a diskette each.

On SINUMERIK 840D MMC 103, SW 4.x and higher, the Win32 variant is installed in all cases.

Insert the first diskette to start the installation.
Enter the following line in Windows service mode (with MMC environment) in the “Start” menu under “Run”:

DOS:	a:\installed
Windows 3.x:	a:\install
Windows 95/98/NT	a:\setup

From SW 3.1 onwards, the remote diagnosis option is available for the SINUMERIK 840D. With remote diagnosis, the MMC 102/103 on the machine is connected to another PC via modem, ISDN card, via a serial interface, network or via Internet. On the PC the same display appears as on the control. You can operate the MMC 102/103 from the PC.

Restrictions regarding the scope of delivery

To limit the scope of the Installation Guide, only one device was described per connection type as an example.

Table 1-1 List of tested communication devices.

Type of communication	Communication device
Analog phone link	USRobotics 56K FAX EXT
Digital phone link	Sportster ISDN TA ext.
Network connection	3COM Etherlink III, 3C509B-Combo

Remote diagnosis functions

The remote diagnosis comprises the following main functions:

- Remote Control
- File Transfer
- Chat

Remote Control

The service engineer (viewer) can set up a direct remote control link which allows him/her to operate the control system (host) interactively using a keyboard and mouse. The same functionality and screen contents are available on the viewer side as on the control system (host).

File Transfer

Files can be transferred between PC and control system in both directions.

Chat

The Chat function can be used for communication between the two operators on the PC and on the MMC 103. The operators can enter texts simultaneously via an input window and read messages from the partner in an output window.

Host 840D

With remote diagnosis, the MMC 103 is designated “Host 840D”. Software with the same name must be installed on the MMC 103. Once this software is activated, the MMC 103 can be operated from the “Viewer-PC”.

Viewer 840D

The PC connected to the control system is designated as the “Viewer 840D”. The viewer PC can be run under both Windows 95/98 and NT 4.0. The installation routine detects automatically which operating system is installed.

Modem

Minimum requirements are a 9600 baud modem. This Manual describes the 3Com / US Robotics 56K EXT modem.

ISDN

We recommend the external ISDN terminal adapter 3Com / US Robotics Sportster. It is treated in the same way as a modem.
For the ISDN card, the ISA card AVM Fritz!Card Classic is recommended. The ISDN card can be used from the new remote diagnosis version V5.1 on the MMC 103.

Network

The remote diagnosis can also be used on the MMC 102/103 via an Ethernet network using the TCP/IP (Windows sockets) protocol. For the network card, Siemens recommends the 3COM Etherlink III Network Card (3C509B-Combo or 3C905B-TX). In addition, an PCI/ISA adapter must be installed on the MMC 102/103.
With the MMC 100.2, no network connection is possible.

Internet

With the MMC 103, V4x. and higher, the remote diagnosis can also be run via the Internet. This requires an Internet provider (e.g., T-Online, AOL) providing a PPP dial-in node (analog or ISDN). In addition, either a modem, an ISDN terminal adapter or an ISDN card is required for the connection from the MMC 103 or PC to the dial-in node. On the Viewer's side, Windows 95/98 or NT must be installed.

Gateway

The remote diagnosis can also be operated over a Windows NT 4.0 gateway. This gateway is the link between the service PC and the MMCs that are connected to the customer Ethernet network. The service PC and the gateway are connected via modem or ISDN. The MMCs are connected to the gateway via Ethernet (TCP/IP). Here there are generally two possibilities:

1. From PC viewer –> gateway host and further from gateway viewer –> MMC host
2. With the gateway as RAS directly from the PC viewer –> MMC host

Keyboard, mouse

For more user-friendly operation of the remote diagnosis on the control, connect a keyboard and a mouse if required. These items of equipment facilitate text inputs in the chat window. A keyboard and possibly a mouse are needed in all cases for installation and configuration. The operation of the remote diagnosis is possible on control systems with OP3X or OP01x also without a keyboard.

1.2 Remote diagnosis for MMC 100.2 (SW 5.1.40 and higher)

The option “Remote diagnosis under MMC 100.2” differs in some regards from the remote diagnosis of the MMC 102/103. The differences are explained in this section. The description and instructions for remote diagnosis above generally apply to both MMC 102/103 and MMC 100.2. The differences are each described separately.

Scope of delivery and installation

In contrast to the MMC 102/103, the MMC 100.2 has no free hard disk capacity and is based on the DOS operating system. For this reason, the remote diagnosis software for the MMC 100.2 is supplied on a PC Card. This contains the complete software for using the remote diagnosis functions and is plugged into the MMC 100.2. The scope of functions of the software for the MMC 100.2 is limited, compared with the MMC 102/103. The same software is used on the viewer (PC) as on the MMC 102/103 (see Section 1.1 Scope of delivery and installation).

Note

If you use a Windows operating system on the Viewer PC for remote diagnosis on the MMC 100.2, the DOS software must be installed and operated in the DOS-box of Windows.
The Windows versions of the remote diagnosis cannot be used here.

Functions

The connection of the MMC 100.2 (host) to the PC (Viewer) is **only possible** with a modem. The viewer and host cannot be linked by Internet, network or gateway.

The remote diagnosis option for MMC 100.2 offers the following functions:

- Remote control (cf. MMC 102/103)
- File transfer (cf. MMC 102/103)
- Chat, Dialog; in limited form only via part program

Remote control

The service engineer (viewer) can set up a direct remote control link which allows him/her to operate the control MMC 100.2 (host) interactively using a keyboard and mouse. The same functionality and screen contents are available on the viewer side as on the MMC 100.2 (host).

File transfer

Files can be transferred between the PC and MMC 100.2 via the modem link. The files can only be stored on the MMC 100.2 on Ramdisk E: parallel channel. All other drives have read status only.

Chat

The Chat function is not implemented in the MMC 100.2. The users of the viewer and host can communicate only indirectly by means of a part program.

Host 840D/810D

For the purpose of remote diagnosis, the MMC 100.2 is designated “Host 840D/810D”. Software with the same name must be installed on the MMC 100.2. Once this software is activated, the MMC 100.2 can be operated from the “Viewer-PC”.

Viewer 840D/810D

The PC connected to the control system is designated as the “Viewer 840D/810D”. The viewer PC can be run under Windows 3.1/3.11, Windows 95 or NT 4.0. The DOS variant of the remote diagnosis must be installed.

Modem

Basic prerequisite for communication is a modem with at least 9600 baud, connected to the serial port on the PC and MMC 100.2.

■

Notes

Detailed Description

2

2.1 General

2.1.1 MMC 102/103

Software

The supplied software can be used to install the remote diagnosis on a SINUMERIK 840D control system and to integrate it into the area switchover. The software is supplied on a CD-ROM that contains a set of diskettes for installing the remote diagnosis.

Documentation

The supplied CD-ROM contains the Remote Diagnosis Documentation in pdf format (Adobe Acrobat) in the SIEMENS directory. The CD-ROM can be read and printed using Adobe Acrobat Reader (recommended: V 4.0).

Help function

A comprehensive help function is available to provide assistance for remote diagnosis operation on the MMC host and viewer PC; it is used as customary under Windows. The CD-ROM also contains further help files, such as readme.wri, support.wri, siemens_d.wri and siemens_e.wri, tipps_d.wri, tipps_e.wri, faq_d.wri.

Keyboard, mouse

The remote diagnosis function can be operated on the viewer PC by means of a mouse and a keyboard. A keyboard and possibly a mouse are needed in all cases for installation on the control system (MMC 103). The remote diagnosis should therefore already be configured at start-up. In the event of service, it is then not necessary to change anything in the configuration for the remote diagnosis. The remote diagnosis function can be activated without keyboard and mouse.

Communication Viewer ↔ Host

For communication between the operator on the machine and the service staff on the viewer PC, a so-called chat window is provided. Communication can take place only in the form of text inputs. Here we recommend you use a keyboard.

Second phone link

For larger service activities, a voice connection via a second telephone is recommended.

Input from ext. PC (viewer)

All operating actions (softkeys) can be carried out in the same way as the operating actions on the control system (MMC 103). The MCP cannot be operated.

Caution

In order to activate the machine control panel function, the viewer must request the operator to perform the necessary operations via CHAT or a second phone link.
The control system operator is responsible for ensuring that the requested measures can be implemented without endangering personnel or the machinery.
The operator must refuse to take any measures that are not guaranteed to be safe. The viewer cannot see the state of the machine or its safety equipment.

2.1.2 MMC 100.2

Software

The supplied software must only be installed on the PC. With all Windows operating systems, the installation of the DOS version must be called via “installd”. The DOS version installed can be run either in the DOS box of the Windows operating systems or in the DOS mode of Win95/98.
The software for the SINUMERIK 810/840 D MMC 100.2 is available on the PC card that can be inserted in the MMC. It need merely be configured to suit the individual system.

Help function

Use the Help in the software supplied for the MMC 102/103 on the viewer PC. Due to the operating system, the Help on the MMC 100.2 host is the DOS Help. You can call Help by pressing function key “**F1**” or select it in the “Help” menu.

Keyboard, mouse

The remote diagnosis viewer software can be operated using keyboard and mouse. With the MMC 100.2, only the keyboard can be used.

2.2 Hardware configuration for remote diagnosis

2.2.1 Modem, analog or ISDN

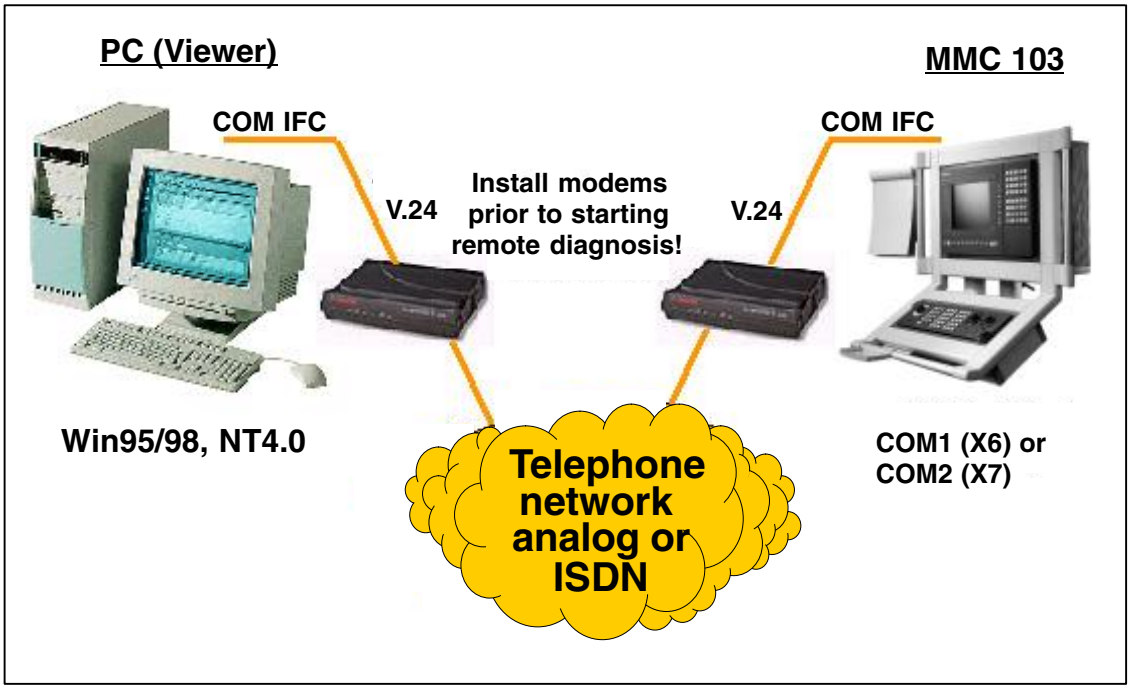


Fig. 2-1 Hardware configuration for remote diagnosis: Modem, analog / ISDN

Hardware configuration

The external PC and the MMC are connected to an analog or ISDN modem via a V.24 interface. Both modems can be connected to one another via an analog or ISDN telephone link.

Requirements of MMC (host)

- MMC 103
- Approx. 7 MB free memory on your hard disk (only MMC 103)
- 1 free COM port:
  In the MMC 103, it must be made sure that the Com port for the modem is not yet occupied by the V.24 or programming device interface.
  To this aim, the following operation must be carried out:
  1. Select the area switchover key.
  2. Select “Services”.
  3. Select both “V.24” and “PG” and check the reserved Com port. If the same Com port as the modem is used, then select “none”.

Requirements of PC (VIEWER)

- IBM AT or 100% compatible PC, Intel 486 or higher
- Windows 95/98 or Windows NT 4.0 operating system
- RAM according to operating system requirements
- Approx. 7 MB hard disk storage
- VGA or SVGA screen

Modem requirements

A modem with 14,400 baud or higher and the correspondingly specific driver must be installed and configured before installing the remote diagnosis tool. We recommend using brand name modems such as 3COM / US-Robotics Sportster Flash or Voice 33.6 PnP External for an analog connection and the 3COM / US-Robotics Sportster ISDN terminal adapter for ISDN connections.

2.2.2 Network (MMC 102/103 only)

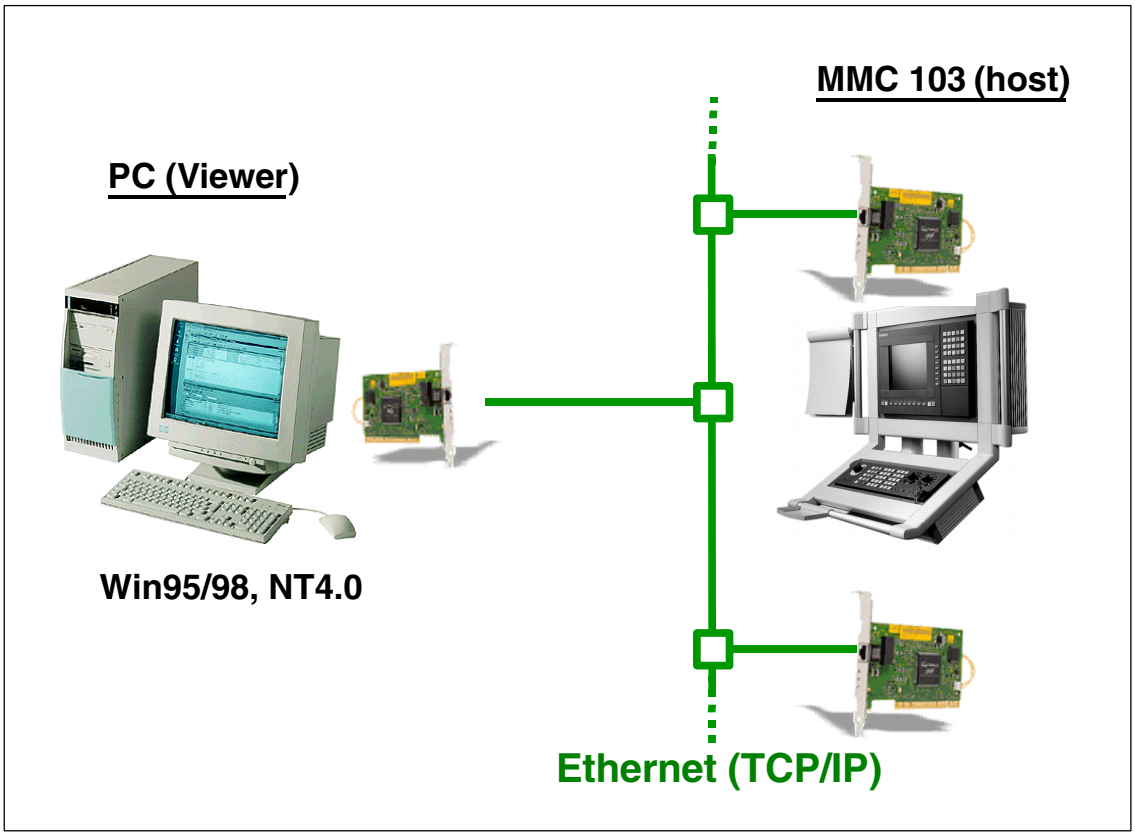


Fig. 2-2 Hardware configuration for remote diagnosis: Network

Hardware configuration

The external PC has a network card. The MMC 102/103 is equipped with an ISA BOX with a network card fitted (we recommend 3COM Etherlink III, 3C509B-Combo). Both computers are connected via an appropriate Ethernet cable.

2.2.3 Internet (MMC 102/103 only)

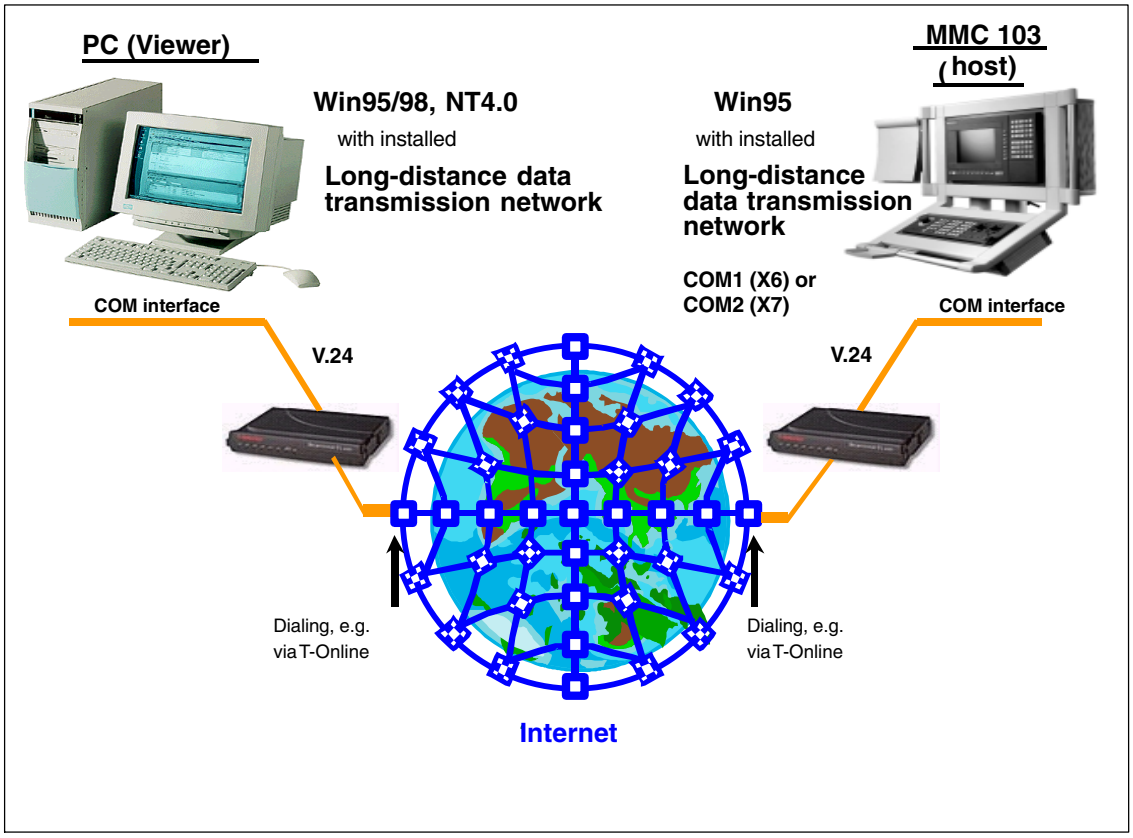


Fig. 2-3 Hardware configuration for remote diagnosis: Internet

Hardware configuration

Precondition for this solution is an MMC 103 with SW 4.x. In addition, an Internet provider with a PPP (point-to-point protocol) dial-in node is required on both sides. This documentation describes an Internet connection via the German provider T-Online.

2.2.4 Windows with gateway (MMC 102/103 only)

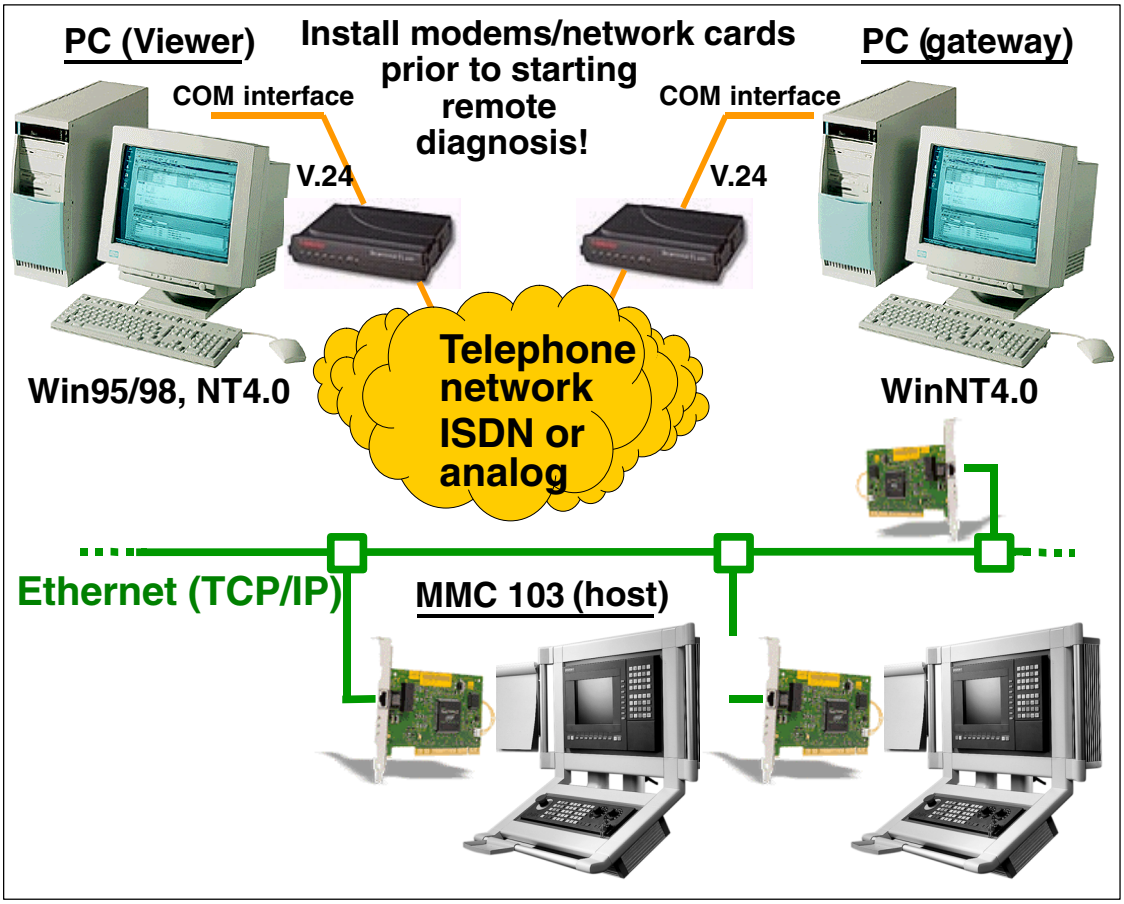


Fig. 2-4 Hardware configuration for remote diagnosis: Windows with gateway

Hardware configuration

The basis of this solution is a gateway PC which must run under Windows NT 4.0 workstation or server.

There are two possibilities for the remote diagnosis connection via the gateway:

1. From PC viewer –> gateway host and further from gateway viewer –> MMC host (cascading)
2. With the gateway as the RAS server remote diagnosis directly from the PC viewer –> MMC host In this case, the Gateway PC must be configured as the RAS server.

In the case of the 2nd solution, the licenses for an additional host and viewer are not required, and the performance is a bit better.

Note

Only the first solution is described in this Documentation.

2.3 Software installation and start-up of the communications I/O

2.3.1 Installing and configuring an analog modem

Installation the modem on MMC 103

MMC 103 (host) specific steps:

1. Start the MMC 103 in Windows service mode. (Items 2 to 5 only apply when the MMC 103 is generally restarted. If the computer was shutdown using the menu “Start”, “Exit”, the MMC starts again automatically. Respond to the one prompt displayed by selecting “2” (start Windows; change ini files). Otherwise continue with point 6).
2. Switch the control system on. When the text “Starting Windows 95” appears on the screen, press the key “6”. The control’s service menu is overlaid.
3. Select key “4” from the menu.
   ⇒ Start Windows (service menu)
   You will be prompted to enter a password: “passwd:”
   (The password authorizes to perform essential intervention on the control system. It corresponds to one of the following access levels:
   – System
   – Manufacturer
   – Service).

Note

The password must be entered in uppercase letters.

---

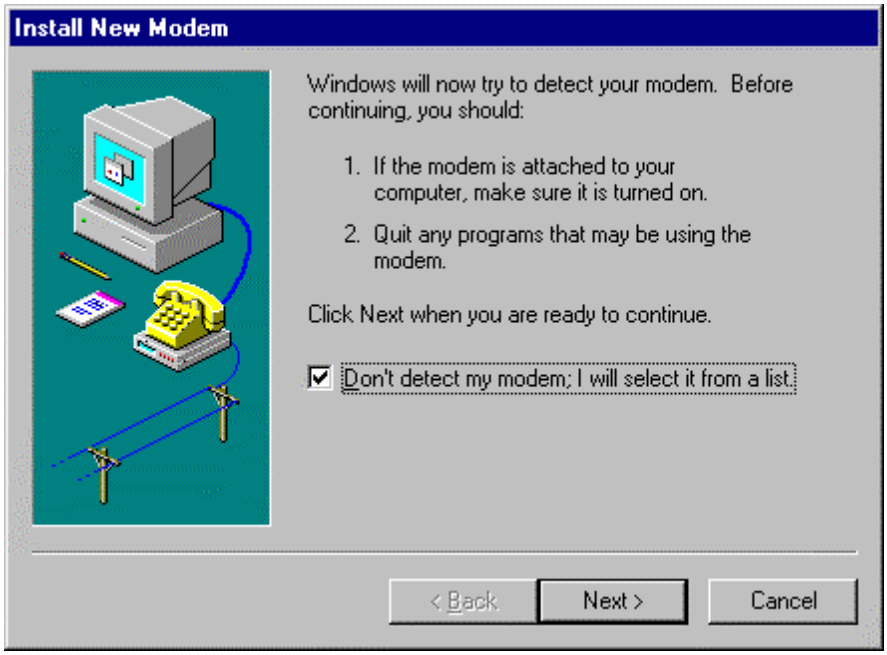
4. Another menu is displayed. Choose key “2”.
   ⇒ Windows (Changing Environment for MMC2)
5. The Windows operating system is started.

Installing the modem under Win95/98/NT4.0

General steps (viewer/host):

6. Call the installation program:

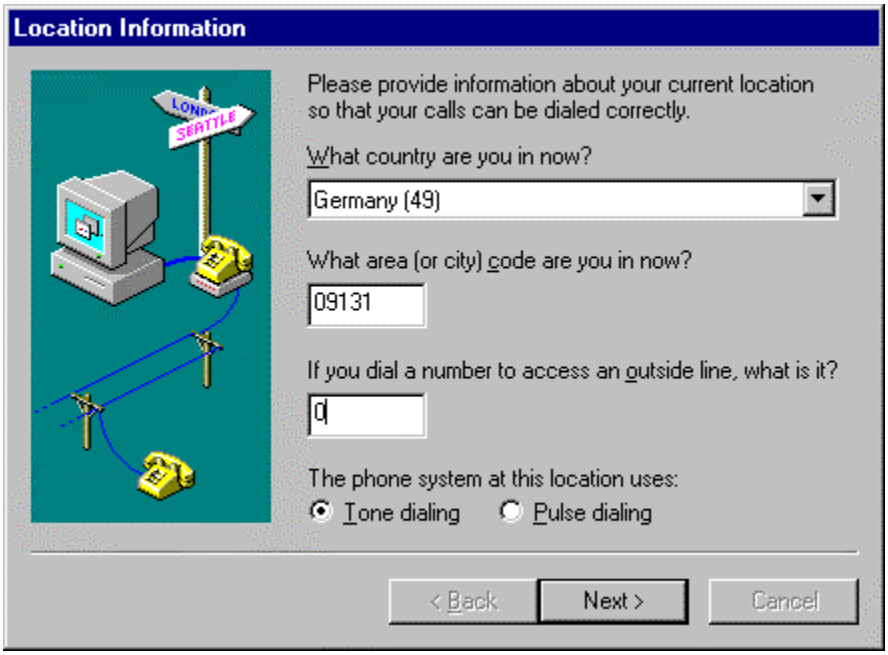
   Double-click on the modem icon to start the installation program via “Start/Settings/Control Panel”.
7. Select type of modem (only Win98):
   From the dialog box “Install New Modem”, choose the option “Add”. Click “Next” to continue the installation.
8. Modem Detection:
   Choose “Select Modem (Don’t detect my modem; I will select it from a list)” to save time (use the TAB key for selection without using the mouse; press the spacebar to confirm your selection).



9. Setting up a new modem:

   Insert the driver diskette supplied with the modem, e.g. “Windows95 Drivers, V2.2, PN:2.018.010-E” and click on “Have Disk...”.
   Enter “a:\” as source in the next input field.
   Select the driver “U.S. Robotics 56K FAX EXT”. If it is the first time that a modem is installed on your control system, in addition, you will be asked on which COM port you wish to install the modem. Choose a free COM port.
   If you then click “Next”, the data are copied into the appropriate directory.

10. My locations:



In this box, enter country code, area code, extension and MVF [tone dialing)]. Select “Next”.

Select “Finish”.
A message is displayed stating whether the modem configuration was successful.

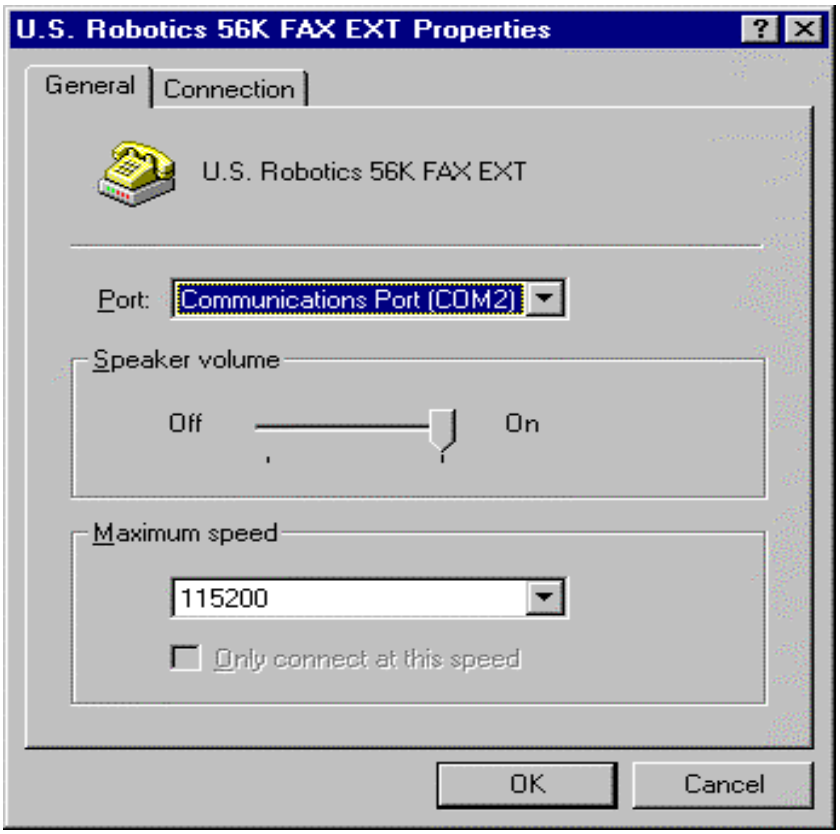
Windows 95 modem settings

If you wish to modify your modem settings later, there are two possibilities to start the configuration tool for the modem.

1. Boot the MMC in Windows service mode and, as in **Win95/98 and NT4.0** choose the menu “Start/Programs/Settings/Control Panel”; then double-click on the Modem icon.
2. Activate the appropriate password in the MMC. In MMC mode, configure the modem by choosing the menu “Configure/Options” from “Diagnostics” → “Remote Diagnosis”. Then click on the “Configure” button alongside “Modem”.

Configuring the modem

Then you can configure the modem as follows:



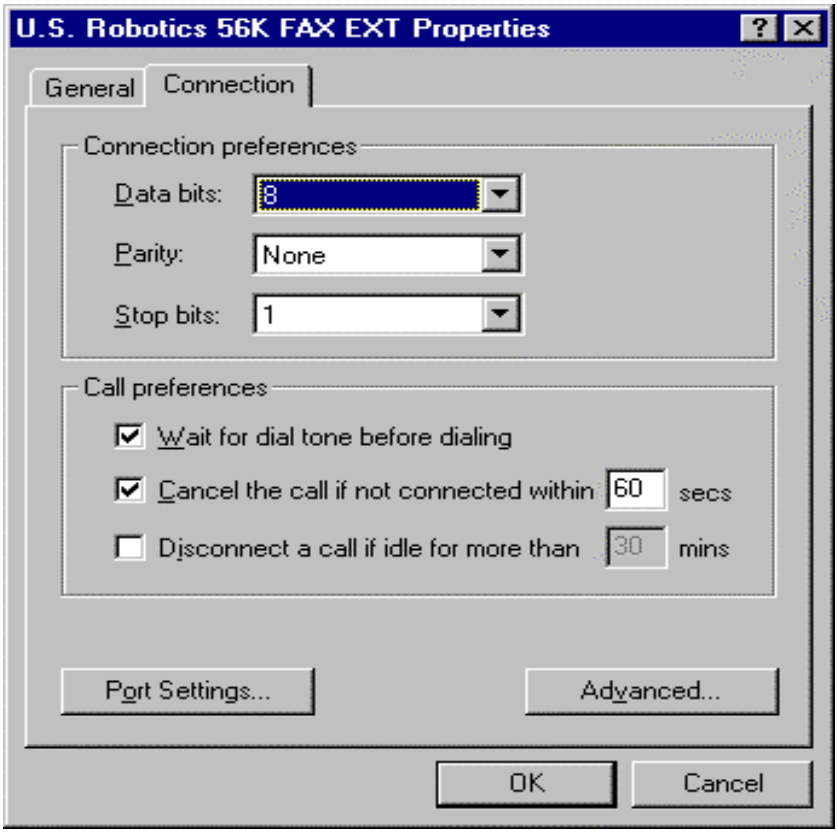
The settings:

- V.24 interface (COM1, COM2, ...)
- Modem volume
- Baud rate (transfer speed)

can be adapted in this dialog box.

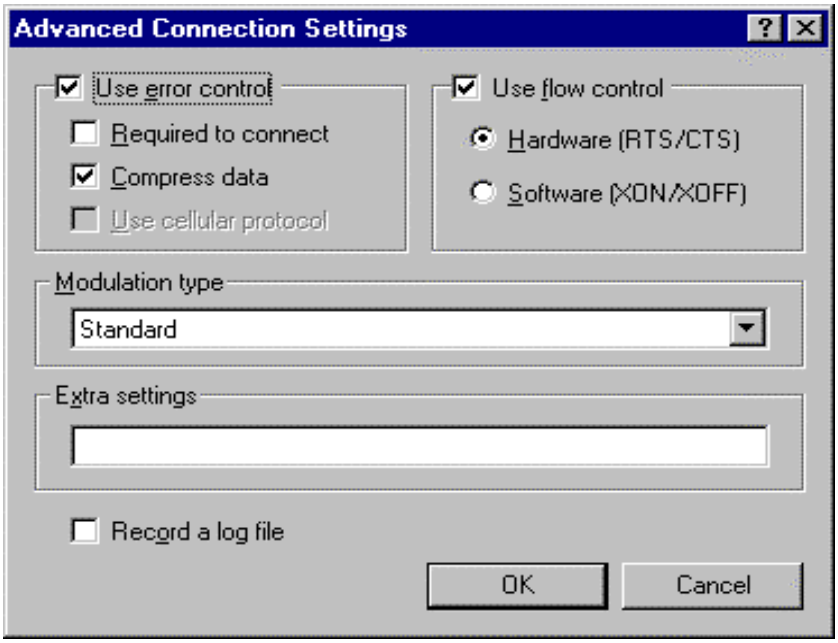
Note

Make absolutely sure that the modem settings of viewer and host are the same.
Otherwise, it is possible that no connection is set up.

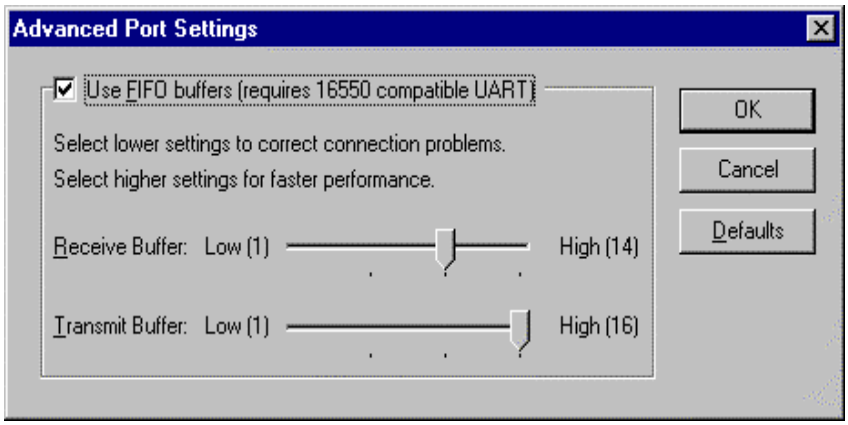


The setting “Wait for dial tone before dialing” must be deactivated when using a PABX.

The settings for “Port Settings...” and “Advanced...” must be made as follows:



In the event that problems occur during connection setup, it is advisable to activate the “Record a log file” field. The cause will be logged in the file “c:\win95\modemlog.txt”.



2.3.2 Installing and configuring the ISDN terminal adapter

The following description only applies to the 3COM / US-Robotics Sportster TA terminal adapter.

Installing the ISDN terminal adapter on MMC 103

MMC 103 (host) specific steps:

1. Start the MMC 103 in Windows service mode. (Items 2 to 5 only apply when the MMC 103 is generally restarted. If the computer was shutdown using the menu “Start”, “Exit”, the MMC starts again automatically. Respond to the one prompt displayed by selecting “2” (Windows; (Changing Environment for MMC2). Otherwise continue with point 6).
2. Switch the control system on. When the text “Starting Windows 95” appears on the screen, press the key “6”. The control’s service menu is overlaid.
3. Select key “4” from the menu.
   ⇒ Start Windows (service menu)
   You will be prompted to enter a password: “passwd:”
   (The password authorizes to perform essential intervention on the control system. It corresponds to one of the following access levels:
   – System
   – Manufacturer
   – Service).

Note

The password must be entered in uppercase letters.

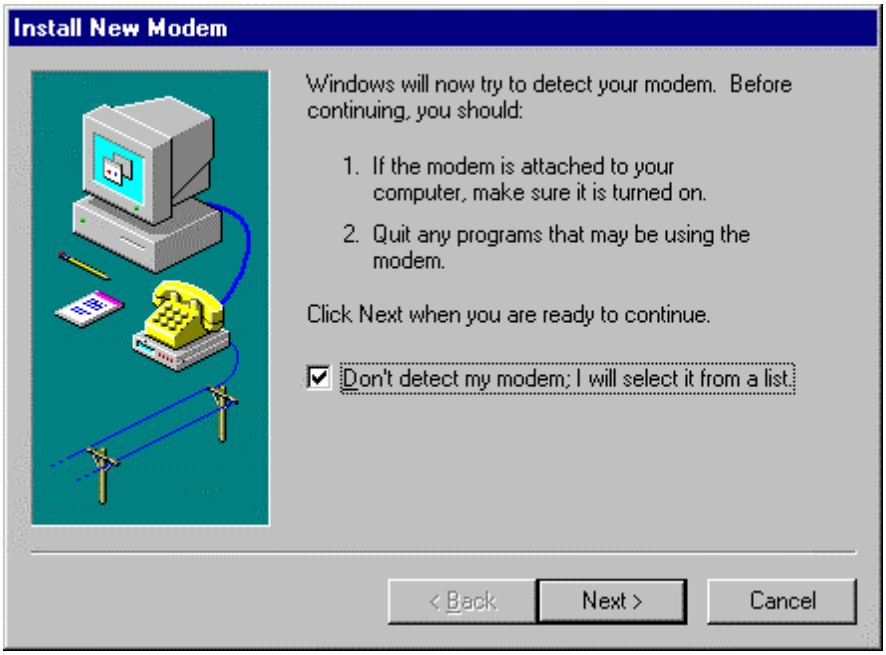
---

4. Another menu is displayed. Choose key “2”.
   ⇒ Windows (Changing Environment for MMC2)
5. The Windows operating system is started.

Installing the modem under Win95/98/NT4.0

General steps (viewer/host):

6. Call the installation program:
   Double-click on the modem icon to start the installation program via "Start/Settings/Control Panel".
7. Select type of modem (only Win98):
   From the dialog box "Install New Modem", choose the option "Add". Click "Next" to continue the installation.
8. Modem Detection:
   Choose "Select Modem (Don't detect my modem; I will select it from a list)" to save time (use the TAB key for selection without using the mouse; press the spacebar to confirm your selection).



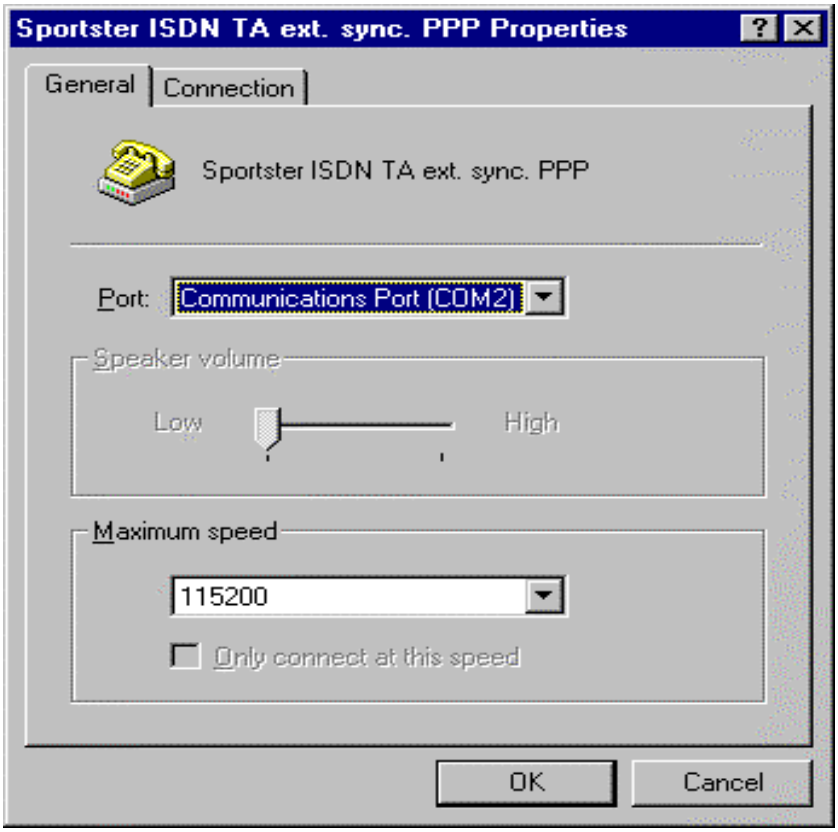
9. Setting up a new modem:
   In the left menu select US-Robotics, Inc. Insert the diskette supplied with the modem, e.g. "Windows95 Drivers, V2.2, PN:2.018.010-E" and click on "Have Disk...".
   Enter "a:\" as source in the next input field.
   Select the driver "Sportster ISDN TA ext. sync. PPP". If it is the first time that a modem is installed on your control system, in addition, you will be asked on which COM port you wish to install the modem. Choose a free COM port.
   If you then click "Next", the data are copied into the appropriate directory.
10. My Location:
    In this box, enter country code, area code, extension and MVF [tone dialing)].
11. Select "Finish".
    A message is displayed stating whether the modem configuration was successful.

Windows 95 ISDN modem settings

If you wish to modify the settings of the ISDN terminal adapter later, you can start the configuration tool for the terminal adapter as follows:

Boot the MMC in Windows service mode and, as in **Win95/98 and NT4.0** choose the menu “Start/Programs/Settings/Control Panel”; then double-click on the Modem icon.

Then you can configure the modem as follows:



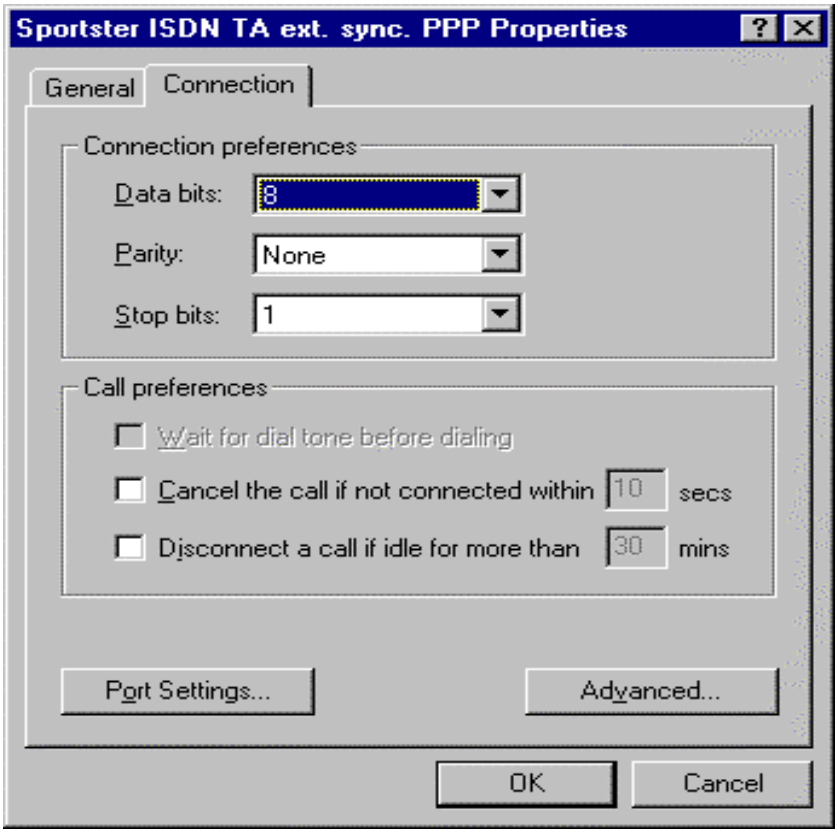
The settings:

- V.24 interface (COM1, COM2, ...)
- Baud rate (transfer speed)

can be adapted in this dialog box.

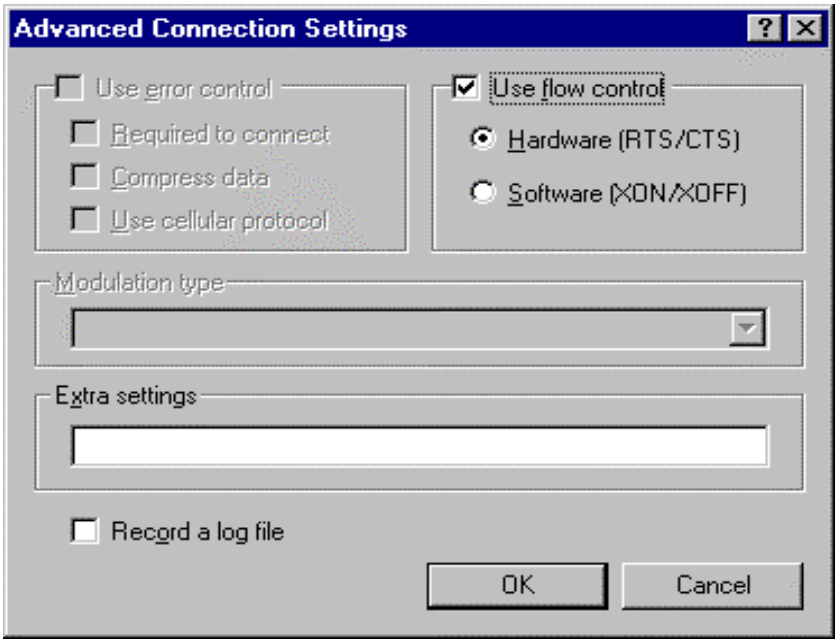
Note

Make absolutely sure that the modem settings of viewer and host are the same. Otherwise, it is possible that no connection is set up.

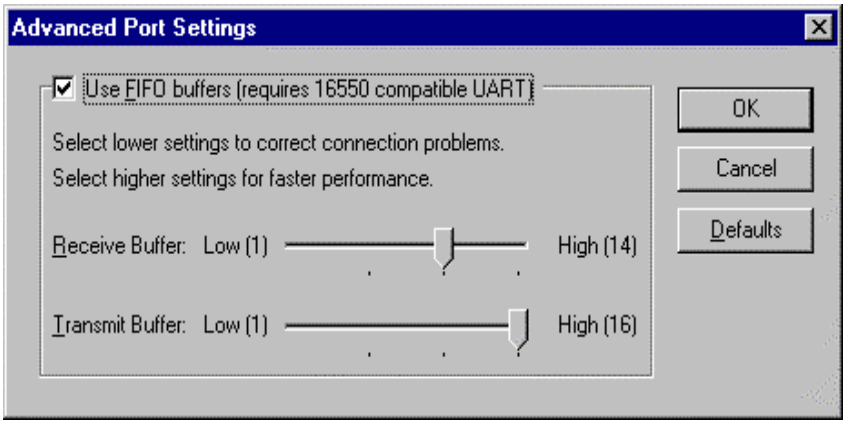


The setting “Wait for dial tone before dialing” must be deactivated when using a PABX.

The settings for “Port Settings...” and “Advanced...” must be made as follows:



In the event that problems occur during connection setup, it is advisable to activate the “Record a log file” field. The cause will be logged in the file “c:\win95\modemlog.txt”.



The same US-Robotics ISDN TA, for example, can be used on the viewer side. In this case, the same must be configured similarly, depending on the viewer version used (Win 3.1X, 95, 98, NT). It is also possible to use the modem entry “AVM ISDN – ISDN (X.75)” under Windows 95/98 on the viewer side in conjunction with the AVM Fritz!Card.

Note

On the MMC 103, the Fritz!Card only operates satisfactorily via the CAPI2.0 interface from version 4.3 onwards.

2.3.3 Installing and configuring the ISDN Fritz!Card

The use of the ISDN Fritz!Card (ISA and PCI) is possible on the **MMC103** from version 4.3 onwards.

Installing the ISDN Fritz!Card on the MMC 103, version 4.3 and higher

Before you can start with the installation on the MMC 103, the following steps must be carried out:

MMC 103 (host) specific steps:

1. Start the MMC 103 in Windows service mode. (Items 2 to 5 only apply when the MMC 103 is generally restarted. If the computer was shutdown using the menu “Start”, “Exit”, the MMC starts again automatically. Respond to the one prompt displayed by selecting “2” (Windows; (Changing Environment for MMC2). Otherwise continue with point 6).
2. Switch the control system on. When the text “Starting Windows 95” appears on the screen, press the key “6”. The control’s service menu is overlaid.
3. Select key “4” from the menu.
   ⇒ Start Windows (service menu)
   You will be prompted to enter a password: “passwd:”
   (The password authorizes to perform essential intervention on the control system. It corresponds to one of the following access levels:
   – System
   – Manufacturer
   – Service).

Note

The password must be entered in uppercase letters.

---

4. Another menu is displayed. Choose key “2”.
   ⇒ Windows (Changing Environment for MMC2)
5. The Windows operating system is started.

Installing the ISDN Fritz!Card under Win95/98/NT4.0

General steps (viewer/host):

6. Carry out the installation:
   For the description of the installation, refer to the documentation of your Fritz! Card, since both the ISA and the PCI ISDN Fritz!Card can be used.

2.3.4 Installing and configuring a network

Installing the Ethernet card 3COM Etherlink III Combo on MMC 102/MMC 103

Exit Windows via the Program Manager (up to V3.x) or, with V4.x and higher, use “Start” (CTRL+ESC)→ “Shut Down”→ “Shut down the computer?”→ “Yes”.
The following prompt is displayed:
“Save Windows Environment for next MMC Start [Y,N]?”
Enter “Y” (Yes).
Select “5” (Return to Main Menu).
Select “3” (DOS Shell)
Enter “passwd:”
An MS-DOS Shell is opened.

Configuring the Ethernet card with PnP disabled and MMC-BIOS up to V2.12

Insert the driver diskette supplied with the card into the floppy drive that is connected to a PC/programming device either locally or via Interlink.
Enter “a:\INSTALL” if you are using a local floppy disk drive.
Enter “e:\INSTALL” if you are using a floppy disk drive connected via Interlink.
Select “Y=Agree to License”.
Select “[ENTER]=Continue”.
Select “Configuration/Diagnostic/Troubleshooting”.
Select “Configuration and Diagnostic Program”.
The adapter is automatically detected. Configure it as follows:
Select “Install” –> “Configure Adapter”.

Make the following settings in the dialog boxes:
I/O Base Address: Select “300h”.
The next setting for the IRQ is important as IRQ 10 is already assigned.
Interrupt Request Level: Select “11” .
Select Boot PROM: “Disabled”.
Transceiver Type: According to the network connection cable used, e.g.:
“On-Board Coax (BNC)” with an RG58 (Cheapernet) network cable.
Do not use “Auto select”, as this can cause problems!
Network Driver Optimization: Select “Windows or OS/2 Client”.
Set Maximum Modem Speed: “9600 baud”.
Plug and Play Capability: “Disabled”.
Full Duplex: “Disabled”.
Activate “OK”.
The configuration is stored in the Etherlink III.
Press “ESC” 3 times to exit the program.

Exit the DOS Shell with “Exit”.
Select “4” (Start Windows (Service Mode)).
Type “passwd:”.
Select “2” (Windows (Changing INI files for MMC2)).
Windows is started. Now you can carry out instructions in the next chapters.

Configuring the Windows network:
If you encounter problems when configuring the network, please refer to the c:\windows\network.wri file.
Check the Windows start entry in the c:\tools\loadmmc2.bat boot file:
With “C:\WINDOWS\WIN /N :” delete the option “/N” (network No) if necessary!
Alternatively, you can insert “/N” if no network is connected, without removing the network configuration.

Load the network drivers for Windows for Workgroups up to MMC 102/103, V3.6

From the Main Program Group in the Windows Setup, choose the subitem “Change Network Settings” from the menu “Options”.
Click on the “Networks” command button. A new screen form is overlaid.
Select “Install Microsoft Windows network” and confirm with “OK”.
Click on the “Drivers” command button. A new screen form is overlaid.
Activate “Add Network Adapter”. A new screen form is overlaid.
Select “Unlisted or Updated Network Adapter” and confirm with “OK”.

Note

If any problems occur with access to the diskette, switch off the MMC 102/103 and then on again and continue the configuration with starting the DOS-Shell using the key “3”.

Insert disk 1/2 labeled ’3COM EtherDisk for Etherlink III Adapter V4.2 or higher.

Enter “A:\WFW311” and confirm with “OK” if you are using a local floppy disk drive.
Enter “E:\WFW311” and confirm with “OK” if you are using a floppy disk drive connected via Interlink.

Installing the drivers from MMC V6.1

Select “3Com EtherLink III (3C5X9)” and click “OK” to confirm.
Select “3Com EtherLink III (3C5X9) [NDIS2/NDIS3]” and confirm with “Setup”.
Set Base I/O Port (hex) to “0x0300” and confirm with “OK ”
Confirm all menus with “Close” or “OK” until the following prompt is displayed.
For “Microsoft Windows Network Names” make, for example, the following entries :

USER name: SIN840D
Workgroup: MyCompany
Computer name: SIN840D

Confirm the message “The files for Enhanced Mode Protocol Manager are currently installed on your Computer” with “Yes to All”.

After the installation you must return to Windows. Do not start Windows again as the settings will be lost if you do!

MMC 103, V4.2 and higher:
Installing the Windows 95 network drivers:
Configure the following via “Start” (CTRL+ESC) → “Settings” → “Control Panel” → “Add New Hardware”:
Select “Next”.
“The question “Do you want Windows to search for new Hardware?” is displayed. Reply with “No” and confirm with “Next”.
Select “Network Adapters”.
Insert the diskette labeled ’3COM EtherDisk for Etherlink III Adapter V5.0 or higher’.
Select “Have Disk”. Under “Copy Manufacturer’s files from:” enter “A:\” and confirm with “OK”.
Select “3Com EtherLink III (3C509/3C509b) in ISA mode”and press “OK” to confirm your selection.
Accept the setting selected by Windows 95 with “Next” (the setting is corrected later).
Confirm the message “Please insert the disk labeled ’Windows 95 CD-ROM’, and then click OK by clicking on “OK”.
A new box is overlaid. Under “Copy files from:” enter “C:\W95INST” and confirm with “OK”.
Respond to the prompt “Do you want to keep this file?” with “Yes” for mapi32.dll and vredir.vxd.
Select “Finish”.

You are asked “Do you want to shut down your computer now?”. Reply “No”.

Shut down Windows via “Start” (CTRL+ESC)→ “Shut Down”→ “Shut down the computer?”→ “Yes”.
The following prompt is displayed: “Save Windows Environment for next MMC Start [Y,N]?”. Confirm with “Y” (Yes).
Select “2” (Overwrite your last saved Windows Environment with the current one).
Select “2” (Windows (Changing Environment for MMC2)).
Wait until Windows 95 is started with active network.

If the box “Enter Network Password” is displayed, enter the following:

User name:	e.g.: chris
Password:	e.g.: RETURN for no password
Confirm new password:	e.g.: RETURN

Correcting the Windows 95 network driver configuration:
Configure the following via “Start” (CTRL+ESC) → “Settings” → “Control Panel” → “Add New”:
Select “3Com EtherLink III (3C509/3C509b) in ISA mode”. Under “Properties” click on the “Resources” tab (use the TAB key to change to the relevant tab and use the cursor keys to select).
Under “Configuration type:” select “Basic Configuration 0”; under “I/O address range” enter “300 – 30F” and confirm both settings with “OK”.

You are asked “Do you want to shut down your computer now?”. Reply “No”.

TCP/IP on MMC 102

Installing the TCP/IP protocol

For installation, the MS TCP/IP 32 stack is required. For further information please refer to the appropriate contact person.

Setting up a TCP/ IP protocol on MMC 103

1. From the “Start” menu, choose → “Settings” → “Control Panel” “Network”.
2. Click on “Add” and select “Protocol” in the overlaid dialog box.
3. In the second dialog box, select “Microsoft” in the left selection field and “TCP/IP” in the right one.
4. Confirm with OK to add the new network protocol.
5. Start the PC again to validate the changes.
6. From the “Start” menu, choose → “Settings” → “Control Panel” “Network”. You can view or edit the IP address by double-clicking on the “Network components” symbol in the display window or by selecting “Properties”. This IP address is required by the viewer for connection setup.
7. Save all current settings, or else they will be lost at restart. Power down the computer via “Shut down” and “Shut down Computer” in the Start menu.
8. The following prompt is displayed: “Save Windows Environment for Next MMC start Y, N]?”

!

Important

Confirm by typing “Y”.

Another question is displayed:
“...overwrite them with the current one ?”

Respond with “1” (Overwrite...). You can start Windows again or boot the control in MMC mode. (Alternatively continue with point 7).

To be on the safe side, press “2” (Start Windows) again.
Shut down Windows again via “Start” and “Shut Down”. Respond to both queries as described above. The backup is saved a second time and the configurations are also saved should an error occur.

2.3.5 Installing and configuring the Internet (for MMC 103, SW 4.2 and higher)

Configuring the long-distance data transmission network for T-Online PPP Internet access

Prerequisite for a remote diagnosis via the Internet is the connection setup via a long-distance data transmission network (long-distance data transmission / Internet). This is set up under Windows 95 on the viewer PC and in Windows service mode on the MMC 103. The following two requirements must be met before the actual Internet access can be configured under Windows 95:

- A modem, ISDN terminal adapter or ISDN card
- Long-distance data transmission network

A new connection is established using the long-distance data transmission network. The installation is completed once this connection has been set up.

Windows 95 can establish long-distance connections (long-distance data transmission) only via a modem. For the installation instructions of the modem, please refer to the supplied Manual or follow the steps mentioned in **Section 2.2.1**.

Modem, external ISDN terminal adapter or ISDN card

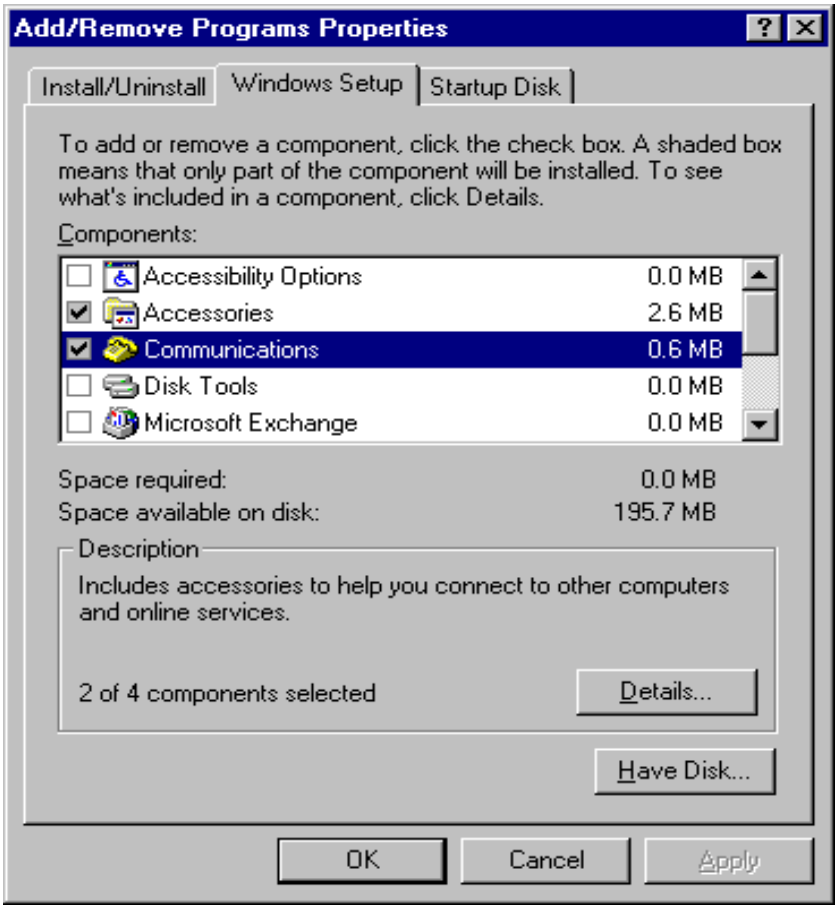
An analog modem is not necessary if an ISDN access and an ISDN card already exist. Since Windows 95 can directly serve only modem connection, a modem must be simulated for the PC. In general, appropriate drivers are either supplied with the ISDN card or can be obtained from the manufacturer.

The company AVM supplies a driver that works with many ISDN cards. This CAPIPORT driver is available for their Fritz!Card. It can also be used with TELES 16.0 and TELES 16.3 PnP. The German Telekom's Teledat 150, which is also from AVM, operates with the AVM CAPIPORT driver.

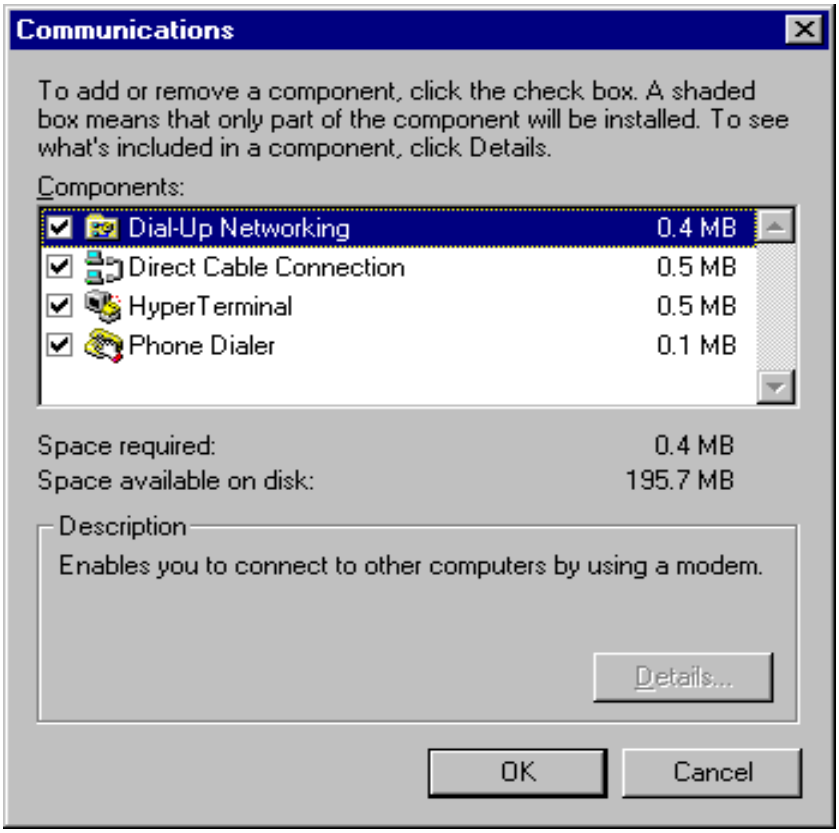
Long-distance data transmission network

Install the Windows 95 long-distance data transmission network. To do this, start the MMC 103 in Windows service mode with MMC 102 environment. Then proceed as follows:

- Activate "Control Panel" via "Start" and "Settings".
- Select the option "Add/Remove Programs" and click on "Windows Setup".
- Select "Communications" by double-clicking on it.



- Select “Communications” by double-clicking on it.

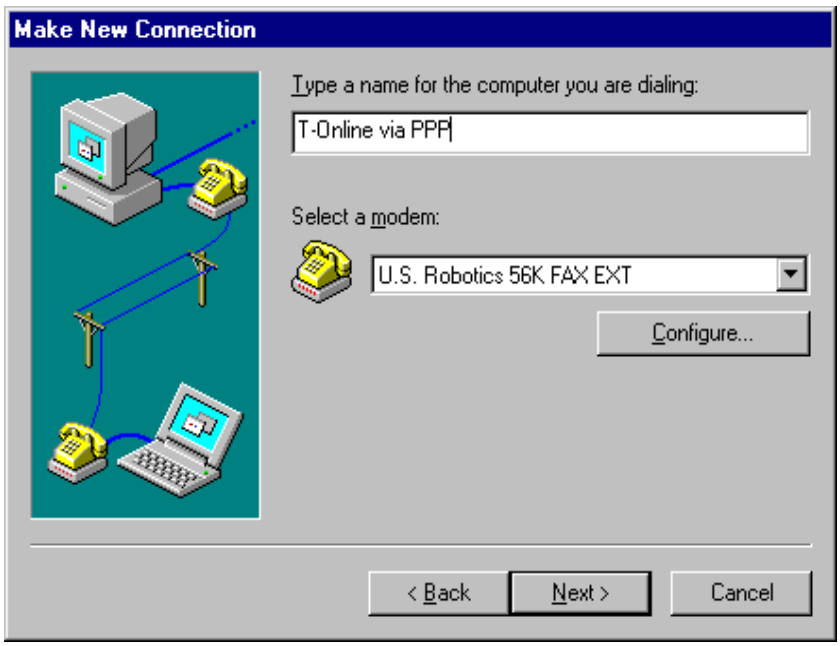


- Activate “Dial-Up Networking” and confirm with “OK”.
- The long-distance data transmission network is installed and can be opened by double-clicking on it under My Computer.

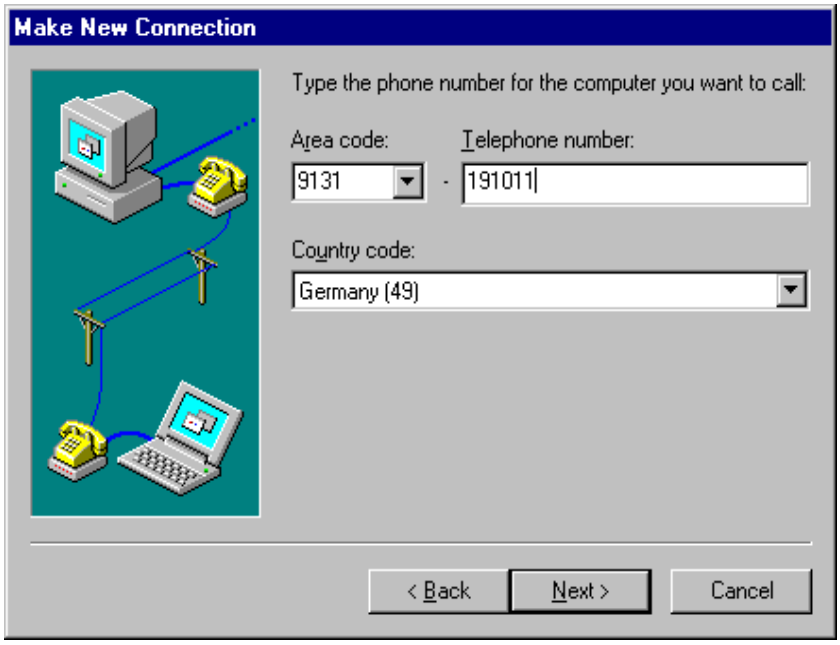
The TCP/IP protocol is not automatically added when the long-distance data transmission network is first installed. Do this now via “Control Panel” → “Network” → “Add” → “Protocol” → “Manufacturer Microsoft” → “TCP/IP”.

New Connection

Click the icon “Create new Icon” and make a new connection via “Start” (Ctrl + Esc) → “Programs” → “Accessories” → “Dial-Up Networking”:



Select a name for the new connection and select a modem.

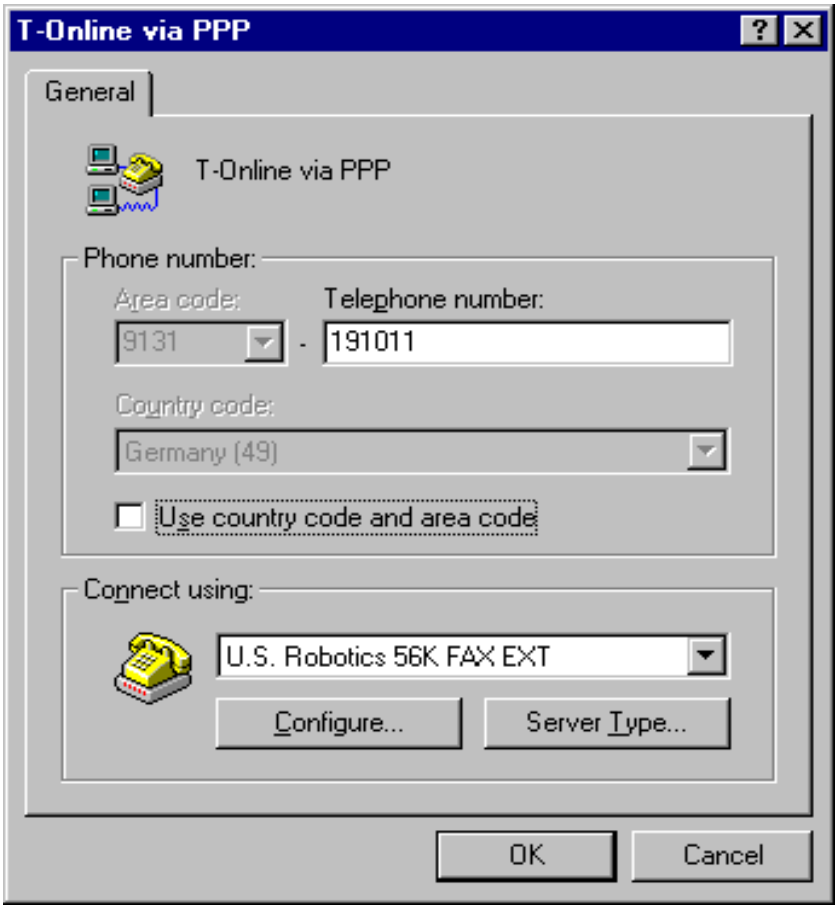


Enter the area code without leading zero, the country code for Germany and 191011 as telephone number for T-Online PPP (also without leading zero).

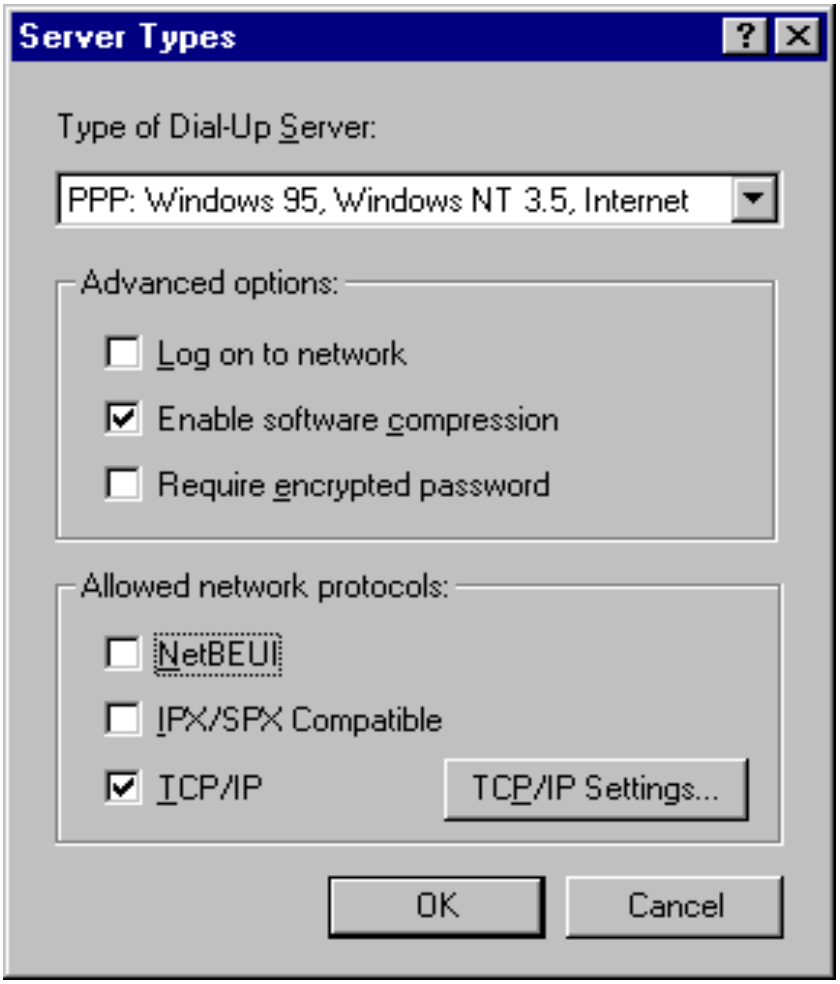
The area code is only of importance should the local access node fail temporarily. You can also change the configuration later (after setting up the connection) to select a cheaper tariff.

Click on “Next” and in the overlaid box on “Finish”. The new connection is stored, but must still be configured before use.

In the “Dial-Up Networking” folder (which is probably still open), right mouse-click on the newly created connection to open the object menu. Select “Properties” and make the following changes to the initial parameter settings.



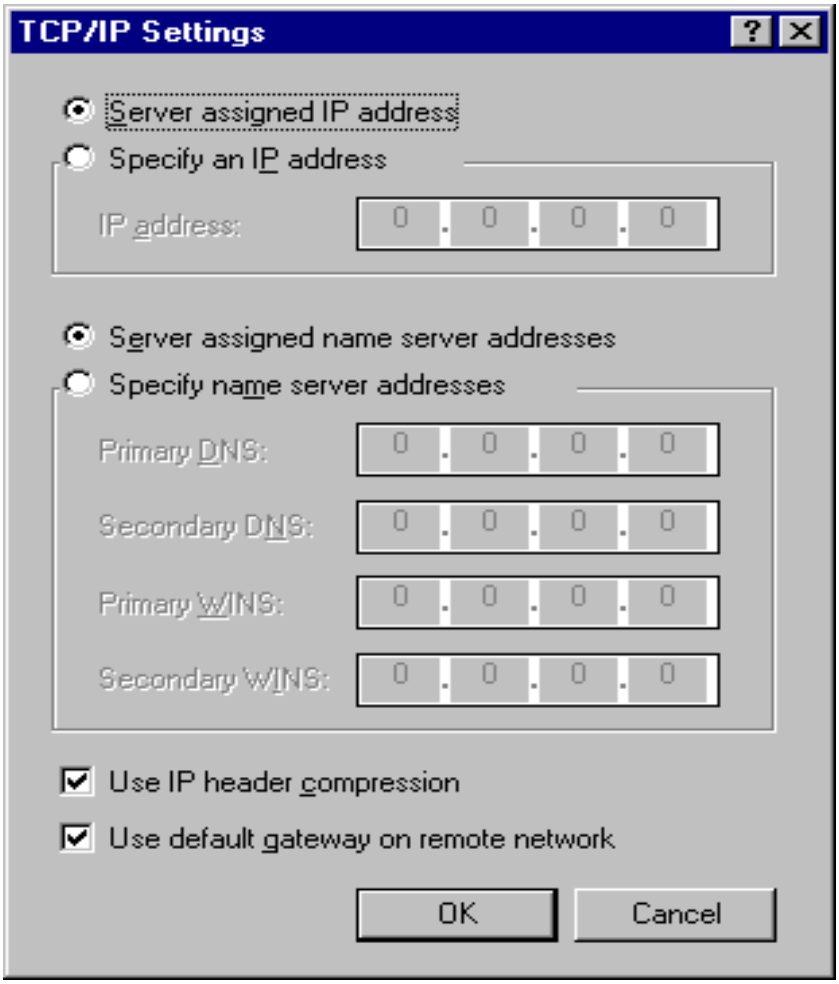
- Deactivate “Use country code and area code”, as now the number 091011 should be available for T-Online in all areas in Germany. The call number must have a leading zero, or two leading zeros if required, if you are using a PABX.
- Change the modem settings by activating “Configure...”.
- Make the following settings under “Server Type...”:



- Type of dial-up server: “PPP: Windows 95”
- Advanced options:
  Log on to network: OFF
  Enable software compression: ON
  Request encrypted password: OFF
- Allowed network protocols: “TCP/IP”

Caution

Only activate “Log on to network” if you are working in a local network.
All standalone users (without SINCOM or SINDNC) are much faster if **“Log on to network”** is deactivated.

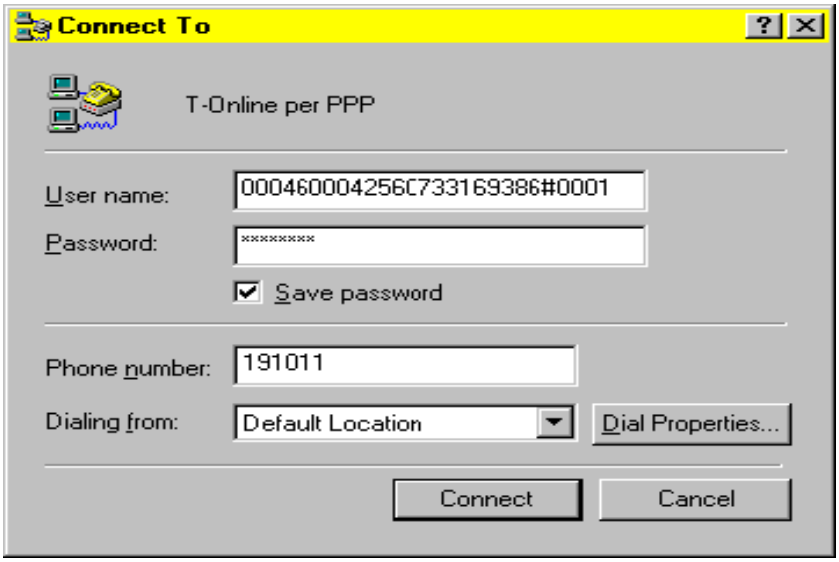


- Do not set the IP address. T-Online operates with dynamically allocated IP addresses that are assigned by the server.
- Do not enter the name server address. It is determined by the server at connection setup.
- Activate “Use IP header compression” and
- “Use default gateway on remote network”.

Close any open dialog boxes with “OK”.

Configuring access to T-Online

Except the access data to T-Online, the newly created connection is now configured. The access data is composed of different numbers that are contained on the order confirmation for your T-Online access. Enter the required data when you establish the first connection with T-Online.
The user name consists of several numbers:



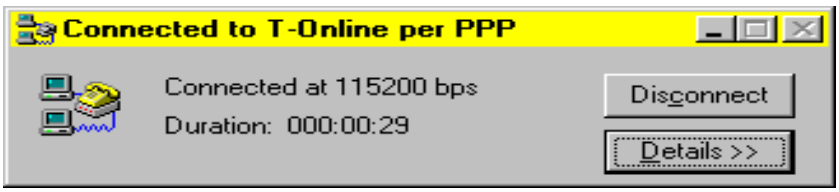
- Twelve-digit connection ID (in this case 000460004256)
- T-Online No. (here: 0733169386). If the number is shorter than 12 digits, it must be followed by “#”.
- Number for other user (here: 0001)

Password

Enter the personal password of your T-Online connection, which is to be found on the order confirmation.
You can activate “Save password” to save the encrypted password in a system file. This way it is not necessary to enter the password for every connection setup.

Call number

The call number can be modified here once more if the number 0191011 is not accepted.
Click on “Connect” to set up a connection between the MMC 103 and T-Online. Small status menus are displayed containing a log of what is currently happening. Lastly the following dialog box is overlaid:



The connection has been set up successfully. The remote diagnosis software can be operated over the Internet.

Integration into the MMC user interface

integration of the Internet dialing via the long-distance data transmission network using the example of a T-Online PPP access. The following files must be loaded:

c:\add_on\regie.ini

[TaskConfiguration]

Task11 = name := oemframe, CmdLine := “C:\\reachout\\internet.bat”, Timeout := 50000,
HeaderOnTop := False, Preload := False

c:\add_on\language\re_gr.ini

[HSoftkeyTexts]

HSK11 = “Internet Connection”

c:\add_on\language\re_uk.ini

[HSoftkeyTexts]

HSK11 = “Internet Connection”

c:\Reachout\internet.bat

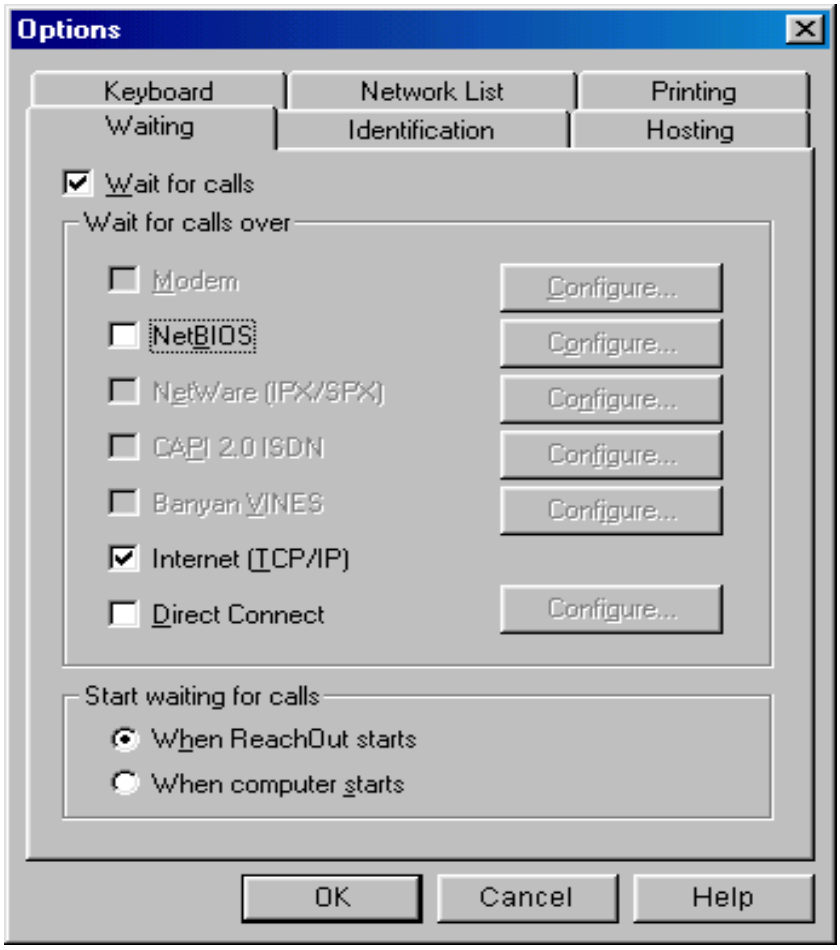
```
@echo off
rem Set up dial-up connection to T-Online via PPP
rem
rundll rnaui.dll,RnaDial T-Online per PPP
rem Wait until connection established
rem
echo
echo Press RETURN (INPUT) once connection setup
echo successful. If required, switch to the MS-DOS window first
echo via ALT + TAB.
pause
echo
echo Determine your dynamically assigned IP address
echo This address is needed on the viewer side for connection
echo setup.
netstat –r
echo
echo Please pass the IP address displayed in the first line
echo e.g.: 193.159.44.76 on to your colleague on the viewer side.
echo It is needed there for connection setup.
echo Once this is done, press RETURN (INPUT). If required, switch
echo to the MS-DOS window first via ALT + TAB.
echo
pause
rem Start the remote diagnosis
rem c:\Reachout\Hostmenu.exe
```

If softkey 11 (2nd softkey level of MMC user interface) is not suitable, you can change it in the above files (regie.ini, re_gr.ini, re_uk.ini). For the file c:\Reachout\internet.bat, it is necessary to select “Close” in Windows service mode via “Properties” –> “Program” when terminating.

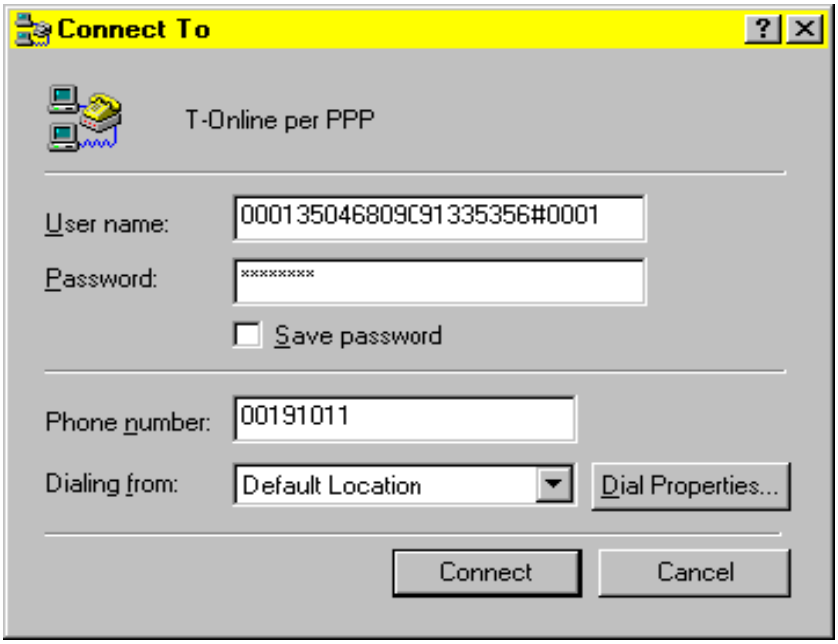
Configuring the host

In the file internet.bat, the entry “rundll rnaui.dll,RnaDial T-Online per PPP” must possibly adapted to the configured long-distance data transmission connection. Example: The dial-up connection has been named “T-Online per PPP”. Start the configuration as follows: “Start-up” → “MMC” → “Options” → “Remote diagnosis”.

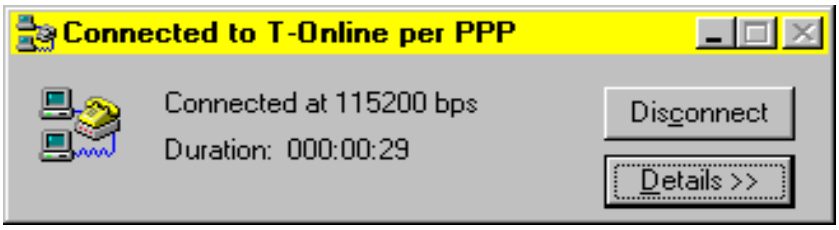
On the MMC 103, start the remote diagnosis configuration menu and select “Internet” as the connection type. Start the remote diagnosis as usual.



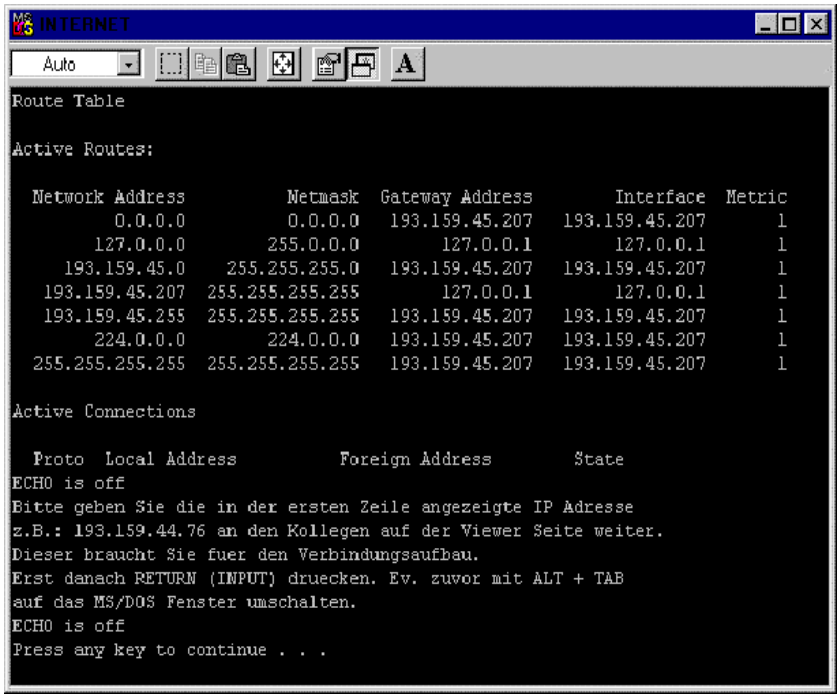
Establish Internet connection



Use softkey 11 “Internet Connection” to establish a connection to the Internet and start remote diagnosis via “Diagnosis” → “Remote Diagnosis”. Select “Connect to” to set up the modem/ISDN connection to the Internet PPP dial-up node. The following window is displayed:



After the T-Online PPP connection has been set up, the local dynamically assigned IP address is determined. With the MMC you can change to the window displayed below entering “Alt” and “Tab” on the full keyboard.



Follow the instructions in the MS-DOS window. Pass the dynamically assigned IP address on to your colleague on the viewer side. The IP address is the last group of digits in the first line (here: 193.159.45.207). With the IP address, your colleagues in the service center can set up access to the control over the viewer.

Starting

Starting the host remote diagnosis on MMC103
Start the host via “Diagnosis” → “Remote Diagnosis”.

Operation on the viewer side

Establish the Internet connection via the entry configured in the long-distance data transmission network. Start the appropriate viewer and connection setup to the host.
Select “Internet” as connection type in the configuration. Enter the above IP address (e.g. 193.159.45.207) at connection setup in the field “Network Name”.

2.3.6 Installing and configuring Windows with gateway (only MMC 102/103)

Installing Windows NT with gateway (only MMC 102/103)

There are only two possibilities of remote diagnosis connection via the Gateway:

1. From PC viewer → gateway host and further from gateway viewer → MMC host (cascading).
   The installation and configuration is similar to the installation of the individual components (viewer → host) of the cascade.
2. With gateway as the RAS server, remote diagnosis directly from PC viewer → MMC host. In this case, the gateway PC must be configured as the RAS server.

In the case of the 2nd solution, the licenses for an additional host and viewer are not required, and the performance is a bit better.

Note

Only the first solution is described in this Documentation.

2.4 Software installation and start-up on the 840D with MMC 103, V4.x and higher

Installing remote diagnosis on MMC 103

The installation of the remote diagnosis must be carried out both on a PC (viewer) and on the control system (host). Before the remote diagnosis is installed, the communication devices (such as modem, network card...) should be installed (see Section 2.3).

For the option “Remote Diagnosis”, the software package “Host 840D Remote Diagnosis” is offered. The companion installation program is used to install the remote diagnosis software on the control system (MM103). During the installation, it is imperative to configure an initial user with administrator rights and an appropriate password. If password protection was enabled on the host, a login should be set up on the host for the user on the viewer side, since this is required for establishing the connection on the viewer PC during the remote diagnosis session.

Note

To start up the remote diagnosis, an MFII keyboard must be connected to the MMC 103.

Delivery form

The remote diagnosis is supplied on a CD-ROM. The CD-ROM contains one folder each with installation disks for the operating systems DOS, Win3.11, Win95/98/NT4.0.

Table 2-1 List of CD-ROM folders for the individual operating systems

Folder	Operating system
’DOS’	DOS
’Win16’	Win3.11
’Win32’	Win95/98, NT4.0

The disks 0 to 4 in the folder ’Win32’ contain the host for the control system and Win32 Viewer. If the control system has no CD-ROM drive, but, instead of that, a floppy disk drive, the individual directories ’Disk 0, 1, 2, 3, 4’ must be copied to 3.5” diskettes prior to the installation from the CD-ROM. If the control system also has no floppy disk drive, the installation can also be carried out via Interlink or via a network.

Note

The CD-ROM contains the host for 840D MMC 102/103 and the host/viewer for Windows 3.1/3.11, Win95/98 and NT4.0 (see also Brief Description).

2.4.1 Installing and configuring remote diagnosis from a CD-ROM drive on a PC/PG via the network

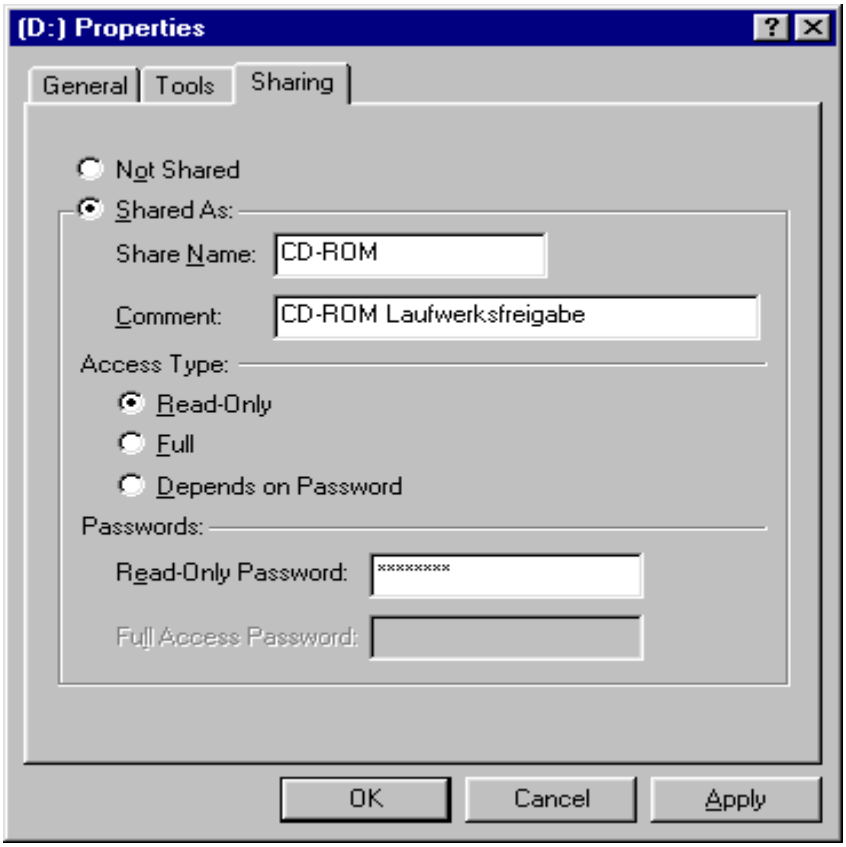
If the MMC 102/103 is already connected to a network, the installation is also possible from the CD-ROM drive of a PC/PG (programming device) of the same network. To do so, the corresponding drive of the PC/PG must be enabled in Windows Service Mode and mapped via the control system's Explorer.

Sequence on the PC/PG

1. Start the Explorer, click with the left mouse button on the CD-ROM drive and choose "Sharing..." from the menu displayed.

Note

If you cannot find the menu entry "Sharing...", select the setting "File" and "Print Sharing..." via "Start" → "Settings" → "Control Panel" → "Network".
In the new dialog box, select the first item "I want to able to give others access to my files."; then click OK to close all open boxes.



2. Settings in the Properties box of the CD-ROM drive:
   A. Select “Shared As”.
   B. Assign a name for the drive.
   C. Assign an access authorization.
   D. If desired assign a password and confirm the box with “OK”.
3. Settings in the Properties box of the CD-ROM drive:

Sequence on the MMC 102/103:

1. Boot the control system in Windows Service Mode.
2. Start the Explorer and select the computer with the CD-ROM drive in the network environment.
3. Click with the left mouse button on the drive with the name “CD-ROM” and choose “Map Network drive...”) from the menu displayed.
4. Click “OK” to confirm the new dialog box.

Note

Make sure that the setting “Reconnect at logon” is not active.
Otherwise, the drive will be connected with each logon.

The remote diagnosis software can now be installed using the newly created drive as described in Section 2.4.3.

2.4.2 Installing and configuring remote diagnosis from a floppy disk drive on a PC/PG via V.24

Sequence

Connect a notebook or programming device (PG) to the X6 interface (COM1) via a zero-modem cable.
Boot the notebook in DOS mode and start the InterServer program (Intersvr.exe).
Switch on the control.
The floppy disk drive will now be detected by the control system as a drive and can be addressed by the control system, e.g. under e:\.
To complete the installation, proceed as with installation via a separate drive (see Section 2.4.3).

2.4.3 Installing and configuring remote diagnosis from a floppy disk drive directly on the control system with MMC 103, V4x and higher

Sequence

Connect MFII keyboard!

1. Switch the control system on. When the text “Starting Windows 95” appears on the screen, press the key “6”. The control’s service menu is overlaid.
2. Select key “4” from the menu.
   ⇒ Start Windows (Service menu)
   You are prompted for your password.
   “passwd:”
   (The password authorizes you to make significant changes in the control. It corresponds to one of the following access levels:
   – System
   – Manufacturer
   – Service).

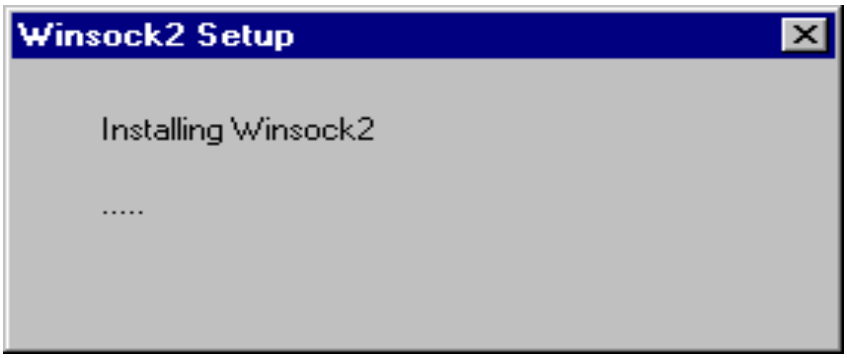
Note

The password must be entered in uppercase letters.

---

3. Another menu is displayed. Now select key “2”.
   ⇒ Start Windows (Changing INI-Files for MMC2)
4. The Windows operating system is started.
   Insert the first diskette (remote diagnosis for WIN32 + DOS Disk0) into the floppy disk drive.
   With V4.x and higher, call the “Start” program menu by pressing “Ctrl” and “Esc” and use the cursor keys to select “Run”.
   Type a:\setup32 at the prompt, if your floppy disk drive is drive “a”.
   Otherwise type the appropriate letter for the drive. (“z” with Interlink). Press “Return” to continue.
5. Installing the remote diagnosis software:
   If Winsock2 is not yet installed on the control system, Winsock2 will be installed prior to the remote diagnosis installation.

   The following box is displayed:



Confirm the next box with “OK”.

The installation is continued.

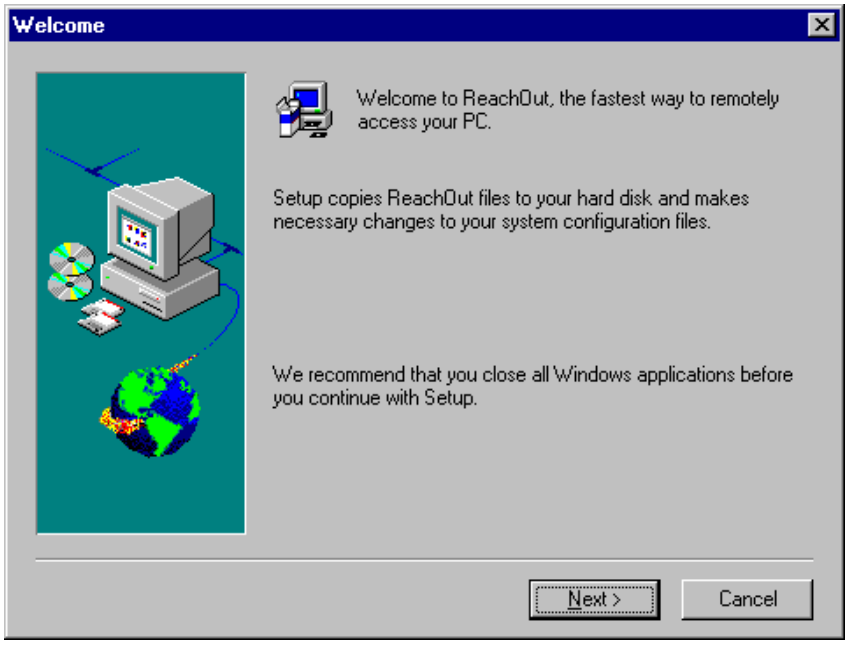
Note

It may possibly take 20 seconds until the next window appears.

5.a) Welcome dialog: Type ’Next’ to continue with setup.

Note

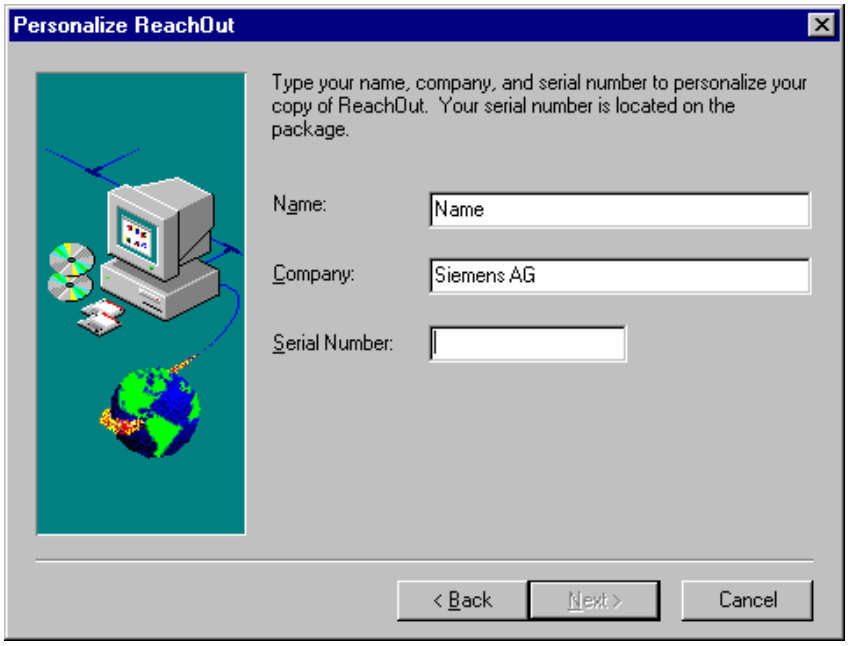
Use the TAB key to change between the different selection fields and press Return to confirm. Select a menu by means of keyboard input of “Alt” and the letter that is underlined in the menu name.



5.b) License Agreement: Read it carefully and click “Yes” to confirm.

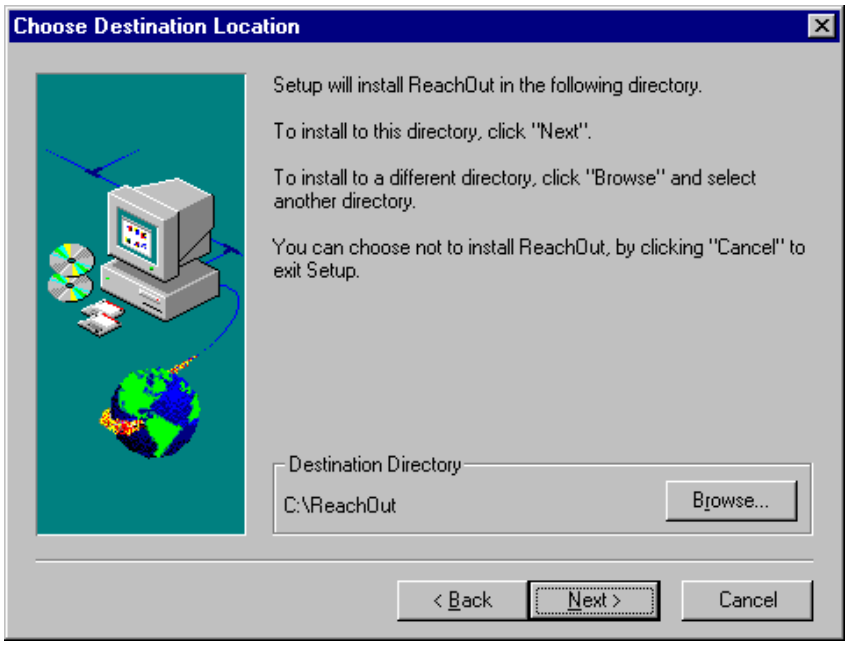
5.c) User information and serial number:

You will be asked to enter end user information and serial number.

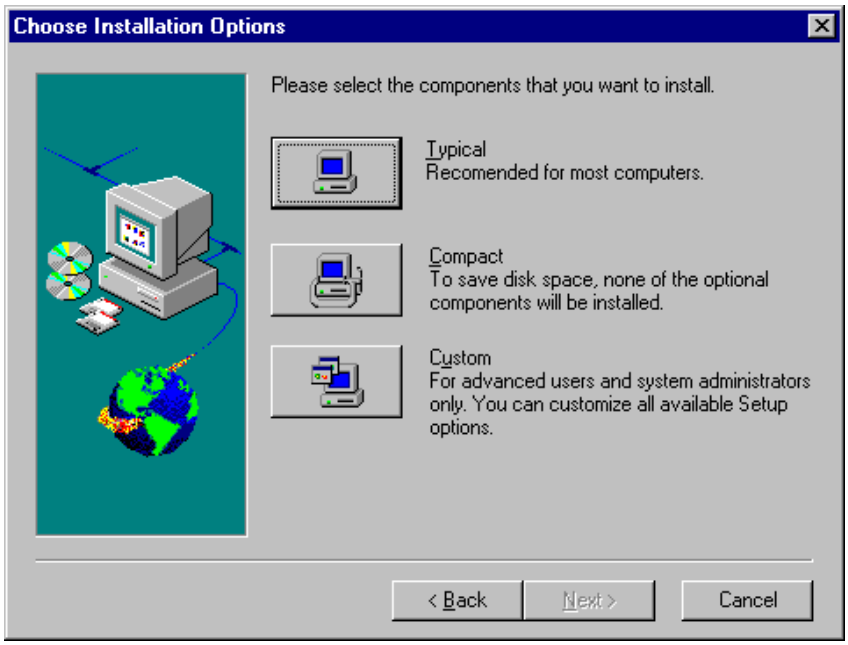


Type the serial number supplied with the remote diagnosis software and choose “Next”.

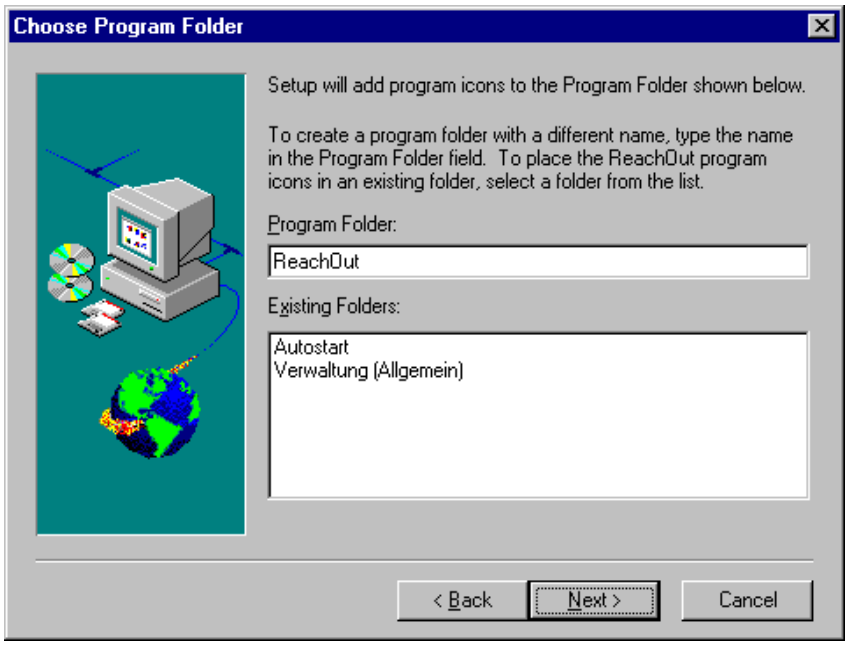
5.d) Path specification:
Choose 'Next' to continue with setup without changing the default directory "C:\ReachOut".



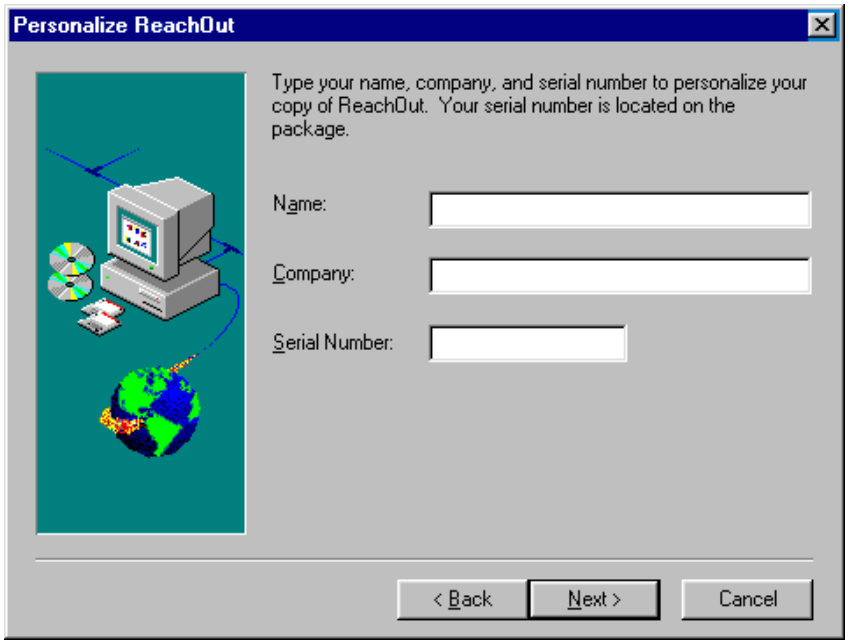
5.e) Select type of installation:
Select 'Typical' for a standard installation.



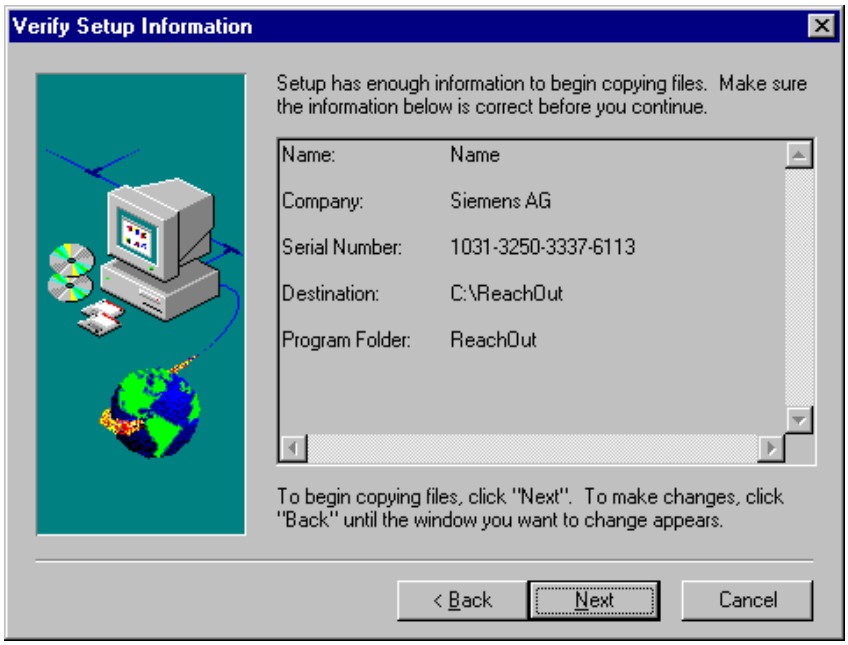
5.f) Choose a program group:
Select a program group where you wish the program icons to be stored and then click 'Next'.



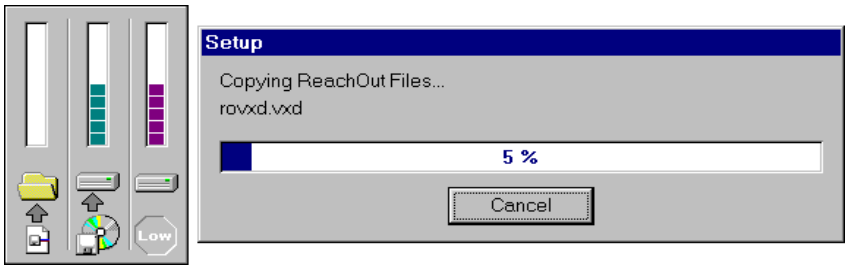
5.g) Enter your personal data:
This dialog box will only appear if the remote diagnosis is reinstalled or installed for the first time.
Enter your personal data and the serial number of the remote diagnosis and click 'Next' to confirm.



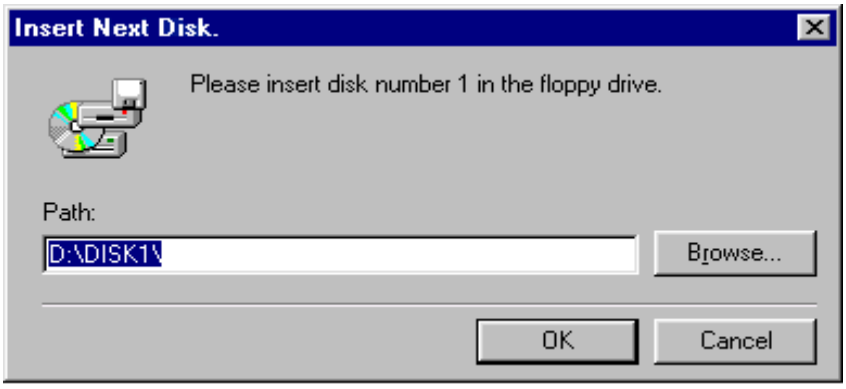
5.h) Verify the setup information:
If the information is correct, click 'Next' to continue with the setup; otherwise, click 'Back' and correct the information.



5.i) Copy the data onto the MMC:
The remote diagnosis directory is created, and the files are copied.

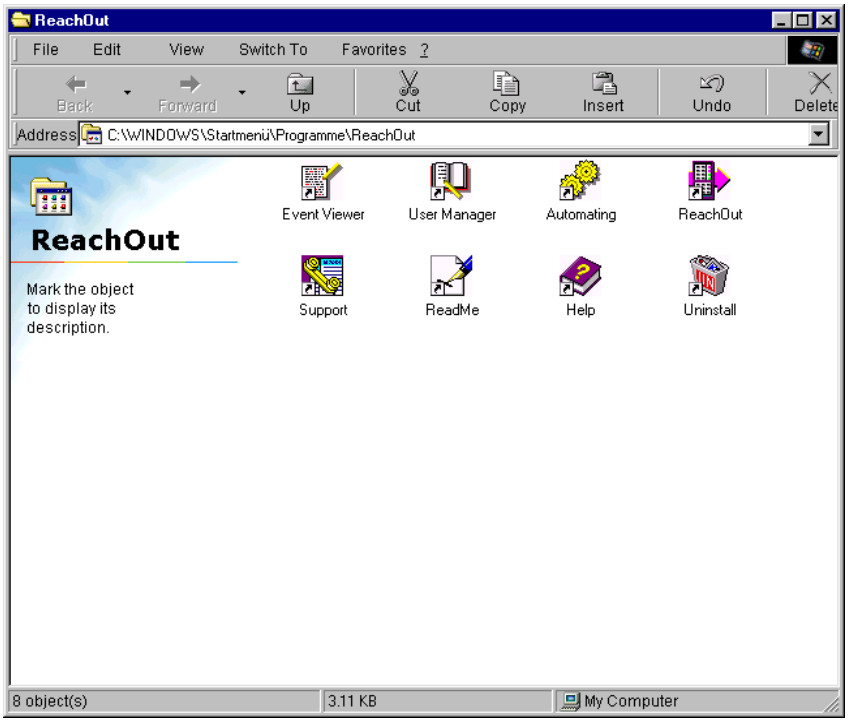


After the first diskette (Disk0), the setup program will ask you to insert the Disk#1.



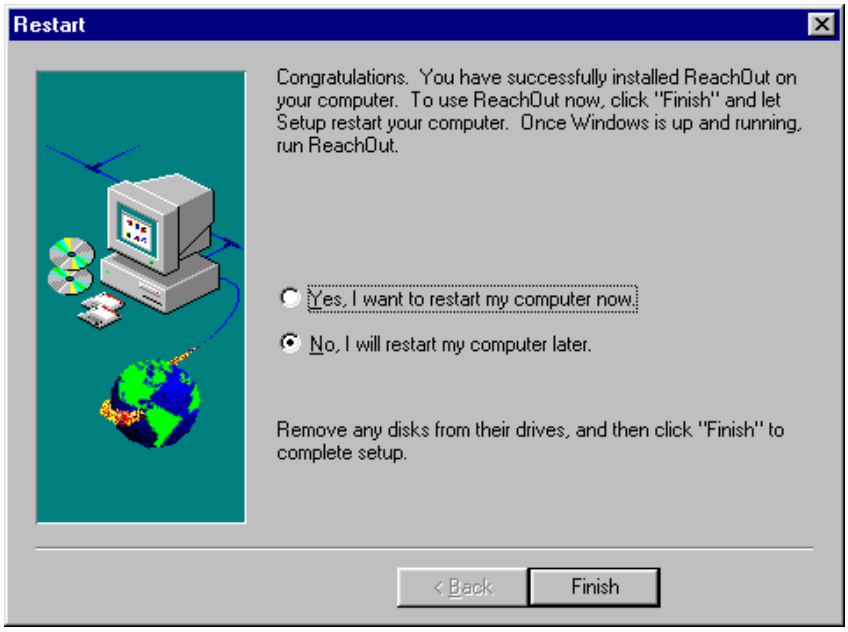
Insert the disk into the drive and click “OK”.
repeat this procedure for Disks #2, #3 and #4.

5.j) End of installation:
At the end of installation, first the “ReachOut” window with all program icons is opened.



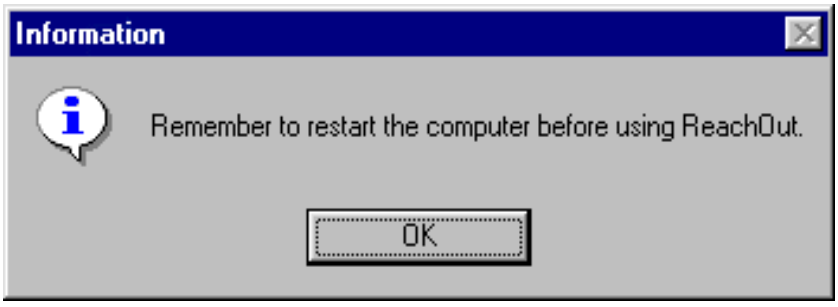
The you will be asked whether you wish to restart Windows immediately.

If no TCP/IP has been installed on the MMC, an information is displayed after installation of the remote diagnosis software that remote diagnosis requires WinSock2 Patch for the TCP/IP functionality in case a network connection is to be added to the computer at a later date. The window must be confirmed with “OK”. Otherwise, there are no other consequences regarding the functionality of the remote diagnosis. Immediately afterwards, the computer asks you if you want to restart Windows immediately.



Note

From the dialog box “Restart”, choose the option “No, I will restart my computer later” and quit the box with “Finish”.
Confirm the next following message box “Information” with “OK”, shut down the MMC103 V4.x using the start menu “Shut down” and by selecting the option “Shut down Computer” in the next following box. Confirm the two next following queries with “Y”.



Before you can use the remote diagnosis, you must configure it according to the description in Section 2.4.3.

2.4.4 Starting up the host on the MMC 103, V4.x and higher

A connection between viewer and host can only be established if the host is made ready to receive after starting the remote diagnosis on the MMC and then selecting and correctly configuring the appropriate communication devices.

To this aim, proceed as follows:

Sequence

1. Switch the control system on. When the text “Starting Windows 95” appears on the screen, press the key “6”. The control’s service menu is overlaid.
2. Select key “4” from the menu.
   ==> Start Windows (Service menu)
   You are prompted for your password.
   “passwd:”
   (The password authorizes you to make significant changes in the control. It corresponds to one of the following access levels:
   – System
   – Manufacturer
   – Service).

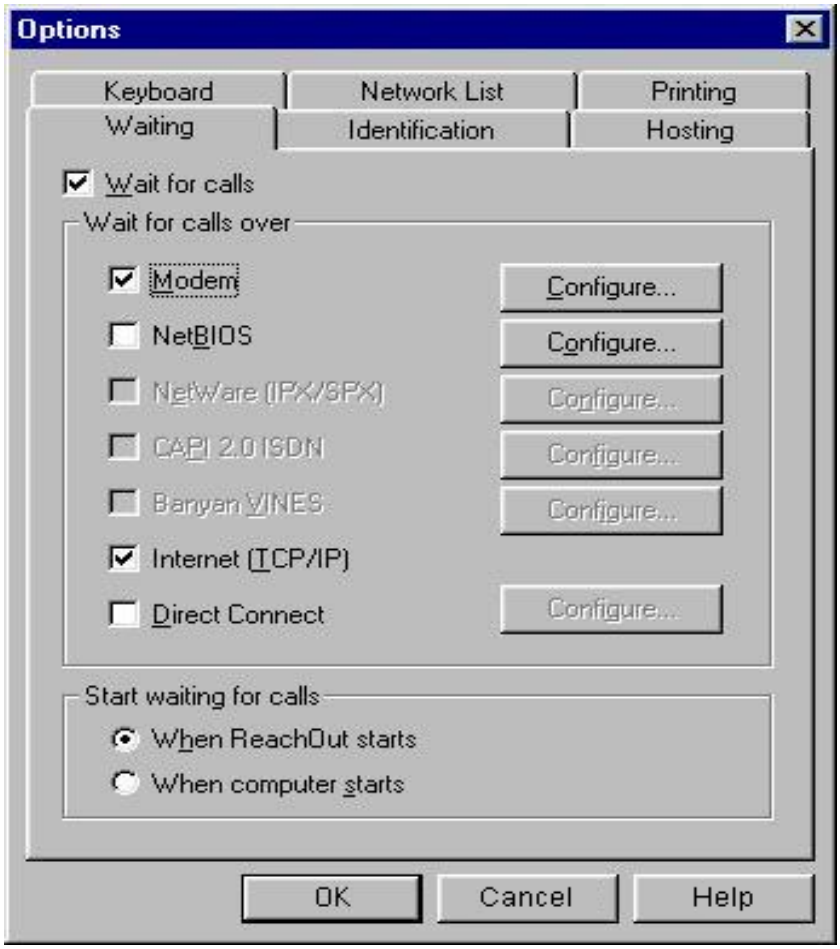
Note

The password must be entered in uppercase letters.

---

3. Another menu is displayed. Now select key “2”.
   ⇒ Start Windows (Changing INI-Files for MMC2)
4. The Windows operating system is started.
5. If the remote diagnosis was not started automatically, start it by clicking on the following icon:

6. Select the menu option “Configure/Options...”.



7. Make the host ready to receive:
   Select the option “Wait for Calls”.

8. Choose a communication device:
   Select the used communication device and, if necessary, choose “Configure” to adapt the device settings.

Note

In the “Options” dialog box, only devices can be selected, which have already been installed on the MMC. For this reason, first install and configure the communication devices to be used (see Section 2.2) and the install and configure the remote diagnosis.

---

9. Time when the host is ready to receive:
   For “Start waiting for calls”, choose the option “When ReachOut starts” to make sure that the host is ready to receive after starting the remote diagnosis.

2.4.5 Uninstalling the remote diagnosis software on MMC 103, V4.x and higher

MMC 103 V4.x and higher

Use “Changing Environment for MMC2” to change to Windows Service Mode.
Start the Uninstall procedure via “Start” (Ctrl + Esc) –> “Programs” –> “ReachOut” –> “Uninstall” –> “Yes”. The question “... Delete all remaining files ...” may be displayed. Respond with “Yes”.
Shut down Windows service mode via “Start” (Ctrl + Esc) –> “Shut Down” –> “Shut down the Computer?” –> “Yes”.
Save the environment with:
“Save Windows Environment for ...” –> “Yes” –> “1.” –>
“2” Windows (changing...)
Save the environment again with:
“Start” (Ctrl + Esc) –> “Shut Down” –> “Shut down the computer?” –> “Yes” –>
“Save Windows Environment for ...” –> “Yes” –> “1” –>
“2” Windows (changing...)

All remote diagnosis files and all entries in the SYSTEM.INI file are removed.

2.5 Software installation and start-up on the 840D with MMC 102/103 V3.x

2.5.1 Installing and configuring remote diagnosis (Win16 version) from a floppy disk drive directly on the control system with MMC 103/103, V3x

Sequence

Connect MFII keyboard!

1. Switch the control system on. Press key “6” when the text “Starting Windows 95” (or with Software Version 3.x “Starting MS-DOS”) is displayed on the screen. The control’s service menu is overlaid.
2. Select key “4” from the menu.
   ⇒ Start Windows (Service menu)
   You are prompted for your password.
   “passwd:”
   (The password authorizes you to make significant changes in the control. It corresponds to one of the following access levels:
   – System
   – Manufacturer
   – Service).

Note

The password must be entered in uppercase letters.

---

3. Another menu is displayed. Now select key “2”.
   ⇒ Start Windows (change ini Files)
4. The Windows operating system is started.
   Insert the first diskette (remote diagnosis for WIN + DOS Disk 12) into the floppy disk drive.
   With V4.x call the “Start” program menu by pressing “Ctrl” and “Esc” and use the cursor keys to select “Run”.
   With V3.x call the “File” program menu by pressing “ALT” and “F” and use the cursor keys to select “Run”.
   Type a:\install at the prompt, if your floppy disk drive is drive “a”. Otherwise type the appropriate letter for the drive. Press “Return” to continue.
5. Installing the remote diagnosis software:

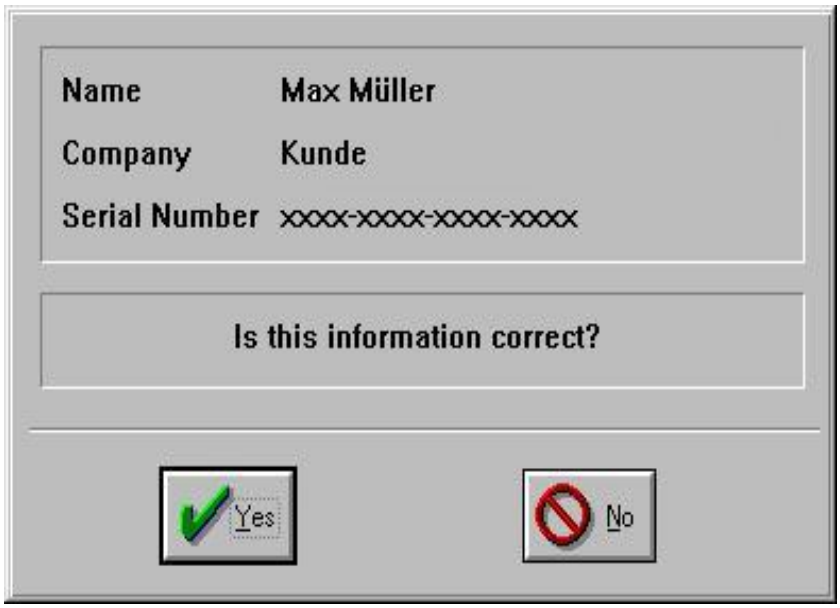
Note

Use the TAB key to change between the different selection fields and press Return to confirm. Select a menu by means of keyboard input of “Alt” and the letter that is underlined in the menu name.

5.a) User information, type of installation and path

This display prompts you for end user information and serial number input.

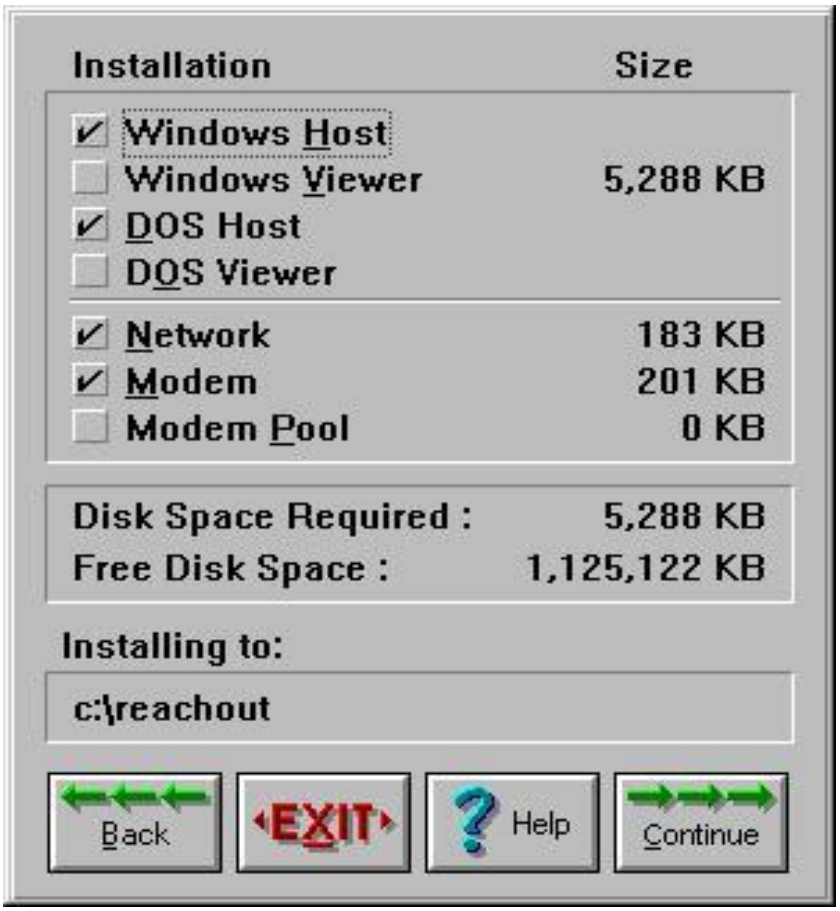
Enter the serial number supplied with the remote diagnosis software and select “Continue”.



Confirm with “Yes”.

Enter the directory the software is to be installed in. The standard setting is the c:\reachout directory, this is the only directory that the installation software copies and handles data in. This directory must NOT be changed! Select “Custom”.

5.b) Selecting the software components to be installed.

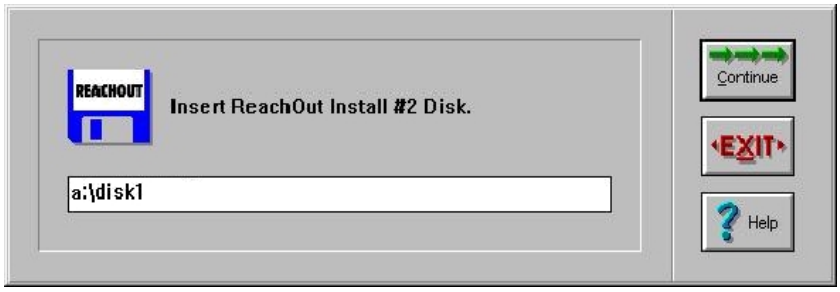


On the SINUMERIK 840D MMC 102/103, select the Windows and DOS host only. Deactivate the “Modem Pool” component.

5.c) Copying data to the MMC

When the settings are made in the “Custom” display, the remote diagnosis directory is created on the MMC and the files are copied.

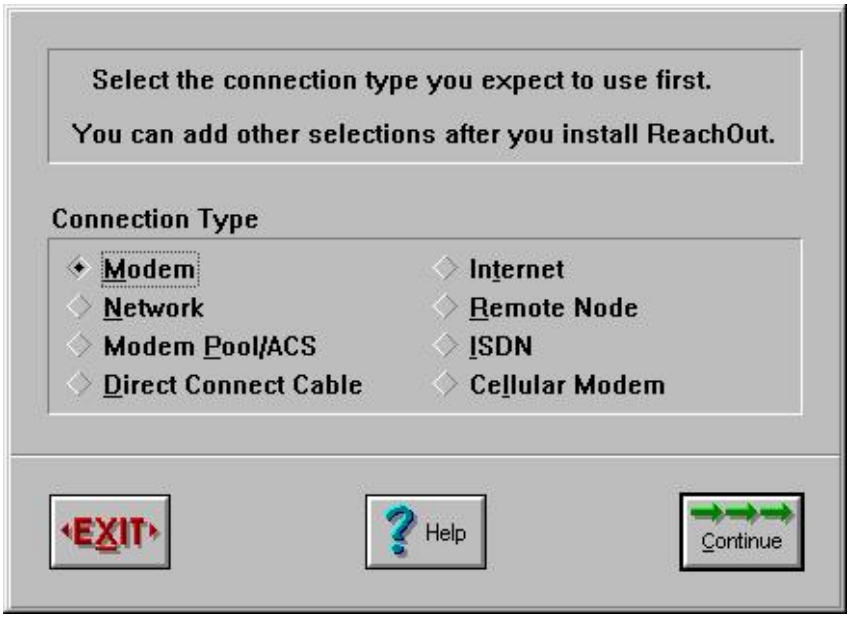
The setup program prompts you to insert #2 Disk.



Insert #2 Disk into the drive, type “disk2” instead of “disk1” and click on “Continue”. Repeat this procedure for #3 Disks.

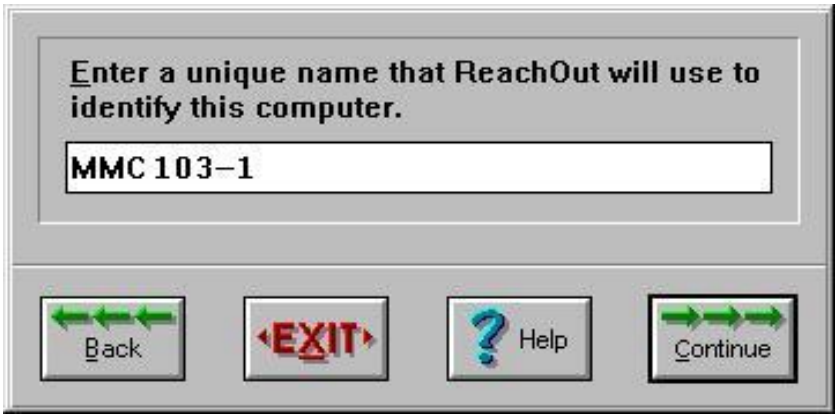
Wait until all files have been copied.

5.d) Configuring the type of connection



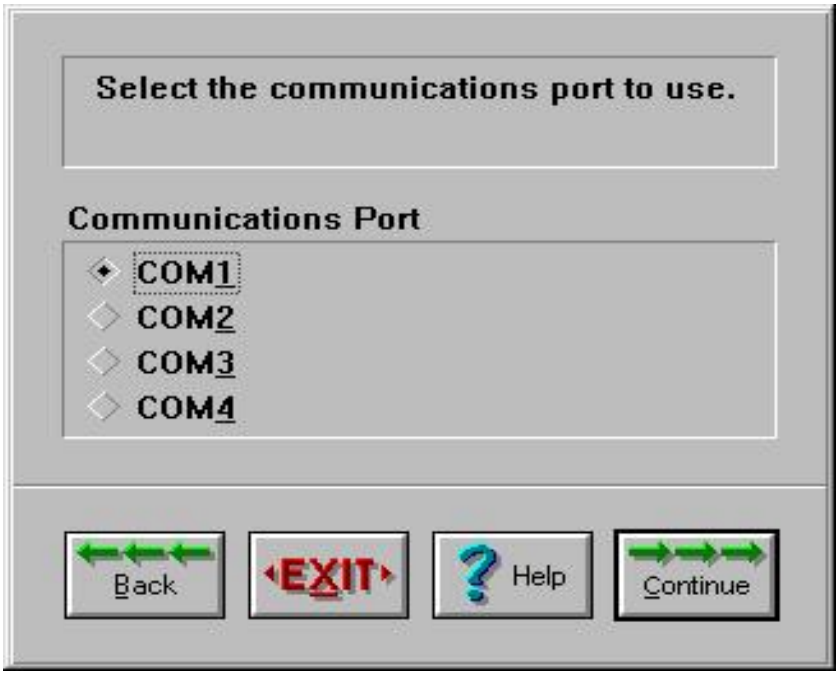
Select the connection type. The default setting is “Modem”. If the connection is to be set up via ISDN, a network or the Internet, please go to the appropriate page in this documentation.

5.e) Host identification name



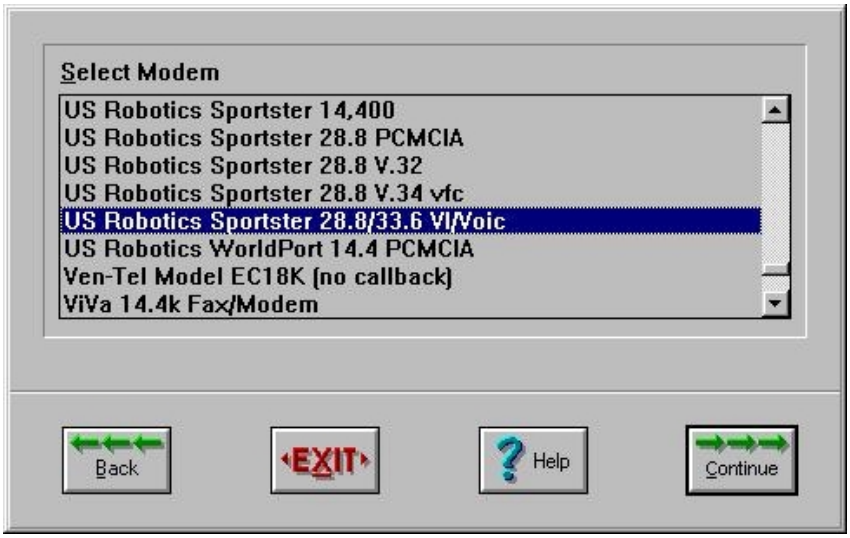
Allocate an unambiguous identification name for the remote diagnosis. The name is displayed on the PC (viewer) at connection setup. If several MMC 102/103’s are installed at a customer’s, it is advisable to use unambiguous names (e.g. stocktaking number, machine number etc.).

5.f) Communications port



Check the checkbox with the number of the interface your modem is connected to. On the MMC 102/103 COM1 corresponds to connector X6, COM2 corresponds to connector X7.

5.g) Modem type



Select the modem that is connected. If it is not listed, select a similar one or the “Hayes AT compatible” modem driver. The modem configuration can be changed or expanded at any time later.

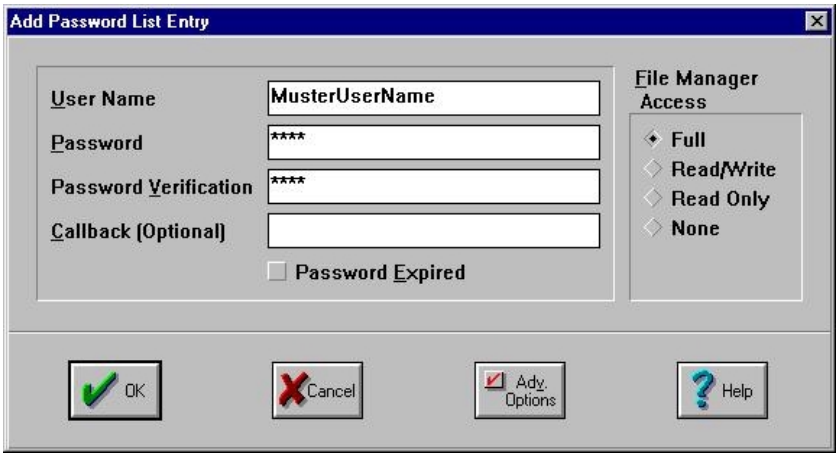
5.h) Automatic start

You are prompted whether the remote diagnosis is to be included in the Windows autostart group. Reply that you don't want this to happen. The remote diagnosis must be activated separately via menu commands specifically set up for this purpose. You can read in the Tips_x.wri file how to start the remote diagnosis automatically.

5.i) Password and user

A user and password can be configured at installation. The operator on the viewer side has to know this password as it is required for the remote diagnosis session on the viewer PC after connection setup.

Confirm with "OK".



Under “Callback” you can make optional settings, e.g. if you want to configure a callback number for security reasons. (See Tips_x.wri file.)

5.j) End of installation

Note

At the end of the installation procedure for remote diagnosis, the system asks you whether you want to restart Windows. Select “Return to Windows” and shutdown the MMC 103 with V4.x via the start menu “Shut down” and “Shut down computer”, or on the MMC 102 with V3.6 via “File” in the Program Manager.

6. After Windows has been shutdown, the following prompt is displayed: “Save Windows Environment for Next MMC start [Y, N]?”

!

Important

Respond to the prompt “Save Windows Environment for Next MMC start [Y, N]?” by pressing “Y”.

Another question is displayed:
“...overwrite them with the current one ?”
Respond with “1” (Overwrite...). You can start Windows again or boot the control in MMC mode. (Alternatively continue with point 7.)
To be on the safe side, press “2” (Start Windows) again.
Shut down Windows again via “Start” and “Shut Down”. Respond to both queries as described above. The backup is saved a second time and the configurations are also saved should an error occur.

7. Switch off the control.
8. Start the control.
9. Should an error message (the COM port used by the remote diagnosis is already assigned) appear when starting the remote diagnosis, please proceed as follows:
   Start the Services area application.
   First activate the “Interface” softkey, then the vertical softkey “V.24”.
   Change the option “Interface: COM1” into “Interface: none”.
   You can use the interface COM1 (on the control X6).
   Activate the vertical softkey “Programming unit”.
   Change the option “Interface :COM2” into “Interface: none”.
   You can use the interface COM2 (on the control X7).
   The changes are immediately effective unless a V.24 job is being processed; in that case they are valid after the V.24 job is completed.
   Enable the interface used by the remote diagnosis. If necessary, you may have to undo the procedure.
10. The installation procedure for remote diagnosis is successfully completed.

2.5.2 Uninstalling the remote diagnosis software on MMC 102/103

MMC 102/103 with V3.x

Use “Changing Environment for MMC2” to change to Windows Service Mode.

Start the Uninstall procedure via “ReachOut” → “Uninstall” → “Yes” in the Program Manager and respond to “Restart Windows now?” with “No”.
Exit Windows Service Mode in the Program Manager via “File” → “Exit Windows” → “Yes”.
Save the modified INI files with:
“Save INI-Files for...” → “Y” → “2” → “5” Return to Main Menue
Remove the remaining files (only required with remote diagnosis V3.X) as follows:
“3” DOS Shell
“cd windows”
“del_rch.bat”
The message “Batch File missing” can be ignored.

2.6 Start-up and software installation on the MMC 100.2

Installing the software on the MMC 100.2

The MMC 100.2 has no free memory medium on which the software could be installed. For this reason, the “ReachOut” software is supplied pre-installed on a PC card. This card need only be plugged into the MMC 100.2 and configured. The installation routine includes setting the modem and configuring the user with the relevant password.

Note

When installing the remote diagnosis software, an MFII keyboard can be connected to the MMC 100.2. to facilitate data entry.

2.6.1 Installing the PC card on the MMC 100.2

Interfaces on MMC 100.2

The ports for the interfaces are located on the rear panel of the MMC 100.2. To install the remote diagnosis software, two serial V.24 interface ports (COM 1 and COM 2) are required, one for the modem and the other to connect the PC card.

Connecting the PC card

The slot for the PC card is activated prior to start-up. A row of coding switches are located above the slot for the PC card.

1. Set microswitch 0 to position 1. All other switches must be set to 0.
2. Insert the PC card with the openings upwards into the intended slot.
3. Connect the cable between the modem and MMC 100.2 to one of the two serial ports on the MMC (COM 1 or COM 2).

2.6.2 Installing and configuring the software

The software “ReachOut” required to operate the remote diagnosis is completely installed on the PC card. It need only be configured to match the user’s system before the software is used for the first time.

Call configuration mode

The PC card is installed in the MMC 100.2 and the modem connected. Please proceed as follows to configure the card:

1. Switch the control system on.
2. Press key “**6**” as soon as the Copyright line containing the serial number is displayed on the screen. The configuration menu appears.

This menu offers you a selection of different actions and you can select the function of your choice with keys “**1**” to “**4**”.

Key	Description of configuration menu
	ReachOut teleservice is enabled at Power On The current status of remote diagnosis is displayed (corresponds to key “4”)
1:	**Update PC–IN (internal EPROM, COM 1)** Updating the control software via COM 1 (not required for remote diagnosis)
2:	**Update PC–IN (internal EPROM, COM 2)** Updating the control software via COM 1 (not required for remote diagnosis)
3:	**ReachOut Setup** Call the configuration software for remote diagnosis
4:	**Disable Reachout at Power On** Remote diagnosis will **not** be started with next Power ON

3. Press key “**3**” to go to the menu for configuring the PC card.

Note

The first line prior to the configuration menus contains the information whether remote diagnosis is enabled or not. If, for example, the line “ReachOut is enabled at Power On” is displayed before the menu, the **remote diagnosis** is enabled with each Power ON (key “4”).
In this case, the configuration menus only offer “Disable ReachOut at Power On” (key “5”) to disable the remote diagnosis with the next start.

Setup ReachOut

Configure the existing hardware in the ReachOut setup menu and define your settings for operation. For initial start-up of the remote diagnosis, you must enter the following settings:

- Select the COM port of the modem
- Set the modem type
- Select the modem transfer rate
- Select the phone line type
- Enter the PCs for connections
- Define the password and access rights
- Enter a telephone number for the Call Back function
- Reboot the system

Selecting menu items

The ReachOut menus for the MMC 100.2 are selected with the arrow keys on the operator panel. Each selection is displayed on a yellow background. The selected item can be activated by pressing the Enter key.

Note

If you have not connected an MMII keyboard to the MMC 100.2, you can quit the menus using the keys "**X**" (E**x**it) or "**Cancel**". On the operator panel, the "**Esc**" key is the "**Cancel**" key. The function keys "**F1**" to "**F8**" are activated via the horizontal keys.

Help

The main "Help" menu contains a brief description of each menu. You can also display direct help for any subject by pressing function key "**F1**". Use the arrow keys to page through the help topics.

Select the COM port

The MMC 100.2 has two COM ports. Check first which of the two ports the modem is connected to. You can set the appropriate COM port in the menu "Comm Port".

1. Select "Preferences" –> "Communication Settings" in the ReachOut menu bar. A new form with a new menu is opened in which you can enter the "Communication Settings".
2. In this new window, select "Comm Settings" –> "Comm Port" to display a selection list.
3. Select either serial interface COM 1 or COM 2 depending on the modem connection.
4. Confirm your selection with the Enter key. The menu is closed.

Note

Both the configuration for **modem type**, **data transfer rate** and **connection type** and for Comm Port is carried out in the screen form “Communication Settings” .

Modem type

ReachOut offers a selection list of typical modems to choose from in the screen form for the Communication Settings in the menu “Modem Select”.

1. Select “Comm Settings” –> “Modem Select” in the “Communication Settings” screenform. A selection list of modems is displayed.
2. Select the connected modem and appropriate transfer rate in the “Modem Table”.

Note

If the modem you are using is not contained in the selection list, select “A Hayes Compatible” modem with the data transfer rate of the modem you are using. If you have selected the modem type, the software expects that a modem is installed and turned on.

3. Confirm your selection with the Enter key. The menu is closed.

Data transfer rate

ReachOut automatically sets the highest possible transfer rate for the selected modem. You can change the current transfer rate in the “Data Speed” menu in the “Communications Settings” form.

1. Select “Comm Settings” –> “Data Speed”. A selection list of transfer rates is displayed.
2. Use the arrow keys to select a transfer rate.
3. Confirm your selection by pressing the Enter key. The menu is closed.

Note

When modems from different manufacturers are used, the same data transfer rate must be set for both viewer and host; otherwise, possibly no connection can be established.

Phone line type

Set the modem to tone or pulse dialing in the “Dialing Prefix” menu in the Communication Settings screenform.

1. Select menu items “Comm Settings” –> “Dialing Prefix” to call the “Enter Phone Dialog Prefix” menu.
2. Enter the prefix for the desired dialing signal type in the text line.
   ATDT: Tone dialing
   ATDP: Pulse dialing

Note

If you use the Call Back option on private branch exchanges, the string for tone dialing must be **ATX3DT**, with pulse dialing **ATX3DP** to ensure that the dialing tone is ignored. For more detailed information on the prefix **ATXn**, please refer to your Modem Manual.

You can enter additional parameters for PBX systems. Descriptions of these can be found directly in the “Enter Phone Dialog Prefix” menu.

3. Confirm your selection with the Enter key. The menu is closed.
4. Press the “**Cancel**” or “**Esc**” key to exit the Communication Settings screenform. The “Save Current Settings” option window is displayed.
5. Press key “**Y**” to save your settings and return to the main screenform of ReachOut.

Making a connection to a PC

In the menu “Phone Book”, select the PCs with which you want to establish connections.

1. Select “Preferences” –> “Edit Phone book” in the ReachOut menu bar. The “Call-Out Phone book” screen form opens.
2. Press key “**A**” (**A**dd New Entry) to enter a new phone connection.
3. Enter a “Description” and “Phone Number”. You can also fill in the “User Name” and “Password” fields if you wish.
4. Confirm you inputs by pressing the Enter key.
5. Repeat the operation if you wish to enter further phone connections.
6. Press key “**X**” (E**x**it) to close the menu.

Note

Use the “E” key (Edit Entry) to modify existing entries. Use the “D” key (Delete Entry) to delete existing entries.

Password and Call Back

Use the “Edit Passwords” menu to define which access rights the user of the viewer PC will have when contacting the host. Each user can also be assigned his/her own password. Only users entered in the list can activate the remote diagnosis function.

1. Select “Security” –> “Edit Passwords” in the ReachOut menu bar. The screenform headed “Password List for this computer when it is a host” is then displayed.
2. Press key “**A**” (**A**dd Password) to record a new user with password. The “Add New Password” screenform is opened.
3. Enter the name and password in text lines “User Name” and “Password”.

Select menu item “File Access” to display a selection list in which you can assign varying levels of access rights to the user.

4. If you wish to grant full access rights to the user, select “Full Access” in the list. Select the appropriate entry for restricted rights.

Call Back

For the Call Back mode, you can enter the phone numbers of the viewer PC. If the host dialed from the viewer PC, no direct connection is established with active Call Back. The host will call back the logged-on viewer PC only with the stored phone number. The remote diagnosis system is not activated until a connection has been successfully set up with Call Back.

Call Back ensures that computers without access authorization cannot perform remote diagnosis. Only viewer PCs and users entered in the host with their phone numbers will be accepted after successful Call Back.

5. Type the phone number of the viewer PC when working with Call Back.
6. Confirm your entries by pressing the “Enter” key and return to the main screenform of ReachOut.

Defining options for log-on

The logging on process for authorized users can be configured in the “Password Options” menu. You can, for example, define whether passwords are needed or set the maximum number of log-on attempts.

1. Select “Security” –> “Password Options” in the ReachOut menu bar. The “Password Options Menu” screenform is opened.
2. You can activate or deactivate the relevant configuration by pressing keys “**1**” to “**4**”.
3. Press the “**Cancel**” or “**Esc**” key to go to the ReachOut main screenform.

Caution

The other setting options may not be used (e.g. “Reboot upon Disconnect”)!

Completing the configuration

The basic configuration of ReachOut is complete once you have made these settings. When the system is restarted next time, it will be configured for the remote diagnosis.

1. Press the "**Cancel**" or "**Esc**" key to exit ReachOut. You return to the configuration submenu of ReachOut.

Key	Description of configuration submenu
	ReachOut teleservice Enabled at Power On
0:	**Exit** Exit configuration menu without saving settings
1:	**Save Setup, Reboot** Save new configuration and restart control system
3:	**ReachOut Setup** Return to Setup menu
5:	**Disable Reachout at Power On** ReachOut will **not** be started when the power is next switched on.

Note

Please make sure that the modem is turned on when the remote diagnosis is started.

2. Press key "**1**" to write the configuration to the PC card and restart. After restart, the remote diagnosis will be available on the MMC 100.2.

Reboot

The configuration menu of ReachOut will not be displayed after rebooting. The PC card is configured and ReachOut will be activated and ready for remote diagnosis every time the control is started.

Note

After an update, the control software of ReachOut must be called again with "3". Close the configuration software without making any changes. As soon as the configuration menu is displayed, press key "0" and close the configuration menu again. A new menu is displayed which you can close by pressing key "1" (Save Setup, Reboot).
Updating the control software might have altered the settings in autoexec.bat and config.sys. Calling the configuration submenu of ReachOut again will update the settings on the PC card.

2.6.3 Activating remote diagnosis later

If the remote diagnosis software is not activated on the host, you can also activate the software after start. The function can be called as a softkey on the MMC 100.2 control.

1. Press softkey “Diagnosis”.
2. Press softkey “Remote diagnosis”. The message “Starting remote diagnosis, restarting control unit; continue with OK” appears.
3. Confirm restart of the control unit by selecting “OK”. ReachOut is activated and the control unit restarted. The NCK is not affected by this process and is not restarted.

2.6.4 Quitting remote diagnosis

Depending on the configuration mode, the function can be quitted as follows:

Mode “Disable ReachOut at Power On” is active:

- HW-Reset of MMC

Mode “Disable ReachOut at Power On” is active:

- Turn off the control system (with POWER ON, remote diagnosis is immediately active in this mode)

2.6.5 Uninstalling the remote diagnosis

Sequence

To uninstall the remote diagnosis software from the MMC 100.2, proceed as follows:

1. Remove the PC card and
2. turn the 0 switch in the card slot to “0”.

2.7 Installing the remote diagnosis software on the PC (viewer)

Preparation

Before you install the remote diagnosis software on your PC, observe the following:

- Determine which viewer version is required. If an MMC 100.2 is used as the host, the DOS version of the viewer must be installed.
- Erase every other remote control software from the PC.
- If you are using a modem, make sure that it is configured correctly and connected to the telephone system.
- Make a note of the modem type, model number and transmission rate.
- Locate the V.24 interface (e.g. COM1 or COM2) to which the modem is connected.
- Determine under which operating system your viewer is operated.

2.7.1 Installing viewer for Windows 95/98/NT4.0

Installation

The software package for the remote diagnosis on the PC is called “Remote diagnosis Viewer 840D”. The supplied installation batch file “SetupW32.bat” can be used to set up the tool on your PC under Windows as follows:

1. Insert the CD into the CD-ROM drive.
2. Select “Run” in the “File” menu in the Program Manager.
3. Type the command line “E:\SetupW32.bat” (“E” is the drive letter for the CD-ROM which can vary).
4. Confirm with “OK”.
5. Follow the further instructions to complete the installation. The steps are the same as required for installation on the control system (see **Section 2.4.1**).

Click on the Help symbol if you need assistance during installation. The help function is available at all times.

Note

To install the remote diagnosis under **Windows NT4.0**, administrator rights are required.

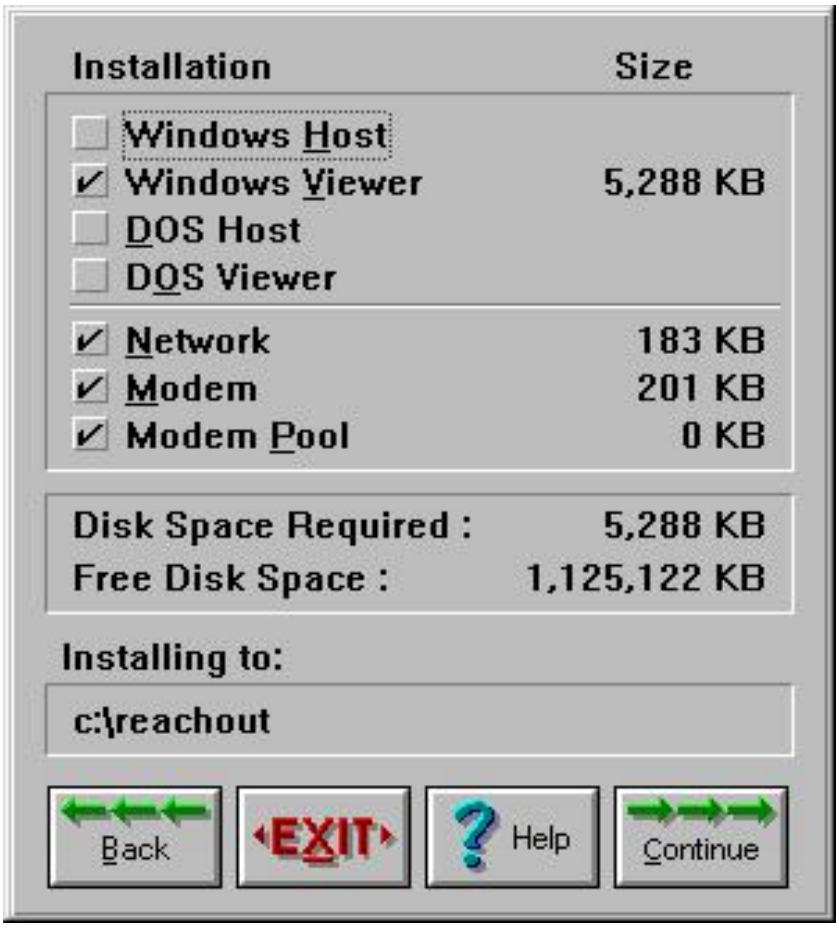
2.7.2 Installing viewer for Windows 3.1, Windows for Workgroups 3.11

Installation

The software package for the remote diagnosis on the PC is called “Remote diagnosis Viewer 840D”. The supplied installation program Install.exe can be used to set up the tool from the CD-ROM drive on the PC under Windows 3.x as follows:

1. Insert the CD into the CD-ROM drive.
2. Select “Run” in the “File” menu in the Program Manager.
3. Type the command line “E:\SetupW32.bat” (“E” is the drive letter for the CD-ROM which can vary).
4. Confirm with “OK”.
5. Follow the further instructions to complete the installation. The steps correspond to the steps for installation on the control.

Click on the Help symbol if you need assistance during installation. The help function is available at all times.



Set the viewer options as displayed in the screenshot.

2.7.3 Installing viewer for MS-DOS (required if MMC 100.2 is used as the host)

Installation

Use the installation program installd.exe to install the DOS version on the viewer PC:

1. Insert the CD-ROM into the CD-ROM drive.
2. Under Windows 95, 98, 3.11, NT, change to the MS-DOS prompt to open a DOS window.
3. Change to the directory Dos/Disk1 of the CD-ROM and type “installd.exe”. The installation program for the remote diagnosis for DOS is started.

Note

For the remote diagnosis of the viewer PC with MMC 100.2, none of the following Windows variants Windows 3.11, 95, 98 NT of the viewer version of the remote diagnosis can be used. Only the DOS variant may be installed. In the above mentioned Windows variants, it can be started in a DOS box.
Background: MMC 100.2 use the DOS operating system.

Installation procedure

You are guided through the installation routine by a menu-driven menu. Please proceed as follows for a standard installation:

1. Enter your name, company name and serial number in the first screenform of “ReachOut Dos Install”. Confirm your inputs by pressing function key “**F10**”. Another prompt concerning the correctness of entries is displayed.
2. Confirm your inputs again with “**F10**” if you are certain that they are correct. A new screen form “Select the correct Path Names below” is displayed.

By default, the remote diagnosis software is installed on C:\reachout. ReachOut stores all data in this path during operation.

3. Confirm the standard path name by pressing function key “**F10**”. Screenform “**F**ull (complete) or **C**ustom Installation?” is displayed.
4. Press key “**C**” to customize the installation. Screenform “Install for Host/Viewer?” then appears.
5. Press key “**3**” to install the software for the viewer. Screenform “Install for which environment?” is displayed.
6. Press key “**2**” to install only the DOS version of ReachOut. Screenform “Select Communications” appears.
7. Press key “**3**” to install communication via modem. The screenform closes and all necessary files are copied onto your computer by the installation routine.
8. During the installation, you will be asked to insert diskette 2 (disk 13). Insert diskette 2 (disk 13) in drive A: and confirm by pressing the “**Enter**” key. Do the same with diskette 3 (disk 14).

Note

If you install the program from the CD-ROM, specify the new directory on the CD-ROM as the source when changing the diskette, e.g. with diskette 2, type d:\disk2. Drive letter d: stands for the CD-ROM drive (may vary).

After all files have been copied onto the hard disk, all you need to do is configure the connection. The screenform headed “Type of Modem Support” appears.

9. Press key “**1**” to configure a connection with an analog modem. Screenform “Enter Computer Name” is displayed.
10. Enter the name of the viewer PC and confirm with “**Enter**”. This name will be used thereafter for remote diagnosis with ReachOut. Screenform “Communication Settings” is displayed.
11. Configure the modem you have installed in the “Comm Settings” menu.

Note

You will find a description of the main configuring steps under configuration of the modem for the host in Section 2.3.3 Configure modem, under paragraph headings “Select COM port ”, “Modem type”, “Data transfer rate” and “Phone line type”.

12. Select “File” –> “Save Configuration” when you have finished configuring the modem and connection.
13. Press the “Esc” key to close the “Communication Settings” menu. Message box “ReachOut Installation Complete” is displayed.
14. The installation of the remote diagnosis for the viewer is completed.

2.8 Connection setup

2.8.1 General on connection setup with viewer for Win32 (Windows 95/98/NT)

Connection setup with several connections

The Windows 32 variant supports connection setup with several computers to or from a single PC.

- A single viewer PC can be connected to several waiting computers. It is possible to use Chat, the Explorer or “RemoteControl” simultaneously or separately.
- A single computer can set up connections under Windows 95/98/NT and wait for connections.

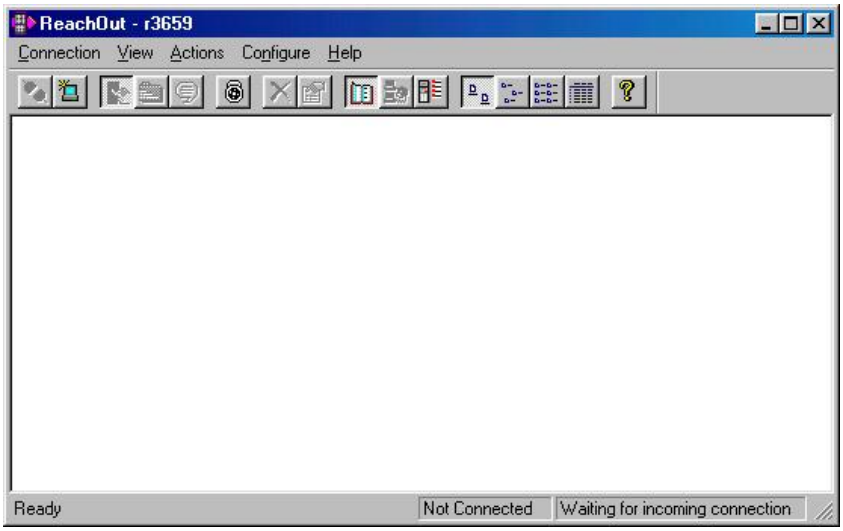
Note

If the control system has no mouse and no “ALT” key, the host on the control system can only be quitted via the viewer.

2.8.2 Connection setup with viewer for Windows 95/98 (MMC 102/103)

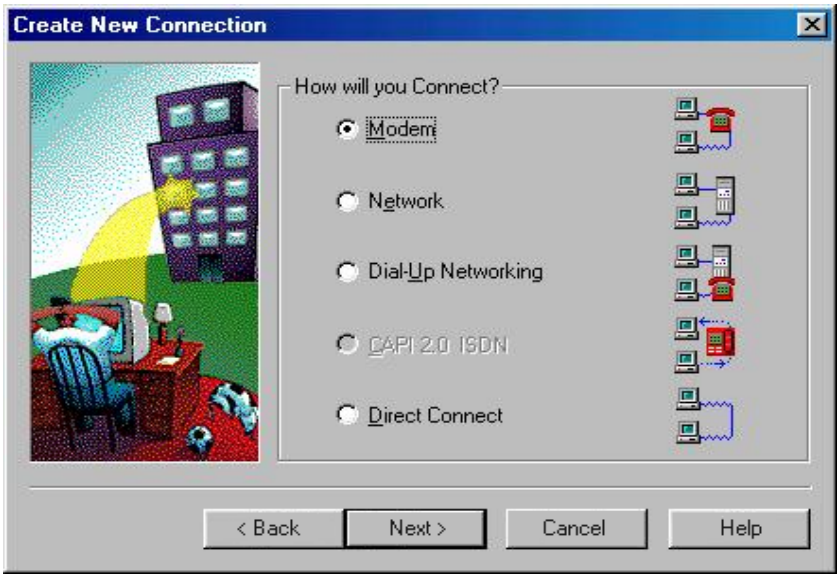
Connection setup with viewer and modem for Windows 95/98

Start “ReachOut” under Windows 95. The following window is overlaid:



Click on “Connection” to create a new connection. The input window “Create New Connection” is overlaid.

Enter the name of your connection. The default setting is “My connection”.



In the next input window select the connection type (over which medium the connection is to be set up). The default setting is “Modem”.

As an alternative, you can also activate “Network” with TCP/IP protocol if the MMC can be reached over a network, Internet or RAS gateway.

Enter the area code and the telephone number in the last window.

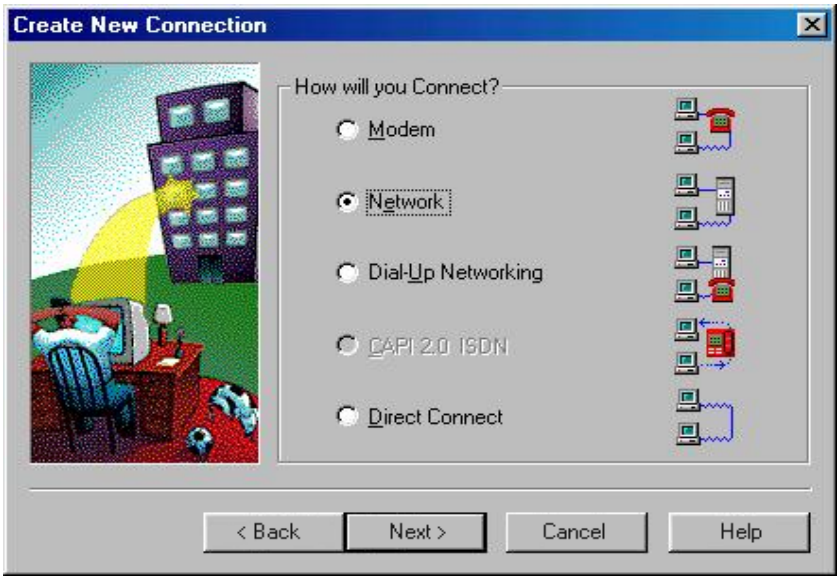
Activate the connection by double-clicking on the symbol that is created in the first screen.

Note

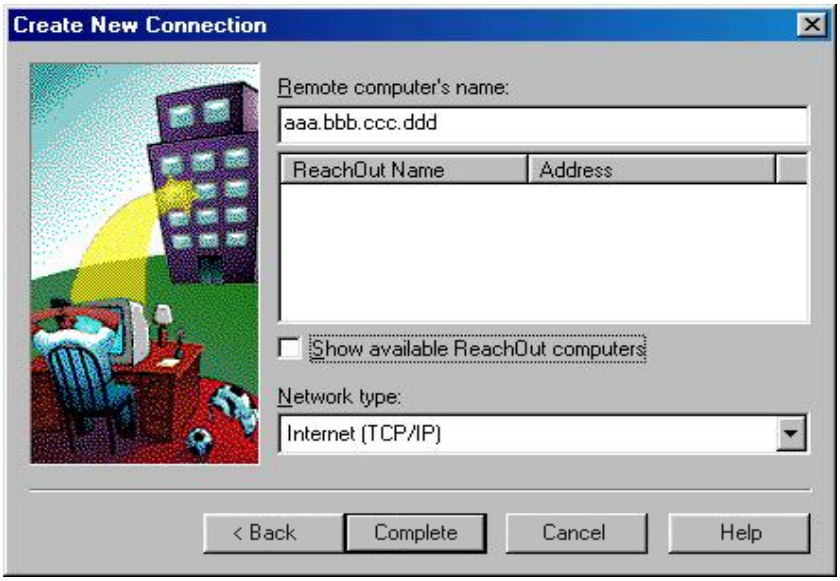
You can adapt the configuration of the modem later using the button “Configure”. But in this case, the settings you have made will be other than those made in the dialog box “**Control Panel/Modems/Properties”.**

Connection setup with viewer via network (TCP/IP)

Use the same procedure as with a modem connection until the bottom box appears.



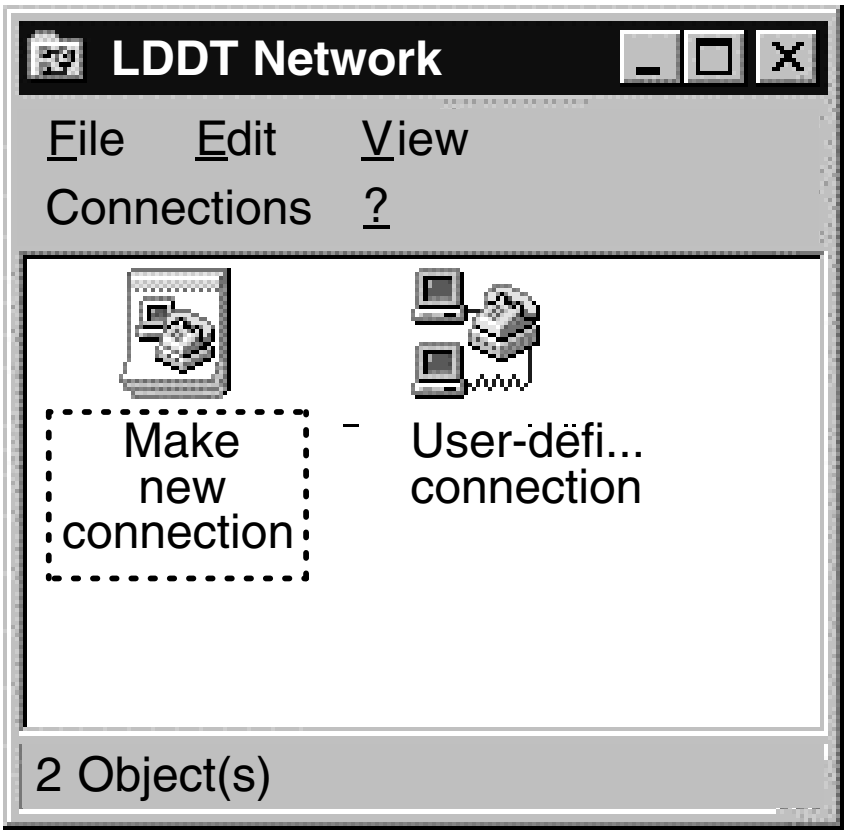
Select “Network”.



Enter the IP address of the remote computer you want to be connected to under “Remote computer’s name”.

Connection setup with viewer via Internet

Start the long-distance data transmission network (LDDT) by double-clicking on the folder “My Computer” or use “Start” → “Programs” → “Accessories” → “Communications”



Create a new connection by double-clicking on the “Make New Connection” icon. On the viewer side perform the steps that are listed under Section 2.3.6 up to and including “Integration into the MMC user interface”. A second icon is displayed with the default name “My Connection” or the selected name. Activate the connection by double-clicking on the appropriate icon.

If dial-up networking and the Internet connection have been set up on the MMC103, you can set up the remote diagnosis connection over network (TCP/IP), as described under “Connection setup with viewer over network (TCP/IP)”.

Use the IP address that was dynamically assigned to the MMC 103 if available.

2.8.3 Connection setup with viewer for Windows NT (MMC 102/103)

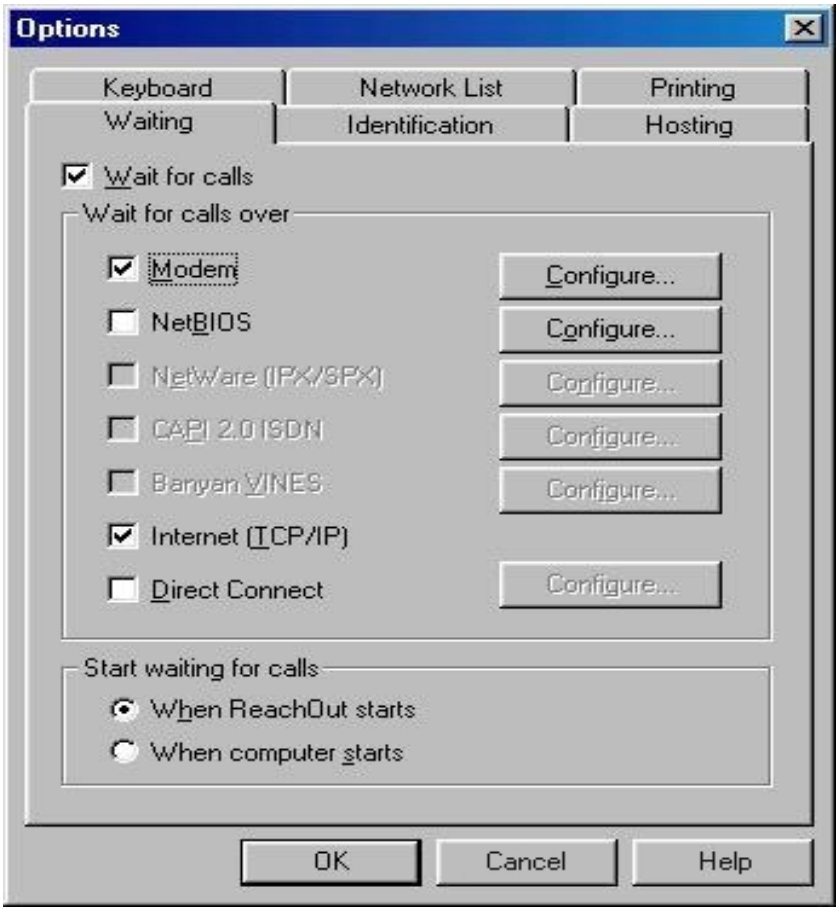
Configuration as gateway

Several controls are connected with a Windows NT computer via a network. You can access the NT computer host from any viewer PC via modem or ISDN. Using the NT viewer you can operate one of the MMC103 hosts from the NT computer host.

Preparations

No additional configuration is required on the individual controls. All controls must be configured as described under a network.

The first viewer PC from which the connection is to be set up to the individual controls must be a Windows NT computer.
Install the remote diagnosis software ReachOut for Windows NT on the computer. In general, installation, configuration and operation are conducted in the same way as for Windows 95.
Configure the individual connections via “Connection”, “New connection”.
Enter the name of the new connection and select the type of connection.
Enter the telephone number or IP address according to the type and click on “Finish”.
Under “Configure”, “Options” select the desired connections and configure them. At least two types of connections must be set up.



The entry “NetBIOS” should not be enabled.
Enable “When computer starts” if the remote diagnosis function is to be started automatically at system boot.

Connection setup

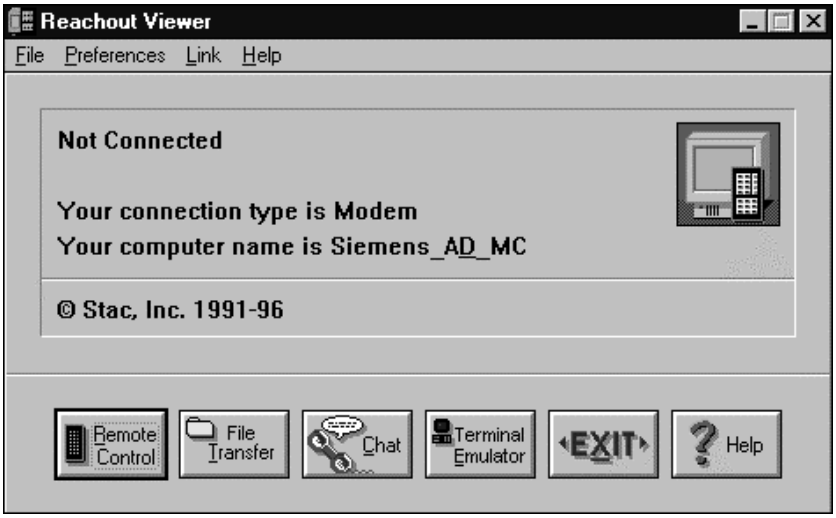
Power up the Windows NT computer and start the remote diagnosis software.
Dial up the Windows NT computer from a second PC which also has the remote diagnosis software installed on it.
Once the connection has been set up, you can access other possible connections from the NT computer too.

Note

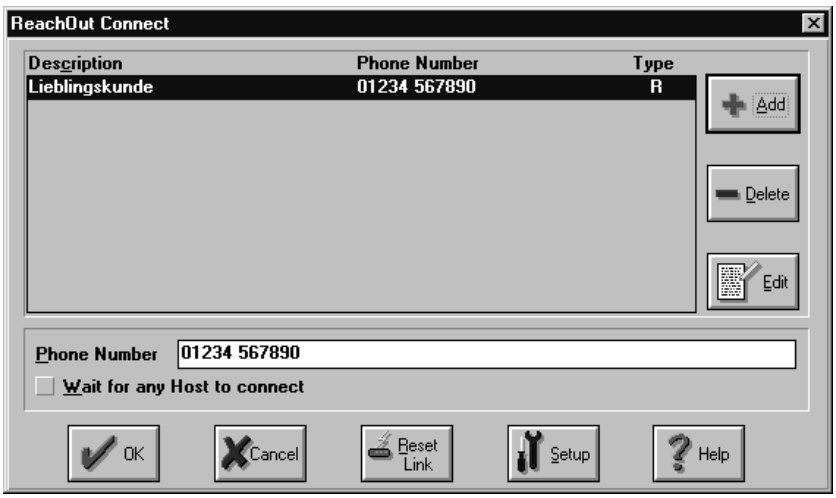
The connection between the viewer PC and NT computer must be different from the connection between the NT computer and the control. A modem can only set up one connection.

2.8.4 Connection setup with viewer for Windows 3.1X (MMC 102/103)

Start Windows 3.1X. In the Program Manager click on the icon “ReachOut” and on “ReachOut Viewer” in the overlaid window.



Start the remote diagnosis via “Remote Control”.



Select the desired subscriber and click on “OK” to establish the connection.

You can add other customer entries by clicking on “Add” and change the entries via “Edit”.

2.8.5 Connection setup with viewer in MS-DOS (MMC 100.2)

The MMC 100.2 is based on the MS-DOS operating system. The ReachOut software for the viewer is also available as an MS-DOS program. If you use a Windows operating system on the viewer PC, you must use the DOS version of the remote diagnosis.

Connection setup in MS-DOS

To set up the connection between the viewer and host via a modem, please proceed as follows:

1. Start ReachOut under MS-DOS or from the DOS prompt of Windows by calling the file gvm.bat from the ReachOut directory. ReachOut for the viewer is loaded.
2. Choose → “Connect using Phone book” from the “Link” menu. Every entered host is displayed with name and telephone number.
3. Select the host of your choice from the list using the cursor keys and confirm by pressing the “Enter” key. The host is selected.

Or:

1. Choose → “Contact to Host PC” from the “Link” menu. Screenform “Enter Phone Number for Host PC” is displayed.
2. Enter the telephone number of the desired host and confirm by pressing the “Enter” key. The host is selected.

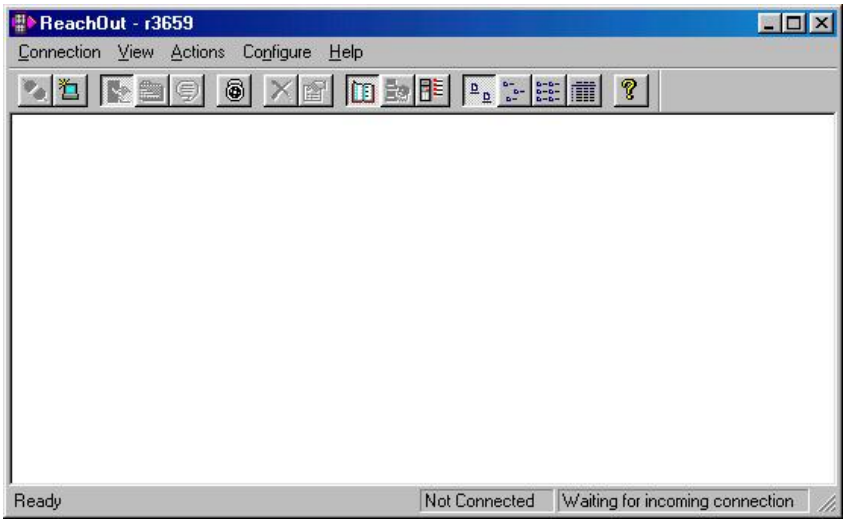
2.9 Functions of the viewer PC, version Win32

ReachOut main screen of the Win32 version:

The new version does not contain any additional functions. But there are some improvements, compared with older versions.

Improvements

1. The remote diagnosis can now display even 24-bit truecolor (to be set on the host) on the viewer.
2. Under WinNT, a connection can be established only with appropriate user rights. With earlier versions, this was not possible under WinNT.



2.9.1 Remote control

Operating the MMC 102/103

In “Remote Control” mode, you can see the screen of the MMC 102/103 on your PC screen and operate the MMC using mouse and keyboard. You can activate softkeys by double-clicking on them with the mouse or by means of the function keys. With the horizontal softkeys, SK1 to SK8 correspond to function keys F1 to F8, “Recall” corresponds to F9 and “Operating area switchover” to F10. The vertical softkeys are reached, starting from the top, by means of Shift + F1 to Shift + F8. You can operate all areas of the MMC 102/103 from the viewer PC.

Disconnecting

To break off the connection, simultaneously press the left and right Shift keys on the viewer. The remote diagnosis screen is overlaid (Windows 3.1, 3.11).
Click on the disconnect symbol and confirm that you want to hang up in the next window. The control remains ready to receive so that the connection can be set up again.
Select the icon that is relevant to this connection for Windows 95/NT and right mouse-click on it. Activate the “Disconnect” menu command from the overlaid menu.

Note

If the control system has no mouse and no “ALT” key, the host on the control system can only be quitted via the viewer.

2.9.2 File transfer

What does file transfer do?

The file transfer function makes it possible to exchange files between the viewer PC and the control. In this case, the viewer PC is referred to as the “Local” and the MMC 102/103 as the “Host”. The transmission rate is approximately 6–8 minutes for 1 MB at 9.600 baud.

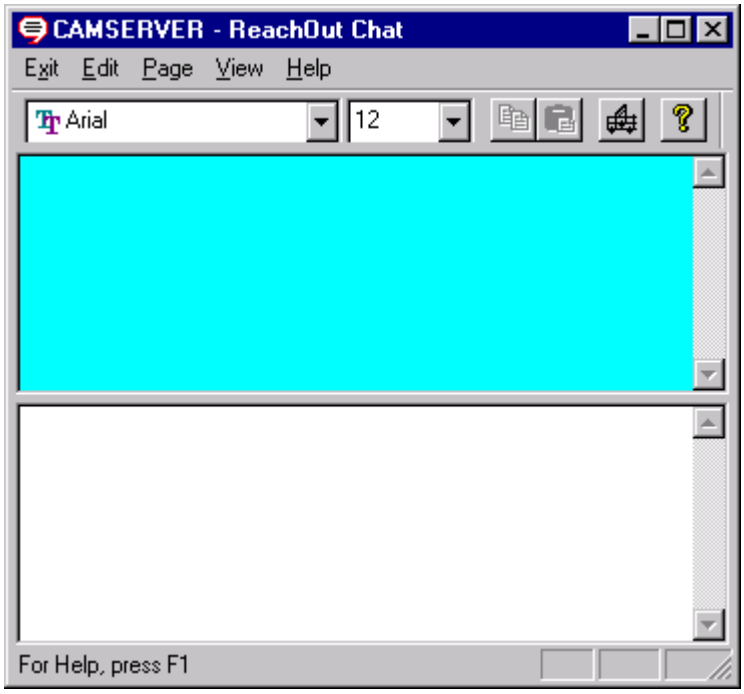
How to cancel the function

You can return to the remote diagnosis selection screen by activating “Exit” in the “File” menu or by means of the key combination Alt + F4.

2.9.3 Chat for MMC 102/103 (communication via text input window)

Chat screen under Windows 95

The chat window allows both operators to communicate by typing in text at the same time. Click on “Chat” to open the Chat window.



Note

On the MMC keyboard you can select “Chat” via the End key and activate it with the Input key.
While a connection is active, you can select the main screen on the MMC via “Diagnosis” → “Remote Diagnosis”.

Exiting the function

The Chat window is closed with “Exit”.

Note

This is not possible on the MMC without the MFII keyboard. In this case, the operator on the viewer side must close the Chat window and the window on the MMC is then closed automatically.

2.9.4 Terminal emulation

This function is not required for remote diagnosis.

2.9.5 Exit

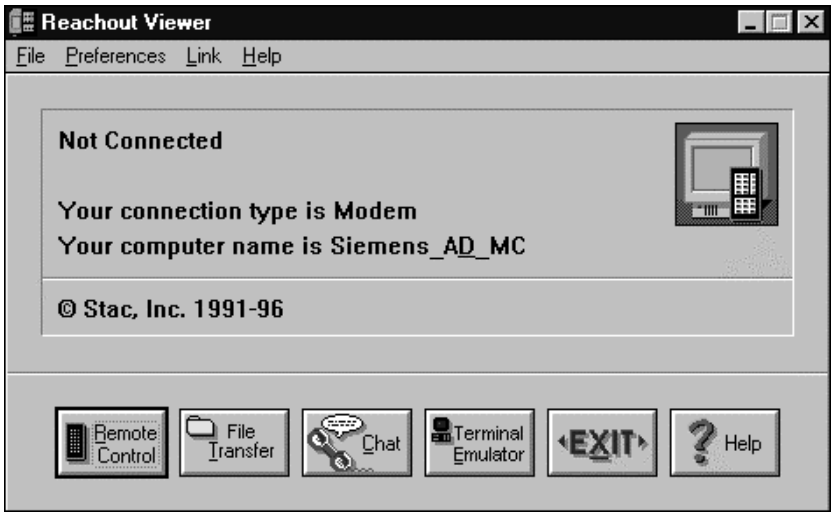
You can leave the menu by activating “Exit”. The program is terminated only if the connection has been ended beforehand.

2.9.6 Help

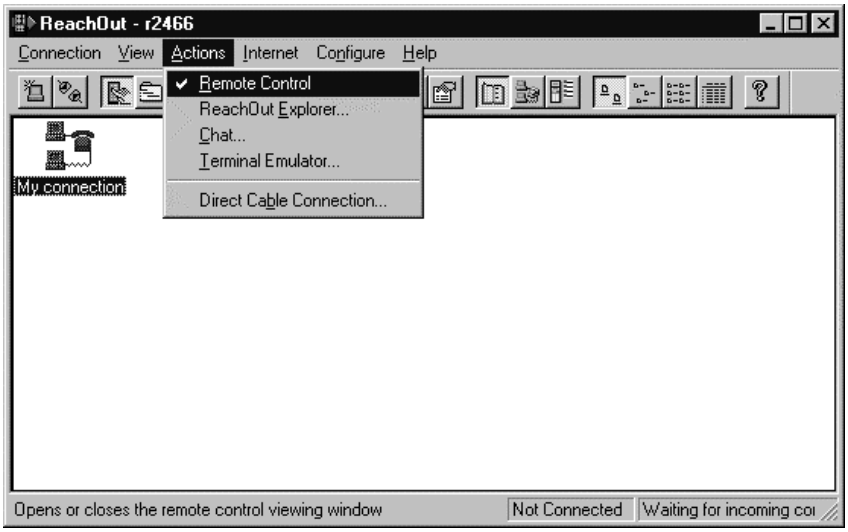
A comprehensive help menu is available to assist at every important stage of operation.

2.10 Functions of viewer PC, version Win16

ReachOut main screen with Windows 3.X:



ReachOut main screen with Windows 3.X:



2.10.1 Remote control

Operating the MMC 102/103

In “Remote Control” mode, you can see the screen of the MMC 102/103 on your PC screen and operate the MMC using mouse and keyboard. You can activate softkeys by double-clicking on them with the mouse or by means of the function keys. With the horizontal softkeys, SK1 to SK8 correspond to function keys F1 to F8, “Recall” corresponds to F9 and “Operating area switchover” to F10. The vertical softkeys are reached, starting from the top, by means of Shift + F1 to Shift + F8. You can operate all areas of the MMC 102/103 from the viewer PC.

Disconnecting

To break off the connection, simultaneously press the left and right Shift keys on the viewer. The remote diagnosis screen is overlaid (Windows 3.1, 3.11).
Click on the disconnect symbol and confirm that you want to hang up in the next window. The control remains ready to receive so that the connection can be set up again.
Select the icon that is relevant to this connection for Windows 95/NT and right mouse-click on it. Activate the “Disconnect” menu command from the overlaid menu.

2.10.2 File transfer

What does file transfer do?

The file transfer function makes it possible to exchange files between the viewer PC and the control. In this case, the viewer PC is referred to as the “Local” and the MMC 102/103 as the “Host”. The transmission rate is approximately 6 – 8 minutes for 1 MB at 9,600 baud.

How to cancel the function

You can return to the remote diagnosis selection screen by activating “Exit” in the “File” menu or by means of the key combination Alt + F4.

2.10.3 Chat for MMC 102/103 (communication via text input window)

Chat screen under Windows 3.11

The chat window allows both operators to communicate by typing in text at the same time. Click on “Chat” to open the Chat window.

Note

On the MMC keyboard you can select “Chat” via the End key and activate it with the Input key. While a connection is active, you can select the main screen on the MMC via “Diagnosis” → “Remote Diagnosis”.

Exiting the function

The Chat window is closed with “Exit”.

Note

This is not possible on the MMC without the MFII keyboard. In this case, the operator on the viewer side must close the Chat window and the window on the MMC is then closed automatically.

2.10.4 Terminal emulation

This function is not required for remote diagnosis.

2.10.5 Exit

You can leave the menu by activating “Exit”. The program is terminated only if the connection has been ended beforehand.

2.10.6 Help

A comprehensive help menu is available to assist at every important stage of operation.

2.11 Function of viewer PC (MMC 100.2)

The remote diagnosis for the MMC 100.2 provides the same functions as for the MMC 102/103. However, the Chat function is restricted in the MMC 100.2 version. Communication is only indirectly possible via a part program.

2.11.1 Remote control

Operation of the MMC 100.2

In “Remote control” mode, the screen contents on the MMC 100.2 (host) appears on your screen. You can control the host functions via the viewer PC keyboard. In this instance, softkeys are activated via function keys. Horizontal softkeys SK1 to SK8 are operated via function keys F1 to F8. Vertical softkeys are operated from top to bottom via Shift + F1 to Shift + F8. “Recall” represents function key F9 and “Operating area switchover” F10. In other words, you have control over all host functions from the viewer PC.

Clear the connection

If you wish to clear the connection between the viewer and host, you must press the left-hand and right-hand shift keys together. The window is closed and you return to the ReachOut interface.

In ReachOut, change to the menu “Link” → “Disconnect from Host-PC”. The connection to the MMC 100.2 is ended.

Note

With older MMC versions (< SW 5), the function “Hardware rely” must possibly be enabled in the viewer under “Preferences–>Keyboard Handling”.

2.11.2 File transfer

Meaning

File transfer allows you to transfer data from the viewer to the host and from the host to the viewer. The transferred data are stored in the clipboard on drive e: on a host with MMC 100.2. A directory named **“transfer”** is provided for this purpose on drive **e:**.

Transfer of data

To transfer data you must open the File Manager:

1. Select “File” –> “Invoke File Manager” on the ReachOut interface. An Explorer with two windows is opened.

The directory of the viewer PC is displayed in the left-hand window of the Explorer. The host directory is displayed in the right-hand window. Files can be copied and moved between the host and viewer or deleted. Files can only be written to drive e: on the host. All other drives have read status only.

2. Close the File Manager by selecting “Exit” –> “Yes”. You then return to the ReachOut interface.

Transfer directory

The transfer of data between the viewer and host is organized via a transfer directory. Data from the viewer or host are stored initially in this directory. To transfer part programs, for example, the data are copied into the transfer directory from the viewer PC where they can then be processed from the host.

To call data stored in the transfer directory from the host, proceed as follows:

1. Select softkey “Services” on the host. A new screenform appears.
2. Select softkey “Clipboard”.
3. Select softkey “Transfer directory”. The transfer directory is opened.

The functions “Copy and Paste”, “Create archive”, “Delete” and “Refresh-Dir” are available on the vertical softkeys.

2.11.3 Chat

Meaning

Direct CHAT is not supported.
The Chat function is possible for the MMC 100.2 only with restrictions if first a part program, such as CHAT1.MPF, is edited in a remote diagnosis session. This part program should be deleted at the end of the session.

Caution

Do not start the CHAT function from the remote diagnosis menu!

■

Supplementary Conditions 3

3.1 General supplementary conditions

Availability of the function

The function “Remote diagnosis” is available for:

- SINUMERIK 840D with MMC102/103, SW 3.1 and higher
- SINUMERIK 840D with MMC100.2, SW 5.1 and higher

The following is not supported:

Remote diagnosis

- up to SW 4: Windows 98
- up to remote diagnosis SW 4: AGP graphics cards

■

Notes

Tips and Tricks

4

This Section contains some helpful tips and tricks that might be useful.

4.1 Cascading of Win-Viewer to the DOS-HOST via “Gateway”

A connection to the control system can be established to several computers using remote diagnosis V5.1 ⇒ **cascading**.

To do so, the viewer version of the remote diagnosis must be installed on all computers, with exception of the control system. The remote diagnosis software on the intermediate computers in the direction towards the control system operate as a viewer and in the direction towards the end viewer as a host, which must be set to the Wait state.

The cascading always starts from the control system and then successively continues from one intermediate computer to the next to the end viewer.

It is also possible to cascade between different operating systems. The cascading of a Win32 viewer to a DOS host is shown below. For obvious reasons, different communication links must be used for viewer and host on the intermediate stations. The example below shows the intermediate computer directly connected to the end viewer via a TCP/IP network and directly to the control system via RS232. It is also possible to use the other admitted communication links.

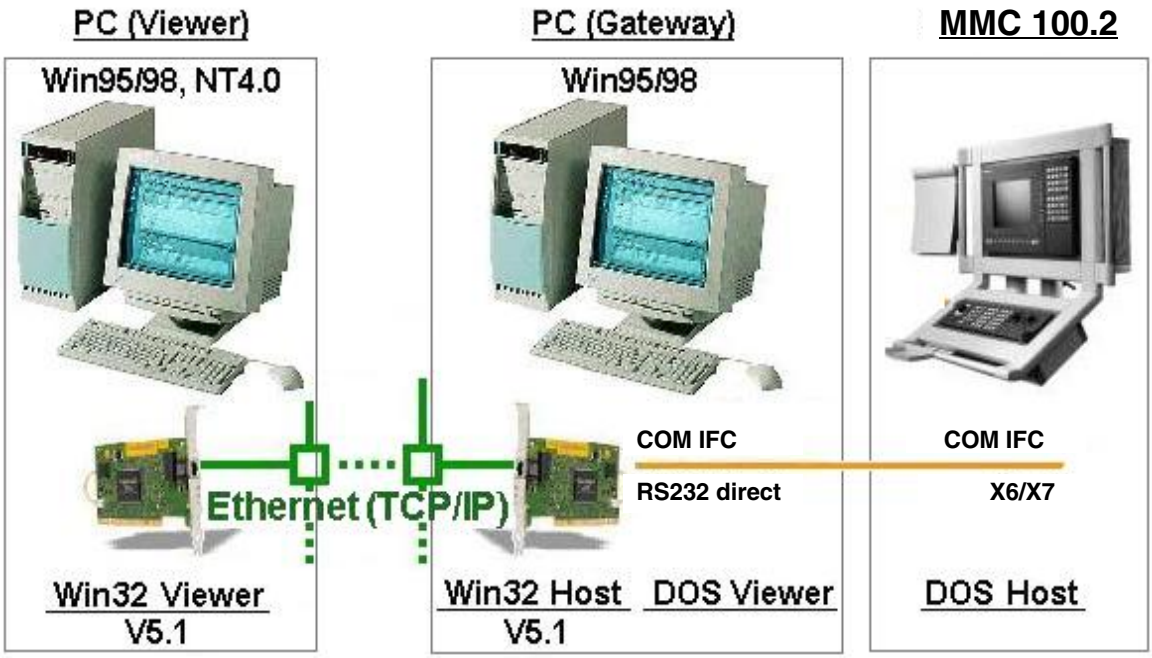


Fig. 4-1 Cascading the remote diagnosis via a gateway

The following possibilities were tested and have been found to be operative.

Machine	Operating system	ReachOut version	Supplementary conditions
MMC 100.2	MS-DOS	DOS host	No
Gateway PC (1)	Win95/98	DOS viewer + Win32 host V4.1/V5.1	The DOS viewer must run in a DOS window (no fullscreen)
Viewer PC (2)	Win95/98 NT	Win32 viewer V4.1/V5.1	First the connection MMC 100.2 ↔ gateway PC must be established before this connection may be established.

Caution

The Win NT operating system may not be run on the gateway PC (1). This mode does not allow 16-bit graphics resolution of the DOS viewer in a DOS window.

4.2 Automated installation of remote diagnosis

Automated installation of remote diagnosis

For remote diagnosis V5.1, installation scripts, such as the Windows Notepad, can be created for automated installation. If different users require different installation routines, several files can be created, whereby these must be started specifically in the Windows command line by specifying the file name of the script after the setup command.
If the same installation routine is sufficient for all users, you should create a script file with the name “ROINST”, which is automatically executed by the SETUP.EXE file and therefore need not be specifically initiated by the user. The script “ROINST” must be stored in the same directory as the setup programs to ensure that this file can be called automatically.

Note

To provide completely automated installation, you should copy all installation disks from the CD-ROM to a network drive or to diskettes. The installation script created can then be stored on the first diskette. Furthermore, the SETUP.ISS file that is contained on the first diskette must be adapted as described in this Section.

If special script names are used, the SETUP program must be started with the attribute “**SCRIPT =** Script name”.

The following syntax applies to the SETUP command:

SETUP [TYPE = [HOST |VIEWER] [; PUBLIC |SHARED] [SCRIPT = script filename] [script parameter 1] [script parameter 2] ... [script parameter n]

In this case, the command line would be used for starting the setup with the script file “HOST” which will contain the settings for installing the host version of remote diagnosis:

“SETUP SCRIPT=HOST”

Note

Under WinNT, this parameter only applies to version 8.30 or higher.

Sequence

Creating an installation script

1. Use a text editor to create a new file
2. Type the desired script commands and settings in the file.

When creating the script for a network installation, make sure that the parameter **PUBLIC** is not deleted; otherwise, you will have no safety during the entire installation of the remote diagnosis.

3. Save the file.

If you wish the script to be started automatically during the remote diagnosis, the script file must be called “ROINST” (without file extension) and be saved in the same folder as the setup program.

Note

Make sure that the **HIDDEN** switch for settings to be removed from the **SETUP.ISS** file is set in the installation script. Otherwise, you must enter these values in dialog boxes.

Parameter list

AUTOSTARTHOST=yes |no	determines whether the remote diagnosis is set to the host wait mode automatically during start.
COMPANYNAME=siemens	specifies the name of the company for which the remote diagnosis is licensed. If you do not want to specify a company name, use “COMPANYNAME=”.
CONNSHORTCUTS	Use this command to create shortcut icons for all remote diagnosis link icon files (*.RCO) that have been found in the [DISK1] folder during installation.
EXECUTE=program_name	starts the program with the name “program _name” after the installation of the remote diagnosis has been completed and Windows has been restarted.
GROUP=folder_name	specifies the name of the folder that will contain the remote diagnosis icons. If the GROUP command is missing, Setup will create a folder with the default name “ReachOut”.
GROUP_ID=group_ID	(only available from version 8.3 onwards.) specifies the name of the remote diagnosis group to which the computer belongs in a NetWare or Banyan VINES network.

HIDDEN	hides the display of the installation process on the display. If an installation script is processed using this command, a search for required parameters is carried out in the SETUP.ISS file contained in the same folder. Certain installation parameters in this file do not have default values; they must therefore be specified by the user. Parameters in the SETUP.ISS file, for which values must be specified: **AdminName=** User name for the initial user of the remote diagnosis with administrator rights. **Important:** Users who install the remote diagnosis with these parameters must know the assigned name. **AdminPass=** Password for the initial user with administrator rights for the application. **Important:** Users who install the remote diagnosis with these parameters must know the assigned name. **SzName=** A name that is stored in the Windows registry of the remote diagnosis. **SzCompany=** The company name that is stored in the Windows registry of the remote diagnosis. **SzSerial=** The serial number of the remote diagnosis, which is stored in the Windows registry of the remote diagnosis. These values are used on all computers on which the remote diagnosis is installed.
ICONS=icon,icon_description	places the shortcut icons in the ReachOut program folder. After installation, these shortcuts are to be found in the Windows “**Start**” menu. If you do not wish to use these icons, use “**ICONS=**” or use the “**NOGROUP**” command. **Example:** ICONS=REACHOUT.EXE,ReachOut;REACHOUT.HLP,ReachOutHelp;ROUSERS.EXE,ReachOut User Manager;

LOCALLOGON=yes |no	defines whether users must log into the remote diagnosis when starting the remote diagnosis. (only available from version 8.02 onwards.)
NAME=computer_name	defines the name used for the remote diagnosis to identify the computer on which the remote diagnosis is installed. If you do not wish to use a name, type "**NAME=**".
NODESKTOPICON	(only available from version 8.02 onwards.)
NETWORK=protocol	defines the network type via which the remote diagnosis can communicate. You can define more than one network if you use the command "**NETWORK**" several times. **Example:** NETWORK=Winsock NETWORK=NetBios **Parameter values that can be used:** **Banyan** – Banyan Vines network **IPX** – Novell Netware SPX or IPX using the Broadcast communication (the same as with **SPX**) **NetBios** – NetBios networks **IPX** – Novell Netware SPX or IPX using the SAP communication **IPX** – Novell Netware SPX or IPX using the Broadcast communication (the same as with **IPX**) **Winsock** – TCP/IP networks, including the Internet
NOGROUP	has the effect that no Windows program folder is created for the remote diagnosis in the **Start** menu.
PUBLIC	installs only the configuration files of the remote diagnosis. The remote diagnosis is started from the folder (usually a network or a shared folder) from which it was installed. **Tip:** Make sure that the remote diagnosis is installed in the source folder completely. The parameter **Public** cannot be used when the program is installed from disks.

REM	identifies a comment line that is not processed by the remote diagnosis.
RESTARTWINDOWS	exits Windows and restarts it after installation.
SERVER	defines the name of the NetWare server with which the remote diagnosis is to be linked. If the remote diagnosis is installed on a computer not integrated into the network, this parameter will have no effect.
SHARED	installs the remote diagnosis as a shared program. A shared installation is a central copy from which safety settings for all users can be made, who have carried out a Public installation from the shared copy. The shared installation is usually carried out on a shared network drive.
TARGET	defines the folder in which the remote diagnosis is to be installed. If this parameter is omitted from the script, the remote diagnosis is automatically installed in the folder "C:\PROGRAM FILES\REACHOUT" provided that the path "C:\PROGRAM FILES" exists. Otherwise, the user will be prompted to specify a folder name.
TYPE=Viewer/Host	defines whether a computer does calls or waits for calls from remote computers.

USERNAME=username	specifies the name of the person for whom the remote diagnosis is licensed. **Tip:** This is not the user name assigned with a password entry.
WAITON=protocol	(only from version 8.02 onwards); defines the connection types via which the remote diagnosis expects a call in advance. You can specify several connection types. Example: WAITON=SPX;TELEPHONY;WINSOCK **Parameter values that can be used:** **Banyan** – Banyan Vines networks **CAPI** =CAPI connection **DIRECT** – direct connection or zero modem cable **NetBios** – NetBios networks **IPX** – Novell Netware SPX or IPX using the Broadcast communication (the same as with **IPX**) **TELEPHONY** – modem **WINSOCK** – TCP/IP networks including Internet

Example of a script

```
REM installation script "ROINST"
AUTOSTARTHOST=no
COMPANYNAME=SIEMENS AG
CONNSHORTCUTS
EXECUTE=C:\MyBatchfile.BAT
GROUP=remote diagnosis
REM GROUP_ID=group_ID
HIDDEN
REM
ICONS=REACHOUT.EXE,ReachOut;REACHOUT.HLP,ReachOut-Help;ROEVE
NTS.EXE,Event Viewer;ROUSERS.EXE,UserManager;
ROSCRUI.EXE,Automating;SUPPORT,WRI, SUPPORT;
README.WRI,ReadMe;C:\Windows\uninst.exe -fC:\remote
diagnosis\
DeIsL1.isu-cC:\remote diagnosis\ROUInst.DLL,UnInst;
REM LOCALLOGON=yes or no
NAME=ExampleComputer
REM NODESKTOPICON
NETWORK=Winsock
REM NOGROUP
REM PUBLIC
RESTARTWINDOWS
SERVER
REMSHARED
TARGET=C:\remote diagnosis
Type=Viewer
USERNAME=Mustermann
WAITON=Winsock
```

Using this script, the remote diagnosis will be installed as a viewer/host with the company name "SIEMENS AG", the user name "Mustermann", the computer name "ExampleComputer" in the folder "C:\remote diagnosis" and will create the entry "remote diagnosis" in the **START** menu. The installation is carried out completely hidden in the background. The user need not carry out any entries. This is done by the command "HIDDEN". The required data, such as the user name of the initial user with administrator rights, the corresponding password and also the serial number are to be found in the file SETUP.ISS. The default values in this file can be modified and the serial number be inserted (see example).

Example of a SETUP.ISS file

```
; This script runs only in conjunction with a regular
; ReachOut installation script that contains the
; "HIDDEN" command. If the "HIDDEN" command is not
; specified, this SETUP.ISS file is ignored.
;
; This script will install ReachOut. You must restart
; the computer manually before using ReachOut.
; Changing BootOption=0 to BootOption=3 will
; cause an automatic restart.
;
; To use this script you _must_ modify the AdminName=,
; AdminPass=, szName=, szCompany=, and szSerial= lines
; below and provide valid entries.  szName and szCompany
; cannot be empty, and szSerial must be a valid number
; or the installation will not complete.

[InstallShield Silent]
Version=v3.00.000
File=ResponseFile
[Application]
Name=ReachOut
Version=8.0
Company=Stac

[DlgOrder]
Dlg0=SdWelcome-0
Dlg1=SdLicense-0
Dlg2=SdRegisterUserEx-0
Dlg3=SdAskDestPath-0
Dlg4=SdAskDestPath-1
Dlg5=SdSelectFolder-0
Dlg6=SdShowDlgEdit3-0
Dlg7=SdStartCopy-0
Dlg8=SdFinishReboot-0
Dlg9=MessageBox-0
Count=10

[SdWelcome-0]
Result=1
[SdLicense-0]
Result=1

[SdRegisterUserEx-0]
szName=Mustermann_ISS
szCompany=SIEMENSAG_ISS
szSerial=xxxx-xxxx-xxxx-xxxx    ← Enter the serial number here!
Result=1

[SdAskDestPath-0]
szDir=C:\Program Files\ReachOut
Result=1

[SdSelectFolder-0]
szFolder=ReachOut
Result=1
```

```
[SdShowDlgEdit3-0]
AdminName=ADMINISTRATOR ← User name with administrator rights!
AdminPass=PASSWORD    ← Password for administrator rights!
Confirm=PASSWORD    ← Confirm the password for administrator
rights!
Result=1

[SdStartCopy-0]
Result=1

[SdFinishReboot-0]
Result=1
BootOption=0
[MessageBox-0]
Result=1
```

After installation, the computer is rebooted, and the batch file “MyBatchfile.BAT” (it could also be an EXE file) is executed. Since the command “**ICONS**” is provided with comments, all icons of the remote diagnosis will be created. If certain icons are not created, remove the comment mark “REM” before the “**ICONS**” and the appropriate entries for the icons.

Note

If the icon “Uninst” is created using the “**ICONS**” command, the inverted commas must be removed from the string of the shortcut for the icon in the Properties; otherwise, the **UNINSTALL.EXE** will not be found.

The host is configured such that it is not automatically set to the Wait status after starting the remote diagnosis. It must be started manually with “Winsock” (TCP/IP, incl. Internet) set as the default setting.

4.3 Adapting the Remote Diagnosis window

In most cases, the Remote Diagnosis window on the MMC is hidden by the header. To adapt the window, it is possible to specify size and position of the window via the registry file.

Sequence

1. Boot the MMC in Windows Service Mode.
2. In Notepad, type the following text:

```
REGEDIT4
[HKEY_CURRENT_USER\Software\Stac\ReachOut\Reachout]
“X”=dword:00000001
“Y”=dword:000000bc
“Width”=dword:0000019a
“Height”=dword:000000fa
```

3. Save this as a file with the extension “.reg”.
4. Now, start the Explorer and start the created file by double-clicking on it.

The position and size specifications will be entered in the Registry. When the remote diagnosis is started next time, it is positioned and proportioned according to the values entered above.

If size and position do not correspond to your wishes, you can adapt the values accordingly. The values must be specified as hexadecimal values.

4.4 Repeated restart request

Repeated restart request

After the remote diagnosis has been installed, it can occur that a restart is required although Windows has already been restarted.

This behavior can be caused by the file “REBOOT” in the ReachOut directory. To eliminate this problem, use the Explorer to delete this file.

4.5 Tips for solving this problem with modem connections

Modems_e.wri

Tips for eliminating problems with modem connections under Windows are to be found in the file “Modems_e.wri” on the CD-ROM in the directory “siemens/Tipps”.

4.6 Blue screen after booting when the connection is established for the first time

Bluescreen Service 3 when establishing a connection under V4.x

The “blue screen” that appears after booting when establishing an RO connection for the first time; the following message appears:

Invalid VxD dynamic link call from ROVxD(01) + 000A3A8 to device 000C, service 3 ....

can be bypassed by disabling the cursor.
This can be done as follows:

1. Boot in Windows Service Mode
2. Edit the “system.ini” in the directory “**C:\Win95**”
3. Comment the following entries with “**;**”:
   a. Section:

      [boot]
      ...
      ;mouse.drv=mouse.drv

   b. Section

      [386Enh]
      ...
      mouse=*vmouse
4. Save the file and save the computer via **Start** → **Shut down ...** and “**Y**” “**Y**”.

Another possible problem solution is the installation of a serial mouse in Windows Service Mode.

Bluescreen Service 11 when establishing a connection under V4.x

The “blue screen” that appears after booting when establishing an RO connection for the first time; the following message appears:

Invalid VxD dynamic link call from ROVxD(01) + 000CC00 to device 001D, service 11 ....

is bypassed by loading a new keyboard driver. To overcome the problem, the MMC software must be upgraded to MMC V04.04.24.

■

Signal Descriptions 5

No signals are required at the NCK-PLC interface for this function.

Example 6

– None –

Data Fields, Lists 7

No signals or machine data are required for this function.

7.1 Alarms

A more detailed description of the alarms which may occur is given in

References: /DA/, Diagnostic Guide

or in the online help in systems with MMC 102/103.

Notes

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

Manual Travel and Handwheel Travel (H1)

■

Brief Description

1

Setting up machine

Even on modern, numerically controlled machine tools, a facility must be provided that allows the operator to traverse the axes manually. This is especially necessary when a new machining program is being set up and the machine axes have to be moved with the traversing keys on the machine control panel or with the electronic handwheel. Where coordinate offset or rotation is selected, handwheel jogging can even be performed in the transformed workpiece coordinate system.

Retraction of tool

After a program interruption caused, for example, by NC-STOP, RESET or power failure, the machine operator must retract the tool manually from its current machining position. This is usually done by operating the traversing keys in JOG mode. The transformations and coordinate systems used for machining must be activated while this is done.

Contents

The following Description of Functions illustrates the options and characteristics associated with the JOG traverse mode.

- Continuous jogging in jog or continuous mode (in JOG)
- Incremental jogging (INC) in jog or momentary-trigger mode (in JOG)
- Traversing the axes using electronic handwheels (accessories) (in JOG)
- Handwheel override in AUTOMATIC (path setting and velocity override)

DRF

The differential resolver function generates an additional incremental zero offset in AUTOMATIC mode via the electronic handwheel. With this function it is possible, for example, to compensate for tool wear within a programmed block.

■

Notes

Detailed Description 2

2.1 General

2.1.1 General characteristics of manual traverse in JOG

The following is a description of the characteristics which generally apply to JOG mode (irrespective of the type selected).

JOG mode

JOG mode must be active if the axes are to be traversed manually (referred to as “Manual traverse” below). The PLC receives the interface signal “Active mode: JOG” (DB11, DBX4.2) when the operating mode is activated.

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

Machine functions

There are several JOG variants (so-called “machine functions”) within the JOG mode:

- Continuous jogging (JOG CONT)
- Incremental jogging (JOG INC)
- Jogging with the handwheel

Handwheel operation

The handwheel operation is also active with the following functions:

- Operating mode JOG-REPOS for moving the geometry and machine axes
- Operating mode AUTOMATIC, for moving out a DRF displacement
- with path override
- when moving the backlash point of an oscillation

The machine function is selected via the PLC interface. A separate PLC interface exists for both the machine axes (axis-specific) and the geometry axes (channel-specific).

Simultaneous traversal

All axes can be traversed simultaneously in JOG.
There is no interpolation between several axes traversed simultaneously.

Velocity

The velocity for JOG traversal is determined by the following value settings depending on the feed mode:

- When linear feedrate (G94) is active
  (SD: JOG_REV_IS_ACTIVE (revolutional feedrate in JOG) = 0):
  - with general SD: JOG_SET_VELO (JOG velocity with G94) or
    for rotary axes
    with the general SD: JOG_ROT_AX_SET_VELO (JOG velocity for rotary axes)
  - or with axis-specific MD: JOG_VELO (jog velocity), only if SD: JOG_SET_VELO = 0.
- When revolutional feedrate (G95) is active
  (SD: JOG_REV_IS_ACTIVE (revolutional feedrate in JOG) = 1):
  - with general SD: JOG_REV_SET_VELO (JOG velocity G95)
  - or with axis-specific MD: JOG_REV_VELO (revolutional feedrate in JOG), only if SD: JOG_REV_SET_VELO = 0.

The default setting for feedrate velocity is mm/min and for revolutional feedrate rpm.

Rapid traverse override

If the rapid traverse override key is pressed at the same time as the traversing keys, then the movement is executed at the rapid traverse velocity set in axis-specific MD: JOG_VELO_RAPID (axis velocity in JOG mode with rapid traverse override) (or in the case of revolutional feedrate, set in MD: JOG_REV_VELO_RAPID).

Feedrate override

The velocity at which axes traverse in JOG can also be influenced by the axial feedrate override switch provided that interface signal “Axial feedrate override active” (DB31, ... DBX1.7) is set.
The assignment of percentages to the individual settings of the feedrate override switch is programmed via machine data.
At switch position 0% the axis is not traversed if 0 has been entered in the associated machine data. IS “Axial feedrate override active” has no meaning at switch position 0%.
Instead of the position on the feedrate override switch (Gray code), the value in percent (0% to 200%) can be set by the PLC. Again, the selection is made via the machine data.

References: /FB/, G1, “Feeds”

Acceleration

Acceleration in manual traverse mode also takes place according to a programmed acceleration characteristic. The acceleration characteristic applicable in JOG mode for a single axis is defined in MD: AX_JERK_DEFAULT (initial setting of axial jerk limitation).

References: /FB/, B2, “Acceleration”

Display

The JOG basic display appears on the screen when JOG mode is selected. This basic display contains the position, feedrate, spindle and tool values.

For information about the individual displayed values see:

References: /BA/, Operator's Guide

Coordinate systems

The operator has the option of traversing axes in different coordinate systems in JOG mode. The following coordinate systems are available:

- Basic coordinate system; each axis can be traversed manually
- Workpiece coordinate system; only geometry axes can be traversed manually (channel-specific)

Geometry axes

In JOG traversing mode, an axis can be traversed as either a machine axis (axis-specific) or as a geometry axis (channel-specific). The characteristics of the machine axes are dealt with in the following description. The special features of traversing geometry axes in JOG mode are described in more detail in Section 2.8.1.

Manual traversal of spindle in JOG

Spindles can also be traversed manually in JOG mode. Essentially the same conditions apply as for manual traverse of axes. Spindles can be traversed in JOG mode using the traverse keys continuously or incrementally, in continuous-trigger or momentary-trigger mode, or with the handwheel. The mode is selected and activated via the axis/spindle-specific PLC interface as for the axes. The axis-specific machine axes also apply to the spindles. The special features of traversing spindles manually are described in more detail in Section 2.8.2.

2.1.2 Control of JOG traverse functions via PLC interface

MMC/NCK/PLC interface

Most individual functions required for manual traversal in JOG are activated via the PLC user interface. The machine-control manufacturer can adapt the JOG mode functions to the machine tool through the PLC user program depending on the configuration of the NC system.

Machine control panel

The signals between the machine control panel and the individual PLC/NCK interface data blocks are transferred by the PLC user program on a machine-specific basis. The PLC user programs define the assignment of the direction keys on the machine control panel to the axis/spindle (machine axes, geometry axes) traversing keys.

The following machine control panel signals are of particular importance to manual traverse:

- JOG mode (selection)
- Machine function INC1 ...
- Direction keys
- Feedrate override and spindle speed override

For further information on signals sent from the machine control panel see:

References: /FB/, P3, “Basic PLC Program”

Selection of machine function

The machine functions available in JOG mode can be selected from the following locations:

- From the machine control panel e.g. user DB interface
- From the PLC user program PLC/NCK interface

The PLC user program transfers the machine function pending at the machine control interface to the relevant PLC/NCK interface. The axis-specific NCK/PLC interface (DB 31, ... signals see Section 5.3) is used for a machine axis/spindle and the channel-specific NCK/PLC interface (DB21, ... signals see Section 5.2) is used for a geometry axis.

2.1.3 Control response at power On, mode change, reset, block search, repositioning

Any reset yields an abort with braking ramp of a traversing movement triggered by handwheel operation.

Selection of MCP

The following example shows the sequence of operations for selecting the “continuous” machine function for a machine axis of the machine control panel.

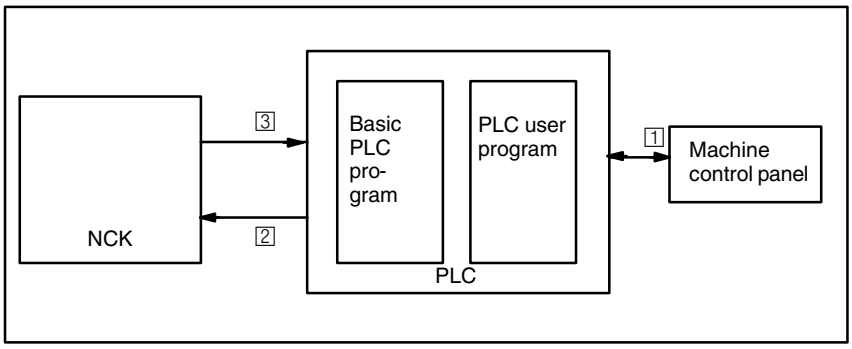


Fig. 2-1 Sequence of operation for selecting a machine function from the machine control panel

Sequence of operation

1 The operator selects the machine function “Continuous JOGGING” on the machine control panel for a machine axis.

2 IS “Machine function”
The PLC program (basic and user program) logically combines this IS and sends the request “Machine function continuous” (DB31, ... DBX5.6) to the NCK interface.
Before this happens, the PLC user program first checks that this request is permissible with regard to the current machine status.

3 IS “Active machine function”
The control selects the machine function internally. As soon as the machine function “JOG continuous” is active (DB31, ... DBX65.6), a signal is returned from the NCK to the PLC.

For further information on signal transmission between the machine control panel and the PLC see:

References: /FB/, P3, “Basic PLC Program”

2.2 Continuous jogging

Selection

Continuous mode in JOG mode is selected via the PLC interface (IS “Machine function: Continuous” (DB21-28 DBX13.6, ff)). As soon as continuous mode is active, a signal is returned to the PLC with IS “Active machine function: Continuous” (DB21-28 DBX41.6, ff).

Traversing keys +/–

The “plus” and “minus” traversing keys are selected to move the relevant axis in the appropriate direction.
If both keys are pressed at the same time no movement takes place or a moving axis is stopped.

!

Important

When the control is switched on, axes can be traversed to the limits of the machine because they have not yet been referenced. Emergency limit switches might be triggered as a result.
The software limit switches and the working area limitation are not yet active.

Travel command +/–

As soon as a traverse request for an axis is active (e.g. after selection of a traverse key), the IS “Travel command+” or “Travel command–” (DB21, ... DBX40.7 or DBX40.6) is sent to the PLC (depending on selected traverse direction).

2.2.1 Distinction between jog mode and continuous mode

Selection

In JOG mode we distinguish between traversing in jog mode and continuous mode. The selection is made in the general SD: JOG_CONT_MODE_LEVELTRIGGRD (JOG continuous in jog mode) and is active for all axes.

Default setting

Jog mode is the default setting.

Continuous traversal in jog mode

Function

In jog mode (default setting) the axis traverses for as long as the traverse key is held down if no axis limitation is reached first. When the traverse key is released the axis is decelerated to zero speed and the movement comes to an end.

Continuous traversal in continuous mode

Function

When the traverse key is pressed and released (first rising edge) the axis starts to traverse at the set velocity in the desired direction. This movement is continued after the traverse key is released. The movement of the axis is either stopped by the operator or because of a response in the control (e.g. software limit switch reached).

Warning

If “continuous” mode is selected, several axes can by started by pressing and releasing the relevant direction key. Any interlocks must be implemented via the PLC!

Interrupt traversing movement

The operator can use the following methods to interrupt the traversing movement:

- Set feedrate override to 0%
- Axial feed disable (PLC interface signal)
- NC STOP or NC STOP axis/spindle

If the cause of the interruption is removed again, the axis continues to traverse.

Abort traversing movement

The traversing movement can be stopped and aborted by means of the following operator inputs or monitoring functions:

- Same traverse key pressed again (second rising edge)
- Traverse key for the opposite direction pressed
- RESET
- When continuous jogging is deselected
- On reaching the first valid limit

Caution

Software limit switches and working area limitations are only activated after reference point approach.

- When a fault occurs

Note

While an axis is moving, a change of mode from JOG to AUT or MDA is disabled internally.

2.2.2 Special features of continuous jogging

Indexing axes

When an axis that is declared as an indexing axis is traversed in continuous JOG mode, it always traverses to indexing positions. For example, the axis traverses on to the next indexing position in the direction of travel even if the key is released in jog mode.

References: /FB/, T1, “Indexing Axes”

2.3 Incremental jogging (INC)

Programming increments

The traversing path to be traversed by the axis is defined by so-called increments (also termed “incremental dimensions”). Before the machine operator jogs the axis he must set the required increment.
The setting is made on the machine control panel, for example. The IS “Machine function: INC1 to INCvar” (DB31, ... DBB5 ff) associated with the required increment is set by the PLC user program after appropriate logic operation.

Settable increments

The operator can set up to six different increment sizes. These are subdivided into:

- **five fixed increments** whose increment sizes are defined jointly for all axes with the general MD: JOG_INCR_SIZE_TAB (increment size INC/handwheel). The default settings are INC1, INC10, INC100, INC1000 and INC 10000.
- **and a variable increment** (INCvar). The increment setting for the variable increment also applies jointly to all axes and is made in SD: JOG_VAR_INCR_SIZE (size of the variable increment for INC/handwheel).

Increment weighting

Axial MD: JOG_INCR_WEIGHT (weighting of an increment of a machine axis for INC/handwheel) defines the path weighting of **one** JOG increment.

2.3.1 Distinction between jog mode and continuous mode

Selection

In incremental jogging, too, we distinguish between traversing in jog mode and continuous mode. The selection is made in the general MD: JOG_INC_MODE_LEVELTRIGGRD (INC and REF jog mode). Jog mode is the default setting.

Incremental jogging in jog mode

Function

If the traverse key for the required direction (e.g. +) is pressed, the axis begins to traverse the increment set. If the traverse key is released before the increment has been traversed the movement is interrupted and the axis stops. If the same key is pressed again, the axis moves the remaining distance until it is zero. As long as the remaining distance is greater than zero the movement can again be interrupted by releasing the traverse key.
Pressing the key for the opposite direction has no effect until the increment has been completely traversed or the movement has been interrupted.

Aborting the traverse movement

If you do not want to traverse the whole increment, the traverse movement can be aborted with RESET or IS “Delete axial distance-to-go” (DB31, ... DBX2.2).

Incremental jogging in continuous mode

Function

The axis traverses the entire set increment when the traverse key is pressed (on the first rising edge). If the same traverse key is pressed again (second rising edge) before the axis has traversed the increment the traverse movement is interrupted; i.e. not completed.

Interrupting the traverse movement

As for continuous jogging.

Aborting the traverse movement

The traverse movement is stopped and aborted by the following operator action or monitoring functions:

- Same traverse key pressed again (second rising edge)
- Traverse key for the opposite direction pressed
- RESET
- Delete axial distance-to-go (PLC interface signal)
- On reaching the first valid limit

Caution

Software limit switches and working area limitations are only activated after reference point approach.

- on deselection or change of the current increment (e.g. change from INC100 to INC 10)
- on faults (e.g. on cancellation of the servo enable)

Note

While an axis is moving, a change of mode from JOG to AUT or MDA is disabled internally.

Warning

If “continuous” mode is selected, several axes can by started by pressing and releasing the relevant direction key. Any interlocks must be implemented via the PLC!

2.3.2 Special features of incremental jogging

Indexing axes

Regardless of the currently set incremental value, the axis declared as an indexing axis (MD: INDEX_AX_ASSIGN_POS_TAB) traverses to the next highest indexing position after the traverse key “+” is pressed. In a similar way, if the traverse key “–” is pressed it moves to the next lowest indexing position.

References: /FB/, T1, “Indexing axes”

2.4 Handwheel traversal in JOG

Selection

JOG mode must be active. The operator must also set the increment INC1, INC10, ... which applies when jogging with the handwheel. As with incremental jogging the required machine function must be set at the PLC interface accordingly.

Traversing

When the electronic handwheel is turned the associated machine axis is traversed either in the positive or negative direction depending on the direction of rotation.

Traversing path

The traversing path produced by rotation of the handwheel is dependent on the following factors:

- Number of handwheel pulses received at the interface
- Active increment (machine function INC1, INC10, INC100, ... INCvar)
- Handwheel pulse weighting with general MD: HANDWH_IMP_PER_LATCH (handwheel pulses per detent position)
- Evaluation of an increment with INE/handwheel (axis-specific MD: JOG_INCR_WEIGHT).

Travel command +/–

While the axis is moving the IS “Travel command+” or “Travel command-” (DB31, ... DBX64.7 or DBX64.6) is output depending on the direction of movement.
If the axis is already being moved using the direction keys, the handwheel cannot be used. Alarm 20051 “Jogging with the handwheel not possible” is output.

Handwheel connection

Two handwheels can be connected simultaneously. In this way, up to two axes can be traversed by handwheel simultaneously.
Exception: If several axes are assigned to one handwheel, more than two axes can be traversed with handwheels.

Handwheel assignment

A separate axis-specific VDI interface signal is used to make the assignment between a handwheel and a geometry or machine axis.

The axis to be moved as a result of rotating handwheel 1 or 2 can be set:

- Via the PLC user interface with IS “Activate handwheel” (DB31-48, DBX4.0 to DBX4.2)
  (with geometry axis: DB21, ... DBX12.0 – 12.2 ff)
- By menu-guided operation (MMC)
  Operating the softkey **Handwheel** in the JOG mode basic menu displays the window “Handwheel”. Here, every handwheel can be assigned an axis and the handwheel can be enabled or disabled.

The assignment is transferred to the PLC interface through the logic of the PLC user program. In this way, several axes can be assigned to one handwheel simultaneously.

Function

The electronic handwheels (accessories) can be used to simultaneously traverse selected axes manually. The weightings assigned to the divisions of the handwheel are defined by the increment size weighting. Where coordinate offset or rotation is selected, handwheel jogging can even be performed in the transformed workpiece coordinate system.

Handwheel selection from MMC

A separate user interface between the MMC and PLC is provided to allow activation of the handwheel from the operator panel. This interface that the basic PLC program supplies for handwheels 1 and 2 contains the following information:

- The axis numbers assigned to the handwheel IS “Axis number handwheel n” (DB10, DBB100 ff)
- Additional information on machine or geometry axis IS “Machine axis” (DB10, DBX100.7 ff)
- The channel number assigned to the handwheel if a geometry axis has been selected on handwheel selection
  IS “Channel number geometry axis handwheel n” (DB10, DBX97 ff)
- Information that the handwheel is enabled or disabled IS “Handwheel deselected” (DB10, DBX100.6 ff)

The IS “Activate handwheel” is either set to “0” (disable) or to “1” (enable) by the basic PLC program for the defined axis.

Input frequency

The handwheel connections can receive handwheel pulses with a maximum input frequency of 100 kHz.

Velocity

The axis velocity settings SD: JOG_SET_VELO (JOG velocity for G94), MD: JOG_VELO (jogging axis velocity) and SD: JOG_ROT_AX_SET_VELO (JOG velocity for rotary axes) which are active in JOG are also used for jogging with the handwheel.
Because of the limited feedrate, the axis is not able to follow the handwheel turn synchronously, especially in the case of large pulse weighting, and therefore lags behind.

Abortion of traversing movement

The traversing movement is aborted as the result of a RESET or the IS “Axial deletion of distance-to-go” (DB31, ... DBX2.2). The setpoint/actual-value difference is deleted. STOP only interrupts the traversing movement. Any setpoint/actual-value difference is maintained. The distance-to-go is then traversed on START.

Increment value limitation

The operator can delimit the size
(MD: HANDWH_GEOAX_MAX_INCR_SIZE).

The size of the selected increment for machine axes can be delimited with the axial machine data (MD: HANDW_MAX_INCR_SIZE).

A traversing movement defined by the handwheel for a geometry axis is defined by

- traversing path
- size of the variable increment (SD: JOG_VAR_INCR_SIZE)
- geometry axis allocation
  (MD: HANDWH_GEOAX_MAX_INCR_SIZE)

or for a machine axis by

- traversing path
- size of the variable increment (SD: JOG_VAR_INCR_SIZE)
- machine axis allocation
  (MD: HANDWH_MAX_INCR_SIZE)

Movement in the opposite direction

Depending on the machine data $MN_HANDWH_REVERSE, the behavior with a change of the traversing direction is as follows:

- If the handwheel is moved in the opposite direction, the resulting distance is computed and the calculated end point is approached as fast as possible: If this end point is located before the point where the moving axis can decelerate in the current movement direction, the unit is decelerated and the end point is approached by movement in the opposite direction. If this is not the case, the newly calculated end point is approached immediately.
- If the handwheel is moved in the opposite direction by at least the number of pulses indicated in the machine data, the axis is decelerated as fast as possible and all pulses received until the end of interpolation are ignored. That means, another movement takes place only after zero speed (setpoint side) of the axis (new function). This feature is available with SW 3.2 and higher.

Acceleration

The acceleration rate for handwheel traversal is determined by the acceleration characteristic programmed in MD: AX_JERK_DEFAULT (Initial setting of axial jerk limitation).
References: /FB/, B2, “Acceleration”

Response at software limit switches, working area limitation

When axes are traversed in JOG mode, they can traverse only up to the first active limitation before the appropriate alarm is output. Depending on the machine data $MN_HANDWH_REVERSE, the behavior is as follows (as long as the axis on the setpoint side has not yet reached the end point):

- The distance resulting from the handwheel pulses forms a fictitious end point which is used for the subsequent calculations: If this fictitious end point is positioned for example 10 mm behind the limitation, these 10 mm must be traversed in the opposite direction before the axis traverses again. If a movement in the opposite direction shall be performed immediately after a limitation, the fictions distance-to-go can be deleted via delete distance-to-go or deselection of the handwheel allocation.
- All handwheel pulses leading to an end point behind the limitation are ignored. Any movement of the handwheel in the opposite direction results immediately in a movement in the opposite direction, i.e. away from the limitation. This feature is available with Software Version 3.2 and higher.

Limitations

The limitations are also active when jogging with the handwheel. For further information see Section 2.8.3.

Revolutional feedrate

In JOG mode, the operating characteristics of the axis/spindle are also dependent on the values set in setting data JOG_REV_IS_ACTIVE (revolutional feedrate active for JOG).

- If this setting data is active, an axis/spindle is always moved with revolutional feedrate MD JOG_REV_VELO (revolutional feedrate with JOG) or MD JOG_REV_VELO_RAPID (revolutional feedrate with JOG with rapid traverse overlay) depending on the master spindle.
- If the setting data is not active, the behavior of the axis/spindle depends on the setting data ASSIGN_FEED_PER_REV_SOURCE (revolutional feedrate for positioning axes/spindles)
- If the setting data is not active, the behavior of a geometry axis on which a frame with rotation is effective depends on the channel-specific setting data JOG_FEED_PER_REV_SOURCE. (In the operating mode JOG, revolutional feedrate for geometry axes on which a frame with rotation is effective).

2.5 Handwheel override in automatic mode

2.5.1 General functionality

Function

With this function it is possible to traverse jog axes or to change their velocities directly with the handwheel in Automatic mode (Automatic, MDA). The handwheel override is activated in the NC part program using the NC language elements **FD** (for path axes) and **FDA** (for positioning axes) and is **non modal**. With positioning axes, it is possible to activate the handwheel override modally using the traverse instruction POSA. When the programmed target position is reached, the handwheel override is deactivated again.Other axes can interpolate or traverse simultaneously in the same NC block.

The function for concurrent positioning axes can also be activated by the PLC user program.

Distinction

Depending on the programmed feedrate, a distinction is made between the following:

- **Path definition** Axis feedrate = **0** (FDA = 0) and
- **Velocity override** Axis feedrate **> 0** (FD or FDA > 0)

Table 2-1 shows which axis types can be influenced by the function “handwheel override in Automatic mode”.

Table 2-1 Axes that can be influenced by the function “handwheel override in Automatic mode”

Axis type	Velocity override	Path definition
Positioning axis	FDA[AXi] > 0 ; axial	FDA[AXi] = 0
Concurrent positioning axis	Parameter “handwheel override active” = 1 and axis feedrate > 0 from FC 15	Parameter “handwheel override active” = 1 and axis feedrate = 0 from FC 15
Path axis	FD > 0 ; applies to path velocity	Not possible

Path definition

With an axis feedrate setting = 0 (e.g. when FDA[AXi] is programmed as 0), the traversing movement of the positioning axis towards the programmed target position is controlled entirely by the operator rotating the assigned handwheel.

The direction in which the handwheel is turned determines the direction in which the axis traverses. The programmed target position cannot be exceeded during handwheel override. The axis can also be moved in the direction opposite to that programmed, the movement in the opposite direction only being restricted by the axial position limits.

A block transition occurs

- When the axis has reached the programmed target position or
- the distance-to-go is deleted by the axial IS “Delete distance-to-go” (DB31, ... DBX2.2).

From this moment on, the path definition is deactivated and any further handwheel pulses have no effect.

After this, incrementally programmed positions refer to the point of interruption and not to the last programmed position.

Velocity override

With regard to the velocity override, a distinction is made between axis feed and path feed.

- **Override for the axis velocity** (FDA[AXi] > 0):
  The positioning axis is moved to the target position at the programmed axial feedrate. With the assigned handwheel it is possible to increase the axis velocity or to reduce it to a minimum of zero depending on the direction in which the handwheel is turned. The resulting axis feedrate is limited by the maximum velocity. The axis cannot be traversed in the direction away from the target position. As soon as the axis has reached the programmed target position, a block transition occurs.
  In this way, the velocity override is automatically deactivated again and any further handwheel pulses have no effect.
  This also applies to concurrent positioning axes, but the target position and the velocity are set by the PLC.
- **Override for the path velocity** (FD > 0):
  The path axes programmed in the NC block move to the target position at the programmed path feedrate. If the velocity override is active, the programmed path velocity is overridden by the velocity generated with the **handwheel of the 1st geometry axis**. As soon as the programmed target position is reached, a block transition occurs.
  Depending of the direction in which the handwheel is turned, the path velocity is increased or reduced to a minimum of zero. It is not possible to reverse the direction of the movement with the handwheel override.

Example of application

The “Handwheel override in AUTOMATIC mode” function is frequently used on grinding machines. For example, the operator can position the reciprocated grinding wheel on the workpiece using the handwheel (path definition). After scratching, the transverse movement is terminated and the block transition is initiated (by activating “Delete distance-to-go”).

Requirements

In order to activate “Handwheel override in AUTOMATIC mode” the following conditions must have been met

- A handwheel must be assigned to the axis in question.
- Pulse weighting is programmed for the assigned handwheel.

Handwheel assignment

The connected handwheels are assigned to the axes analogously to the procedure described in Section 2.4, i.e. via the PLC user interface with IS “Activate handwheel” (DB31, ... DBX4.0 to DBX4.2) or by means of menu-assisted operator inputs.
If an axis is programmed for the handwheel override and no handwheel is assigned to it, the following cases are possible:

- With velocity override
  The axes are traversed at the programmed velocity. A self-acknowledging alarm is output (without response).
- With path definition
  No traverse movement is performed, because the velocity is zero. A self-acknowledging alarm is output (without response).

Note

When the velocity override is applied to path axes, only the **handwheel of the 1st geometry axis** acts on the path velocity.

Handwheel weighting

The traversing path of the axis that is generated by advancing the handwheel by one detent position is dependent on several factors (see Section 2.4).

- The selected increment size (general MD: JOG_INCR_SIZE_TAB[5] or axial SD: JOG_VAR_INCR_SIZE)
- The weighting of an increment (axial MD: JOG_INCR_WEIGHT)
- Number of handwheel pulses per detent position (general MD: HANDWH_IMP_PER_LATCH)

For example, the axis traverses by 0.001 mm per handwheel detent position if machine function INC1 and the standard setting of the above machine data are selected.

With the velocity override, the velocity results from the path covered using the handwheel within a period of time.

Example

Assumptions:
The operator turns the handwheel at 100 pulses/second.
The selected machine function is INC100.
Machine data specified above for handwheel weighting with default setting
⇒ Handwheel traversing path per second: 10 mm
⇒ Velocity override: 0.6 m/min

PLC interface signals

As soon as the handwheel override takes effect, the following interface signals to the PLC are set to “1”:

- With positioning axes:
  IS “Handwheel override active” (DB31, ... DBX62.1)
- With concurrent positioning axes:
  IS “Handwheel override active” (DB31, ... DBX62.1)
- With path axes:
  IS “Handwheel override active” (DB21, ... DBX33.3)

With the path definition, the IS “Traverse commands +/–” (DB31, ... DBX64.6 and 64.7) are output to the PLC depending on the direction of travel.

Limitations

The axial limitations (SW limit switches, HW limit switches, working area limitations) are effective in conjunction with handwheel override. With the path definition, the axis can be traversed with the handwheel in the programmed direction of travel only as far as the programmed target position.

The resulting velocity is limited by the axial MD: MAX_AX_VELO (maximum axis velocity).

NC STOP/ override = 0

If the feedrate override is set to 0% or an NC STOP is initiated while the handwheel override is active, the following applies:

- With path definition
  The handwheel pulses arriving in the meantime are summated and stored. On NC start or feedrate override > 0%, the stored handwheel pulses are activated (i.e. traversed).
  However, if the handwheel is deactivated first (IS “Activate handwheel n” DB21, ... DBX12/16/20), the stored handwheel pulses are deleted.
- With velocity specification
  The handwheel pulses arriving in the meantime are not summated and are not active.

2.5.2 Programming and activation of handwheel override

General notes

When the handwheel override is programmed with the NC language elements **FD** (for path axes) and **FDA** (for positioning axes), the following points must be observed:

- FDA and FD are **nonmodal.**
  There is an exception with respect to positioning axes: If the traverse instruction POSA is programmed, the handwheel override can be active beyond block boundaries because the block transition is not affected by the positioning axis.
- When the handwheel override is activated with FDA or FD, a **target position** must be programmed in the NC block for the positioning axis or for a path axis. When the programmed target position is reached, the handwheel override is deactivated again.
- It is not possible to program FDA and FD or FA and F in the same NC block.
- The positioning axis must not be an indexing axis.

Positioning axis

Syntax for handwheel override: **FDA[AXi] = [feed value]**

Example 1

Activate velocity override

N10 POS[U]=10 FDA[U]=100 POSA[V]=20 FDA[V]=150 **. . .**

POS[U]=10	Target position of positioning axis U
FDA[U]=100	Activate velocity override for positioning axis U; the axis velocity of U is 100 mm/min
POSA[V]=20	Target position of positioning axis V
FDA[V]=150	Activate velocity override for positioning axis V; the axis velocity of V is 150 mm/min

Example 2

Activate path definition and velocity override in the same NC block

N20 POS[U]=100 FDA[U]= 0 POS[V]=200 FDA[V]=150 **. . .**

POS[U]=100	Target position of positioning axis U
FDA[U]= 0	Activate path definition for positioning axis U;
POS [V]=200	Target position of positioning axis V
FDA[V]=150	Activate velocity override for positioning axis V; the axis velocity of V is 150 mm/min

Path axis

Syntax for handwheel override: **FD = [feed value]**

To program “handwheel override in Automatic mode” for path axes, the following conditions must be fulfilled:

- Active movement commands from group 1: G01, G02, G03, CIP
- Exact stop active (G60)
- Linear feed in mm/min or inch/min active (G94)

These conditions are checked by the control and an alarm is output if any of them is not met.

Example 3

Activate velocity override

N10 G01 X10 Y100 Z200 FD=1500 **. . .**

X10 Y100 Z200	Target position of path axes X, Y and Z
FD=1500	Activate velocity override for path axes; the path velocity is 1500 mm/min

Concurrent positioning axis

The handwheel override for concurrent positioning axes is activated from the PLC via FC15. The appropriate parameter “Handwheel override ON” is set for this purpose.

If the parameter velocity (F_value) is assigned the value 0, the activated handwheel override acts as a path definition (i.e. the feedrate is not derived from the axial MD: POS_AX_VELO (initial setting for positioning axis velocity)).

References: /FB/, P2, “Positioning axes”
/FB/, P3, “Basic PLC program”

2.5.3 Special features of handwheel override in automatic mode

Velocity display

The velocity display for handwheel override shows the following values:

- Set velocity: programmed velocity
- Actual velocity: Resultant velocity including handwheel override

Effect on transverse axes

If the axis is defined as a transverse axis and DIAMON is active, the handwheel pulses are interpreted as diameter values and traversed as such while handwheel override is active.

Dry run feedrate

If the dry run is active (IS “Activate dry run feedrate” (DB21, ... DBX0.6=1)), the dry run feedrate always applies (SD: DRY_RUN_FEED).
In this way, the axis is traversed at dry run feedrate without the handwheel influencing the programmed target position despite the active handwheel override with path definition (FDA[AXi]=0) (i.e. the path definition is not active).

DRF active

When “Handwheel override in automatic mode” is activated it is important to check whether the function DRF is active (IS “Activate DRF” (DB21, ... DBX0.3=1)). In this case, the handwheel pulses would also cause a DRF offset of the axis. The operator must therefore deactivate DRF first (see Section 2.9).

Feedrate override

The feedrate override does not affect the velocity of the movements produced by the handwheel (exception: 0%). It only acts on the programmed feedrate. With path definition and rapid jogging with handwheel, the axis might not follow the rotation of the handwheel synchronously (especially with a large handwheel pulse weighting), so that the axis lags.

2.6 Third handwheel via actual-value input (840D, 810D)

Function

To date:
It is possible to connect two handwheels to the peripheral interface (X121, 37-pin) on the NCU module using the cable distributor, etc.
840D with SW 4.1 and higher, 810D with SW 2.1 and higher:
It is now possible to connect a third handwheel via a 611D actual-value input that can be selected in a machine data.
A third handwheel could be used, for example, as a contour handwheel.

Comparison of the 3 handwheels

All three handwheels are identical in terms of operating procedures and functionality.
The third handwheel differs from the others only in terms of its connection method.

Connecting the handwheel to the actual-value input

The signals from the handwheel (track A, *A, B, *B, 5V and 0V) must be wired to the actual-value input as follows:

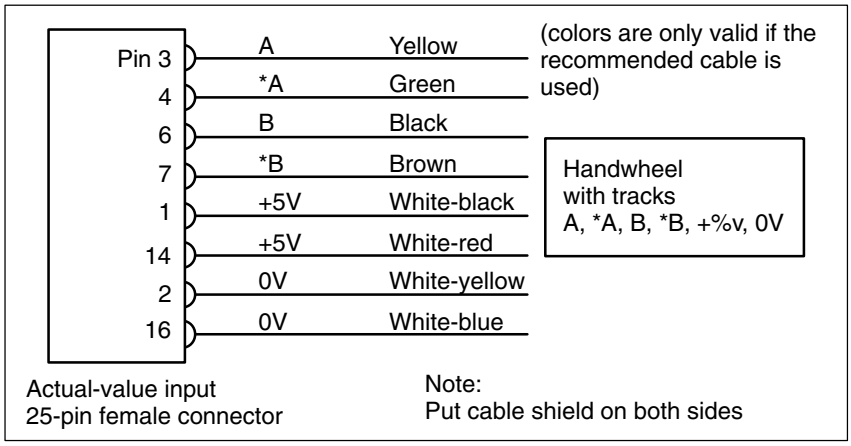


Fig. 2-2 Connecting a handwheel to an actual-value input

Recommendation:

- Use the “Actual-value cable for encoder with voltage signals” (6FX2002–2CG00– ...)
- Separate the cable at the circular connector, remove outer shield and connect to earth potential.
- Apply handwheel signals as shown in Fig. 2-2

Activation, machine data and interface signals

The following machine data and interface signals are required to activate the third handwheel:

- Machine data
  MD 11340: $MN_ENC_HANDWHEEL_SEGMENT_NR
  MD 11342: $MN_ENC_HANDWHEEL_MODULE_NR
  MD 11344: $MN_ENC_HANDWHEEL_INPUT_NR
- Interface signals
  IS “Channel number for handwheel 3” (DB10, DBX99.0, 99.1, 99.2)
  IS “Axis number for handwheel 3” (DB10, DBX102.0 to 2.4)
  IS “Define handwheel 3 as contour handwheel” (DB10, DBX102.5)
  IS “Handwheel 3 selected” (DB10, DBX102.6)
  IS “Machine axis” (DB10, DBX102.7)
  IS “Activate handwheel 3” (DB21, ... , DBX12.2, 16.2, 20.2)
  IS “Handwheel 3 active” (DB21, ... , DBX40.2, 46.2, 52.2)
  IS “Activate handwheel 3” (DB31, ... , DBX4.2)
  IS “Handwheel 3 active” (DB31, ... , DBX64.2)

Supplementary conditions

- The alarm “Handwheel %1 configuration incorrect or inactive” is output to indicate incorrect parameterization of the measuring circuit connection or missing hardware components during POWER ON.
- In contrast to actual-value encoders, no encoder monitors are provided when handwheels are connected. The handwheel pulses disappear in the event of a hardware defect or cable break.
- There is no interlocking to prevent duplicate assignment of an actual-value input, i.e. the input can theoretically be assigned to an actual-value encoder for sensing position or speed or to the “third” handwheel. In this case, “handwheel” pulses are evaluated according to the number of encoder pulses per revolution (coarse increments).
- The third handwheel cannot be operated until the SIMODRIVE 611D bus has powered up.

Machine data

11340 MD number	**$MN_ENC_HANDWHEEL_SEGMENT_NR** Third handwheel: Bus segment		
Default value: 1	Min. input limit: 1	Max. input limit: 1	
Changes effective after Power On	Protection level: 0/0	Unit: –	
Data type: BYTE	Applies from SW version:	840D SW4.1 810D SW2.1	
Significance:	Number of bus segment via which the 3rd handwheel is addressed (encoder connection): 1: 611D drive bus 0, 2, 3: Reserved		
Related to	$MN_ENC_HANDWHEEL_MODULE_NR $MN_ENC_HANDWHEEL_INPUT_NR		

11342 MD number	**$MN_ENC_HANDWHEEL_MODULE_NR** Third handwheel: Drive no./measuring circuit no.		
Default value: 0	Min. input limit: 0	Max. input limit: NCU 572: 15	
Changes effective after Power On	Protection level: 7/2	Unit: –	
Data type: BYTE	Applies from SW version:	840D SW4.1 810D SW2.1	
Significance:	Number of module within a segment ($MN_ENC_HANDWHEEL_SEGMENT_NR) via which the 3rd handwheel is addressed. The logical drive number (see MD 13010: DRIVE_LOGIC_NR) must be entered here on the 611D and the module number on the local bus (count from left to right).		
Special cases, errors, ...	**= 0:** The configuration of a 3rd handwheel is deactivated, in this case the settings in $MN_ENC_HANDWHEEL_SEGMENT_NR and $MN_ENC_HANDWHEEL_INPUT_NR are irrelevant.		
Related to	MD 13010: DRIVE_LOGIC_NR $MN_ENC_HANDWHEEL_SEGMENT_NR $MN_ENC_HANDWHEEL_INPUT_NR		

11344 MD number	**$MN_ENC_HANDWHEEL_INPUT_NR** Third handwheel: Input on module/measuring-circuit card		
Default value: 1	Min. input limit: 1	Max. input limit: 2	
Changes effective after Power On	Protection level: 7/2	Unit: –	
Data type: BYTE	Applies from SW version:	840D SW4.1 810D SW2.1	
Significance:	Number of the input on a module via which the 3rd handwheel is addressed. 840D: 1/2 = upper/lower actual-value input 810D: Always 1		
Related to	$MN_ENC_HANDWHEEL_SEGMENT_NR $MN_ENC_HANDWHEEL_MODULE_NR		

2.7 Contour handwheel / path definition by handwheel (840D, 810D)

Function

When the function is active, the feedrate of the path and synchronous axes can be controlled by a handwheel in AUTOMATIC and MDA modes.

Operating characteristics of function

The following operating characteristics in conjunction with the contour handwheel can be set in MD $MN_HANDWH_TRUE_DISTANCE:

- Path definition:
  Limitation of the axis velocity to the maximum permissible value causes the axes to overtravel. The path defined by the handwheel is traversed, **no pulses are lost**.
- Velocity specification:
  The handwheel specifies only the velocity at which the axes must be traversed. As soon as the handwheel stops, the axes also reach a standstill. The movement is decelerated immediately if no pulses are received from the handwheel within one IPO cycle. As a result, there is **no axis overtravel**. The handwheel pulses do no define the traversing path.

Feedrate

The feedrate in mm/min is dependent on:

- the number of pulses of the selected handwheel within one time period
- the handwheel pulse evaluation method set in MD $MN_HANDWH_IMP_PER_LATCH
- the activated increment (INC1, 10, 100, ...)
- the path weighting of an increment ($MA_JOG_INCR_WEIGHT of first available geometry axis)

The feedrate does not depend on:

- the programmed feed mode (mm/min, mm/rev.)
- the programmed feedrate (resultant velocity might even be higher)
- the rapid traverse rate with respect to G0 blocks
- the override (a setting of 0% is effective, i.e. zero speed)

Travel direction

The travel direction is dependent on the rotational direction:

- **Clockwise rotation:** Axis traverses in the programmed direction
  If the block change criterion (IPO end) is reached, then the next block is inserted (identical response to G60).
- **Anti-clockwise rotation:** Axis traverses in the programmed direction
  In this case, the axis can traverse only up to the start of the next block. The pulses are not picked up if the handwheel continues to rotate.

Activation of function

The function can be activated either by interface signals or by the NC program.

- Activation via interface signal “Activate handwheel x as contour handwheel”
  The function is activated/deactivated via the following interface signal:
  IS “Activate handwheel x as contour handwheel” (DB21, ... DBX30.0, 30.1, 30.2)

- Activation via NC program
  The contour handwheel can be activated non-modally with FD=0 in the NC program, i.e. velocity F from the block in front of the handwheel applies automatically in the following block **without** having to be programmed again.

Note

If the preceding NC blocks do not contain a feedrate, then an appropriate alarm is output.
FD and F in one NC block are mutually exclusive (e.g. generate an alarm).

Simulation of contour handwheel

The contour handwheel can be simulated when it is activated. Once it has been activated via an interface signal, the feedrate is no longer determined by the handwheel, but the programmed feedrate value is applied instead. The direction of travel is specified via an interface signal.

IS “Simulation of contour handwheel” (DB21, ... DBX30.3)
IS “Negative direction simulation of contour handwheel” (DB21, ... DBX30.4)

If	then
the simulation function is deselected,	the current movement is decelerated along a braking ramp
the direction of travel is reversed,	
Note: The override works as for execution of the NC program.	

Supplementary conditions

- **Preconditions**
  Fixed feedrate, dry run feedrate, thread cutting or tapping must not be selected.
- **Limit values**
  The limit values for acceleration and velocity defined in machine data for the relevant axes are applied.
- **Interruption of traversing movement**
  The function remains selected after an NC STOP, but the handwheel pulses are no longer summated and are ineffective (on the condition, however, that MD $MC_HANDWH_CHAN_STOP_COND bit 2 = 1).
- **DRF**
  If selected, a DRF function has an additional path override effect.
- **Channel-specific deletion of distance-to-go**
  This causes the movement initiated by the handwheel to be aborted, the axes are decelerated and the program restarted at the next NC block.
  The contour handwheel then becomes operative again.

Machine data

11346 MD number	**$MN_HANDWH_TRUE_DISTANCE** Path definition or velocity specification by handwheel		
Default value: 1	Min. input limit: 0	Max. input limit: 3	
Changes effective after Power On	Protection level: 7/2	Unit: –	
Data type: BYTE	Applies from SW version:	840D SW4.1 810D SW2.1	
Significance:	Setting the operating characteristics for traversal with handwheel, contour handwheel or FDA=0: **Value = 1:** The handwheel pulses define the traversing path. No pulses are lost. Axis overtravel occurs as a result of limiting the velocity to its maximum permissible value. **Value = 0:** The handwheel pulses specify the axis travel velocity. As soon as the handwheel stops, the axes also reach a standstill. The movement is decelerated immediately if no pulses are received from the handwheel within one IPO cycle. As a result, the axes do not overtravel. The handwheel pulses do no define the traversing path. **Value = 2** The inputs from the handwheel are velocity inputs. As soon as the handwheel stops, the axes also reach a standstill. The movement is decelerated immediately, value = 0, but not via the shortest possible path, but to the next possible path on an imaginery grid.This grid corresponds in each case to a distance travelled by the selected axis per handwheel detent position (see $MA_JOG_INCR_WEIGHT and $MN_JOG_INCR_SIZE_TAB, $MC_HANDWH_GEOAX_MAX_INCR_SIZE, $MA_HANDWH_MAX_INCR_SIZE). The start point of the traversing motion is assumed to be the grid zero point. **Value = 3** The inputs from the handwheel are path inputs. If the axis needs to be decelerated prematurely owing to the settings in other MD ($MN_HANDWH_REVERSE = 0, $MC_HANDWH_CHAN_STOP_COND, $MA_HANDWH_STOP_COND), then, unlike value = 1, the axis is not decelerated via the shortest possible path, but to the next possible point on an imaginery grid.		

Interface signals

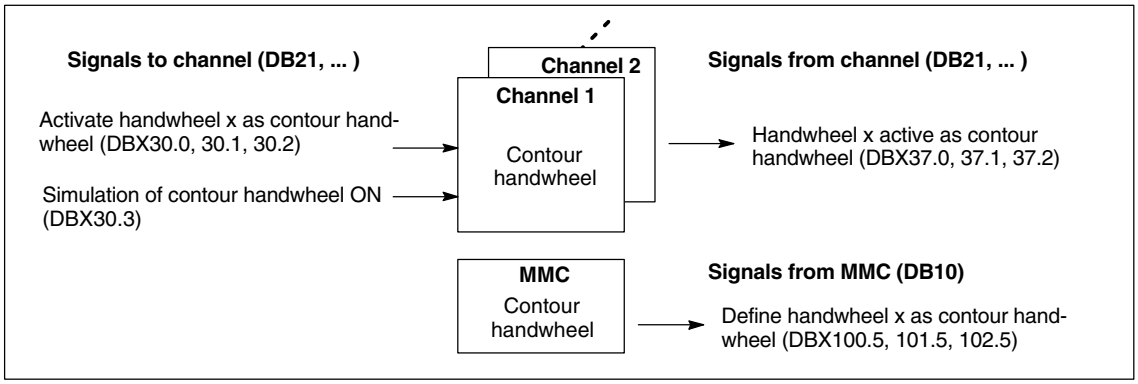


Fig. 2-3 Overview of interface signals for contour handwheel

DB 21, 22, ... **DBX30.0** **DBX30.1** **DBX30.2** Data block	**Activate handwheel 1 as contour handwheel** **Activate handwheel 2 as contour handwheel** **Activate handwheel 3 as contour handwheel** Signal(s) to channel (PLC ——> NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 840D SW4.1, 810D SW2.1
Description	These signals allow one of the three handwheels to be selected/deselected as the contour handwheel. Signal = 1 Handwheel x is selected as the contour handwheel Signal = 0 Handwheel x is deselected as the contour handwheel The contour handwheel can be selected/deselected in the middle of a block. When the handwheel is activated, the axis movement is first decelerated and then traversed as determined by the handwheel. When the handwheel is deactivated, the movement is first decelerated before execution of the NC program continues. If the NC program must not continue until after a new NC-START, then deactivation of the contour handwheel must be linked to an NC-STOP in the PLC user program.	
Special cases, errors, ...	The signal setting remains valid after an NC-RESET.	
Related to	IS “Handwheel x active as contour handwheel” (DB21, 22, ... , DBX37.0, 37.1, 37.2)	

DB 21, 22, ... **DBX30.3** **DBX30.4** Data block	**Simulation of contour handwheel ON** **Negative direction simulation of contour handwheel** Signal(s) to channel (PLC ——> NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 840D SW4.1, 810D SW2.1
Description	To activate/deactivate simulation of the contour handwheel and to set the traversing direction, these signals must be set as follows: Bit 3 Bit 4 Meaning 0 0 Simulation OFF 0 1 Simulation OFF 1 0 Simulation ON, direction as programmed 1 1 Simulation ON, opposite direction to programmed direction During simulation, the feedrate is not determined by the contour handwheel, but the axis is traversed along the contour at the programmed feedrate. If the function is deselected, the current axis movement is decelerated along a braking ramp. When the traversing direction is reversed, the current axis movement is decelerated along a braking ramp and the axis then traversed in the opposite direction.	
Special cases, errors, ...	The simulation function is available only in AUTOMATIC mode and can only be activated if the contour handwheel is already active.	

DB 21, 22, ... **DBX37.0** **DBX37.1** **DBX37.2** Data block	**Handwheel 1 active as contour handwheel** **Handwheel 2 active as contour handwheel** **Handwheel 3 active as contour handwheel** Signal(s) from channel (NCK ——> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 840D SW4.1, 810D SW2.1
Description	These signals indicate which handwheel is selected as the contour handwheel. Signal = 1 Handwheel x is selected as the contour handwheel Signal = 0 Handwheel x is deselected as the contour handwheel	
Special cases, errors, ...	The signal setting remains valid after an NC-RESET.	
Related to	IS “Activate handwheel x as contour handwheel” (DB21, 22, ..., DBX30.0, 30.1, 30.2)	

DB10 **DBX100.5** **DBX101.5** **DBX102.5** Data block	**Define handwheel 1 as contour handwheel** **Define handwheel 2 as contour handwheel** **Define handwheel 3 as contour handwheel** Signal(s) from MMC (MMC –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 840D SW4.1, 810D SW2.1
Description	These signals indicate which handwheel is defined as the contour handwheel via the MMC. Signal = 1 Handwheel x is defined as the contour handwheel via the MMC Signal = 0 Handwheel x is not defined as the contour handwheel To ensure that the handwheel defined via the MMC can operate as the contour handwheel, the appropriate signal must be gated with IS “Activate handwheel x as contour handwheel” (DB21, 22, ... ,DBX30.0, 30.1, 30.2).	
Special cases, errors, ...	Depending on the setting of parameter HWheelMMC in FB1 of the basic PLC program, these signals are supplied by the basic program or must be supplied by the PLC user program.	
Related to	IS “Activate handwheel x as contour handwheel” (DB21, 22, ..., DBX30.0, 30.1, 30.2) FB1 parameter HWheelMMC	

2.8 Special features of JOG mode

2.8.1 Geometry axes in JOG mode

Coordinate systems in JOG mode

In JOG mode the operator can also jog the axes declared as geometry axes in the workpiece coordinate system. Any coordinate offsets or rotations that have been selected remain active.

The special features of jogging geometry axes are described below.

The geometry axes are traversed in the coordinate system that was last activated.

Simultaneous jogging

Only one geometry axis can be jogged continuously or incrementally using the traverse keys. Where an attempt is made to jog more than one geometry axis, alarm 20062 “Axis already active” is output. If the axis is not defined as a geometry axis, alarm 20060 “Axis cannot be traversed as geometry axis” is output.
However, 3 geometry axes can be jogged simultaneously with handwheels 1, 2 and 3.

PLC interface

The geometry axes have their own PLC interface (DB21, ... DBB12-23 and DBB40-56) which contains the same signals as the axis-specific PLC interface.

Feedrate/rapid traverse override

The channel-specific feedrate override switch and rapid traverse override switch are active for jogging geometry axes.

Alarms

Alarm 20062 “Axis already active” is triggered when a geometry axis is being jogged under the following conditions:

- The axis is already being traversed in JOG mode via the axial PLC interface.
- A frame for a rotated coordinate system is already active and another geometry axis in this coordinate system is traversed in JOG mode with the traverse keys.

If the axis has not been defined as a geometry axis, alarm 20060 is triggered if an attempt is made to traverse it as a geometry axis in JOG mode.

Use

Jogging movements for which transformations and frames have to be active.

2.8.2 Special features of spindle jogging

Spindle traversal in JOG mode

Spindles can also be traversed manually in JOG mode. Essentially the same conditions apply as for manual traverse of axes. Spindles can be traversed in JOG mode using the traverse keys continuously or incrementally, in continuous-trigger or momentary-trigger mode, or with the handwheel. The function is selected and activated via the axis/spindle-specific PLC interface in the same way as for the machine axes. The axis-specific machine axes also apply to the spindles.

Spindle mode

The spindle can be jogged in positioning mode (spindle is in position control) or in open-loop control mode.

JOG speed

The speed used for jogging spindles can be defined as follows:

- With the general SD: JOG_SPIND_SET_VELO (JOG speed for spindle) which is activated for all spindles jointly
- Or with machine data JOG_VELO (JOG axis velocity). However, this MD only has an effect if SD: JOG_SET_VELO (JOG velocity for G94) = 0.

The maximum speed for the active gear stage also applies when spindles are traversed in JOG mode.

References: /FB/, S1, “Spindles”

Speed override

The spindle speed override switch can be used to modify the speed of spindles traversed in JOG mode.

JOG acceleration

Because a spindle often uses many gear stages in speed control and position control mode, the acceleration programmed for the current gear stage is always applied in spindle JOG mode.

References: /FB/, S1, “Spindles”

PLC interface signals

When spindles are traversed manually, the PLC interface signals between the NCK and PLC have the same effect as for machine axes. IS “Position reached with exact stop fine or coarse” (DB31, ... DBX60.7 or DBX60.6) is only set if the spindle is in position control mode.
For purely spindle-specific interface signals the following should be noted when traversing spindles in JOG:

- The following PLC interface signals to the spindle have no effect:
  - IS “Invert M3/M4” (DB31, ... DBX17.6)
  - IS “Set direction of rotation ccw” or “Set direction of rotation cw” (DB31, ... DBX18.7 or DBX18.6)
  - IS “Oscillation speed” (DB31, ... DBX18.5)
  - IS “Spindle RESET” (DB31, ... DBX16.7)
- The following PLC interface signals from the spindle are not set:
  - IS “Actual speed cw” (DB31, ... DBX83.7)
  - IS “Spindle in set range” (DB31, ... DBX83.5)

2.8.3 Monitoring functions

Limitations

The following limitations are active in JOG mode:

- Working area limitation (axis must be referenced)
- Software limit switches 1 and 2 (axis must be referenced)
- Hardware limit switches

The control ensures that the traversing movement is aborted as soon as the first valid limitation has been reached. Velocity control ensures that deceleration is initiated early enough for the axis to stop exactly at the limitation position (e.g. software limit switch). Only when the hardware limit switch is triggered does the axis stop abruptly with “Rapid stop”.

Alarms are triggered when the various limitations are reached (alarms 16016, 16017, 16020, 16021). The control automatically prevents further movement in this direction. The traverse keys and the handwheel have no effect in this direction.

!

Important

The software limit switches and working area limitations are only active if the axis is first referenced.

For further information on working area limitations and hardware and software limit switches see:
References: /FB/, A3, “Axis/Contour Tunnel Monitoring, Protection Zones”

Axis retraction

The axis can be retracted from a limitation position by moving it in the opposite direction.

Machine manufacturer

The function for retracting an axis that has reached the limitation position depends on the machine-tool manufacturer. Please refer to the machine-tool manufacturer's documentation!

Maximum velocity and acceleration

The velocity and acceleration values applied in JOG mode are programmed for specific axes via machine data by the start-up engineer. The control limits the value for the values valid for the axes to the maximum velocity and acceleration specifications.

References: /FB/, G2, "Velocities, Setpoint/Actual-Value Systems, Closed-Loop Controls"
/FB/, B2, "Acceleration"

2.8.4 Miscellaneous

Switching modes from JOG→AUT or from JOG→MDA

It is possible to switch operating modes from JOG to AUT or MDA only if all axes in the channel have reached "Exact stop coarse".

References: /FB/, K1, "Mode Group, Channels, Program Operation Mode"

Rotational feedrate active in JOG

In JOG mode, it is also possible to traverse an axis at a rotational feedrate (analogous to G95) referred to the current speed of the master spindle. The function is activated with SD: JOG_REV_IS_ACTIVE (JOG in revolutional feedrate).
The feedrate value (in mm/rev) used can be defined as follows:

- With general SD: JOG_REV_SET_VELO (JOG speed for G95) if this is not equal to 0.
- If the value 0 is set in SD: JOG_REV_SET_VELO, then the rotational feedrate is determined by axial machine data JOG_REV_VELO (rotational feedrate for JOG) or, in the case of rapid traverse override, by JOG_REV_VELO_RAPID.

If a master spindle has not been defined and the axis is to be traversed in JOG at revolutional feedrate, alarm 20055 and for geometry axes alarm 20065 is output.

Transverse axes

If a geometry axis is defined as a transverse axis and radius programming is selected (MD: DIAMETER_AX_DEF (geometry axes with transverse axis function)), the following must be noted when traversing in JOG:

- Continuous jogging:
  There are no differences when a transverse axis is traversed in continuous mode.
- Incremental jogging:
  Only **half the distance** of the selected increment size is traversed. For example, with INC10 the axis only traverses 5 increments when the traverse key is pressed.
- Jogging with the handwheel
  As for incremental jogging, only half the path is traversed per handwheel pulse.

References: /FB/, P1, “Transverse Axes”

2.9 DRF

Function DRF

With the function DRF (Differential Resolver Function) an additional incremental zero offset can be activated with the electronic handwheel in automatic mode during machining.
The same conditions apply to the handwheel assignment, pulse weighting, etc., as for manual traverse with the handwheel in JOG mode (see Section 2.4). In addition, the velocity generated by the handwheel in DRF can be reduced from the JOG velocity with the axial MD: HAND_VELO_OVERLAY_FACTOR (ratio of JOG velocity to handwheel velocity).

DRF offset

The DRF offset is an axis zero offset that is generated by means of DRF (i.e. traversing movement with handwheel in automatic mode). The DRF offset is active in the basic coordinate system.

Caution!

Zero offset with DRF is always active; i.e. for all modes and after RESET. It can, however, be suppressed non-modally in the part program.

References: /PA1/, Programming Guide Fundamentals

Uses

DRF is used for the following applications:

→ Compensating for tool wear within an NC block. Where NC blocks have a long machining time it might be necessary to compensate for tool wear manually within the NC block (e.g. large surface milling machines).

→ Highly precise compensation during grinding

→ Very simple temperature compensation

→ Offsets that are not included in the actual-value display

DRF active

DRF must be active to allow the DRF offset to be modified through traversal with the handwheel. The following conditions must be fulfilled:

- AUTOMATIC mode is selected (channel in RESET/interrupted/active)
- and IS “Activate DRF” (DB21, ... DBX0.3) = 1

DRF offset can be switched off for specific channels by the operator with the program control function. This is reported by the MMC with IS “DRF selected” (DB21, ... DBX24.3) to the PLC. The PLC program (basic PLC program or user program) transfers this interface signal as IS “Activate DRF” after logic combination.

Control of DRF offset

The DRF offset can be modified, deleted or read in the following ways (see Fig. 2-4):

- By the operator by jogging with the handwheel
- By the NC part program (in high-level language)
  - Reading the DRF offset (axis-specific)
  - Deleting the DRF offset for all axes in a channel (command “DRFOF”)

References: /PA1/, Programming Guide Fundamentals

- From PLC user program
  - Reading the DRF offset (axis-specific)

References: /FB/, P3,

“Basic PLC Program”

- From the MMC by the operator
  - Reading the DRF offset (axis-specific)

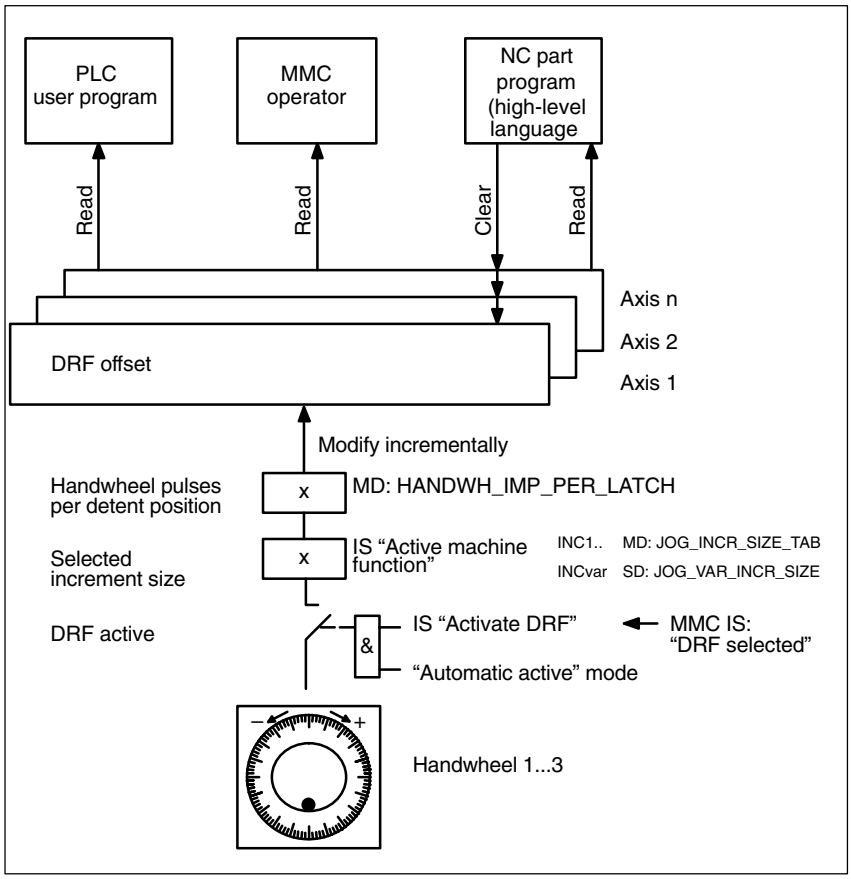


Fig. 2-4 Control of DRF offset

Note

If DRF offset is cleared the axis is not traversed!

Power On

The DRF offset is deleted by a Power On.

Reference point approach

If an axis with a DRF offset is referenced, the offset is deleted during phase 1 of the referencing operation!
It is not possible to specify a DRF offset with the handwheel while an axis is being referenced (e.g. with G74). Alarm 20053 “DRF not possible” is triggered.

Display

The axis position display (ACTUAL POSITION) does not change while an axis is being traversed with the handwheel in DRF. The current DRF offset can be displayed in the DRF window.

2.10 Installation and start-up

Note

Before installation can begin several conditions must be fulfilled. For procedure please see:

References: /IAD/, “Installation and Start-up Guide”
/IAF/, “Installation and Start-up Guide”

Machine/ setting data

The machine and/or geometry axes can be traversed manually only if specific machine/setting data have been preset. The machine and setting data that apply specifically to manual traverse are listed below. A description of these data together with their default settings is given in Section 4.

JOG continuously in jog mode

General SD: JOG_CONT_MODE_LEVELTRIGGRD (JOG continuously in jog mode)

INC and REF in jog mode

General MD: JOG_INC_MODE_LEVELTRIGGRD (INC and REF in jog mode)

Velocity

Axial MD: JOG_VELO (JOG axis velocity)
Axial MD: JOG_VELO_RAPID (JOG rapid traverse)
General SD: JOG_SET_VELO (JOG velocity for G94)
General SD: JOG_ROT_AX_SET_VELO (JOG velocity for rotary axes)

Revolutional feedrate

General SD: JOG_REV_IS_ACTIVE (revolutional feedrate active in JOG)
Axial MD: JOG_REV_VELO (revolutional feedrate for JOG)
Axial MD: JOG_REV_VELO_RAPID (revolutional feedrate for JOG with rapid traverse override)
General SD: JOG_REV_SET_VELO (JOG speed for G95)

Acceleration

Axial MD: AX_JERK_DEFAULT (initial setting for axial jerk limitation)

Incremental/ handwheel	Axial MD:	JOG_INCR_WEIGHT (weighting of an increment for INC/handwheel)
	General SD:	JOG_VAR_INCR_SIZE (size of a variable increment for INC/handwheel)
	Axial MD:	HANDWH_VELO_OVERLAY_FACTOR (ratio JOG velocity to handwheel velocity (DRF))
	General MD:	JOG_INCR_SIZE_TAB [n] (increment size for INC/handwheel)
	General MD:	HANDWH_IMP_PER_LATCH [n] (handwheel pulses per detent position [handwheel number])
Spindle	General SD:	JOG_SPIND_SET_VELO (JOG speed for spindle)

■

Supplementary Conditions 3

Availability of function "Handwheel override in automatic mode"

The function is available on

- SINUMERIK FM-NC with NCU 570, with SW 2 and higher
- SINUMERIK 840D with NCU 571/572/573 with SW 2 and higher

SINUMERIK 840Di handwheels

Two handwheels can be connected to the SINUMERIK 840Di via the MCI Board Extension module on the SINUMERIK 840Di.
The handwheels are connected via the 25-pin cable distributor interface (X121) on the MCI Board Extension module.

Data Descriptions (MD, SD) 4

4.1 General machine data

11300 MD number	**JOG_INC_MODE_LEVELTRIGGRD** INC and REF in jog mode		
Default value: 1	Min. input limit: 0	Max. input limit: 1	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: BOOLEAN	Applies from SW version: 1.1		
Significance:	1: Jog mode for JOG-INC and reference point approach JOG-INC: When the traverse key is pressed in the required direction (e.g. +) the axis begins to traverse the set increment. If the key is released before the increment has been completely traversed, the movement is interrupted and the axis stops. If the same key is pressed again, the axis completes the remaining distance-to-go until it is 0. 0: Continuous mode for JOG-INC and reference point approach JOG-INC: When the traverse key is pressed (first rising edge) the axis traverses the whole set increment. If the same key is pressed again (second rising edge) before the axis has completed traversing the increment, the movement is aborted, i.e. not completed. The differences in axis traversing characteristics between the jog and continuous modes in JOG-INC are described in detail in Section 2.3. For travel behavior in reference point approach see **References:** /FB/, R1, "Reference Point Approach"		
MD irrelevant for	Continuous jogging (JOG continuous)		

11310 MD number	**$MN_HANDWH_REVERSE** Threshold for direction reversal for handwheel		
Default value: 2	Min. input limit: 0	Max. input limit: –	
Changes effective after Power On	Protection level: 2/7	Unit: –	
Data type: BYTE	Applies from SW version: 3.2		
Significance:	0: No immediate movement in the opposite direction >0: Immediate movement in the opposite direction if the handwheel is turned in the opposite direction by at least the number of pulses indicated		

11320 MD number	**HANDWH_IMP_PER_LATCH [n]** Handwheel pulses per detent position [handwheel number]: 0 ... 2		
Default value: 1	Min. input limit: ***	Max. input limit: ***	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	MD: HANDW_IMP_PER_LATCH. The number of pulses generated by the handwheel for each handwheel detent position is entered. The handwheel pulse weighting must be defined for each connected handwheel (1 to 3) separately. When adapted to the control, each handwheel detent position has the same effect as one press of the traverse key in incremental jogging mode. If a negative value is entered the handwheel is active in the reverse direction.		
Related to	MD: JOG_INCR_WEIGHT (weighting of an increment of a machine axis for INC/manual).		

11330 MD number	**JOG_INCR_SIZE_TAB [n]** Increment size for INC/handwheel [increment index]: 0 ... 4		
Default value: 1; 10; 100; 1000; 10000	Min. input limit: 0	Max. input limit: plus	
Changes effective after Power On	Protection level: 2	Unit: Linear axis: mm Rotary axis: degrees	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	In incremental jogging of handwheel jogging the number of increments to be traversed by the axis can be defined by the operator, e.g. via the operator panel. In addition to the variable increment sizes (INCvar) 5 fixed increment sizes (INC...) can also be set. The increment size for each of these 5 fixed increments is defined for all axes by entering values in JOG_INCR_SIZE_TAB [n]. The default setting is INC1, INC10, INC100, INC1000 and INC10000. The entered increment sizes are also active for DRF. The size of the variable increment is defined in SD: JOG_VAR_INCR_SIZE.		
Related to	MD: JOG_INCR_WEIGHT(weighting of an increment for INC/manual) IS “Active machine function: INC1; ...; INC10000” (DB21–28, DBB41 ff) IS “Active machine function: INC1; ...; INC10000” (DB31–48, DBB69).		

4.2 Channel-specific machine data

20620 MD number	**$MC_HANDWH_GEOAX_MAX_INCR_SIZE** Limitation of handwheel increment for geo axes		
Default value: 0	Min. input limit: 0	Max. input limit: –	
Changes effective after Power On	Protection level: 2/7	Unit: mm	
Data type: DOUBLE	Applies from SW version: 3.2		
Significance:	>0: Limitation of the size of the selected increment $MN_JOG_INCR_SIZE[<increment/VDI signal>] or $SN_JOG_VAR_INCR_SIZE for geometry axes 0: No limitation for geometry axes		

20622 MD number	**$HANDWH_GEOAX_MAX_INCR_VSIZE** Path velocity override		
Default value: 500	Min. input limit: 0	Max. input limit: plus	
Changes effective after Power On	Protection level: 2 / 7	Unit: mm/min	
Data type: DOUBLE	Applies from SW version: 3.2		
Significance:	For the velocity override of the path: > 0: Limitation of size of selected increment ($MN_JOG_INCR_SIZE_[<Increment/VDI signal>] or $SN_JOG_VAR_INCR_SIZE) / 1000*IPO sampling time =0: No limitation		

20624 MD number	**$MC_HANDWH_CHAN_STOP_COND** Definition of operating characteristics in jogging with handwheel		
Default value: 0x3FF, 0x3FF, 0x3FF, ...	Min. input limit: 0	Max. input limit: 0xFFF	
Changes effective after Power On	Protection level: 2/7	Unit: –	
Data type: DWORD	Applies from SW version: 3.2		
Significance:	Definition of the behavior of jogging with handwheel on channel-specific VDI interface signals: Bit==0 Interruption or collection of the distances preset via the handwheel Bit==1: Abort of the traversing movement or no collection Bit allocation Bit 0: Mode group stop Bit 1: Mode group stop axes plus spindle Bit 2: NC stop Bit 3: NC stop axes plus spindles Bit 4: Feedrate disable Bit 5: Feedrate override Bit 6: Rapid traverse override Bit 7: Feed stop geometry axis Bit 7=0: Interruption/collection Bit 7=1: Abort traversing movement/no collection Setting for geometry axes Bit 8 = 0 For JOG with handwheel, the maximum possible velocity corresponds to the feedrate set in MD 32020: JOG_AX_VELO for the appropriate machine axis/axes. =1 For JOG with handwheel, the maximum possible velocity corresponds to the feedrate set in MD 32000: MAX_AX_VELO for the appropriate machine axis/axes. Bit 9 = 0 The override is active in JOG mode with handwheel. = 1 The override is always assumed to be 100% for JOG mode with handwheel regardless of how the override switch is set. Exception: The override 0% is always active. Setting for DRF for all axes of the channel Bit 10 = 0 With DRF MD 11310: HANDWH_REVERSE is not active, i.e. the behavior is the same as with MD 11310 = 0. =1 With DRF MD 11310: HANDWH_REVERSE is active. Setting for the contour handwheel Bit 11 = 0 When the contour handwheel is deactivated, program execution is automatically continued. Bit 11 = 1 When the contour handwheel is deactivated, an NC stop is automatically initiated. Only after input of NC-START, can the program execution be continued.		

4.3 Axis/spindle-specific machine data

31090 MD number	**JOG_INCR_WEIGHT** Evaluation of an increment for INC/handwheel		
Default value: 0.001	Min. input limit: ***	Max. input limit: ***	
Changes effective after Power On	Protection level: 2	Unit: Linear axis: mm Rotary axis: degrees	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	The **path of an increment** which applies when an axis is traversed with the JOG keys in incremental mode or with the handwheel is defined in this MD. The path covered by the axis on each increment each time the direction key is pressed or for each handwheel position is defined by the following parameters: • MD: JOG_INCR_WEIGHT (weighting of an increment of a machine axis for INC/handwheel) • Selected increment size (INC1, ..., INCvar) The possible increment stages are defined globally for all axes in MD: JOG_INCR_SIZE_TAB [n] and in SD: JOG_VAR_INCR_SIZE. Entering a negative value reverses the direction of the traverse keys and the handwheel rotation. **SW 5 and higher:** Default settings: JOG_INCR_WEIGHT[0]=0.001 mm (valid in metric measuring system) JOG_INCR_WEIGHT[1]=0.00254 mm (valid in inch measuring system and corresponds to 0.0001 inch)		
MD irrelevant for	Operating modes AUTOMATIC and MDA		
Related to	MD: JOG_INCR_SIZE_TAB SD: JOG_VAR_INCR_SIZE		

32010 MD number	**JOG_VELO_RAPID** Rapid traverse in JOG mode		
Default value: 10000	Min. input limit: 0	Max. input limit: plus	
Changes effective after Power On	Protection level: 2	Unit: Linear axis: mm/min Rotary axis: mm/rev	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	The axis velocity entered here applies when the rapid traverse override key is operated in JOG mode and when the axial feedrate override switch is set to 100%. The value entered must not exceed the maximum permissible axis velocity (machine data MAX_AX_VELO). This machine data is not used for the programmed rapid traverse G00.		
MD irrelevant for	Operating modes AUTOMATIC and MDA		
Related to	MD: MAX_AX_VELO (maximum axis velocity) MD: JOG_REV_VELO_RAPID (revolutional feedrate for JOG with rapid traverse override) IS “Rapid traverse override” (DB21–28, DBX12.5 ff) IS “Feedrate override” (DB21–28, DBB4)		

32020 MD number	**JOG_VELO** Axis velocity in JOG mode		
Default value: 2000	Min. input limit: 0	Max. input limit: plus	
Changes effective after Power On	Protection level: 2	Unit: Linear axis: mm/min Rotary axis: mm/rev	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	The value entered is the velocity traversed in JOG mode when the axial feedrate override switch is on position 100%. This velocity is only used when general setting data JOG_SET_VELO = 0 is set for linear axes and linear feedrate (MD: JOG_REV_IS_ACTIVE = 0) or SD: JOG_ROT_AX_SET_VELO = 0 is set for rotary axes. If this is the case, the axis velocity is active for – continuous jogging – incremental jogging (INC1, ... INCvar) – handwheel jogging The value entered must not exceed the maximum permissible axis velocity (machine data MAX_AX_VELO). If DRF is active, the axis velocity for JOG must be reduced with MD: HANDWH_VELO_OVERLAY_FACTOR. Spindles in JOG mode: This machine data can also be used to define the JOG mode velocity for specific spindles (if SD: JOG_SPIND_SET_VELO = 0). However, the velocity can be modified with the spindle speed override switch.		
Application	If different velocities/speeds have to be set for the individual axes/spindles traversing in JOG mode, this can be done for specific axes in this MD. SD: JOG_SET_VELO must be set to 0!		
Related to	MD : MAX_AX_VELO (maximum axis velocity) MD: JOG_REV_VELO (revolutional feedrate for JOG) MD: HANDWH_VELO_OVERLAY_FACTOR (ratio JOG velocity to handwheel velocity (DRF)) SD: JOG_SET_VELO (JOG velocity for G94) SD: JOG_ROT_AX_SET_VELO (JOG velocity for rotary axes) IS “Feedrate override” (DB21–28, DBB4)		

32040 MD number	**JOG_REV_VELO_RAPID** Revolutional feedrate in JOG mode with rapid traverse override		
Default value: 2,5	Min. input limit: 0	Max. input limit: plus	
Changes effective after Power On	Protection level: 2	Unit: mm/rev	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	The value entered in this MD defines the revolutional feedrate of the axis in JOG mode with rapid traverse override referred to the revolutions of the master spindle. This feedrate is active when SD: JOG_REV_IS_ACTIVE = 1. (Revolutional feedrate active with JOG.)		
MD irrelevant for	SD: JOG_REV_IS_ACTIVE = “0”		
Related to	SD: JOG_REV_IS_ACTIVE (revolutional feedrate for JOG active) MD: JOG_REV_VELO (revolutional feedrate with JOG)		

32050 MD number	**JOG_REV_VELO** Revolutional feedrate in JOG mode		
Default value: 0.5	Min. input limit: 0	Max. input limit: plus	
Changes effective after Power On	Protection level: 2	Unit: mm/rev	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	The value entered in this MD defines the revolutional feedrate of the axis in JOG mode referred to the revolutions of the master spindle. This feedrate is active when SD: Revolutional feedrate active with JOG mode JOG_REV_IS_ACTIVE = 1.		
MD irrelevant for	Linear feedrate; i.e. SD: JOG_REV_IS_ACTIVE = 0		
Related to	SD: JOG_REV_IS_ACTIVE (revolutional feedrate for JOG active) MD: JOG_REV_VELO_RAPID (JOG revolutional feedrate with rapid traverse)		

32080 MD number	**$MA_HANDWH_MAX_INCR_SIZE** Limitation of selected increment		
Default value: 0	Min. input limit: 0	Max. input limit: –	
Changes effective after Reset	Protection level: 2/7	Unit:	
Data type: DOUBLE	Applies from SW version: 3.2		
Significance:	>0: Limitation of size of selected increment $MN_JOG_INCR_SIZE[<Increment/VDI signal>] or $SN_JOG_VAR_INCR_SIZE for the associated machine axis 0: No limitation		

32082 MD number	**$MA_HANDWH_MAX_INCR_VELO_SIZE** Limitation of selected increment for velocity override		
Default value: 500	Min. input limit: 0	Max. input limit: plus	
Changes effective after Reset	Protection level: 2/7	Unit: mm/min	
Data type: DOUBLE	Applies from SW version: 3.2		
Significance:	For the velocity override of positioning axes: >0: Limitation of size of selected increment $MN_JOG_INCR_SIZE[<Increment/VDI signal>] or $SN_JOG_VAR_INCR_SIZE for the associated machine axis 0: No limitation		

32084 MD number	**$MA_HANDWH_STOP_COND** Control of VDI signals in relation to handwheel		
Default value: 0xFF	Min. input limit: 0	Max. input limit: 0xFF	
Changes effective after RESET	Protection level: 2/7	Unit: –	
Data type: DWORD	Applies from SW version: 3.2		
Significance:	Definition of the behavior of the jogging with handwheel on axis-specific VDI interface signals: Bit==0 Interruption or collection of the distances preset via the handwheel Bit==1: Abort of the traversing movement or no collection Bit allocation Bit 0: Feedrate override Bit 1: Spindle speed override Bit 2: Feedrate stop/spindle stop Bit 3: Clamping procedure running (==0 no effect) Bit 4: Controller enable Bit 5: Pulse enable For machine axis Bit 6 = 0 For JOG with handwheel, the maximum possible velocity corresponds to the feedrate set in MD 32020: JOG_VELO for the appropriate machine axis. =1 For JOG with handwheel, the maximum possible velocity corresponds to the feedrate set in MD 32000: MAX_AX_VELO for the appropriate machine axis. Bit 7 = 0 The override is active in JOG mode with handwheel. = 1 The override is always assumed to be 100% for JOG mode with handwheel regardless of how the override switch is set. Exception: The override 0% is always active.		

32090 MD number	**HANDWH_VELO_OVERLAY_FACTOR** Ratio JOG velocity to handwheel velocity (DRF)		
Default value: 0.5	Min. input limit: 0	Max. input limit: plus	
Changes effective after NEW_CONF	Protection level: 2	Unit: –	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	The velocity active with the handwheel in DRF can be reduced in relation to the JOG velocity with this machine data. The following applies for the velocity active with DRF: v_{DRF} = SD:JOG_SET_VELO * MD:HANDWH_VELO_OVERLAY_FACTOR or when SD:JOG_SET_VELO = 0: v_{DRF} = MD:JOG_VELO * MD:HANDWH_VELO_OVERLAY_FACTOR The velocity setting in SD: JOG_ROT_AX_SET_VELO applies for DRF on rotary axes instead of the value in SD: JOG_SET_VELO.		
MD irrelevant for	JOG handwheel		
Related to	MD: JOG_VELO (JOG axis velocity) SD: JOG_SET_VELO (JOG velocity for G94) SD: JOG_AX_SET_VELO (JOG velocity for rotary axes)		

4.4 General setting data

41010 SD number	**JOG_VAR_INCR_SIZE** Size of variable increment for INC/handwheel		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: MMC-MD 9220	Unit: mm or degrees	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	With this setting data the number of increments when variable increment (INCvar) is selected is defined. This increment size is traversed by the axis in JOG mode whenever the traverse key is pressed or the handwheel is turned one detent position and variable increment is selected (PLC interface signal “Active machine function: INC variable” for machine or geometry axesis set to 1). The defined increment size also applies to DRF. **Note:** Please note that the increment size is active for incremental jogging **and** handwheel jogging. So if a large increment value is entered and the handwheel is turned the axis might cover a large distance (depends on setting in MD: JOG_INCR_WEIGHT).		
SD irrelevant for	JOG continuous		
Related to	IS “Active machine function: INC variable” (DB21–28, DBX41.5 ff) or IS “Active machine function; INC variable” (DB31 – 48, DBX 69.5) MD: JOG_INCR_WEIGHT (weighting of an increment for INC/handwheel)		

41050 SD number	**JOG_CONT_MODE_LEVELTRIGGRD** Continuous JOG in jog mode		
Default value: 1	Min. input limit: 0	Max. input limit: 1	
Changes effective immediately	Protection level: MMC-MD 9220	Unit: –	
Data type: BOOLEAN	Applies from SW version: 1.1		
Significance:	1: Jog mode for JOG continuous In jog mode (default setting) the axis traverses for as long as the traverse key is held down and an axis limitation has not been reached. When the key is released the axis is decelerated to zero velocity and the movement is considered completed. 0: Continuous mode for JOG continuous In continuous mode the traverse movement is started with the first rising edge of the traverse key and continues to move when the key is released. The axis can be stopped again by pressing the traverse key again (second rising edge). The differences in axis traversing characteristics between the jog and continuous modes in JOG are described in detail in Section 2.1.		
SD irrelevant for	Incremental jogging (JOG INC) Reference point approach (JOG REF)		

41100 SD number	**JOG_REV_IS_ACTIVE** Revolutional feedrate for JOG active		
Default value: 1	Min. input limit: 0	Max. input limit: 1	
Changes effective immediately	Protection level: MMC-MD 9220	Unit: –	
Data type: BOOLEAN	Applies from SW version: 1.1		
Significance:	1: The axis (machine or geometry axis) is traversed in JOG mode at revolutional feedrate (G95) referred to the revolutions of the main spindle. The revolutional feedrate can be set as follows: – With global SD: JOG_REV_SET_VELO (only active when SD is not equal to 0) – With axial MD: JOG_REV_VELO – With rapid traverse override with axial MD: JOG_REV_VELO_RAPID 0: The axis is traversed in JOG mode at linear feedrate (G94). The revolutional feedrate can be set as follows: – With global SD: JOG_SET_VELO (only active when SD is not equal to 0) – With axial MD: JOG_VELO – With rapid traverse override with axial MD: JOG_VELO_RAPID		
SD irrelevant for	Operating modes AUTOMATIC and MDA		
Related to	SD: JOG_REV_SET_VELO (JOG velocity for G95) MD: JOG_REV_VELO (revolutional feedrate with JOG) MD: JOG_REV_VELO_RAPID (JOG revolutional feedrate with rapid traverse) SD: JOG_SET_VELO (JOG velocity for G94) MD: JOG_VELO (JOG axis velocity) MD: JOG_VELO_RAPID (JOG rapid traverse)		

41110 SD number	**JOG_SET_VELO** JOG velocity for linear axes (for G94)		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective immediately	Protection level: MMC-MD 9220	Unit: mm/min or mm/rev	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	Value not equal to zero: The velocity value entered applies to linear axes traversed in JOG mode if linear feedrate (G94) is active for the relevant axis (MD: JOG_REV_IS_ACTIVE = 0). The axis velocity is active for – continuous jogging – incremental jogging (INC1, ... INCvar) – handwheel jogging The value entered is valid for all linear axes and must not exceed the maximum permissible axis velocity (MD: MAX_AX_VELO). With DRF: If DRF is active, the velocity set in SD:JOG_SET_VELO must be reduced with MD: HANDWH_VELO_OVERLAY_FACTOR. Value equal to zero: If 0 has been entered in the setting data, the active linear feedrate in JOG mode is MD: JOG_VELO “JOG axis velocity”. Each axis can be given its own JOG velocity with this MD (axial MD).		
SD irrelevant for	– For linear axes if SD: JOG_REV_IS_ACTIVE = 1 – For rotary axes (SD: JOG_ROT_AX_SET_VELO is active here)		
Application	The operator can define a JOG velocity for a particular application.		
Related to	SD: JOG_REV_IS_ACTIVE (revolutional feedrate for JOG active) Axial MD: JOG_VELO (JOG axis velocity) Axial MD: MAX_AX_VELO (maximum axis velocity) Axial MD: HANDWH_VELO_OVERLAY_FACTOR (ratio JOG velocity to handwheel velocity (DRF)) SD: JOG_ROT_AX_SET_VELO (JOG velocity for rotary axes)		

41120 SD number	**JOG_REV_SET_VELO** JOG velocity (for G95)		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective immediately	Protection level: MMC-MD 9220	Unit: mm/rev	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	Value not equal to zero: The velocity value entered applies to axes traversed in JOG mode if revolutional feedrate (G95) is active for the relevant axis (MD: JOG_REV_IS_ACTIVE = 1). The axis velocity is active for – continuous jogging – incremental jogging (INC1, ... INCvar) – handwheel jogging The value entered is valid for all axes and must not exceed the maximum permissible axis velocity (MD: MAX_AX_VELO). Value equal to zero: If 0 has been entered in the setting data, the active revolutional feedrate in JOG mode is MD: JOG_REV_VELO “revolutional feedrate with JOG”. Each axis can be given its own revolutional feedrate with this MD (axial MD).		
SD irrelevant for	– For axes if SD: JOG_REV_IS_ACTIVE = 0		
Application	The operator can define a JOG velocity for a particular application.		
Related to	Axial SD: JOG_REV_IS_ACTIVE (revolutional feedrate for JOG active) Axial MD: JOG_REV_VELO (revolutional feedrate with JOG) Axial MD: MAX_AX_VELO (maximum axis velocity)		

41130 SD number	**JOG_ROT_AX_SET_VELO** JOG velocity for rotary axes		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective immediately	Protection level: MMC-MD 9220	Unit: mm/rev	

41130 SD number	**JOG_ROT_AX_SET_VELO** JOG velocity for rotary axes	
Data type: DOUBLE		Applies from SW version: 2.1
Significance:	Value not equal to zero: The velocity entered applies to rotary axes in JOG mode (in continuous mode, in incremental mode, in jogging with handwheel). The value entered is common to all rotary axes and must not exceed the maximum permissible axis velocity (MD: MAX_AX_VELO). With DRF, the velocity set with SD: JOG_ROT_AX_SET_VELO must be reduced with the MD: HANDWH_VELO_OVERLAY_FACTOR. Value equal to zero: If the value 0 is entered in the setting data, the velocity that applies to rotary axes in JOG mode is the axial MD: JOG_VELO (jog axis velocity) In this way, it is possible to define a separate JOG velocity for every axis.	
Application	The operator can define a JOG velocity for a particular application.	
Related to	MD: JOG_VELO (JOG axis velocity) MD: MAX_AX_VELO (maximum axis velocity) MD: HANDWH_VELO_OVERLAY_FACTOR (ratio JOG velocity to handwheel velocity (DRF)	

41200 SD number	**JOG_SPIND_SET_VELO** JOG velocity for spindles		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective immediately	Protection level: MMC-MD 9220	Unit: mm/rev	
Data type: DOUBLE		Applies from SW version: 1.1	
Significance:	Value not equal to zero: The velocity entered applies to spindles in JOG mode if they are traversed manually using the “traversing keys plus and minus”. The velocity is active for – continuous jogging – incremental jogging (INC1, ... INCvar) – handwheel jogging The value entered is valid for all spindles and must not exceed the maximum permissible velocity (MD: MAX_AX_VELO). Value equal to zero: If 0 has been entered in the setting data, the active JOG velocity is MD: JOG_VELO (conventional axis velocity) acts as the JOG velocity for the rotary axis. Each axis can be given its own JOG velocity with this MD (axial MD). When the spindle is traversed in JOG mode, the maximum velocity of the active gear stage (MD: GEAR_STEP_VELO_LIMIT) is taken into account.		
SD irrelevant for	Axes		
Application	The operator can define a JOG velocity for the spindles for a particular application.		
Related to	Axial MD: JOG_VELO (JOG axis velocity) MD: GEAR_STEP_MAX_VELO_LIMIT (maximum velocity of gear stages)		
References	/FB/, S1, “Spindles”		

■

Signal Descriptions

5

5.1 General signals

5.1.1 Signals from NC

DB10 **DBB97, 98, 99** Data block	**Channel number of geometry axis for handwheels 1, 2, 3** Signal(s) from NC (MMC –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Significance of signal	The operator can assign an axis directly to the handwheel (1, 2, 3) on the operator panel. If this axis is a geometry axis (IS “Machine axis” = 0), the assigned channel number for the handwheel in question is transferred to the PLC. In this way, the IS “Activate handwheel” is set for the selected geometry axis in accordance with the state set by the operator (IS “Handwheel selected”). The following codes apply to the channel number: Bit 7 6 5 4 3 2 1 Bit 0 | Channel number 0 0 0 0 0 0 0 0 | – 0 0 0 0 0 0 0 1 | 1 0 0 0 0 0 0 1 0 | 2 With machine axes (IS “Machine axis” = 1), the IS “Channel number geometry axis for handwheel 1, 2, 3” has no meaning. For further information, see IS “Axis number for handwheel 1, 2, 3”.	
Related to	IS “Axis number of handwheel 1, 2, 3” (DB10, DBB100 ff) IS “Handwheel selected” (DB10, DBX100.6 ff) IS “Machine axis” (DB10, DBX100.7 ff) IS “Activate handwheel” (DB21, ... DBX12.0 – 12.2 ff)	
Application	If DB10, DBB97 = 2, then handwheel 1 is assigned to channel 2.	

DB10 **DBB 100; 101; 102, Bits 0–4** Data block	**Axis number for handwheel 1, 2 or 3** Signal(s) from NC (MMC –> PLC)		
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1	
Significance of signal	The operator can assign an axis to every handwheel directly via the operator panel. To do so, he defines the required axis (e.g. X). The basic PLC program provides the number of the axis plus the information 'machine axis or geometry axis' (IS "machine axis") as MMC interface signals. The basic PLC program sets the interface signal "Activate handwheel" for the defined axis. Depending on the setting in MMC interface signal "machine axis" either the interface for the geometry axis or for the machine axis is used. The following must be noted when assigning the axis designation to the axis number: • IS "Machine axis" = 1; i.e. machine axis: The assignment is made via MD: AXCONF_MACHAX_NAME_TAB [n] (machine axis name). • IS "Machine axis" = 0; i.e. geometry axis: The assignment is made via MD: AXCONF_GEOAX_NAME_TAB [n] (geometry axis name in channel). IS "Channel number geometry axis handwheel n" defines the channel assigned to the handwheel. For following codes are used for the axis number:		
Related to	IS "Channel number geometry axis handwheel n" (DB10, DBX97 ff) IS "Handwheel selected" (DB10, DBX100.6 ff) IS "Machine axis" (DB10, DBX100.7 ff) IS "Activate handwheel" (DB21, DBX12.0 to DBX12.2 ff) IS "Activate handwheel" (DB31, DBX4.0 to DBX4.2) MD: AXCONF_MACHAX_NAME_TAB [n] (machine axis name) MD: AXCONF_GEOAX_NAME_TAB [n] (geometry axis name in channel)		

Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Axis number
0	0	0	0	0	–
0	0	0	0	1	1
0	0	0	1	0	2
0	0	0	1	1	3
0	0	1	0	0	4
0	0	1	0	1	5
0	0	1	1	0	6
0	0	1	1	1	7
0	1	0	0	0	8

DB10 **DBX100.6; 101.6; 102.6** Data block	**Handwheel selected (for handwheel 1, 2 or 3)** Signal(s) from NC (MMC –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The operator has selected the handwheel for the defined axis via the operator panel (i.e. activated). This information is made available by the basic PLC program at the MMC interface. This means that the interface signal “Activate handwheel” is set to ’1’ for the defined axis by the basic PLC program. The associated axis is also displayed on the MMC interface (IS: “Axis number” and IS “Machine axis”). As soon as the handwheel is active, the axis can be traversed in JOG mode with the handwheel (IS “Handwheel active”).	
Signal state 0 or signal transition 1 ——> 0	The operator has disabled the handwheel for the defined axis via the operator panel. This information is made available by the basic PLC program at the MMC interface. Now the interface signal “Activate handwheel” can be reset for the defined axis by the basic PLC program.	
Related to	IS “Axis number” (DB10, DBB100 ff) IS “Machine axis” (DB10, DBX100.7 ff) IS “Activate handwheel” (DB21, DBX12.0 – DBX12.2 ff) IS “Handwheel active” (DB21, DBX40.0 to DBX40.2 ff) IS “Activate handwheel” (DB31, DBX4.0 – DBX4.2) IS “Channel number geometry axis for handwheel 1, 2 or 3” (DB10, DBB97 ff)	

DB10 **DBX100.7; 101.7; 102.7** Data block	**Machine axis (for handwheel 1, 2 or 3)** Signal(s) from NC (MMC –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The operator has assigned an axis to the handwheel (1, 2, 3) directly on the operator panel. This axis is a machine axis. For further information see IS “Axis number”.	
Signal state 0 or signal transition 1 ——> 0	The operator has assigned an axis to the handwheel (1, 2, 3) directly on the operator panel. This axis is a geometry axis. For further information see IS “Axis number”.	
Related to	IS “Axis number” (DB10, DBB100 ff) IS “Handwheel selected” (DB10, DBX100.6 ff) NST “Channel number geometry axis for handwheel 1, 2 or 3” (DB10, DBB97 ff)	

5.2 Channel-specific signals

5.2.1 Overview of signals to channel

DB 21, ...	Signals to channel							
DBB	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0		Activate dry run feedrate	Activate M01	Activate single block	Activate DRF	Activate traverse forwards	Activate traverse backwards	Execution from external
12	Geometry axis 1							
	Traverse keys		Rapid traverse override	Traverse key disable	Feed hold	Activate handwheel		
	+	–				3	2	1
13	Geometry axis 1							
	Machine functions							
		Continuous	Variable INC	10000 INC	1000 INC	100 INC	10 INC	1 INC
16	Geometry axis 2							
	Traverse keys		Rapid traverse override	Traverse key disable	Feed hold	Activate handwheel		
	+	–				3	2	1
17	Geometry axis 2							
	Machine functions							
		Continuous	Variable INC	10000 INC	1000 INC	100 INC	10 INC	1 INC
20	Geometry axis 3							
	Traverse keys		Rapid traverse override	Traverse key disable	Feed hold	Activate handwheel		
	+	–				3	2	1
21	Geometry axis 3							
	Machine functions							
		Continuous	Variable INC	10000 INC	1000 INC	100 INC	10 INC	1 INC

5.2.2 Description of signals to channel

DB21, ... **DBX0.3** Data block	**Activate DRF** Signal(s) to channel (PLC –> NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The function DRF is selected. The function can either be selected directly from the PLC user program or from the operator panel via MMC interface signal “DRF selected”. This MMC interface signal is either converted by the basic PLC program or the PLC user program to interface signal “Activate DRF”. As soon as the function DRF is active, DRF offset can be modified in operating modes AUTOMATIC or MDA.	
Signal state 0 or signal transition 1 ——> 0	The function is not selected.	
Signal irrelevant for.....	JOG mode	
Application	The DRF function can be enabled specifically by the PLC user program with IS “Activate DRF”.	
Related to	IS “DRF selected” (DB21, ... DBX24.3)	

DB21, ... **DBB12; 16; 20 Bits 0–2** Data block	**Activate handwheel (1 to 3) for geometry axis (1,2,3)** Signal(s) to channel (PLC –> NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	These machine data determine whether this geometry axis is assigned to handwheel 1, 2, 3 or no handwheel. Only one handwheel can be assigned to an axis at any one time. If several interface signals “Activate handwheel” are set, priority “Handwheel 1” before “Handwheel 2” before “Handwheel 3” applies. **Note:**3 geometry axes can be traversed simultaneously with handwheels 1 to 3!	
Signal state 0 or signal transition 1 ——> 0	Neither handwheel 1, 2 nor 3 is assigned to this geometry axis.	
Application	The PLC user program can use this interface signal to disable the influence of turning the handwheel on the geometry axis.	
Related to	IS “Handwheel active” for geometry axis (DB21, DBX40.7 or DBX40.6 ff)	

DB21, ... **DBX12.4; 16.4; 20.4** Data block	**Traverse key disable for geometry axis (1,2,3)** Signal(s) to channel (PLC –> NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The traverse keys plus and minus have no effect on the geometry axes in question. It is thus not possible to traverse the geometry axis in JOG with the traverse keys on the machine control panel. If the traverse key disable is activated during a traverse movement, the geometry axis is stopped.	
Signal state 0	Traverse keys plus and minus are enabled.	
Application	It is thus possible, depending on the operating mode, to disable manual traverse of the geometry axis in JOG mode with the traverse keys from the PLC user program.	
Related to	IS “Traverse key plus” and “Traverse key minus” for the geometry axis (DB21, DBX12.7 or DBX12.6 ff)	

DB21, ... **DBX12.5; 16.5; 20.5** Data block	**Rapid traverse override for geometry axis (1,2,3)** Signal(s) to channel (PLC –> NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	If interface signal “Rapid traverse override” is set together with “Traverse key plus” and “Traverse key plus”, the geometry key in question traverses at rapid traverse. The rapid traverse feedrate is defined in machine data JOG_VELO_RAPID. Rapid traverse override is active in the following JOG modes: – Continuous jogging – Incremental jogging If rapid traverse override is active, the velocity can be modified with the rapid traverse override switch.	
Signal state 0 or signal transition 1 ——> 0	The geometry axis traverses at the defined JOG velocity (SD: JOG_SET_VELO or MD: JOG_VELO).	
Signal irrelevant for.....	– Operating modes AUTOMATIC and MDA – Reference point approach (JOG mode)	
Related to	IS: “Traverse key plus” and “Traverse key minus” for the geometry axis (DB21, DBX12.7 or DBX12.6 ff)	
References	/FB/, V1, “Feeds”	

DB21, ... **DBB12; 16; 20 Bits 7; 6** Data block	**Plus and minus traverse keys for geometry axis (1,2,3)** Signal(s) to channel (PLC –> NCK)	
Edge evaluation: yes	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The selected geometry axis can be traversed in both directions in JOG mode with the traverse keys plus and minus. Depending on the active machine function and the setting “Jog or continuous mode” (SD: JOG_CONT_MODE_LEVELTRIGGRD for JOG continuous and MD: JOG_INC_MODE_LEVELTRIGGRD for JOG INC), the signal transition will cause different reactions. Case 1: **Continuous jogging with jog mode** The geometry axis traverses in the direction concerned as long as the interface signal is set to 1 (and as long as the axis position has not reached an activated limitation). Case 2: **Continuous jogging with continuous mode** When the signal state first changes from 0 → 1, the geometry axis starts to traverse in the relevant direction. This traversing movement still continues even when the signal state changes from 1 → 0. A new signal state change from 0 → 1 (same travel direction!) wird die stops the traversing movement. Case 3: **Incremental jogging with jog mode** With signal 1 the geometry starts to traverse at theset increment. If the signal changes to the 0 state before the increment is traversed, the signal changes to the 0 state before the increment is traversed, the traversing movement is interrupted. As soon as the signal state changes to 1 again the movement is continued. The geometry axis can be stopped and started several times as described above until the increment has been completelytraversed. Case 4: **Incremental jogging with continuous mode** When the signal state first changes from 0 → 1, the geometry starts to traverse at the set increment. If the signal stateof the same traverse signal changes from 0 → 1, before the geometry axis has traversed the increment, the traverse movement is aborted. The increment is not traversed to the end. If both traverse signals (plus and minus) are set at the same time there is no movement or a current movement is aborted! The effect of the traverse keys can be disabled for every geometry axis individually with the PLC interface signal “Traverse key disable”. **Caution!** In contrast to machine axes, only one axis can be traversed at a time with the traverse keys. Alarm 20062 is triggered if an attempt is made to traverse more than one axis with the traverse keys.	
Signal state 0 or signal transition 1 ——> 0	See cases 1 to 4 above	
Signal irrelevant for.....	Operating modes AUTOMATIC and MDA	
Special cases, errors, ...	The geometry axis cannot be traversed in JOG mode: – If it is already being traversed via the axial PLC interface (as a machineaxis. – If another geometry axis is already being traversed with the traverse keys. Alarm 20062 “Axis already active” is output.	
Related to	IS “Traverse keys plus or minus” for machine axes (DB31, ... BX8.7 or DBX8.6) IS “Traverse key disable for geometry axes” (DB21, ... DBX12.4 ff)	

DB21, ... **DBB13; 17; 21 Bits0–5** Data block	**Machine function for geometry axis (1,2,3)** **INC1, INC10, INC100, INC1000, INC10000, INCvar** Signal(s) to channel (PLC –> NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	This interface signal defines how many increments the geometry axis traverses when the traverse key is pressed or the handwheel is turned one detent position. JOG mode must be active for this (exception: with DRF). The increment size is assigned to the interface signals as follows: – INC1 to INC10000: with general machine data JOG_INCR_SIZE_TAB. – INCvar: with general setting data JOG_VAR_INCR_SIZE As soon as the selected machine function becomes active, this is signalled to the PLC interface (IS “Active machine function INC1;...”). If several machine function signals (INC1, INC...or “Continuous jogging”) are selected at the interface simultaneously, no machine function is activated by the control.	
Signal state 0 or signal transition 1 ——> 0	The machine function in question is not selected. If an axis is currently traversing an increment, this movement is also aborted if this machine function is deselected or switched over.	
Related to	IS “Active machine function INC1,...” for geometry axes (DB21, ... DBB41 ff) IS “Machine function continuous” for geometry axes (DB21, ... DBX13.6 ff)	

DB21, ... **DBX13.6; 17.6; 21.6** Data block	**“Continuous” machine function for geometry axis** **(1, 2, 3)** Signal(s) to channel (PLC –> NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The machine function “Continuous jogging” is selected. The associated geometry axis can be traversed with the traverse keys plus and minus in JOG mode.	
Signal state 0 or signal transition 1 ——> 0	Machine function “Continuous jogging” is not selected.	
Related to	IS “Active machine function INC 1,..., continuous” (DB21, ... DBB41 ff) IS “Machine function INC1,...,INC10000” (DB21, ... DBB13 ff)	

5.2.3 Overview of signals from channel

DB 21–28	Signals from channel							
DBB	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
24 (MMC –> PLC)		Dry run feedrate selected	M01 selected	Single block selected	DRF selected			
33 (MMC –> PLC)					Handwheel override active			
37	Stop at the end of block with SBL suppressed	Read-in enable is ignored	CLC stopped upper limit /TE1/	CLC stopped lower limit /TE1/	CLC active /TE1/	Contour handwheel active		
						Handwheel 1	Handwheel 2	Handwheel 3
40	Geometry axis 1							
	Travel command					Handwheel active		
	plus	minus				3	2	1
41	Geometry axis 1							
	Active machine functions							
		Continuous	Variable INC	10000 INC	1000 INC	100 INC	10 INC	1 INC
46	Geometry axis 2							
	Travel command					Handwheel active		
	plus	minus				3	2	1
47	Geometry axis 2							
	Active machine functions							
		Continuous	Variable INC	10000 INC	1000 INC	100 INC	10 INC	1 INC
52	Geometry axis 3							
	Travel command					Handwheel active		
	plus	minus				3	2	1
53	Geometry axis 3							
	Active machine functions							
		Continuous	Variable INC	10000 INC	1000 INC	100 INC	10 INC	1 INC

5.2.4 Description of signals from channel

DB21, ... **DBX24.3** Data block	**DRF selected** Signal(s) from channel (MMC –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The operator has selected DRF on the operator panel. The PLC program (basic PLC program or user program) transfers this MMC interface signal as IS “Activate DRF” after logical combination. As soon as DRF is active, the DRF offset can be changed in AUTOMATIC or MDA mode using the handwheel assigned to the axis.	
Signal state 0 or signal transition 1 ——> 0	The operator has not selected DRF on the operator panel.	
Signal irrelevant for.....	JOG mode	
Related to	IS: “Activate DRF” (DB21, ... DBX0.3)	

DB21, ... **DBB37 Bits 0–2** Data block	**Contour handwheel active (1 to 3)** Signal(s) from axis/spindle (NCK –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 4.3
Signal state 1 or signal transition 0 ——> 1	These PLC interface signals report whether this geometry axis is assigned to contour handwheel 1, 2 or 3 or to no contour handwheel. Only one contour handwheel can be assigned to an axis at any one time. If several interface signals “Contour handwheel active” are set, priority ’Contour handwheel 1’ before ’Contour handwheel 2’ before ’Contour handwheel 3’ applies. If the assignment is active, the geometry axis can be traversed in JOG mode with the contour handwheel or a DRF offset can be generated in AUTOMATIC or MDA modes.	
Signal state 0 or signal transition 1 ——> 0	Neither contour handwheel 1, 2 nor 3 is assigned to this geometry axis.	
Related to		

DB21, ... **DBB40; 46; 52 Bits 0–2** Data block	**Handwheel active (1 to 3) for geometry axis** Signal(s) from axis/spindle (NCK –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	These PLC interface signals report whether this geometry axis is assigned to handwheel 1, 2 or 3 or to no handwheel. Only one handwheel can be assigned to an axis at any one time. If several interface signals “Activate handwheel” are set, priority “Handwheel 1” before “Handwheel 2” before “Handwheel 3” applies. If the assignment is active, the geometry axis can be traversed in JOG mode with the handwheel or a DRF offset can be generated in AUTOMATIC or MDA modes.	
Signal state 0 or signal transition 1 ——> 0	Neither handwheel 1, 2 nor 3 is assigned to this geometry axis.	
Related to	IS “Activate handwheel” (DB21, DBX12.0 to DBX12.2 ff)	

DB21, ... **DBB40; 46; 52 Bits 6;7** Data block	**Plus and minus travel commands (for geometry axis)** Signal(s) from axis/spindle (NCK –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	A traverse movement of the axis is to be executed in one or the other direction. Depending on the mode selected, the command is triggered in different ways: – JOG mode: with the plus or minus traverse key – REF mode: with the traverse key that takes the axis to the reference point – AUT/MDA mode: the program block containing a coordinate value for the axis in question is executed.	
Signal state 0 or signal transition 1 ——> 0	A travel command in the relevant axis direction has not been given or a traverse movement has been completed. • JOG mode: – The travel command is reset depending on the current setting “jog or continuous mode” (see interface signal “Traverse keys plus and minus”). – While traversing with the handwheel. • REF mode: When the reference point is reached • AUT/MDA modes: – The program block has been executed (and the next block does not contain any coordinate values for the axis in question) – Abort with “RESET”, etc. – IS “Axis disable” is active	
Application	To release clamping of axes with clamping (e.g. on a rotary table). **Note:** If the clamping is not released until the travel command is given, continuous-path operation of these axes is not possible!	
Related to	IS: “Traverse key plus” and “Traverse key minus” for the geometry axis (DB21, DBX12.7 or DBX12.6 ff)	

DB21, ... **DBB41; 47; 53 Bits 0–6** Data block	**Active machine function for geometry axis (1, 2, 3)** **INC1, ..., continuous jogging** Signal(s) from axis/spindle (NCK –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The PLC interface receives a signal stating which JOG mode machine function is active for the geometry axes. The reaction to actuation of the traverse key or rotation of the handwheel varies depending on which machine function is active (see Section 2.2 and 2.3).	
Signal state 0 or signal transition 1 ——> 0	The machine function in question is not active.	
Related to	IS “Machine function INC1,..., continuous jogging” for geometry axis (DB21, ... DBB13 ff)	

DB21, ... **DBX33.3** Data block	**Handwheel override active** Signal(s) from axis/spindle (NCK –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The function “Handwheel override in AUTOMATIC mode” is active for the programmed path axes. Handwheel pulses of the 1st geometry axis function as a velocity override over the programmed path feedrate.	
Signal state 0 or signal transition 1 ——> 0	The function “Handwheel override in AUTOMATIC mode” is not active for the programmed path axes. An active handwheel override is not active if • The path axes have reached the target position • The distance-to-go is deleted by the channel-specific IS “Delete distance-to-go” (DB21, ... DBX6.2) • A RESET is performed	

5.3 Axis/spindle-specific signals

5.3.1 Overview of signals to axis/spindle

DB 31, ...	Signals to axis/spindle							
DBB	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
4	Traverse keys		Rapid traverse override	Traverse key disable	Feed hold Spindle hold	Activate handwheel		
	plus	minus				3	2	1
5	Machine functions							
		Continuous	Variable	10000	1000	100	10	1
			INC	INC	INC	INC	INC	INC

5.3.2 Description of signals to axis/spindle

DB31, ... **DBX4.0; 4.1; 4.2** Data block	**Activate handwheel (1 to 3)** Signal(s) to axis/spindle (PLC –> NCK)		
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1	
Signal state 1 or signal transition 0 ——> 1	This PLC interface signal defines whether this machine axis is assigned to handwheel 1, 2, 3 or no handwheel. Only one handwheel can be assigned to an axis at any one time. If several interface signals “Activate handwheel” are set, priority “Handwheel 1” before “Handwheel 2” before “Handwheel 3” applies. If the assignment is active, the machine axis can be traversed with the handwheel in JOG mode or a DRF offset can be generated in AUTOMATIC or MDA mode.		
Signal state 0 or signal transition 1 ——> 0	Neither handwheel 1, 2 nor 3 is assigned to this geometry axis.		
Application	The PLC user program can use this interface signal to disable the influence of turning the handwheel on the axis.		
Related to	IS “Handwheel active” for geometry axis (DB31, ... DBX64.0 to DBX64.2)		

DB31, ... **DBX4.4** Data block	**Traverse key disable** Signal(s) to axis/spindle (PLC –> NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The traverse keys plus and minus have no effect on the machine axes in question. It is thus not possible to traverse the machine axis in JOG with the traverse keys on the machine control panel. If the traverse key disable is activated during a traverse movement, the machine axis is stopped.	
Signal state 0 or signal transition 1 ——> 0	Traverse keys plus and minus are enabled.	
Application	It is thus possible, depending on the operating mode, to disable manual traverse of the machine axis in JOG mode with the traverse keys from the PLC user program.	
Related to	IS “Traverse key plus” and “Traverse key minus” (DB31, ... DBX4.7 or DBX4.6)	

DB31, ... **DBX4.5** Data block	**Rapid traverse override** Signal(s) to axis/spindle (PLC –> NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	If interface signal “Rapid traverse override” is set together with “Traverse key plus” and “Traverse key minus”, the machine axis in question traverses at rapid traverse. The rapid traverse feedrate is defined in machine data JOG_VELO_RAPID. Rapid traverse override is active in the following JOG modes: – Continuous jogging – Incremental jogging If rapid traverse override is active, the velocity can be modified with the rapid traverse override switch.	
Signal state 0 or signal transition 1 ——> 0	The machine axis traverses at the defined JOG velocity (SD: JOG_SET_VELO or MD: JOG_VELO).	
Signal irrelevant for.....	– Operating modes AUTOMATIC and MDA – Reference point approach (JOG mode)	
Related to	IS “Traverse key plus” and “Traverse key minus” (DB31, ... DBX4.7 or DBX4.6) IS “Axial feedrate/spindle speed override” (DB31, ... DBB0)	

DB31, ... **DBX4.7, 4.6** Data block	**Plus and minus traverse keys** Signal(s) to axis/spindle (PLC –> NCK)	
Edge evaluation: yes	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The selected machine axis can be traversed in both directions in JOG mode with the traverse keys plus and minus. Depending on the active machine function and the setting “Jog or continuous mode” (SD: JOG_CONT_MODE_LEVELTRIGGRD for JOG continuous and MD: JOG_INC_MODE_LEVELTRIGGRD for JOG INCR), the signal transition will cause different reactions. Case 1: **Continuous jogging with jog mode** The machine axis traverses in the direction concerned as long as the interface signal is set to 1 (and as long as the axis position has not reached an activated limitation). Case 2: **Continuous jogging with continuous mode** When the signal state first changes from 0 → 1, the geometry axis starts to traverse in the relevant direction. This traversing movement still continues even when the signal state changes from 1 → 0. A new signal state change from 0 → 1 (same travel direction!) stops the traversing movement again. Case 3: **Incremental jogging with jog mode** With signal 1 the machine axis starts to traverse at theset increment. If the signal changes to the 0 state before the increment is traversed, the traversing movement is interrupted. As soon as the signal state changes to 1 again the movement is continued. The machine can be stopped and started several times as described above until the increment has been completely traversed. Case 4: **Incremental jogging with continuous mode** When the signal state first changes from 0 → 1, the machine axis starts traverse at the set increment. If the signalof the same traverse signal changes from 0 → 1, before the axis has traversed the increment, the traverse movement is aborted. The increment is not traversed to the end. If both traverse signals (plus and minus) are set at the same time there is no movement or a current movement is aborted. The effect of the traverse keys can be disabled for every machine axis individually with the PLC interface signal “Traverse key disable”.	
Signal state 0 or signal transition 1 ——> 0	See cases 1 to 4 above	
Signal irrelevant for.....	Operating modes AUTOMATIC and MDA	
Application	The machine axis cannot be traversed in JOG mode if it is already being traversed via the channel-specific PLC interface (as a geometry axis). Alarm 20062 is triggered.	
Special cases, ...	Indexing axes	
Related to	IS “Traverse keys plus and minus for geometry axes” (DB21, ... DBX12.7 and DBX12.6 ff) IS “Traverse key disable” (DB31, ... DBX4.4)	

DB31, ... **DBX5.6** Data block	**Continuous machine function** Signal(s) to axis/spindle (PLC –> NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The machine function “Continuous jogging” is selected. The associated machine axis can be traversed with the traverse keys plus and minus in JOG mode.	
Signal state 0 or signal transition 1 ——> 0	Machine function “Continuous jogging” is not selected.	
Related to	IS “Active machine function INC 1,..., continuous” (DB31, ... DBB65) IS “Machine function INC1,...,INC10000” (DB31, ... DBB5)	

DB31, ... **DBB5 Bits 0–5** Data block	**Machine function INC1, INC10, INC100, INC1000, INC10000, INCvar** Signal(s) to axis/spindle (PLC –> NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	This interface signal defines how many increments the machine axis traverses when the traverse key is pressed or the handwheel is turned one detent position. JOG mode must be active for this (exception: with DRF). The increment size is assigned to the interface signals as follows: – INC1 to INC10000: with general machine data JOG_INCR_SIZE_TAB. – INCvar: with general setting data JOG_VAR_INCR_SIZE As soon as the selected machine function becomes active, this is signalled to the PLC interface (IS “Active machine function INC1;...”). If several machine function signals (INC1, INC...or “Continuous jogging”) are selected at the interface simultaneously, no machine function is activated by the control.	
Signal state 0 or signal transition 1 ——> 0	The machine function in question is not selected. If an axis is currently traversing an increment, this movement is also aborted if this machine function is deselected or switched over.	
Related to	IS “Active machine function INC1,...” for geometry axes (DB31, ... DBB65) IS “Machine function continuous” (DB31, ... DBX5.6)	

5.3.3 Overview of signals from axis/spindle

DB 31, ...	Signals from axis/spindle							
DBB	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
64	Travel command					Handwheel active		
	plus	minus				3	2	1
65	Active machine functions							
		Continu-ous	Variable INC	10000 INC	1000 INC	100 INC	10 INC	1 INC

5.3.4 Description of signals from axis/spindle

DB31, ... **DBX62.1** Data block	**Handwheel override active** Signal(s) from axis/spindle (NCK –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The function “Handwheel override in AUTOMATIC mode” is active for the programmed positioning axis (FDA[AXi]). Handwheel pulses for this axis either act as a path setting (if FDA=0) or as a velocity override (if FDA>0) over the programmed axis feedrate. The interface signal is also set if “Handwheel override in AUTOMATIC mode” is active for a concurrent positioning axis (with FC15).	
Signal state 0 or signal transition 1 ——> 0	The function “Handwheel override in AUTOMATIC mode” is not active for the programmed positioning axis (or concurrent positioning axis). An active handwheel override is not active if • The positioning axis has reached the target position • The distance-to-go is deleted by the axis-specific IS “Delete distance-to-go” (DB31, ... DBX2.2) • A RESET is performed	

DB31, ... **DBX64.0; 64.1; 64.2** Data block	**Handwheel active (1 to 3)** Signal(s) from axis/spindle (NCK –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	These PLC interface signals provide feedback whether the machine axis is assigned to handwheel 1, 2, 3 or no handwheel. Only one handwheel can be assigned to an axis at any one time. If several interface signals “Activate handwheel” are set, priority “Handwheel 1” before “Handwheel 2” before “Handwheel 3” applies. If the assignment is active, the machine axis can be traversed with the handwheel in JOG mode or a DRF offset can be generated in AUTOMATIC or MDA mode.	
Signal state 0 or signal transition 1 ——> 0	Neither handwheel 1, 2 nor 3 is assigned to this geometry axis.	
Related to	IS “Activate handwheel” (DB31, DBX4.0 to DBX4.2) IS “Handwheel selected” (DB10, DBB100.6 ff)	

DB31, ... **DBX64.7, 64.6** Data block	**Plus and minus traverse keys** Signal(s) from axis/spindle (NCK –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	A traverse movement of the axis is to be executed in one or the other direction. Depending on the mode selected, the command is triggered in different ways: – JOG mode: with the plus or minus traverse key – REF mode: with the traverse key that takes the axis to the reference point – AUT/MDA mode: the program block containing a coordinate value for the axis in question is executed.	
Signal state 0 or signal transition 1 ——> 0	A travel command in the relevant axis direction has not been given or a traverse movement has been completed. • JOG mode: – The travel command is reset depending on the current setting jog or continuous mode" (see interface signal "Traverse keys plus and minus"). – While traversing with the handwheel. – REF mode: When the reference point is reached • AUT/MDA modes: – The program block has been executed (and the next block does not contain any coordinate values for the axis in question) – Abort with "RESET", etc. – IS "Axis disable" is active	
Application	To release clamping of axes with clamping (e.g. on a rotary table). **Note:** If the clamps are not released until the travel command is given, these axes cannot be operated under continuous path control!	
Related to	IS "Traverse key plus" and "Traverse key minus" (DB31, ... DBX4.7 or DBX4.6)	

DB31, ... **DBB65 Bits 0–6** Data block	**Active machine function** **INC1, ..., continuous jogging** Signal(s) from axis/spindle (NCK –> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The PLC interface receives a signal stating which JOG mode machine function is active for the machine axes. The result when the traverse key is pressed or the handwheel is turned depends on the active machine function (see Sections 2.2 and 2.3).	
Signal state 0 or signal transition 1 ——> 0	The machine function in question is not active.	
Related to	IS "Machine function INC1,..., continuous jogging" (DB31, ... DBB5)	

■

Notes

6.7 Example

None

Data Fields, Lists

7.1 Interface signals

DB number	Bit, byte	Name	Reference
Signals to/from NC			
10	97, 98, 99	Channel number for geometry axis handwheel 1, 2, 3	
10	100, 101, 102	Axis number for handwheel 1, 2, 3, handwheel selected and machine axis	
Mode group specific			
11, ...	0.2	JOG mode	K1
11, ...	4.2	Active JOG mode	K1
Channel-specific			
21, ...	0.3	Activate DRF	
21, ...	12.2, 12.1, 12.0 16.2, 16.1, 16.0 20.2, 20.1, 20.0	Activate handwheel 1, 2, 3	
21, ...	12.4, 16.4, 20.4	Traverse key disable	
21, ...	12.5, 16.5, 20.5	Rapid traverse override	
21, ...	12.7, 12.6 16.7, 16.6 20.7, 20.6	Traverse keys plus and traverse keys minus	
21, ...	13, 17, 21	Geometry axis machine function INC1 ... continuous	
21, ...	24.3	DRF selected	
21, ...	40.2, 40.1, 40.0 46.2, 46.1, 46.0 52.2, 52.1, 52.0	Handwheel 1, 2, 3 active	
21, ...	40.7, 40.6 46.7, 46.6 52.7, 52.6	Travel command plus and travel command minus	
21, ...	41, 47, 53	Geometry axis active machine function INC1 ... continuous jogging	

Channel-specific			
21, ...	33.3	Handwheel override active for path axes (SW2 and higher)	
21, ...	30.0 30.1 30.2	Activate handwheel 1 as contour handwheel Activate handwheel 2 as contour handwheel Activate handwheel 3 as contour handwheel	
21, ...	30.3	Simulation of contour handwheel ON	
21, ...	30.4	Negative direction simulation of contour handwheel	
21, ...	37.0 37.1 37.2	Handwheel 1 active as contour handwheel Handwheel 2 active as contour handwheel Handwheel 3 active as contour handwheel	
21, ...	100.5 101.5 102.5	Handwheel 1 active as contour handwheel Handwheel 2 active as contour handwheel Handwheel 3 active as contour handwheel	
Axis/spindle-specific			
31, ...	0	Feedrate/spindle speed override	V1
31, ...	1.7	Override active	V1
31, ...	2.2	Axial delete distance-to-go	V1
31, ...	4.2, 4.1, 4.0	Activate handwheel 1, 2, 3	
31, ...	4.4	Traverse key disable	
31, ...	4.5	Rapid traverse override	
31, ...	4.7, 4.6	Traverse keys plus and traverse keys minus	
31, ...	5.6	Machine function continuous	
31, ...	5.6, 5.5, 5.4, 5.3, 5.2, 5.1, 5.0	Machine function continuous, var. INC, 10000 INC, 1000 INC, 100 INC, 10 INC, 1 INC	
31, ...	60.7, 60.6	Position reached with exact stop coarse/fine	B1
31, ...	64.2, 64,1, 64.0	Handwheel 1, 2, 3 active	
31, ...	64.7, 64.6	Travel command plus and travel command minus	
31, ...	65	Active machine function INC1 ... continuous jogging	
31, ...	62.1	Handwheel override active, for positioning axes and concurrent positioning axes (SW2 and higher)	

7.2 Machine data

Number	Identifier	Name	Reference
General ($MN_ ...)			
10000	AXCONF_MACHAX_NAME_TAB [n]	Machine axis name [n = axis number]	K2
11300	JOG_INC_MODE_LEVELTRIGGRD	INC and REF in jog mode	
11310	HANDWH_REVERSE	Defines movement in the opposite direction	
11320	HANDWH_IMP_PER_LATCH [n]	Handwheel pulses per detent position [n=handwheel number: 0 – 2]	
11330	JOG_INCR_SIZE_TAB [n]	Increment size INC/handwheel (n = increment index: 0 to 4)	
11340	ENC_HANDWHEEL_SEGMENT_NR	Third handwheel: Bus segment	FBMA

General ($MN_ ...)			
11342	ENC_HANDWHEEL_MODULE_NR	Third handwheel: Drive no./measuring circuit no.	FBMA
11344	ENC_HANDWHEEL_INPUT_NR	Third handwheel: Input on module/measuring-circuit card	
11346	HANDWH_TRUE_DISTANCE	Path definition or velocity specification by hand-wheel	FBMA
Channel-specific ($MC_ ...)			
20060	AXCONF_GEOAX_NAME_TAB [n]	Geometry axis in channel [n = geometry axis number]	K2
20100	DIAMETER_AX_DEF	Geometry axes with transverse axis function	P1
20620	HANDWH_GEOAX_MAX_INCR_SIZE	Delimitation of the geometry axis	
20622	HANDWH_GEOAX_MAX_INCR_VSIZE	Path velocity override	
20624	HANDWH_CHAN_STOP_COND	Response to channel-specific VDI interface sig-nals bits 0...7	
Axis/spindle-specific ($MA_ ...)			
30450	IS_CONCURRENT_POS_AX	Default setting at reset: Neutral axis or channel axis	P2
31090	JOG_INCR_WEIGHT	Weighting of an increment for INC/handwheel	
32000	MAX_AX_VELO	Maximum axis velocity	G2
32010	JOG_VELO_RAPID	JOG rapid traverse	
32020	JOG_VELO	Conventional axis velocity	
32040	JOG_REV_VELO_RAPID	Revolutional feedrate for JOG mode with rapid traverse override	
32050	JOG_REV_VELO	Revolutional feedrate for JOG mode	
32060	POS_AX_VELO	Initial setting for positioning axis velocity	P2
32080	HANDWH_MAX_INCR_SIZE	Delimitation of the size of the selected increment	
32082	HANDWH_MAX_INCR_VELO_SIZE	Limitation of selected increment for velocity override	
32084	HANDWH_STOP_COND	Response to axis-specific VDI interface signals bits 0...5	
32090	HANDWH_VELO_OVERLAY_FACTOR	Ratio JOG velocity to handwheel velocity (with DRF)	
35130	GEAR_STEP_MAX_VELO_LIMIT[n]	Maximum velocity for gear stage	S1

7.3 Setting data

Number	Identifier	Name	Refer-ence
General ($MN_ ...)			
41010	JOG_VAR_INCR_SIZE	Size of variable increment INC/handwheel	
41050	JOG_CONT_MODE_LEVELTRIGGRD	JOG continuous mode	
41100	JOG_REV_IS_ACTIVE	Revolutional feedrate with JOG active	

General ($MN_ ...)			
41110	JOG_SET_VELO	JOG velocity for linear axes (for G94)	
41120	JOG_REV_SET_VELO	JOG velocity (for G95)	
41130	JOG_ROT_AX_SET_VELO	JOG velocity for rotary axes	
41200	JOG_SPIND_SET_VELO	JOG velocity for spindle	

7.4 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

■

SINUMERIK 840D/810D/FM-NC Description of Functions, Extension Functions (FB2)

Compensations (K3)

■

Brief Description 1

Purpose of compensation

The accuracy of machine tools is impaired as a result of deviations from the ideal geometry, power transmission faults and measuring system errors. Temperature differences and mechanical forces often result in great reductions in precision when large workpieces are machined.

Some of these deviations can usually be measured during installation and then compensated for during operation on the basis of values read by the positional actual-value encoder and other sensory devices.

Compensations

CNCs provide functions for compensation of the essential causes of error to meet the increasing demand for precision in machine tools.

The following compensation types can be activated for specific axes on the SINUMERIK 840D and FM-NC:

- compensation
- Backlash compensation
- Interpolatory compensation
  - LEC
    (leadscrew error and measuring system error compensation)
  - Beam sag compensation
    (compensation of beam sag and angular errors)
- Following error compensation (dynamic feedforward control)
- Friction compensation (or quadrant error compensation)
  - Conventional friction compensation
  - Quadrant error compensation with neural networks
    (SINUMERIK 840C only)
- Drift compensation for analog set speed (on SINUMERIK FM-NC only)
- Electronic counterweight for drives on SIMODRIVE 611D

These compensation functions can be set for each machine individually with axis-specific machine data.

Interpolatory compensation

The “Interpolatory compensation” function allows position-related dimensional deviations (for example, by leadscrew errors, measuring system errors or sag) to be corrected.

The compensation values are measured during installation and stored in a table as a position-related value. During operation the axis is compensated between interpolation points during linear interpolation.

Friction compensation

The “friction compensation” function is particularly effective in achieving a significant improvement in contour accuracy in circular contour machining operations. If the direction of rotation of an axis changes, contour errors occur when the velocity equals zero (quadrant transition point) because of the changing friction conditions. “Friction compensation” (also called “Quadrant error compensation”) compensates for this error reliably the first time the contour is machined.

A **neural network** integrated in the SINUMERIK 840D adapts the optimum parameters in a self-learning process to compensate for friction, backlash or torsion. The system allows for simple, automatic re-optimization at any time.

The friction compensation system is installed most simply with a circularity test. The circular contour is followed and the actual position deviations from the programmed radius (most especially at the quadrant transition points) are measured and then displayed graphically. The circularity test is an “installation tool” function.

Activation

The compensations are active in all operating modes of the control as soon as the input data are available. Any compensations that require the position actual value are not activated until the axis reaches the reference point.

Position display

The normal actual-value and setpoint position displays ignore the compensation values and show the position values of an “ideal machine”. The compensation values are output in the “Service axes” display in the “Diagnosis” operating area.

■

Detailed Description 2

2.1 Temperature compensation

2.1.1 General

Deformation due to temperature effects

Heat generated by the drive equipment or high ambient temperatures (e.g. caused by sunlight, drafts) cause the machine base and parts of the machinery to expand. The degree of expansion depends on the temperature and the thermal conductivity of the machine parts.

Effects

Owing to the thermal expansion of the machinery, the actual positions of the axes change depending on temperature. Since this phenomenon impairs the accuracy of the machined workpieces, it is possible to compensate such temperature-related changes in actual value position (so-called temperature compensation).

Sensor technology

Apart from the position actual values supplied by existing encoders, temperature compensation functions generally require a number of additional temperature sensors to acquire a temperature profile.
As temperature-related changes take a relatively long time to have an effect, acquisition and preprocessing of the temperature profile can be executed by the PLC in one-minute cycles.

Error curves

In order to implement temperature compensation, the actual-value offsets over the positioning range of the axis must be measured at a given temperature (T) and plotted. This produces an error curve for this temperature value. Error curves must be produced for different temperatures.

Error curve characteristic

The error curve characteristic shown in the figure below is frequently obtained.

If a position reference point P_0 is chosen for the axis, an offset in the reference point (corresponds to the “position-independent component” of the temperature compensation) can be observed as the temperature changes, and because of the change in length an additional offset in the other position points which increases with the distance to the reference point (corresponds to the “position-dependent component” of the temperature compensation).

The error curve for a given temperature T can generally be represented with sufficient accuracy by a straight line with a temperature-dependent gradient and reference position (see figure below).

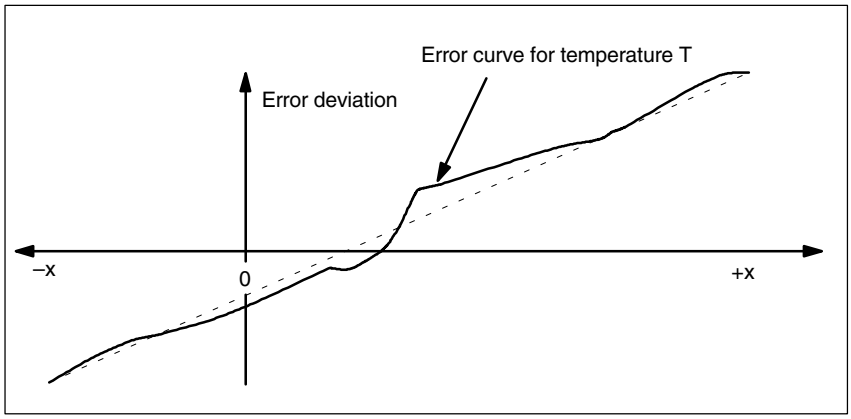


Fig. 2-1 Example of an error curve for heat expansion

Compensation equation

The compensation value ΔK_x is calculated on the basis of current actual position P_x of this axis and temperature T according to the following equation:

$$\Delta K_x = K_0 (T) + \tan\beta (T) * (P_x - P_0)$$

Key to letters (see figure below):

ΔK_x	Temperature compensation value of axis at position Px
K_0	Position-independent temperature compensation value of axis
P_x	Actual position of axis
P_0	Reference position of axis
$\tan\beta$	Coefficient of the position-dependent temperature compensation (corresponds to the gradient of the approximated error line)

The compensation values are acquired in interpolation cycles. If the compensation value ΔK_x is positive, the axis moves in the negative direction.

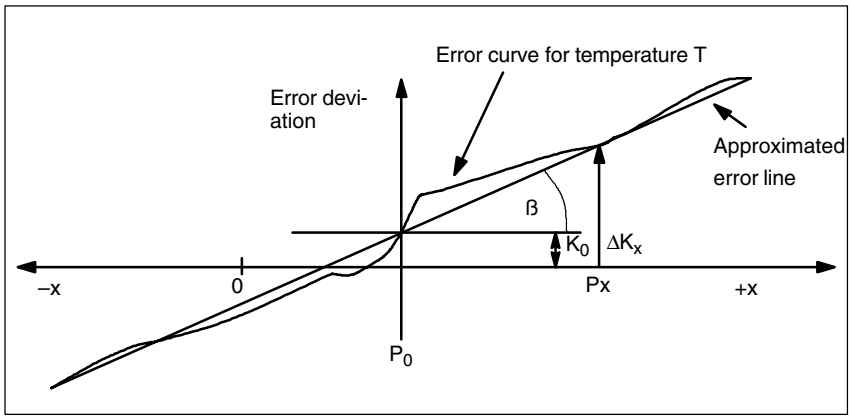


Fig. 2-2 Approximated error line for temperature compensation

Temperature fluctuations

Since the approximated error line applies only to the instantaneous temperature value, the parameters of the error lines that are newly generated when the temperature rises or falls must be sent to the NCK again. Only in this way can expansion due to heat be compensated for effectively.

2.1.2 Temperature compensation parameters

Temperature-dependent parameters

Error curves for different temperatures can be defined for each axis, as illustrated in the figure above. For each error curve the following parameters must be determined and then entered in the setting data:

- Position-dependent temperature-compensation value K_0
  SD 43900: TEMP_COMP_ABS_VALUE
- Reference position P_0 for position-dependent temperature compensation
  SD 43920: TEMP_COMP_REF_POSITION
- Slope tanβ for position-dependent temperature compensation SD 43910: TEMP_COMP_SLOPE

Activate temperature compensation

Temperature compensation can be activated for every axis by means of axial MD 32750: TEMP_COMP_TYPE. The type of temperature compensation to be applied can also be selected.

Table 2-1 MD 32750: TEMP_COMP_TYPE

MD 32750: TEMP_COMP_TYPE	Meaning	Associated parameters
0	No temperature compensation active	_____
1	Position-independent temperature compensation active	SD 43900: TEMP_COMP_ABS_VALUE
2	Position-dependent temperature compensation active	SD 43920: TEMP_COMP_REF_POSITION SD 43910: TEMP_COMP_SLOPE
3	Position-independent and position-dependent temperature compensation active	SD 43900: TEMP_COMP_ABS_VALUE SD 43920: TEMP_COMP_REF_POSITION SD 43910: TEMP_COMP_SLOPE

Activation

The following conditions must be fulfilled before temperature compensation can be applied:

1. The option must be enabled.
2. The compensation type must be selected (MD 32750: TEMP_COMP_TYPE).
3. The parameters for the compensation type are defined.
4. The axis must be referenced (IS “Referenced/synchronized 1 or 2” DB31 to 48, DBX60.4 or 60.5 = ‘1’).

As soon as these conditions are fulfilled, the temperature compensation value for the current position actual value is added to the setpoint in all modes and the machine axis is traversed.

If the reference position is subsequently lost again, e.g. because the encoder frequency has been exceeded (IS “Referenced/Synchronized 1 or 2” = 0), then the compensation processing routine is aborted.

Modify parameters

When temperature T changes, the parameters which are temperature-dependent, i.e. (K_0, tanβ and P_0), also change and can thus always be overwritten by the PLC.

It is thus possible for the machine-tool manufacturer to represent the mathematical and technological relationship between the axis positions and temperature values via the PLC user program and thus calculate the various parameters for the temperature compensation. The temperature parameters are transferred to the NCK with various Services (FB2 (GET) “Read data” and FB3 (PUT) “Write data”).

For more information on handling and parameterization of FB2 and FB3 see:
References: /FB/, P3 “Basic PLC Program”.

Monitoring functions

Axial MD 32760: COMP_ADD_VELO_FACTOR (velocity violation due to compensation) can be set to limit the maximum compensation value that can be added to the specified velocity value in each IPO clock cycle.

This machine data limits the maximum gradient of the error curve. If the maximum gradient is exceeded, the compensation value is limited in the control.

Smooth compensation value

To prevent overloading of the machine or tripping of monitoring functions in response to step changes in the above parameter settings, the compensation values are distributed among several IPO clock cycles by an internal control function as soon as they exceed the maximum compensation value specified for each cycle (MD 32760: COMP_ADD_VELO_FACTOR).

Position display

The normal actual-value and setpoint position displays ignore the compensation values and show the position values of an ideal machine.

Display of compensation values

The total compensation value calculated from the temperature and sag compensation functions belonging to the current actual position is output in the “Service axes” display in the “Diagnosis” operating area.

Example

Installation of the temperature compensation is described below using the example of a Z axis on a lathe.

Determine error curve

In order to determine the temperature-dependent error characteristic of the Z axis, proceed as follows:

- Constant heating by traversing the axis across the whole Z axis traversing range (in the example: from 500 mm to 1500 mm)
- Measuring the axis position in distances of 100 mm
- Measuring the actual temperature at the leadscrew
- Executing a traversing measuring cycle every 20 minutes

The mathematical and technological relationship and the resulting parameters for temperature compensation are derived from the recorded data. The error deviations measured in relation to the axis position measured by the NC are displayed graphically in the figure below.

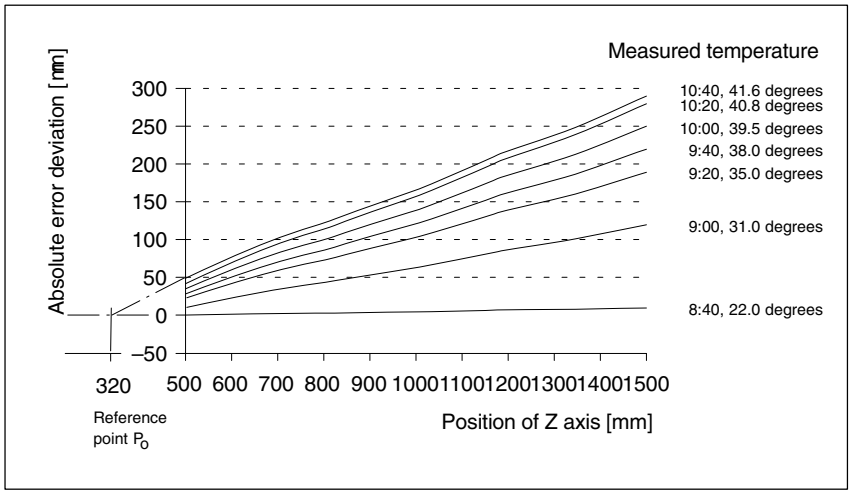


Fig. 2-3 Error curves determined for the Z axis

Setting parameters

The temperature compensation parameters must now be set on the basis of the measurement results (see figure above).

Reference position P_0

As the figure above illustrates, there are basically two methods of configuring reference position P_0:

1. $P_0 = 0$ with position-independent temperature compensation value $K_0 \neq 0$
2. $P_0 \neq 0$ with position-independent temperature compensation value $K_0 = 0$

In our example, variant 2 is chosen, where the position-independent temperature compensation value is always 0. The temperature compensation value therefore only consists of the position-dependent components. The following parameters result:

- MD 32750: TEMP_COMP_TYPE = 2 (only position-dependent temperature compensation is active)
- P_0 = 320 mm —> SD 43920: TEMP_COMP_REF_POSITION = 320

Coefficient tanβ (T)

In order to determine the dependency of coefficient tanβ of the position-dependent temperature compensation on the temperature, the error curve gradient is plotted against the measured temperature (see figure below).

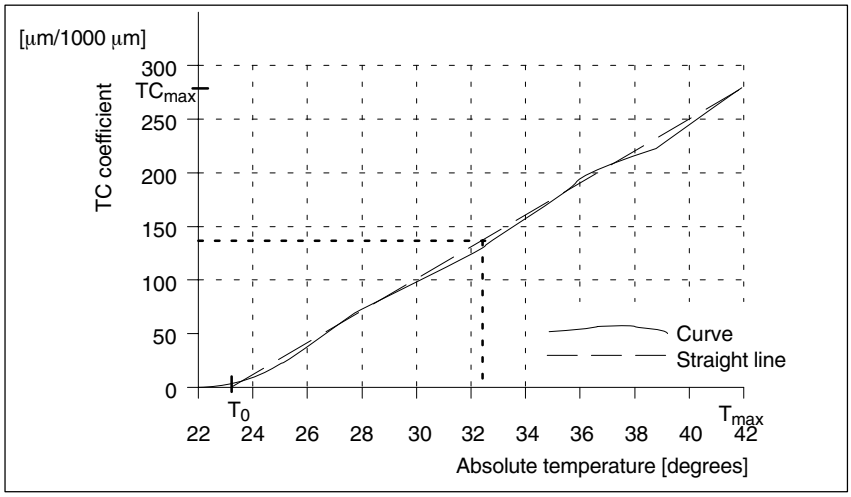


Fig. 2-4 Characteristic of coefficient tanβ as a function of measured temperature T

Depending on the resulting line, the following dependency on T results for the coefficient tanβ:

$$\tan\beta\ (T) = (T - T_0) * \frac{TK_{max}}{T_{max} - T_0}$$

where T_0 = Temperature where the position-dependent error = 0
T_{max} = Maximum measured temperature
TC_{max} = Temperature coefficient for T_{max}

The results are as follows with reference to the figure above:

for T_0 = 23 degrees
T_{max} = 42 degrees
TC_{max} = 270 µm/1000 mm

Therefore:

tanβ(T) = (T – 23 degrees) * 14.21 [µm/1000 mm]
Example: for T = 32.3 degrees —> tanβ = 132 µm/1000 mm

The coefficient tanβ (T) for every measured temperature T can be calculated easily in the PLC according to the above equation and
then transferred to the NCK as SD 43910: TEMP_COMP_SLOPE.

2.2 Backlash compensation

Mechanical backlash

Slight backlash generally occurs in the power train between a moving machine part and its drive (e.g. leadscrew) since an unacceptably high level of machine wear would occur if the mechanical components were to be set to be absolutely free of backlash.
For this reason backlash can arise between the machine part and the measuring system.

Effects

Mechanical backlash on axes/spindles with indirect measuring systems causes the travel path to be falsified. For example, when an axis reverses direction, its travel distance is reduced or increased by an amount corresponding to the backlash (see the two figures below).

Compensation

To compensate for backlash, the axis-specific actual value is corrected by the amount of backlash every time the axis/spindle changes direction.

This quantity can be entered in MD: 32450 BACKLASH during installation of every axis/spindle. If there is a second measuring system installed for the axis/spindle, the relevant backlash values must be entered for each measuring system.

In SW 5 and later, the backlash can be weighted by a factor as a parameter set function. The weighting factor is set in MD 32452: BACKLASH_FACTOR to between 0.01 and 100.0, default setting is 1.0.
Example application: Compensation of gear-stage-related backlash.

Activation

Backlash compensation is always active in all operating modes after reference point approach.

Position display

The normal actual-value and setpoint position displays ignore the compensation values and show the position values of an “ideal machine”.

Display of compensation values

The compensation value applying to the current actual position is output as the total compensation calculated from “LEC” and “backlash compensation” in the “Service axes” display in the “Diagnosis” operating area.

Positive backlash

The encoder “leads” the machine part (e.g. table). Since the actual position acquired by the encoder also “leads” the real actual position of the table, the table travels too short a distance (see figure below). The backlash compensation value must be entered as a **positive** value here (= normal case).

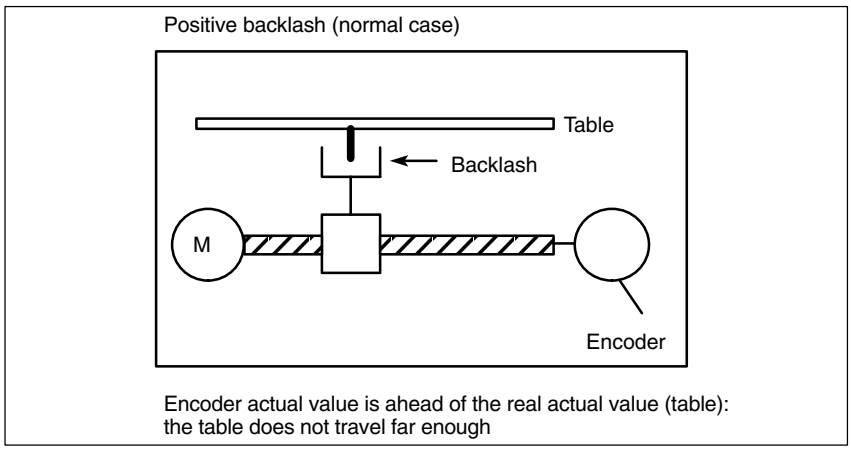


Fig. 2-5 Positive backlash (normal case)

Negative backlash

The encoder “lags behind” the machine part (e.g. table); the table then travels too far (see figure below). The correction value entered is **negative**.

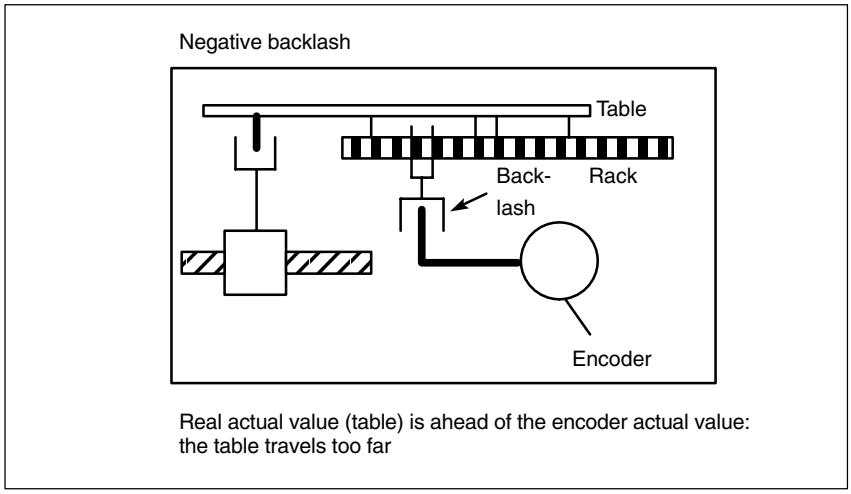


Fig. 2-6 Negative backlash

2nd measuring system

If there is a second measuring system for the axis/spindle, a backlash compensation must be entered for this too. As the second measuring system is mounted in a different way from the first measuring system, the backlash can be different from that of the first measuring system.

When the measuring system is switched over the associated compensation value is always activated.

Note

The actual-value difference between the two encoders must not exceed the value stored in axis-specific MD 36500 ENC_CHANGE_TOL (maximum tolerance for position actual-value changeover).

2.3 Interpolatory compensation

2.3.1 General

Compensation methods

The following compensation methods are applied in order to implement “interpolatory compensation”:

1. “Leadscrew error compensation” or “Measuring system error compensation”, which from now on will be referred to as **leadscrew error compensation (LEC)**.
2. Beam sag compensation or angularity error compensation (Software Version 2 and higher), which from now on will be referred to as **beam sag compensation**.

Many of the characteristics of these two compensation methods are identical and are therefore described in the next Section “General notes”.

Terms

The following terms are used in the description of “Interpolatory compensation”:

Compensation value	The difference between the axis position measured by the position actual-value encoder and the required programmed axis position (= axis position of the ideal machine). The compensation value is often also referred to as the “correction” value.
Basic axis	Axis whose setpoint or actual value position forms the basis of the calculation of a compensation value.
Compensation axis	Axis whose setpoint or actual position is modified by a compensation value.
Interpolation point	A position of the base axis and the corresponding compensation value of the compensation axis.
Correction table	Table containing interpolation points.
Compensation relations	Assignment of the base axis and the corresponding compensation axis and the reference to the corresponding compensation table.

Leadscrew and measuring system errors

The measuring principle of “indirect measurement” on NC-controlled machines is based on the assumption that the leadscrew pitch is constant at any given point within the traversing range so that the actual axis position can be derived from the position of the drive spindle (ideal case).

However, manufacturing tolerances result in dimensional deviations of varying degrees of severity on spindles (so-called “leadscrew errors”).

To these are added the deviations caused by the measuring system used (differing divisions) and by the way the measuring system is mounted on the machine (measuring system errors) and any machine-dependent sources of error.

Sag errors

Weight can result in position-dependent displacement and inclination of moved parts since it can cause machine parts and their
guides to sag (see Fig. 3.2).

Large workpieces, too, e.g. cylinders, sag under their own weight.

Angularity errors

If moving axes are not positioned in exactly the required angle (e.g. perpendicular) with respect to one another, increasingly serious positioning errors will occur as the deviation from zero point becomes greater.

Compensation table

Since the deviations in dimension caused by the phenomena described above have a direct effect on workpiece machining accuracy, they need to be compensated by appropriate position-dependent correction values. The compensation values are derived from measured error curves and entered in the control in the form of compensation tables during installation. A separate table must be created for each compensation relation.

The compensation values and additional table parameters are entered in the compensation tables using special system variables.

Note

Compensation tables can be loaded only if MD 32700: ENC_COMP_ENABLE (interpolatory compensation)=0 and/or MD 32710: CEC_ENABLE (enable beam sag compensation) are set to zero.

Input of compensation table

The size of the compensation table, i.e. the number of interpolation points, must first be defined in a machine data – a Power On must then be executed.

Compensation tables can be loaded to the backed-up NC user memory by two different methods.

- The compensation values are loaded when an NC program with the compensation tables is started.
- The compensation values can also be loaded by transferring the tables from a PC via the serial interface on the MMC.

Note

Once the size of the compensation tables has been defined in machine data, the NC generates the tables after the next Power On. The default setting for these tables is “0” .

The compensation tables can be output from the “Services” operating area via the serial interface on the MMC and loaded back after editing.

These compensation values are not lost when the control is switched off because they are stored in the non-volatile user memory. They can be updated if necessary (e.g. as a result of re-measuring because of machine ageing).

Caution

When MD 18342: MM_CEC_MAX_POINTS[t] (max. number of interpolation points of beam sag comp., SRAM) or MD 38000: MM_ENC_COMP_MAX_POINTS (number of interpolation points for interpolatory comp., SRAM) is changed, the buffered NC user memory is re-initialized when the system powers up. All user data of the battery-buffered user memory (e.g. drive and MMC machine data, tool offsets, part programs, compensation tables etc.) are deleted.

References: /FB/, S7, “Memory Configuration”

Archiving

Compensation tables are not saved with the series start-up file.

To archive compensation tables, they must be output via the serial interface on the MMC. The following compensation types can be selected for archiving in the operating area “Services”, “Data OUT”

- LEC/measuring system error compensation (%_N_AX_EEC_INI)
- Beam sag/angularity compensation (%_N_AX_CEC_INI)
- Quadrant error compensation (%_N_AX_QEC_INI)

Compensation tables can also be saved as an archive file with an MMC 102/103.

Linear interpolation between interpolation points

The traversing path to be compensated delineated by the start and end positions is divided up into several (number depends on error curve shape) path segments of equal size (see figure below). The actual positions that limit these sub-paths are designated “interpolation points”. A compensation value must be entered for each interpolation point (actual position) during installation. The compensation value applied between 2 interpolation points is generated on the basis of **linear interpolation** using the compensation values for the adjacent interpolation points (i.e. adjacent interpolation points are linked along a line).

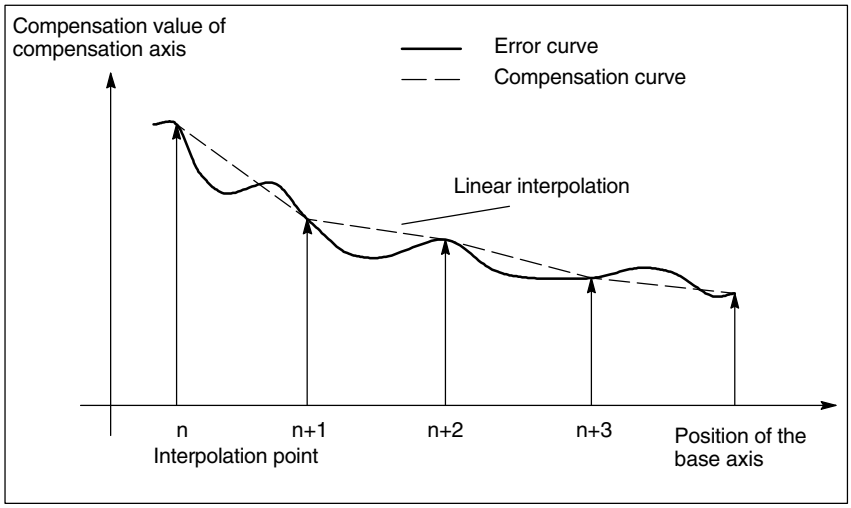


Fig. 2-7 Linear interpolation between the interpolation points

Compensation value at reference point

The compensation table should be structured such that the compensation value at the reference point is “zero”.

2.3.2 Measuring system error compensation (MSEC)

Function

The leadscrew error compensation function is part of the measuring system error compensation system.
In “Measuring system error compensation” (from now on referred to as **MSEC**), the base and compensation axes are **always identical**. It is therefore an **axial compensation** for which a definition of the base axis and compensation axis in the compensation table is not necessary.

The principle of the MSEC is to modify the axis-specific position actual value by the assigned compensation value in the interpolation cycle and to apply this value to the machine axis for immediate traversal. A positive compensation value causes the corresponding machine axis to move in the negative direction.

The magnitude of the compensation value is not limited and is not monitored. In order to avoid impermissibly high velocities and accelerations caused by compensation, small compensation values must be selected. Large compensation values can cause other axis monitoring functions to output alarms (e.g. contour monitoring, velocity setpoint limitation).

If the axis to be compensated has a 2nd position measuring system, a separate compensation table must be created and activated for each measuring system. The correct table is **automatically** used when switching between measuring systems.

Activation

The “MSEC” does not become active until the following conditions are fulfilled:

- The compensation values are stored in the NC user memory and active (after Power On).
- The function has been activated for the machine axis concerned (MD 32700: ENC_COMP_ENABLE [e] = 1). If compensation is required for a 2nd measuring system, this must also be enabled with the above machine data (e = 0: 1. measuring system; e = 1: 2. measuring system).
- The axis has been referenced (IS: “Referenced/synchronized 1 or 2” DB31, ... DBX60.4 or 60.5 = ‘1’).

As soon as these conditions have been fulfilled, the axis-specific actual value is altered by the compensation value in all modes and traversed by the machine axis immediately.

If the reference is then lost, e.g. because the encoder frequency has been exceeded (IS “Referenced/synchronized 1 or 2”=‘0’), compensation processing is deactivated.

Compensation interpolation points

For every machine axis and for every measuring system (if a 2nd measuring system is installed), the number of reserved interpolation points of the compensation table must be defined and the necessary memory reserved in MD 38000: MM_ENC_COMP_MAX_POINTS.

MD 38000: $MA_ MM_ENC_COMP_MAX_POINTS [e,AXi]

with: AXi = axis name e.g. X1, Y1, Z1

e = measuring system (e = 0: 1. measuring system; e = 1: 2. measuring system)

$$\text{MM\_ENC\_COMP\_MAX\_POINTS}[e,AXi] = \frac{\$AA\_ENC\_COMP\_MAX[e,AXi] - \$AA\_ENC\_COMP\_MIN[e,AXi]}{\$AA\_ENC\_COMP\_STEP[e,AXi]} + 1$$

Compensation table

The position-related compensation values are stored in the form of system variables for the relevant axis in the compensation table.

The following measuring-system-specific parameters must be set for the table (see figure below):

- **Compensation value for interpolation point N in compensation table** ($AA_ENC_COMP [e,N,AXi])

  For every individual interpolation point (axis position) the compensation value must be entered in the table.

  Interpolation point N is limited by the number of possible interpolation points in the relevant compensation table (MD 38000: MM_ENC_COMP_MAX_POINTS).

  The magnitude of the compensation value is not limited.
  Permissible limit of N: $0 \leq N < \text{MM\_ENC\_COMP\_MAX\_POINTS} - 1$

Note

The first and last compensation values remain active over the entire traversing range, i.e. these values should be set to “0” if the compensation table does not cover the entire traversing range.

- **Distance between interpolation points** ($AA_ENC_COMP_STEP[e,AXi])
  The distance between interpolation points corresponds to the distance between the compensation values in the relevant compensation table (see above for meaning of e and AXi).

- **Initial position** ($AA_ENC_COMP_MIN[e,AXi])
  The initial position is the axis position at which the compensation table for the relevant axis begins (≙ interpolation point 0).

  The compensation value for the initial position is
  $AA_ENC_COMP_STEP[e,**0**,AXi)].

  For all positions smaller than the initial position the compensation value of interpolation point zero is used (does not apply for table with modulo).

- **End position** ($AA_ENC_COMP_MAX[e,AXi])
  The end position is the axis position at which the compensation table for the relevant axis ends (≙ interpolation point k).

  The compensation value for the end position is
  $AA_ENC_COMP_STEP[e,**k**,AXi)].

  The compensation value of interpolation point k is used for all positions larger than the end position (exception: table with modulo functions).

  The number of required interpolation points is calculated as follows:

  $$k = \frac{\$AA\_ENC\_COMP\_MAX - \$AA\_ENC\_COMP\_MIN}{\$AA\_ENC\_COMP}$$

  With $0 \preceq k <$ MD 38000: MM_ENC_COMP_MAX_POINTS

  The following conditions apply to interpolation point k:

  – With k = MD 38000: MM_ENC_COMP_MAX_POINTS – 1

    ⇒ the compensation table is fully utilized!

  – With k = MD 38000: MM_ENC_COMP_MAX_POINTS – 1

    ⇒ the compensation table is not fully utilized; compensation values entered in the table greater than k have no effect.

- With k = MD 38000: MM_ENC_COMP_MAX_POINTS – 1

  ⇒ the compensation table is limited internally by reducing the end position; the compensation values greater than k are not used.

- **Compensation with modulo function**
  ($AA_ENC_COMP_IS_MODULO[e,AXi])
  When the compensation is activated with a modulo function, the compensation table is repeated cyclically, i.e. the compensation value at location $AA_ENC_COMP_MAX (≙ interpolation point $AA_ENC_COMP[e,k,AXi]) is followed immediately by the compensation value at location $AA_ENC_COMP_MIN (≙ interpolation point $AA_ENC_COMP[e,0,AXi]).

  For rotary axes with modulo 360° it is therefore suitable to program 0° as the initial position position ($AA_ENC_COMP_MIN) and 360° as the end position ($AA_ENC_COMP_MAX).

The compensation values entered for these two positions should be the same as otherwise the compensation value jumps from MAX to MIN at the transition point and vice versa.

$AA_ENC_COMP_IS_MODULO[e,AXi] = 0:	Compensation without modulo function
$AA_ENC_COMP_IS_MODULO[e,AXi] = 1:	Compensation with modulo function

Caution

When the compensation values are entered it is important that all interpolation points be assigned a position value within the defined range (i.e. no gaps). Otherwise, the previous valid position value is used for these interpolation points!

Note

Table parameters which contain position information are not automatically converted at measuring system change (change in MD 10240: SCALING_SYSTEM_IS_METRIC) in Software Version 4 and lower. The position information is always interpreted in the current measuring system. Conversions must be conducted externally.

In Software Version 5 and higher, by setting MD 10260: CONVERT_SCALING_SYSTEM=1. External conversion is no longer necessary.

References: /FB1/, G2, Chapter 2

Example

The following example shows compensation value inputs for machine axis X1.

```
%_N_AX_EEC_INI
CHANDATA (1)
$AA_ENC_COMP[0,0,X1]        = 0.0      ; 1. compensation value
                                         (≙interpolation point 0) +0µm
$AA_ENC_COMP[0,1,X1]        = 0.01     ; 2. compensation value
                                         (≙interpolation point 1) +10µm
$AA_ENC_COMP[0,2,X1]        = 0.012    ; 3. compensation value
                                         (≙interpolation point 2) +12µm
:
$AA_ENC_COMP[0,800,X1]      = –0.0     ; last compensation value
                                         (≙interpolation point 800)
$AA_ENC_COMP_STEP[0,X1] = 1.0          ; Distance between interpolation
                                         points 1.0 mm
$AA_ENC_COMP_MIN[0,X1]      = –200.0   ; Compensation begins
                                         at –200.0 mm
$AA_ENC_COMP_MAX[0,X1]  = 600.0        ; Compensation ends at +600.0 mm
$AA_ENC_COMP_IS_MODULO[0,X1] = 0;        Compensation without
                                         modulo function
M17
```

In this example, the number of compensation interpolation points set in MD 38000: MM_ENC_COMP_MAX_POINTS $\geq$ must be 801 or else alarm 12400 “Element does not exist” will be output.

The compensation table for this example requires at least 6.4 Kbytes of the non-volatile NC user memory (8 bytes per compensation value).

3

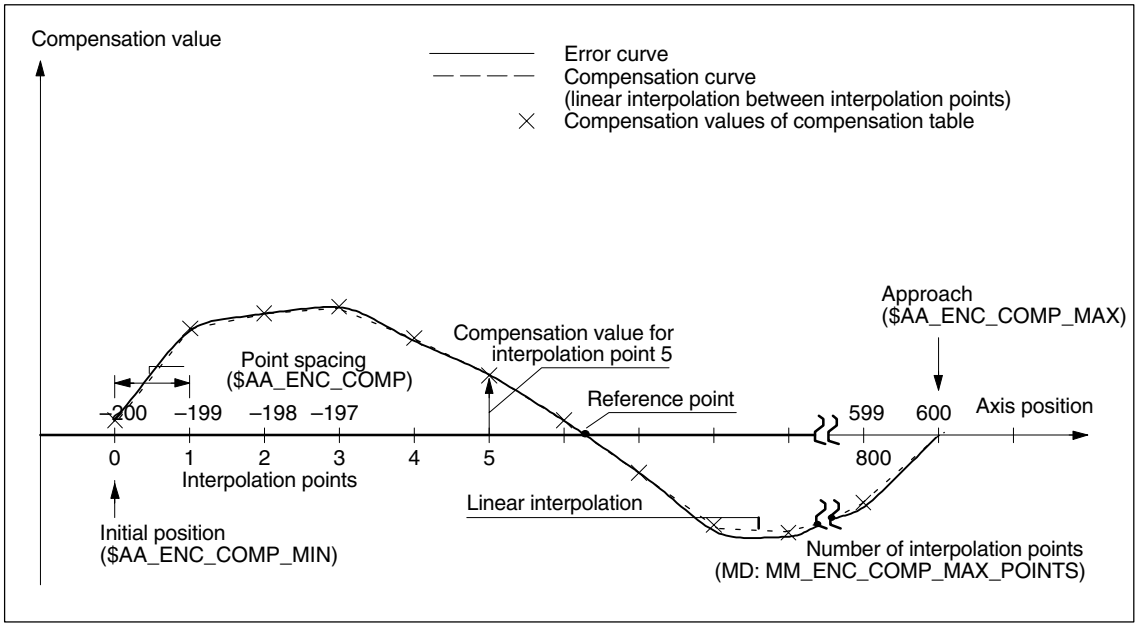


Fig. 3-1 Compensation table parameters (system variables for MSEC)

2.3.3 Sag compensation and angularity error compensation

Function

In contrast to the MSEC, the base and compensation axes need **not be identical** for “Sag compensation” or “Angularity error compensation”, requiring an axis assignment in every compensation table.

In order to compensate for sag of one axis (base axis) which results from its own weight, the absolute position of another axis (compensation axis) must be influenced. “Sag compensation” is therefore an **inter-axis compensation**.

As illustrated in the figure below, the further the machining head moves in the negative Y1 axis direction, the more the cross-arm sags in the negative Z1 axis direction.

The error must be recorded in the form of a compensation table that contains a compensation value for the Z1 axis for every actual value position in the Y1 axis. It is sufficient to enter the compensation values for the interpolation points.

When the Y1 axis traverses, the control calculates the corresponding compensation value in the Z1 axis in interpolation cycles performing linear interpolation for positions between the interpolation points. This compensation is sent to the position control loop as an additional setpoint. A positive compensation value causes the corresponding machine axis to move in the negative direction.

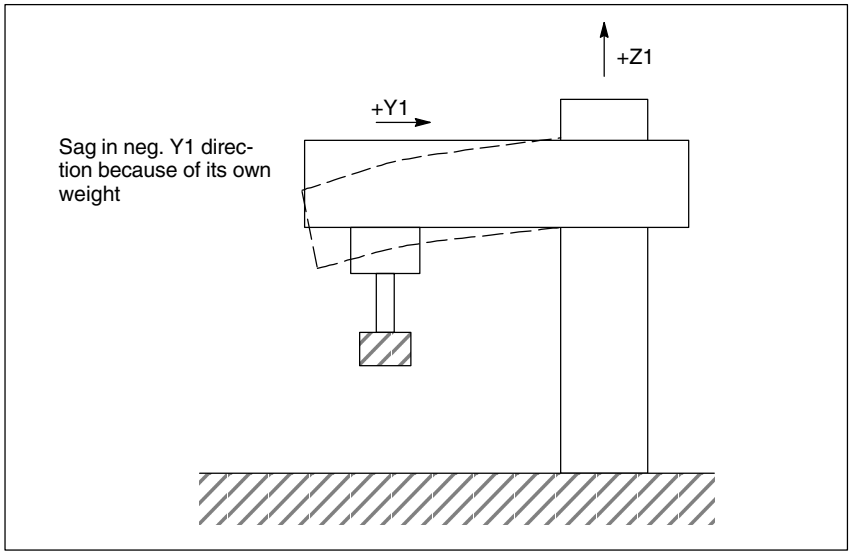


Fig. 3-2 Example of sag caused by own weight

Depending on the requirement, several compensation relations can be defined for one axis. The total compensation value results from the sum of all the compensation values of this axis.

Setting options

The many ways in which the compensation value for sag compensation can be produced/influenced are listed below (see figure below).

1. An axis can be defined as the input variable (base axis) for **several** compensation tables (settable via system variables).
2. An axis can be defined as the recipient of the output variable (compensation axis) of **several** compensation tables (settable via system variable). The total compensation value is derived from the sum of the individual compensation values.

   The following definitions apply for the maximum number of possible compensation tables:

   – Maximum number of tables available in total for all axes:
     2 * maximum number of axes of system
   – Maximum number of tables that can affect one compensation axis:
     1 * maximum number of axes of the system
3. An axis can be both a base axis and a compensation axis at any one time. The programmed (required) position setpoint is always used to calculate the compensation values.
4. The range of influence of the compensation (starting and end position of the base axis) and the distance between the interpolation points can be defined for every compensation table (settable via system variables).
5. Compensation can be direction-dependent (settable via system variables).
6. Every compensation table has a modulo function for cyclic evaluation (settable via system variables).
7. A weighting factor by which the table value is multiplied (definable as a setting data which can therefore be altered by the part program, PLC or the user at any time) can be introduced for every compensation table.
8. Compensation tables can be multiplied in pairs (settable via system variables). The product added to the total compensation value of the compensation axis.
9. The compensation can be activated in the following ways:
   – With MD 32710: CEC_ENABLE [AXi] the sum of all compensation relations is enabled for machine axis AXi.
   – With SD 41300: $SN_CEC_TABLE_ENABLE[t], evaluation of the compensation table [t] is enabled.

     It is thus possible, for example, to alter the compensation relations either from the part program or from the PLC user program (e.g. switching over the tables), depending on the machining requirements.
10. In SW 5 and higher, when MD 10260: CONVERT_SCALING_SYSTEM=1 is set, the axial MD 32711: CEC_SCALING_SYSTEM_METRIC becomes effective. The measuring system for all tables effective for this axis is set in this machine data. Hereby all position entries are interpreted together with the calculated total compensation value in the configured measuring system. External conversions of position information are no longer necessary with a measuring system change.

Note

No compensation table becomes active until the base axis and compensation axis have been referenced.

Monitoring

To avoid excessive velocities and acceleration rates on the machine axis as a result of applying sag compensation, the total compensation value is monitored and limited to a maximum value. The maximum compensation value is set in axial MD 32720: CEC_MAX_SUM on an axis-specific basis.

If the resulting total compensation value is greater than the maximum value, alarm 20124 “Sum of compensation values too high” is output. Program processing is not interrupted. The compensation value output as an additional setpoint is limited to the maximum value.

Alteration of the total compensation value is also limited axially. When limit value MD 32730: CEC_MAX_VELO is exceeded, alarm 20125 “Compensation value changed too quickly” is output; again program processing is continued. The path not covered because of the limitation is made up as soon as the compensation value is no longer subject to limitation.

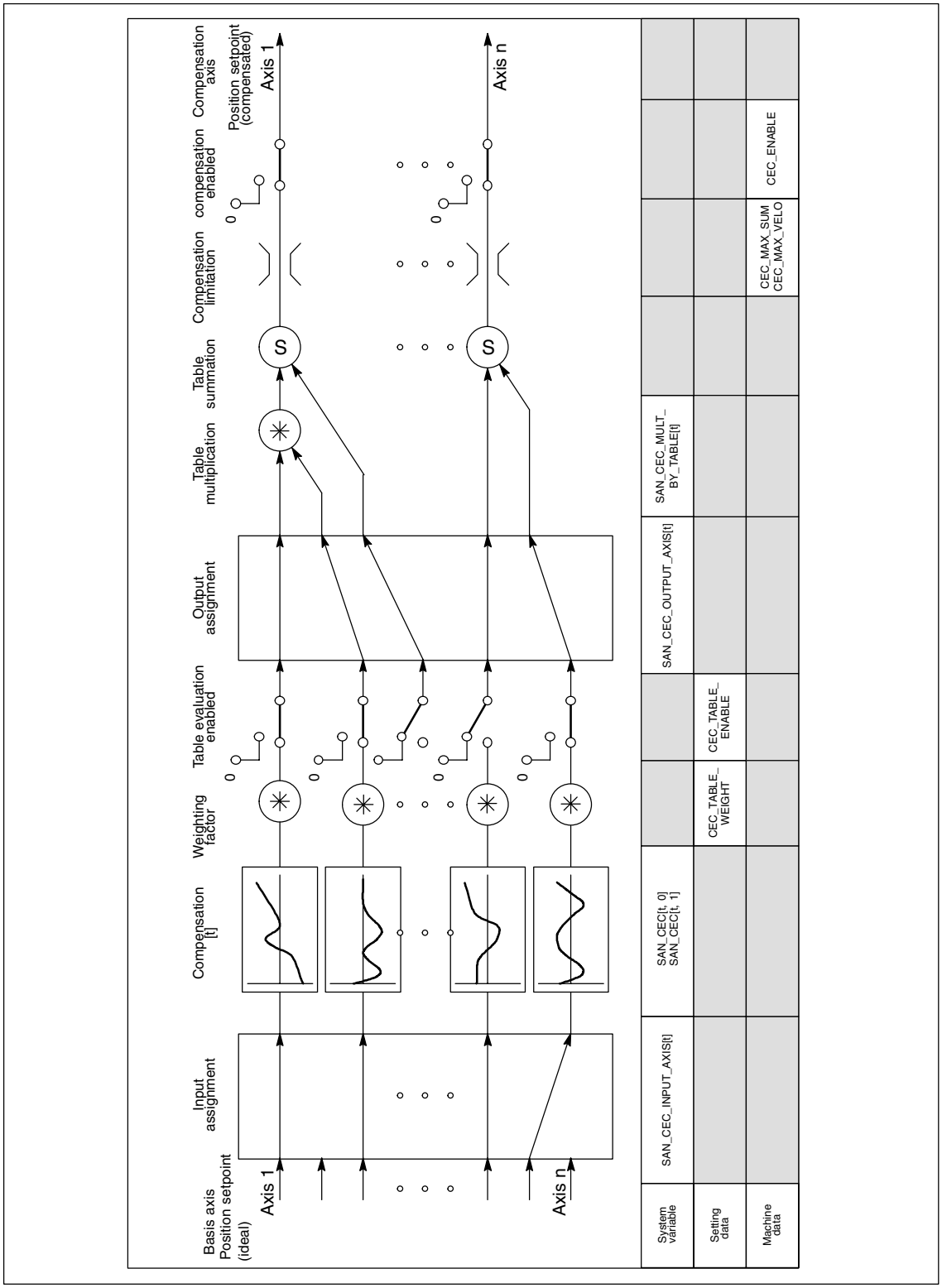


Fig. 3-3 Generation of compensation value for sag compensation

Complex compensation

Since it is possible to use the position of an axis as the input quantity (base axis) for several tables, to derive the total compensation value of an axis from several compensation relationships (tables) and to multiply tables, it is also possible to implement sophisticated and complex beam sag and angularity error compensation systems.

This function also makes it possible to deal with different error sources efficiently. For example, it is possible to combine a table with a modulo function for a periodic recurring error component with a second table without a modulo function for an aperiodic error component for the same axis.

Leadscrew errors can also be compensated with this function by parameterizing an identical axis for the base and compensation axes. However, in contrast to the MSEC, measuring-system switchovers are not automatically registered in this case.

Activation

The beam sag compensation function does not become active until the following conditions are fulfilled:

- The option “Interpolatory compensation” has been enabled.
- The function has been activated for the relevant machine axis (compensation axis).
  (MD 32710: CEC_ENABLE [AXi] = 1).
- The compensation values have been stored in the non-volatile NC user memory and are active (after Power On).
- Evaluation of the relevant compensation table has been enabled (SD 41300: CEC_TABLE _ENABLE [t] = 1)
- The current measuring system of the base and compensation axes has been referenced (IS: “Referenced/Synchronized 1 or 2” DB31, ... DBX60.4 or 60.5 = ‘1’).

As soon as these conditions have been fulfilled the setpoint position of the compensation axis is altered in all modes with reference to the setpoint position of the base axis and the corresponding compensation value and is then immediately traversed by the machine axis.

If the reference is then lost, e.g. because the encoder frequency has been exceeded (IS “Referenced/Synchronized 1 or 2” = ‘0’), compensation processing is deactivated.

Compensation interpolation points

The number of required interpolation points in the compensation table must be defined for every compensation relationship and the requisite memory space reserved in general MD 18342: MM_CEC_MAX_POINTS.

MD 18342: $MN_MM_CEC_MAX_POINTS[t]

with: [t] = Index of compensation table
with ($0 \leq t < 2$ * maximum number of axes)
i.e. t = 0: 1. compensation table
t = 1: 2. compensation table etc.

$$\text{MM\_CEC\_MAX\_POINTS[t]} = \frac{\$\text{AN\_CEC\_MAX[t]} - \$\text{AN\_CEC\_MIN[t]}}{\$\text{AN\_CEC\_STEP[t]}} + 1$$

Table parameters

The position-related corrections for the relevant compensation relationship are stored as system variables in the compensation table.

The following parameters must be set for the table (see Fig. 3.1):

- **Compensation value for interpolation point N in compensation table [t]** ($AN_CEC [t, N])

  The compensation value of the compensation axis must be entered in the table for each individual interpolation point (position of the base axis).

  Interpolation point N is limited by the number of possible interpolation points in the relevant compensation table (MD 18342: MM_CEC_MAX_POINTS).

  Permissible range of N: $0 \leq N <$ MD 18342: MM_CEC_MAX_POINTS

- **Base axis** ($AN_CEC_INPUT_AXIS[t])
  Name of machine axis whose setpoint is to be used as the input for the compensation table [t].

- **Compensation axis** ($AN_CEC_OUTPUT_AXIS[t])
  Name of machine axis to which the output of the compensation table [t] is to be applied.

Note

In multi-channel systems the “general axis identifiers” AX1... must be preset, if the identifiers of machine axis and channel axis are identical.

- **Distance between interpolation points** ($AN_CEC_STEP[t])
  The distance between interpolation points defines the distance between the input values for the compensation table [t].

- **Initial position** ($AN_CEC_MIN[t])
  The initial position is the position of the base axis at which the compensation table [t] begins (≐ interpolation point 0).

  The compensation value for the initial position is $AN_CEC [t,**0**].
  The compensation value of interpolation point 0 is used for all positions smaller than the initial position (exception: table with modulo functions).

- **End position** ($AN_CEC_MAX[t])
  The end position is the position of the base axis at which the compensation table [t] ends (≐ interpolation point k).

  The compensation value for the end position is $AN_CEC [t,**k**].

  The compensation value of interpolation point k is used for all positions larger than the end position (exception: table with modulo functions).

  The number of required interpolation points is calculated as follows:

  $$k = \frac{\$AN\_CEC\_MAX[t] - \$AN\_CEC\_MIN[t]}{\$AN\_CEC\_STEP[t]}$$

  With $0 \leq k <$ MD 18342: MM_CEC_MAX_POINTS

  The following conditions apply to interpolation point k:

  – With k = MD 18342: MM_CEC_MAX_POINTS – 1
    ⇒ the compensation table is fully utilized!

– With k < MD 18342: MM_CEC_MAX_POINTS – 1
  ⇒ the compensation table is not fully utilized; the entered compensation values greater than k have no effect.
– With k > MD 18342: MM_CEC_MAX_POINTS – 1
  ⇒ the compensation table is limited in the control by reducing the end position; the compensation values greater than k are not used.

- **Direction-dependent compensation** ($AN_CEC_DIRECTION[t])
  This system variable can be used to define whether the compensation table [t] should apply to both travel directions of the base axis or only either the positive or negative direction.
  0: Table affects both traversing directions of the base axis
  1: Table only affects the positive traversing direction of the base axis
  –1: Table only affects the negative traversing direction of the base axis
  Possible applications: Position-dependent backlash compensation can be implemented using two tables, one of which affects the positive traversing direction, the other of which affects the negative traversing direction of the same axis.
- **Table multiplication** ($AN_CEC_MULT_BY_TABLE[t])
  This option allows the compensation values of any table to be multiplied with those of another (or with themselves). The product is added as an additional compensation value to the total compensation value of the compensation table.
  Syntax: $AN_CEC_MULT_BY_TABLE[$t_1$] = t_2
  t_1 = Index of table 1 of the compensation axis
  t_2 = Number of table 2 of the compensation axis.
  It is important to ensure that the number and index of the **same table** are different!
  The general rule is: Table number = table index + 1
- **Compensation with modulo function** ($AN_CEC_IS_MODULO[t])
  When the compensation with modulo function is activated, the compensation table is repeated cyclically, i.e. the compensation value at location $AN_CEC_MAX[t] (interpolation point $AN_CEC[t,k]) is followed immediately by the compensation value at location $AN_CEC_MIN[t] (interpolation point $AN_CEC[t,0]).
  These two compensation values should be the same as otherwise the compensation value jumps from MAX to MIN at the transition point and vice versa.
  $AN_CEC_IS_MODULO[t] = 0: Compensation without modulo function
  $AN_CEC_IS_MODULO[t] = 1: Compensation with modulo function
  If modulo compensation is to be implemented with a modulo rotary axis as base axis, the compensation table used has to be modulo calculated as well.
  Example:
  MD 30300: IS_ROT_AX[AX1] = 1: Rotary axis
  MD 30310: ROT_IS_MODULO[AX1] = 1: Modulo 360°
  $AN_CEC_INPUT_AXIS[0]=AX1
  $AN_CEC_MIN[0]=0.0
  $AN_CEC_MAX[0]=360.0
  $AN_CEC_IS_MODULO[0]=1

Note

Table parameters which contain position information are not automatically converted at measuring system change (change in MD 10240: SCALING_SYSTEM_IS_METRIC) in SW 4 and lower. The position information is always interpreted in the current measuring system. Conversions must be conducted externally.

In SW 5 and higher, by setting MD 10260: CONVERT_SCALING_SYSTEM=1 you can configure the measuring system via MD 32711: CEC_SCALING_SYSTEM. External conversions of position information are no longer necessary with a measuring system change.

References: /FB1/, G2, Chapter 2

Example of table

The following example shows a compensation table for sag compensation of the Y1 axis. Depending on the position of the Y1 axis, a compensation value is applied to the Z1 axis. The 1st compensation table (t = 0) is used for this.

```
%_N_NC_CEC_INI
CHANDATA (1)
$AN_CEC [0,0]                  = 0        ; 1. compensation value (≙interpolation point 0) for Z1: ±0µm
$AN_CEC [0,1]                  = 0.01     ; 2. compensation value (≙interpolation point 1) for Z1: +10µm
$AN_CEC [0,2]                  = 0.012    ; 3. compensation value (≙interpolation point 2) for Z1: +12µm
:
$AN_CEC [0,100]                = 0        ; last compensation value (≙interpolation point 101) for Z1: ±0µm
$AN_CEC_INPUT_AXIS[0]          = (AX2)    ; Base axis Y1
$AN_CEC_OUTPUT_AXIS[0]         = (AX3)    ; Compensation axis Z1
$AN_CEC_STEP[0]                =  8       ; Distance between interpolation points 8.0 mm
$AN_CEC_MIN[0]                 = –400.0   ; Compensation begins at –400 mm
$AN_CEC_MAX[0]                 = 400.0    ; Compensation ends at Y1 = +400 mm
$AN_CEC_DIRECTION[0]           = 0        ; Table applies to both directions of travel of Y1
$AN_CEC_MULT_BY_TABLE[0] =                ;
$AN_CEC_IS_MODULO[0]           = 0        ; Compensation without modulo function
M17
```

In this example, the number of compensation interpolation points set in MD 18342: MM_CEC_MAX_POINTS [0] must be ≙101; otherwise alarm 12400 is output.

The compensation table for this example requires at least 808 bytes of non-volatile NC user memory.

Table multiplication

With the table multiplication function, any table can be multiplied with any other table (i.e. even with itself). The multiplication link is established using the system variables described above.

The following example for the compensation of machine foundation sagging illustrates an application of table multiplication.

On large machines, sagging of the foundations can cause inclination of the whole machine. For the boring mill in the second figure below, for example, it is determined that compensation of the X1 axis is dependent both on the position of the X1 axis itself (since this determines angle of inclination β) and on the height of the boring mill (i.e. the position of the Z1 axis).

To implement compensation, the compensation values of the X1 and Z1 axes must be multiplied according to the following equation (see figure below):

$$\Delta X1 = Z1 * \sin\beta(X1) \approx Z1 * \beta(X1)$$

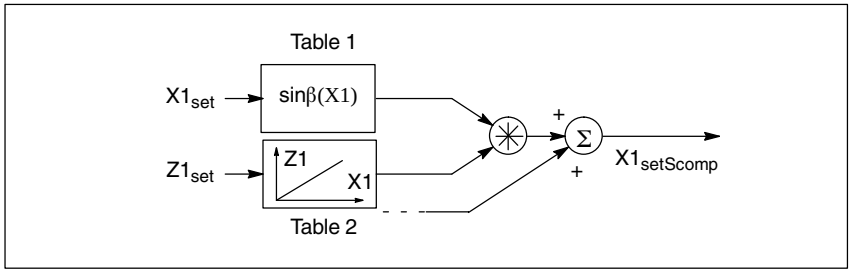


Fig. 3-4 Table multiplication

Compensation table 1 (table index = 0) describes the reaction of axis X1 on axis X1 (sine of the position-dependent tilting angle β(X1)).

Compensation table 2 (table index = 1) describes the reaction of axis Z1 on axis X1 (linear).

In table 1, the multiplication of table 1 (index = 0) with table 2 is to be selected:

$AN_CEC_MULT_BY_TABLE[0] = 2

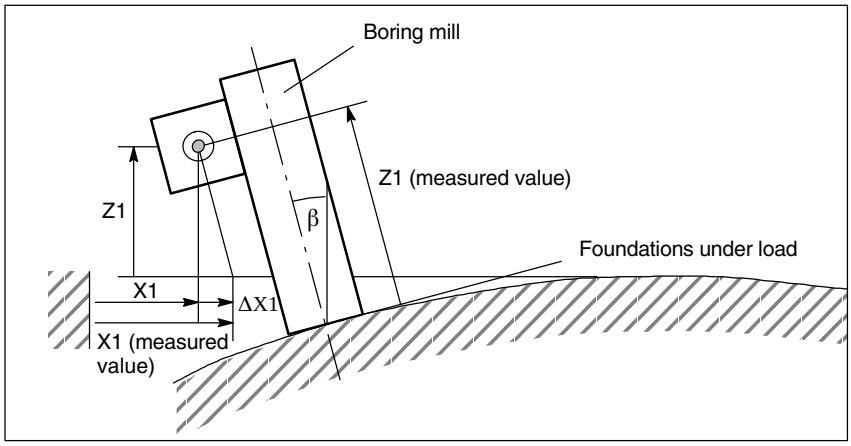


Fig. 3-5 Compensation of sag in a machine base

Example: Input of compensation values in a grid structure

The compensation values of the z axis sag on flat bed machines are often measured in practice at various points as a function of the x and y coordinates. Where such conditions need to be met, it is useful to enter the measured compensation values according to a grid-type distribution. The interpolation points with the relevant compensation values are positioned on the intersections of the grid (x–y plane). Compensation values between these interpolation points are interpolated linearly by the control.

The following example explains in more detail how sag and angularity compensation can be implemented by a grid of 4 x 5 (lines x columns) in size. The size of the whole grid is 2000x900mm^2. The compensation values are each measured in steps of 500mm along the x axis and 300mm along the y axis.

Note

The maximum dimensions of the grid (number of lines and columns) depends on the following points:

No. of lines:	Dependent on number of axes in the system (dependent on NCU type)
No. of columns:	Dependent on the maximum number of values which can be entered in a compensation table (up to a total of 2000 values)

Caution

The number of lines and columns is set in MD 18 342: MM_CEC_MAX_POINTS. The machine data is memory-configuring.

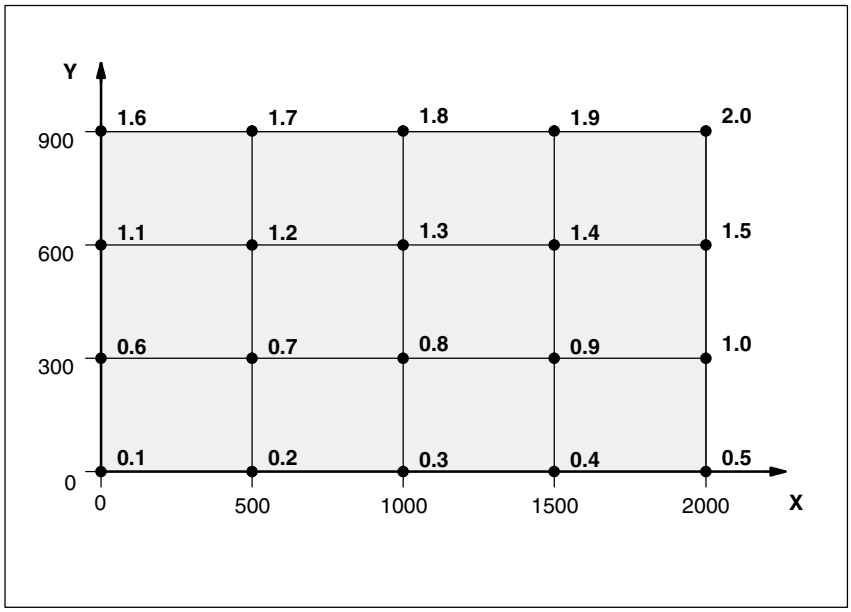


Fig. 3-6 Compensation values of z axis with chessboard-like distribution of x-y plane

Basic principle

The compensation values cannot be entered directly as a 2-dimensional grid. Compensation tables in which the compensation values are entered must be created first.

A compensation table contains the compensation values of one line (four lines in the example, i.e. four compensation tables). Compensation values 0.1 to 0.5 are entered in the first table in the example and compensation values 0.6 to 1.0 in the second (see Fig. 3-6). Compensation tables are referred to below as f tables and the table values as f_i(x) (i = number of table).

The compensation values of f tables are evaluated by multiplying them with other tables. The latter are referred to below as g tables and their values as g_i(y). The number of f tables and g tables is equal (four in the example).

In g tables, one compensation value in each table is set to 1 and all the others to 0. The position of compensation value 1 within the table is determined by the table number. In the first g table, compensation value 1 is positioned at the first interpolation point and, in the second g table, at the second interpolation point, etc. By multiplying g tables with f tables, the correct compensation value in each f table is selected by multiplying it with 1. All irrelevant compensation values are concealed through multiplication with 0.

Using this scheme, compensation value D_z at position (x/y) is calculated according to the following equation:

$$D_z(x/y)=f\_1(x)*g\_1(y) + f\_2(x)*g\_2(y) + ...$$

When the compensation value for the current position of the machine spindle is calculated, the f table values are multiplied by the g table values according to this rule.

Applied to the example, this means, for instance, that compensation value $D_z(500/300)$ is calculated by multiplying each of the function values f_i(500) in the f tables with the function values g_i(300) in the g tables:

$$D_z(500/300)= f\_1(1000)*g\_1(300) + f\_2(1000)*g\_2(300) + f\_3(1000)*g\_3(300) + f\_4(1000)*g\_4(300)$$

$$D_z(500/300)= 0.2*0 + 0.7*1 + 1.2*0 + 1.7*0 = 0.7$$

(for functions values, see also f and g tables in program code)

Program code

The application example described above can be achieved with the following part program code:

```
$MA_CEC_ENABLE[Z1]        = FALSE  ; Deactivate the compensation by
                                     setting to FALSE, allowing the table
                                     values to be altered without gener-
                                     ating alarm 17070.
NEWCONF                            ; Activate $MA_CEC_ENABLE
```

```
;Define values f_i(x) in the f tables:
;Function values f_1(x) for table with index [0]
$AN_CEC [0.0]          =0.1
$AN_CEC [0.1]          =0.2
$AN_CEC [0.2]          =0.3
$AN_CEC [0.3]          =0.4
$AN_CEC [0.4]          =0.5

;Function values f_2(x) for table with index [1]
$AN_CEC [1.0]          =0.6
$AN_CEC [1.1]          =0.7
$AN_CEC [1.2]          =0.8
$AN_CEC [1.3]          =0.9
$AN_CEC [1.4]          =1.0

;Function values f_3(x) for table with index [3]
$AN_CEC [2.0]          =1.1
$AN_CEC [2.1]          =1.2
$AN_CEC [2.2]          =1.3
$AN_CEC [2.3]          =1.4
$AN_CEC [2.4]          =1.5

;Function values f_4(x) for table with index [3]
$AN_CEC [3.0]          =1.6
$AN_CEC [3.1]          =1.7
$AN_CEC [3.2]          =1.8
$AN_CEC [3.3]          =1.9
$AN_CEC [3.4]          =2.0

;Enable evaluation of f tables with compensation values
$SN_CEC_TABLE_ENABLE[0] =TRUE
$SN_CEC_TABLE_ENABLE[1] =TRUE
$SN_CEC_TABLE_ENABLE[2] =TRUE
$SN_CEC_TABLE_ENABLE[3] =TRUE

;Define weighting factor of f tables
$SN_CEC_TABLE_WEIGHT[0] =1.0
$SN_CEC_TABLE_WEIGHT[1] =1.0
$SN_CEC_TABLE_WEIGHT[2] =1.0
$SN_CEC_TABLE_WEIGHT[3] =1.0


;Changes to the following table parameters take effect after power ON.
;Define base axis X1
$AN_CEC_INPUT_AXIS[0]     =(X1)
$AN_CEC_INPUT_AXIS[1]     =(X1)
$AN_CEC_INPUT_AXIS[2]     =(X1)
$AN_CEC_INPUT_AXIS[3]     =(X1)

;Define compensation axis Z1
$AN_CEC_OUTPUT_AXIS[0]   =(Z1)
$AN_CEC_OUTPUT_AXIS[1]   =(Z1)
$AN_CEC_OUTPUT_AXIS[2]   =(Z1)
$AN_CEC_OUTPUT_AXIS[3]   =(Z1)
```

```
;Define distance between interpolation points for compensation values in f
tables
$AN_CEC_STEP[0]          =500.0
$AN_CEC_STEP[1]          =500.0
$AN_CEC_STEP[2]          =500.0
$AN_CEC_STEP[3]          =500.0

;Compensation starts at X1=0
$AN_CEC_MIN[0]           =0.0
$AN_CEC_MIN[1]           =0.0
$AN_CEC_MIN[2]           =0.0
$AN_CEC_MIN[3]           =0.0

;Compensation ends at X1=2000
$AN_CEC_MAX[0]           =2000.0
$AN_CEC_MAX[1]           =2000.0
$AN_CEC_MAX[2]           =2000.0
$AN_CEC_MAX[3]           =2000.0

;Values of f tables with index [t1] are multiplied by values in g tables
;with the number [t2]
;in accordance with the rule of calculation specified above
$SN_CEC_MULT_BY_TABLE_[0]=5
$SN_CEC_MULT_BY_TABLE_[1]=6
$SN_CEC_MULT_BY_TABLE_[2]=7
$SN_CEC_MULT_BY_TABLE_[3]=8

;Define the g table values for g_i(y) :
:Function values g_1(x) for table with index [4]
$AN_CEC [4.0]            =1.0
$AN_CEC [4.1]            =0.0
$AN_CEC [4.2]            =0.0
$AN_CEC [4.3]            =0.0

;Function values g_2(x) for table with index [5]
$AN_CEC [5.0]            =0.0
$AN_CEC [5.1]            =1.0
$AN_CEC [5.2]            =0.0
$AN_CEC [5.3]            =0.0

;Function values g_3(x) for table with index [6]
$AN_CEC [6.0]            =0.0
$AN_CEC [6.1]            =0.0
$AN_CEC [6.2]            =1.0
$AN_CEC [6.3]            =0.0

;Function values g_4(x) for table with index [7]
$AN_CEC [7.0]            =0.0
$AN_CEC [7.1]            =0.0
$AN_CEC [7.2]            =0.0
$AN_CEC [7.3]            =1.0

;Enable evaluation of g tables with compensation values
$SN_CEC_TABLE_ENABLE[4] =TRUE
$SN_CEC_TABLE_ENABLE[5] =TRUE
$SN_CEC_TABLE_ENABLE[6] =TRUE
$SN_CEC_TABLE_ENABLE[7] =TRUE
```

```
;Define weighting factor for g tables
$SN_CEC_TABLE_WEIGHT[4] =1.0
$SN_CEC_TABLE_WEIGHT[5] =1.0
$SN_CEC_TABLE_WEIGHT[6] =1.0
$SN_CEC_TABLE_WEIGHT[7] =1.0

;Changes to the following table parameters take effect after power ON.
;Define basic axis Y1
$AN_CEC_INPUT_AXIS[4]      =(Y1)
$AN_CEC_INPUT_AXIS[5]      =(Y1)
$AN_CEC_INPUT_AXIS[6]      =(Y1)
$AN_CEC_INPUT_AXIS[7]      =(Y1)

;Define compensation axis Z1
$AN_CEC_OUTPUT_AXIS[4]   =(Z1)
$AN_CEC_OUTPUT_AXIS[5]   =(Z1)
$AN_CEC_OUTPUT_AXIS[6]   =(Z1)
$AN_CEC_OUTPUT_AXIS[7]   =(Z1)

;Define distance between interpolation points for compensation values in
g tables
$AN_CEC_STEP[4]              =300.0
$AN_CEC_STEP[5]              =300.0
$AN_CEC_STEP[6]              =300.0
$AN_CEC_STEP[7]              =300.0

;Compensation starts at Y1=0
$AN_CEC_MIN[4]               =0.0
$AN_CEC_MIN[5]               =0.0
$AN_CEC_MIN[6]               =0.0
$AN_CEC_MIN[7]               =0.0

;Compensation ends at Y1=900
$AN_CEC_MAX[4]               =900.0
$AN_CEC_MAX[5]               =900.0
$AN_CEC_MAX[6]               =900.0
$AN_CEC_MAX[7]               =900.0

$MA_CEC_ENABLE[Z1]           =TRUE    ; Re-activate compensation
NEWCONF

;Execute a program test to check effectiveness of compensation.
G01 F1000 X0 X0 Z0 G90
R1=0 R2=0
LOOP_Y:
LOOP_X:
STOPRE
X=R1 Y=R2
M0                                    ; Wait to check the CEC value
R1=R1+500
IF R1 <=2000 GOTOB LOOP_X
R1=0
R2=R2+300
IF R2<=900 GOTOB LOOP_Y
```

Note

You can read the compensation value under variable “Sag + temperature compensation” on the MMC. To do so, select softkey “Diagnosis” followed by softkey “Service axis”. The currently effective compensation value is displayed next to the “Sag + temperature compensation” variable.

```
;The Power On machine data are set to prepare
;the table configuration
;cec.md:
;Set option data for start-up

;Define the number of interpolation points in the compensation tables
;Machine data is memory-configuring.
$MN_MM_CEC_MAX_POINTS[0]=5
$MN_MM_CEC_MAX_POINTS[1]=5
$MN_MM_CEC_MAX_POINTS[2]=5
$MN_MM_CEC_MAX_POINTS[3]=5
$MN_MM_CEC_MAX_POINTS[4]=4
$MN_MM_CEC_MAX_POINTS[5]=4
$MN_MM_CEC_MAX_POINTS[6]=4
$MN_MM_CEC_MAX_POINTS[7]=4

$MA_CEC_MAX_SUM[AX3]=10.0        ; Define the max. total compensation
                                   value
$MA_CEC_MAX_VELO[AX3]=100.0      ; Limit the max. changes to the total
                                   compensation value
M17
```

2.3.4 Special features of interpolatory compensation

Measurement

The “Measurement” function supplies the compensated actual positions (ideal machine) required by the machine operator or programmer.

TEACH IN

The “TEACH IN” function also uses compensated position values to determine the actual positions to be stored.

Software limit switches

The ideal position values (i.e. the position actual values corrected by the MSEC and backlash compensation functions) are also monitored by the software limit switches.

Position display

The position actual-value display in the machine coordinate system shows the ideal (programmed) actual position value of the axis (ideal machine).

In the axis/spindle service display (operating area Diagnosis) the positional value determined by the measuring system plus the sum of the backlash compensation and leadscrew error compensation is displayed (= actual position value measuring system 1/2).

Compensation value display

The following compensation values are also output in the “Axes” service display (Diagnosis operating area):

Axes service display	Meaning
Absolute compensation value measuring system 1 or 2	Display value corresponds to the total compensation value calculated from “MSEC” and “Backlash compensation” for the current actual position of the axis (measuring system 1 or 2).
Compensation value beam sag/temperature	Display value is the sum of the compensation values from “beam sag compensation” and “temperature compensation” for the current actual position of the axis.

References: /FB1/, D1, “Diagnostic Tools”

Reference point loss

If the reference point for the base axis is lost (IS: “Referenced/Synchronized 1 or 2” DB31, ... DBX60.4 or 60.5 = ‘0’), the MSEC and backlash compensation functions are deactivated in the affected axes. When the reference point is reached these compensations are automatically switched on again.

Protection of access to compensation tables

There is currently no protection against access to the compensation tables.

Controller enable

As a result of the compensation relationship, a traversing movement by the base axis may also cause the compensation axis to move, making it necessary for controller enable signals to be set for these axes (PLC user program). Otherwise the compensation only has a limited effect.

Travel commands to PLC

The traversing signals are output in response to every change in compensation value to give the axis the opportunity to open its brakes (which, for example, operate as a function of travel commands).

2.4 Following error compensation (feedforward control)

2.4.1 General

Axial following errors

The axial following error can be reduced almost to zero with the help of the feedforward control. This feedforward control is therefore also called “following error compensation”.

The following error causes undesired velocity-dependent contour errors especially during acceleration at contour curves, e.g. arcs and corners.

Feedforward control methods

The following feedforward control methods can be used to implement “following error compensation”:

- Velocity feedforward control (SINUMERIK 840D and FM-NC) (included in the basic version)
- Torque feedforward control (on SINUMERIK 840D) (option)

The selection is made in MD 32620: FFW_MODE (feedforward control method).

Note

The torque feedforward control method cannot be used for the combination SINUMERIK 840Di and SIMODRIVE 611U drive.

Activation/ deactivation in part program

The feedforward control can also be activated and deactivated by means of the following high-level language elements in the part program:

FFWON	Feedforward control ON
FFWOF	Feedforward control OFF

The default setting (i.e. M30 even after Reset) is entered in channel-specific MD 20150: GCODE_RESET_VALUES (initial setting of G groups).

MD 32630: FFW_ACTIVATION_MODE defines for each axis whether the feedforward control can be switched on with FFWON or off with FFWOF.

FFWON and FFWOF are used to activate and deactivate respectively the feedforward control of all axes/spindles in the channel for which MD 32630: FFW_ACTIVATION_MODE = 1 is set (as well as MD 32620: FFW_MODE = 1 or 2).

MD 32630:FFW_ACTIVATION_MODE should therefore have identical settings for axes that interpolate with each other.

The feedforward control should only be switched on or off while the axis/spindle is stationary to prevent jerk. This is the responsibility of the programmer.

Conditions

The following points should be noted before the feedforward control is applied:

- Rigid machine behavior
- Precise knowledge about the machine dynamic response
- No sudden changes in the position and speed setpoints

Optimization of control loop

The feedforward control is set on an axis/spindle-specific basis. First of all, the current control loop, speed control loop and position control loop must be set to an optimum for the axis/spindle.

References: /IAF/, Installation and Start-Up Guide SINUMERIK FM-NC
/IAD/, Installation and Start-Up Guide SINUMERIK 840D

Activation

MD 32620: FFW_MODE must first be set to select the desired feedforward control mode.

0 = No feedforward control

1 = Speed feedforward control (default setting)

2 = Torque feedforward control (only possible with SINUMERIK 840D)
The option must be enabled before torque feedforward control is selected.

Parameter assignments

The feedforward control parameters must then be assigned to the relevant axis/spindle and then entered in the machine data.

2.4.2 Speed feedforward control

In the case of speed feedforward control, a velocity setpoint is also applied directly to the input of the speed controller (see figure below). This additional setpoint can be weighted by a factor that must equal approximately “1” as standard.

In order to achieve a correctly set speed feedforward control, the equivalent time constant of the speed control loop must be determined exactly and entered as a machine data.

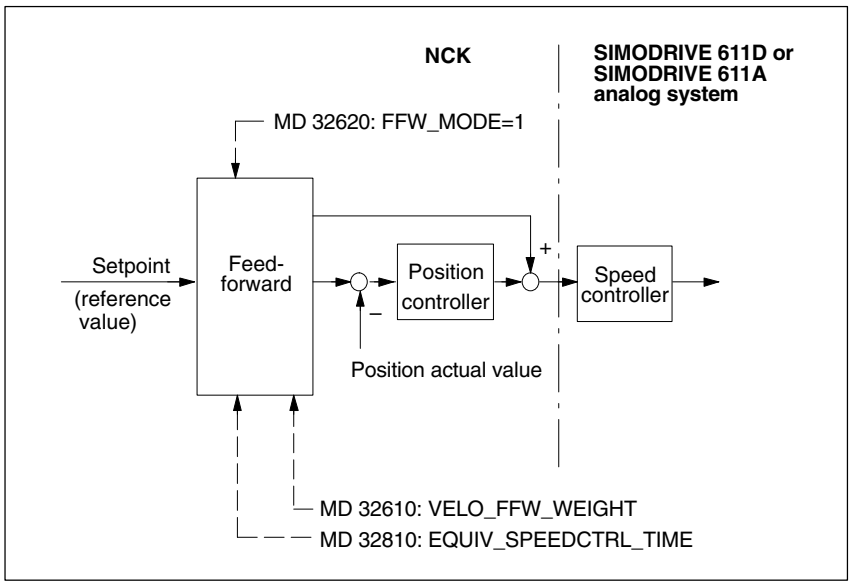


Fig. 3-7 Speed feedforward control

Parameters

The following axis-specific parameters must be defined for the speed feedforward control during installation:

- MD 32610: VELO_FFW_WEIGHT
  Feedforward control factor
- MD 32810: EQUIV_SPEEDCTRL_TIME
  Equivalent time constant of the closed speed control loop

Parameters for speed feedforward control

MD 32810: EQUIV_SPEEDCTRL_TIME
Equivalent time constant of closed speed control loop

The equivalent time constant of the closed speed control loop is determined by measuring the step response of the speed control loop. With the 611D, the settling process can be displayed using the installation tools.

References: /IAF/, “Installation and Start-Up Guide”
/IAD/, “Installation and Start-Up Guide”

The equivalent time constant of the speed control loop can also be generated from the position control cycle (=basic system cycle x factor for position control cycle) plus the speed setpoint filter (drive machine data 1500 ... 1521).

On the SINUMERIK FM-NC, for example, a speed setpoint jump can be generated using a battery box or an analog function generator. The equivalent time constant must be determined as exactly as possible.

MD 32610: VELO_FFW_WEIGHT
Feedforward control factor for speed feedforward control

If the control loop for axis/spindle is optimally set and the equivalent time constant has been determined exactly, the feedforward control factor will be approximately **1**. Therefore the initial value to be entered in the machine data is 1 (= standard default setting).

With this value the following error will be reduced to nearly zero (i.e. control deviation is 0) when speed is constant. This should be checked by making positioning movements and looking at the actual resulting control deviation shown on the service display.

References: /FB/, D1, “Diagnostic Tools”

Fine adjustment

By making fine adjustments to the values set in MD 32610: VELO_FFW_WEIGHT and MD 32810: EQUIV_SPEEDCTRL_TIME, it is possible to set the desired response for the relevant axis/spindle.

This is done by traversing the axis/spindle at a constant velocity and checking the affect of the changes made in the machine data in the service display “Control deviation”. The adjustment criterion for the speed feedforward control is “control deviation” = 0.

Case 1: When the axis is traversed in the positive direction the “control deviation” displays a **positive** value.
⇒ The equivalent time constant of the speed control loop or the feedforward factor is **too small**

Case 2: When the axis is traversed in the positive direction the “control deviation” displays a **negative** value.
⇒ The equivalent time constant of the speed control loop or the feedforward control factor is **too large**

A small acceleration and a large feedrate should be chosen so that the values can be easily read on the service display. This produces very long acceleration phases from which it is easy to read off the control deviation.

An example with axis X:

MD 32300: MAX_AX_ACCEL = 0,1 ; m/s^2
MD 32000: MAX_AX_VELO = 20000,0 ; mm/min

; Part program for setting the equivalent time constant

```
G1 F20000
FFWON
LOOP:
X1000
X0
GOTOB LOOP
M30
```

Example for active speed feedforward control of axes 1, 2 and 3.

Equivalent time constant of the speed control loop
(MD 32810: EQUIV_SPEEDCTRL_TIME) for

- Axis 1: 2 ms
- Axis 2: 4 ms (dynamically the slowest axis)
- Axis 3: 1 ms

The values for the time constant of the dynamic response adaptation
(MD 32910: DYN_MATCH_TIME) are then as follows for:

- Axis 1: 2 ms
- Axis 2: 0 ms
- Axis 3: 3 ms

References: /FB/, G2, “Velocities, Setpoint/Actual-Value Systems, Closed-Loop Control”

Lead time with speed setpoint

Transfer of speed setpoints to the drive can be set in machine data MD 10082 and MD 10083.

Note

It is only possible to fix the lead time for output of speed setpoints with the digital 611D drives.

The lead time for output of speed setpoints is determined in MD 10082: CTRLOUT_LEAD_TIME. The larger the value entered, the sooner the drive transfers the speed setpoints.

This means that:

- 0 %: Setpoints are transferred at the beginning of the next position control cycle
- 50 %: Setpoints are already transferred after execution of half of the position control cycle

A reasonable lead time can only be determined by measuring the maximum position control calculating time. MD 10083: CTRLOUT_LEAD_TIME_MAX suggests a value measured by the control. As this is a net value, it is advisable for the user to provide for a safety allowance of, for example, 5 %.

Note

If lead times that are input are too high, this can cause output of drive alarm 300506.

The input value is rounded to the next lower speed controller pulse rate in the drive. If the speed controller pulse rate settings of the drives are different, changing the value will not necessarily lead to the same degree of controller improvement for all configured drives.

2.4.3 Torque feedforward control (not 840Di)

In the case of torque feedforward control, an additional current setpoint proportional to the torque is applied directly to the current controller input (see figure below). This value is formed using the acceleration and moment of inertia.

In order to achieve a correctly set torque feedforward control, the exact equivalent time constant must be determined and entered in the machine data.

Because of the direct current setpoint injection, torque feedforward control is only possible with digital drives (SINUMERIK 840D).

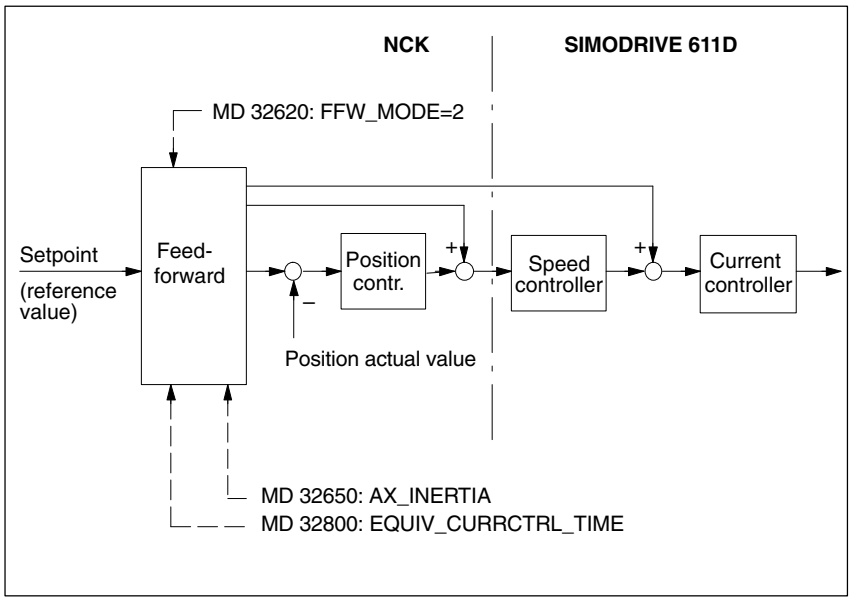


Fig. 3-8 Torque feedforward control

Application

Torque feedforward control is required to achieve high contour accuracy where the demands on the dynamics are great. If set correctly, the following error can almost be completely compensated even during high acceleration.

Parameters

The following axis-specific parameters must be defined during installation for torque feedforward control:

- MD 32650: AX_INERTIA
  Moment of inertia of the axis for torque feedforward control (from the point of view of the drive)
- MD 32800: EQUIV_CURRCTRL_TIME
  Equivalent time constant of current control loop
- SIMODRIVE 611D-MD 1004: CTRL_CONFIG
  Configuration structure
  Set bit 0 = “1” (torque feedforward control active)

Parameters for torque feedforward control

(available only on SINUMERIK 840D)

SIMODRIVE 611D MD 1004: CTRL_CONFIG configuration structure

The torque feedforward control is activated in the SIMODRIVE 611D with bit 0 = “1”.

MD 32800: EQUIV_CURRCTRL_TIME
Equivalent time constant of closed current control loop

The equivalent time constant of the closed current control loop is determined by measuring the step response of the current control loop. With the SIMODRIVE 611D, the settling process can be displayed using the installation tools.

In addition, the current setpoint of the 1st drive of each module on the DA converter of the module is output so that it can also be observed with an oscilloscope.

References: /IAD/, “Installation and Start-Up Guide”

The equivalent time constant must be determined as exactly as possible.

MD 32650: AX_INERTIA
Total moment of inertia of the axis

The total moment of inertia (moment of inertia of drive + load referred to the motor shaft) of the axis must be determined and entered in the MD for torque feedforward control.

1 to 2 times the SIMODRIVE 611D-MD 1117: MOTOR_INERTIA (motor moment of inertia) is the recommended initial value setting for MD 32650: AX_INERTIA.

Fine adjustment

By making fine adjustments to the values set in MD 32800: EQUIV_CURRCTRL_TIME and MD 32650: AX_INERTIA, it is possible to set the desired response for the relevant axis/spindle.
Because acceleration is very fast the service display cannot be used to finely adjust the parameters. In the case of the SIMODRIVE 611D, for example, changes made to the machine data should be checked by recording the following error from an analog setpoint output (this can only be done with the installation tool).

It is important to observe the following error against constant travel even when the axis/spindle is accelerating.
The adjustment criterion for torque feedforward control is
Following error $\approx$ 0

Case 1: When the axis is traversed in the positive direction the following error shows a **positive** value.
⇒ The values entered for the equivalent time constant of the current control loop or for the moment of inertia of the axis are **too small**

Case 2: When the axis is traversed in the positive direction the following error shows a **negative** value.
⇒ The values entered for the equivalent time constant of the current control loop or for the moment of inertia of the axis are **too large**

Setting for interpolating axes

The feedforward control parameters must be set optimally for each axis even in the case of interpolating axes. The axes can have **different** feedforward control settings.

Check contour monitoring

The two equivalent time constants (MD 32810: EQUIV_SPEEDCTRL_TIME and MD 32800: EQUIV_CURRCTRL_TIME) influence the contour monitoring which should therefore subsequently be checked.

References: /FB/, A3, “Axis Monitoring, Protection Zones”

Effect on servo gain factor

When the feedforward control is set correctly, the response to setpoint changes in the controlled system under speed feedforward control is as dynamic as that of the speed control loop or, under torque feedforward control, as that of the current control loop, i.e. the servo gain factor set in MD 32200: POS_CTRLGAIN has very little influence on the response to setpoint changes (e.g. corner errors, overshoots, circle/radius errors).

On the other hand, feedforward control does not affect the disturbance characteristic (synchronism). In this case, the factor set in MD 32200: POS_CTRLGAIN is the active factor.

Service display “Servo gain factor”

When a feedforward control is active, the servo gain of the axis (corresponds to servo gain factor applied to response to setpoint changes) shown in the Service display is very high.

Dynamic response adaptation

For axes that interpolate with one another, but with different axial control loop response times, dynamic response adaptation can be used to achieve identical time responses of all axes to ensure optimum contour accuracy without loss of control quality.

When a feedforward control is active, the difference between the equivalent time constants of the “slowest” speed or current control loop for the relevant axis must be entered as the time constant of the dynamic response adaptation (MD 32910: DYN_MATCH_TIME).

2.5 Friction compensation (quadrant error compensation)

2.5.1 General

Function

Friction occurs predominantly in the gearing and guideways. Static friction is especially noticeable in the machine axes. Because it takes a greater force to initiate a movement (breakaway) than to continue it, a greater following error occurs at the beginning of a movement.

The same phenomenon occurs on a change of direction where static friction causes a jump in frictional force. If, for example, one axis is accelerated from a negative to a positive velocity, it sticks for a short time as the velocity passes through zero because of the changing friction conditions. With interpolating axes, changing friction conditions can cause contour errors.

Quadrant errors

This behavior is particularly apparent on circular contours on which one axis is moving at maximum path velocity and the other is stationary at quadrant transitions. With the aid of friction compensation these so-called “quadrant errors” can be almost completely eliminated.

Principle

Measurements on machines have shown that contour errors caused by static friction can be effectively compensated by the injection of an additional setpoint pulse with the correct sign and amplitude.

Friction compensation methods

One of two friction compensation methods can be selected on the SINUMERIK 840D (MD 32490: FRICT_COMP_MODE “Type of friction compensation”):

- Conventional friction compensation (MD 32490: FRICT_COMP_MODE = 1)

With this type, the intensity of the compensation pulse can be set according to the characteristic as a function of the acceleration. This characteristic must be determined and parameterized during start-up using the circularity test. The procedure for this is relatively complicated and requires some experience.

Conventional friction compensation can also be used with SINUMERIK FM-NC.

- Quadrant error compensation with neural networks
  (option on SINUMERIK 840D) (MD 32490: $MA_FRICT_COMP_MODE = 2)

To simplify start-up, the compensation characteristic no longer has to be set manually by the start-up engineer but is determined automatically during a training phase and then stored in the non-volatile user memory.

The neural network can reproduce a compensation curve of much better quality and precision.

The function also allows simple automatic re-optimization directly at the machine.

Circularity test

The friction compensation function (both conventional and neural friction compensation) can be started up most easily by means of a circularity test. This is done by following a circular contour, measuring the actual position and representing the deviations from the programmed radius (especially at the quadrant transition points) graphically. The measurements are recorded using a “Trace” that is stored in the passive file system.

This circularity test is a function of the “Installation tool”. With the MMC101 or MMC102/103, this function can be selected directly in the Diagnosis area.

See Section 2.7 for more detailed information about the circularity test.

2.5.2 Conventional friction compensation

Method of friction compensation

Conventional friction compensation is selected by entering the value 1 in MD 32490: FRICT_COMP_MODE (friction compensation type).

Amplitude adaptation

In many cases, the injected amplitude of the friction compensation value does not remain constant over the whole acceleration range. For example, for optimum compensation with high accelerations, a smaller compensation value must be injected than for smaller accelerations. For this reason, friction compensation **with adapted injection amplitude** can be activated in cases where high accuracy is required (see figure below). The function is activated axis-specifically in MD 32510:FRICT_COMP_ADAPT_ ENABLE = 1 (adaptation for friction compensation active).

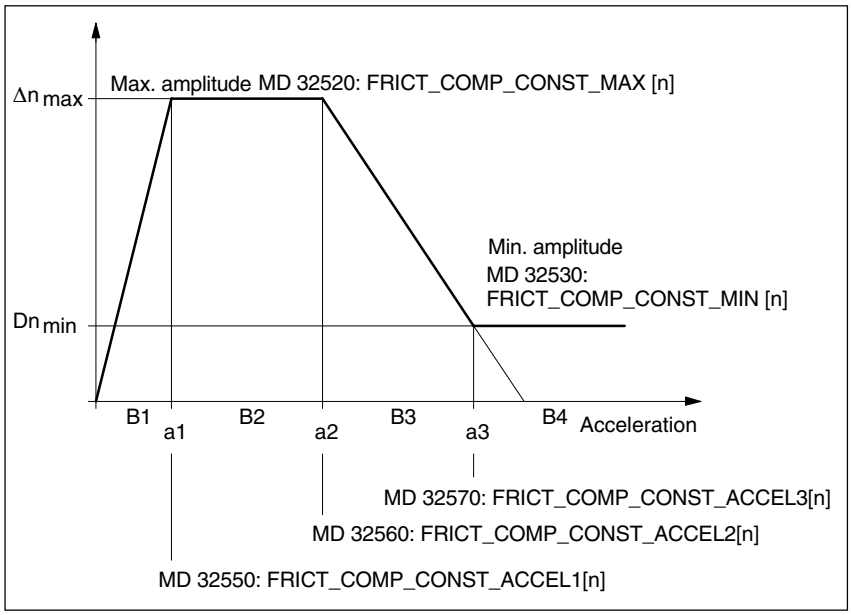


Fig. 3-9 Typical curve for friction compensation with amplitude adaptation

The adaptation characteristic is divided into four ranges (a different injection amplitude Δn is applied in each range):

B1:	for $a < a_1$	$\Delta n = \Delta nmax * a / a_1$
B2:	for $a_1 \leq a \leq a_2$	$\Delta n = \Delta nmax$
B3:	for $a_2 < a < a_3$	$\Delta n = \Delta nmax * (1 - (a - a_2) / (a_3 - a_2))$
B4:	for $a \geq a_3$	$\Delta n = \Delta nmin$

Characteristic parameters

The parameters of the adaptation characteristic in the figure above must be entered as machine data for specific axes.

Δn	=	Injection amplitude of the friction compensation value
Δn_{max}	=	Maximum friction compensation value MD 32520: FRICT_COMP_CONST_MAX [n]
Δn_{min}	=	Minimum friction compensation value MD 32530: FRICT_COMP_CONST_MIN [n]
a_1	=	Adaptation acceleration value 1 for friction compensation MD 32550: FRICT_COMP_ACCEL1 [n]
a_2	=	Adaptation acceleration value 2 for friction compensation MD 32560: FRICT_COMP_ACCEL2 [n]
a_3	=	Adaptation acceleration value 3 for friction compensation MD 32570: FRICT_COMP_ACCEL3 [n]

Note about shape of characteristic

In special cases, the calculated characteristic may deviate from the typical shape illustrated in the figure above.
In some cases, the value for Δn_{min} (MD 32530: FRICT_COMP_CONST_MIN) may even be greater than Δn_{max} (MD 32520: FRICT_COMP_CONST_MAX).

2.5.3 Start-up of conventional friction compensation

Circularity test

The friction compensation function can be started up most easily by means of a circularity test. Here, deviations from the programmed radius (especially at the quadrant transitions) can be measured and displayed while traversing a circular contour.

Step-by-step start-up

The conventional friction compensation function must first be selected.
(MD 32490: FRICT_COMP_MODE=1).

The friction compensation value mainly depends on the machine configuration. Installation is performed in two stages.

- Stage 1: Calculation of the compensation values without adaptation
- Stage 2: Calculation of the adaptation characteristic (if the friction compensation is dependent on the acceleration and the results of stage 1 are not satisfactory).

Installation stage 1: Friction compensation without adaptation

1. Circularity test without friction compensation

A circularity test without friction compensation
(MD 32500: FRICT_COMP_ENABLE = 0) should be performed first.
The procedure for performing a circularity test is described in Section 2.7.

A typical characteristic of quadrant transitions without friction compensation is shown in the figure below.

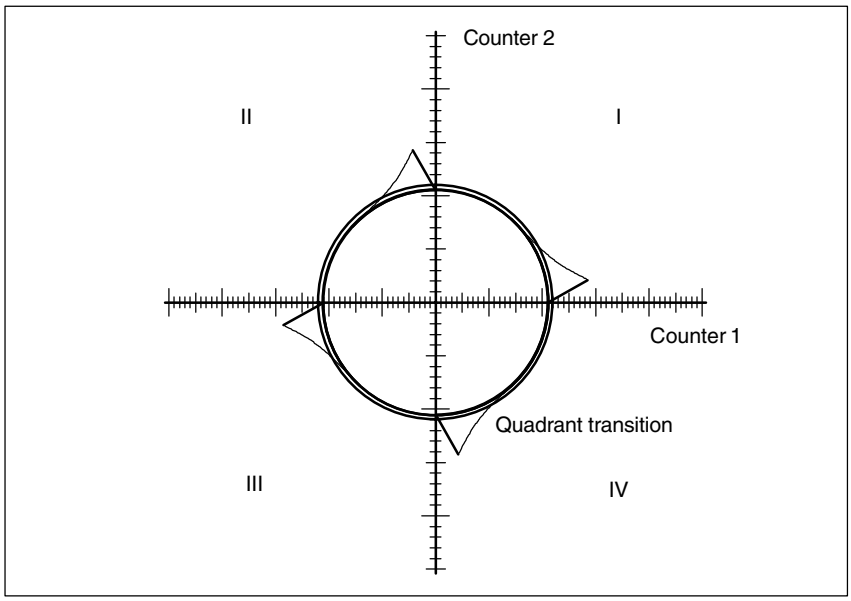


Fig. 3-10 Uncompensated radius deviation at quadrant transitions

2. Enabling the friction compensation

After this, the friction compensation must be activated for the axis/spindle in question.

Activate friction compensation
⇒ MD 32500: FRICT_COMP_ENABLE[n] = 1

3. Deactivate adaptation

In order to start up friction compensation without adaptation, the adaptation must be deactivated.

Deactivate adaptation
⇒ MD 32510: FRICT_COMP_ADAPT_ENABLE[n] = 0

4. Determine compensation parameters

Friction compensation without adaptation is defined by the following parameters:

1. MD 32520: FRICT_COMP_CONST_MAX [n] friction compensation value (amplitude) in [mm/min]
2. MD 32540: FRICT_COMP_CONST_TIME [n] friction compensation time constant in [s]

These two parameters are changed until the circularity test produces minimum or no deviations from the programmed radius at the quadrant transitions (see the next 4 figures).
The tests must be performed with different radii and velocities (typical values for the application of the machine).

Start value

A relatively low injection amplitude plus a time constant of a few position controller cycles should be entered as the start value when measuring commences.

Example:

MD 32520: FRICT_COMP_CONST_MAX [n] = 10 (mm/min)
MD 32540: FRICT_ COMP_TIME [n] = 0.008 (8 ms)

The effect of changing the parameters must be checked using the measured and plotted circles.

Averaging

If it is not possible to determine a common compensation time constant for the varying radii and velocities, then the average of the calculated time constants must be worked out.

Good friction compensation setting

When the friction compensation function is well set, quadrant transitions are no longer noticeable (see figure below).

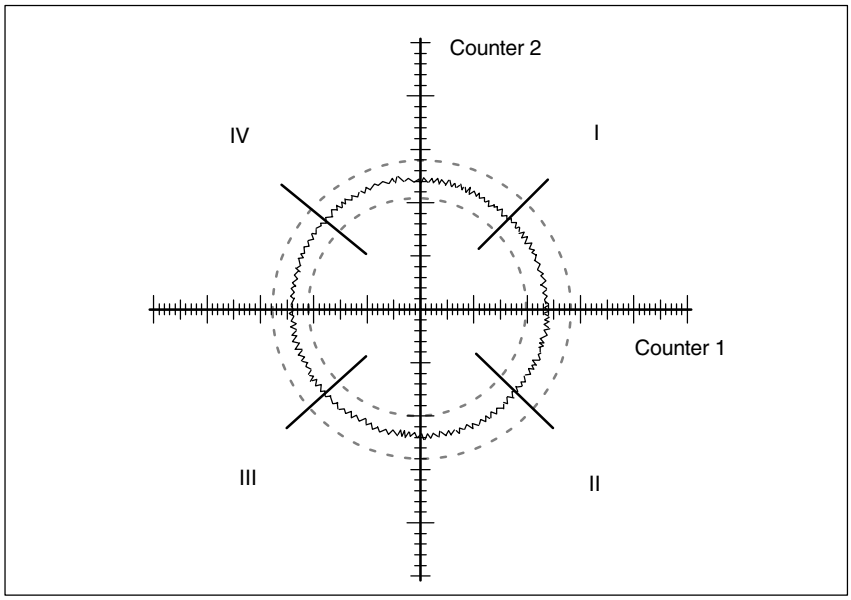


Fig. 3-11 Quadrant transitions with correctly set friction compensation

Amplitude too low

When the injection amplitude setting is too low, radius deviations from the programmed radius are compensated inadequately at quadrant transitions during circularity testing (see figure below).

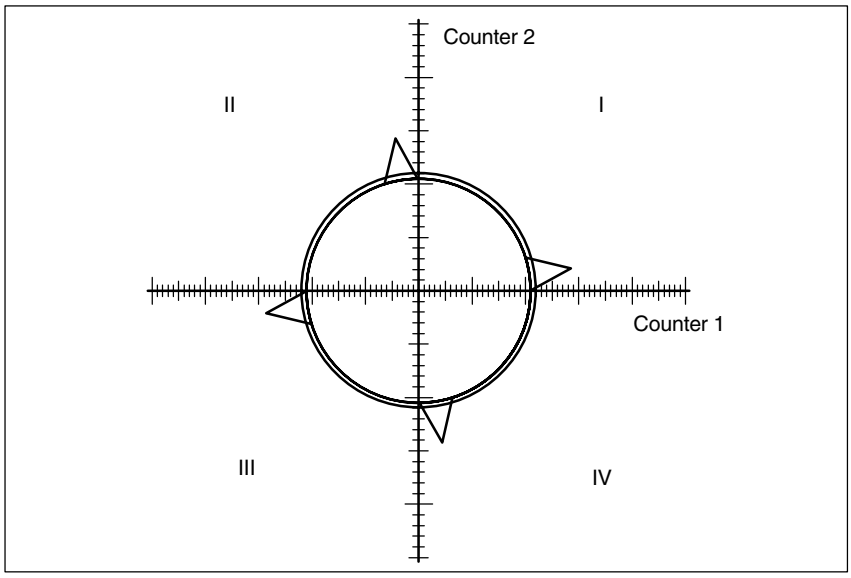


Fig. 3-12 Amplitude too low

Amplitude too high

When the injection amplitude setting is too high, radius deviations at quadrant transitions are manifestly overcompensated at quadrant transitions (see figure below).

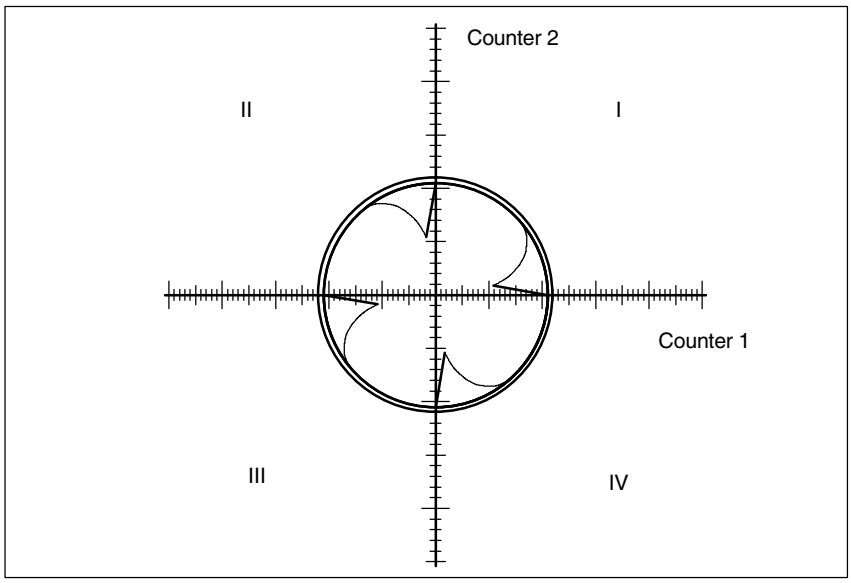


Fig. 3-13 Amplitude too high

Time constant too low

When the compensation time constant settings are too low, radius deviations are compensated briefly at quadrant transitions during circularity testing, but are followed immediately again by greater radius deviations from the programmed radius (see figure below).

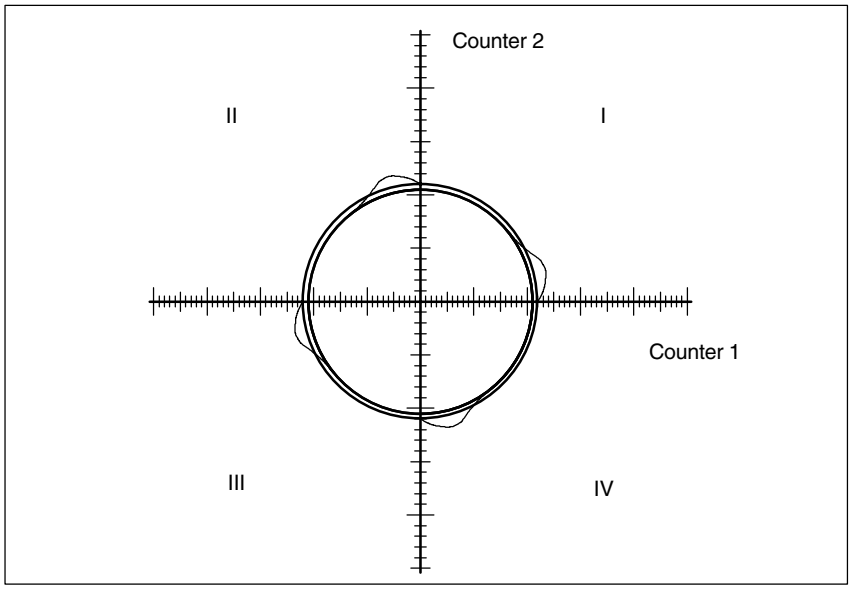


Fig. 3-14 Compensation time constant too small

Time constant too high

When the compensation time constant settings are too high, radius deviations are compensated at quadrant transitions during circularity testing (on condition that the optimum injection amplitude has already been calculated), but the deviation in the direction of the arc center increases significantly after quadrant transitions (see figure below).

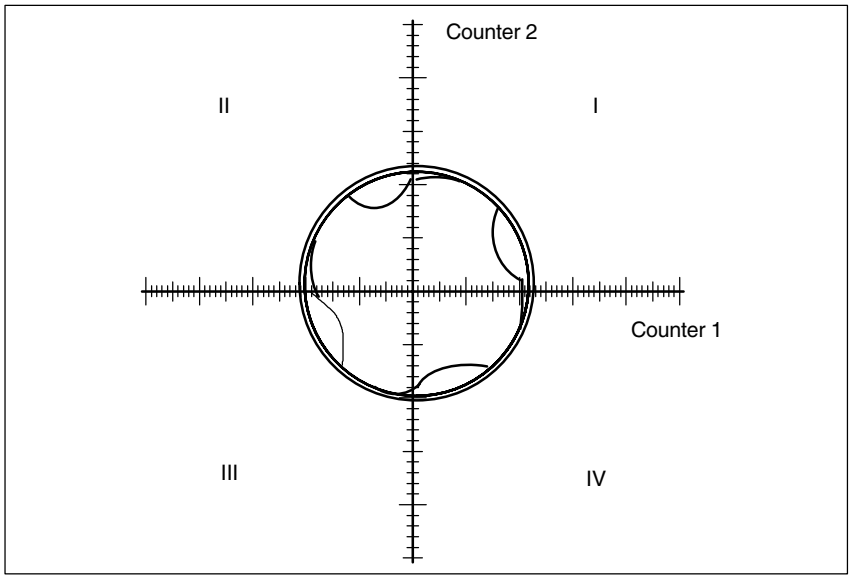


Fig. 3-15 Compensation time constant too large

Adaptation yes/no?

If, with the time constant and the constant injection amplitude determined, a good result is achieved both in the circularity test and in positioning over the whole working area (i.e. for all radii and velocities of relevance), curve adaptation will not be necessary.

However, if the friction compensation turns out to be dependent on the acceleration, the adaptation characteristic must be calculated in second stage (see stage 2: Friction compensation with adaptation).

Installation stage 2: Friction compensation with adaptation

Application

Whenever friction compensation depends on the acceleration and the required results cannot be obtained with constant injection amplitude, adaptation must be used.

In order to obtain optimum compensation over the whole of the working range of the friction feedforward control where high demands are made on accuracy, the acceleration dependency of the compensation value must be calculated. To achieve this, the dependency must be measured at various points in the working range between acceleration zero and the maximum planned acceleration. The adaptation characteristic derived from the measurement results is then entered in the above machine data axis-specifically.

1. Determining the adaptation characteristic

For different radii and velocities ...

1. ... it is necessary to determine the required injection amplitudes
2. ... it is necessary to check the compensatory effect of the injection amplitudes using the circularity test
3. ... it is necessary to log the optimum amplitudes.

The adaptation characteristic (for example, see Fig. 3.8) is defined completely by determining the parameters specified in Section 2.5.2. However, many more measured values must be obtained for checking purposes. It must be ensured that there is a sufficiently large number of interpolation points for small radii at high speed. The size of the curves must be obtained by plotting.

2. Determining acceleration values

During circular movement, the axial acceleration values are calculated using the radius r and the traversed velocity v according to the formula
$a = v^2/r$.
Using the feedrate override switch it is easy to vary the velocity and therefore axial acceleration value a.

Acceleration values a_1, a_2 and a_3 for the adaptation characteristic must be entered in MD 32550: FRICT_COMP_ACCEL1 to MD 32570: FRICT_COMP_ACCEL3 in compliance with the condition $a_1 < a_2 < a_3$. If the curve is wrongly parameterized, the alarm 26001 “Parameterization error for friction compensation” is output.

Example of characteristic settings

1. Calculation of acceleration
   The axial acceleration during the passage through zero of the speed for a circular path is calculated using the formula $a = v^2/r$.
   With the radius r = 10 mm and a circular velocity of v = 1 m/min (=16.7 mm/s) the acceleration is therefore $a = 27.8\ mm/s^2$.
2. Input of the curve knee points
   The following accelerations were calculated to be the curve knee points:
   $a_1 = 1.1\ mm/s^2$; $a_2 = 27.\ 8\ mm/s^2$; $a_3 = 695\ mm/s^2$

The following values are therefore entered in the machine data in this order:

MD 32550: FRICT_COMP_ACCEL1 [n] = 0.0011 [m/s^2]
MD 32560: FRICT_COMP_ACCEL2 [n] = 0.0278 [m/s^2]
MD 32570: FRICT_COMP_ACCEL3 [n] = 0.695 [m/s^2]

For example, the following values were calculated for the injection amplitudes:

MD 32520: FRICT_COMP_CONST_MAX [n] = 30 [mm/min]
MD 32530: FRICT_COMP_CONST_MIN [n] = 10 [mm/min]

Note

If the results obtained at very low velocities are not satisfactory, then the computational resolution for linear positions MD 10200: INT_INCR_ PER_MM or for angular positions MD 10210: INT_INCR_PER_DEG must be increased.

See also MD 32580: FRICT_COMP_INC_FACTOR
(weighting factor of friction compensation value for short traversing movements).

2.6 Neural quadrant error compensation

2.6.1 Fundamentals

Principle of QEC

As explained in Section 2.5, the purpose of quadrant error compensation (QEC) is to reduce contour errors occurring during reversal as the result of drift, backlash or torsion. Compensation is effected through prompt injection of an additional speed setpoint (see figure below).

In conventional QEC, the intensity of the compensation pulse can be set according to a characteristic as a function of the acceleration. This characteristic must be calculated and parameterized by a circularity test during start-up (see Fig. 3.8). The procedure for this is relatively complicated and requires some experience.

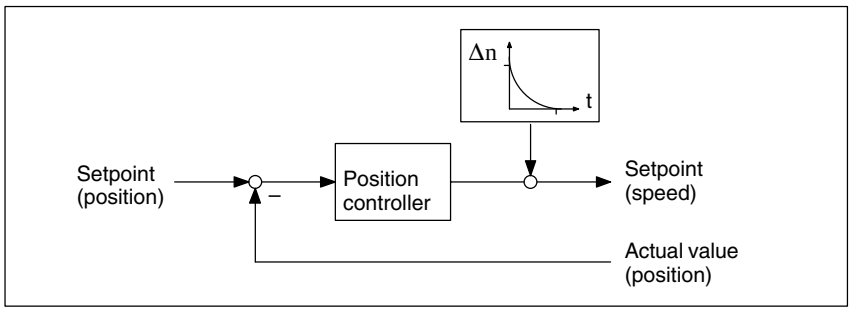


Fig. 3-16 Injection of an additional speed setpoint pulse

Advantages of QEC with neural network

On the SINUMERIK 840D with Software Version 2 or higher, the characteristic block that used to be manually parameterized can now be replaced by a neural network. This has the following advantages:

- Start-up has been simplified because the compensation characteristic no longer needs to be set manually by the start-up engineer but is **determined automatically** during a learning phase.
- The characteristic for a manually parameterized friction compensation is approximated by a polygon with 4 straight lines (see Fig. 3.8). For improved precision, the neural network can **reproduce the real curve much better**.

  The resolution of the characteristic curve can be adapted to the precision requirements and a directional quality of the compensation amplitude can be considered.

  In addition to the compensation amplitude, it is possible to adapt the decay time to the acceleration in special cases.
- The system permits **simple automatic re-optimization on site** at any time.

Requirements for neural QEC

An essential requirement for implementing QEC with neural network is that the errors occurring on the workpiece at quadrant transitions are detected by the measuring system. This is only possible either with a direct measuring system, with an indirect measuring system with clear reactions of the load on the motor (i.e. rigid mechanics, little backlash) or with suitable compensation. With indirect measuring systems, any backlash that might occur must be compensated by backlash compensation.

Learning/working phases

QEC with neural network involves the following two phases:

- **Learning phase**
  During the learning phase, a certain pattern of behavior is memorized in the neural network. The relation between the input and output signals is learnt. The result is the learnt compensation characteristic that is stored in the non-volatile user memory. Activation and deactivation of the learning process is programmed in the NC part program using special high-level language commands.
- **Working phase**
  During the working phase, additional speed setpoint pulses are injected in accordance with the learnt characteristic. The stored characteristic does not change during this phase.

The learning phase can be executed for several (up to 4) axes at the same time. For further information about training the neural network, see Section 2.6.3.

The learning and working phases and the resulting neural QEC are purely axial. There is no mutual influence between the axes.

Saving characteristic values

On completion of the learning phase, the calculated compensation data (characteristic values in user memory) including the network parameters (QEC system variables) must be saved in a file selected by the operator. As a standard, these files are called “AXn_QEC.INI”.

Loading characteristic values

These saved and learned compensation data can be loaded back directly to the user memory in the same way as part programs.

When the part program containing the tables is loaded, the compensation values are transferred to the NC user memory.
The characteristic values become effective only after compensation has been enabled.

Characteristic values **cannot** be written when the compensation function **is active** (MD 32500: FRICT_COMP_ENABLE must be set to 0 and activated).

With QEC: The QEC must be enabled (and activated) by setting MD 32500: FRICT_COMP_ENABLE = 1 (QEC active).

Recommended procedure for start-up

As mentioned above, the neural network integrated in the control automatically adapts the optimum compensation data during the learning phase.

The axis involved must perform reversals with acceleration values constant section by section. Before activation of the learning phase, the parameters of the neural network (QEC system variables) must be pre-assigned in accordance with the requirements.

In order to simplify start-up as much as possible, NC programs are provides as reference examples.

As described in Section 2.6.4, the start-up engineer must first learn the characteristic for the axes using these reference examples and the recommended QEC parameter values and check the contour accuracy achieved using the circularity test (see Section 2.7). If the results do not meet the requirements, re-optimization must be performed changing the parameters appropriately (see Sections 2.6.2, 2.6.3 and 2.6.5) (i.e. relearning).

2.6.2 Parameterization of neural QEC

Machine data

The basic configuring data for the neural QEC are stored as machine data.

- MD 32490: FRICT_COMP_MODE
  Method of friction compensation (2 = neural QEC)
- MD 32500: FRICT_COMP_ENABLE
  Friction compensation active
- MD 32580: FRICT_COMP_INC_FACTOR
  Weighting factor for friction compensation value for short traversing blocks
- MD 38010: MM_QEC_MAX_POINTS
  Maximum number of compensation values for QEC with neural networks

With these machine data, the neural QEC is activated as soon as the memory space is reserved in the non-volatile RAM. The procedure and assignment is described in Section 2.6.4 “Start-up” or in Chapter 4.

All other data are set using system variables.

QEC system variables

The data for parameterizing the neural network are defined as system variables that can be written and read by an NC program. The following system variables are used for parameterization of the neural network:

- **$AA_QEC_COARSE_STEPS** “Coarse quantization of the characteristic”

  This parameter defined the coarse quantization of the input signal and is therefore the resolution of the characteristic. The larger the value that is selected, the higher the memory requirement and the greater the length of time required for the training phase. See the end of this section for more information.

  Value range: 1 to 1024; recommended value: 49

- **$AA_QEC_FINE_STEPS** “Fine quantization of the characteristic”

  This parameter defines the fine quantization of the input signal and is therefore the resolution of the characteristic. The larger the value that is selected, the higher the memory requirement.

  Value range: 1 to 16; recommended value: 8

- **$AA_QEC_DIRECTIONAL** "Directionality"

  This parameter defines whether the compensation is to be injected directionally or not. If activated, a separate characteristic is determined and stored for each acceleration direction. Because two characteristics are used, double the memory space must be reserved in the non-volatile user memory.

  Value range: TRUE/FALSE; recommended value: FALSE

- **$AA_QEC_LEARNING_RATE** "Learning rate for the active learning phase"

  With the learning rate it is possible to determine how quickly the optimum characteristic is to be learnt in the active learning phase of the neural QEC. This value is a weighting factor with which it is possible to define to what extent the deviations affect the injection amplitude. With values greater than 100%, the characteristic is learnt more quickly but too high learning rate values (weighting factors) can cause instability (two-step response).

  A small learning rate is recommended for re-learning processes during normal operation (<50%) otherwise the characteristic is changed on every little disturbance when the speed passes through zero.

  Value range: > 0%; $\leq$ 500%; recommended value: 50%

- **$AA_QEC_ACCEL_1 / _2 / _3** "Acceleration limit values for the characteristic ranges 1 / 2 / 3"

  The acceleration characteristic is divided into three ranges. A different quantization of the acceleration stages applies to each range. In the low acceleration range, an especially high resolution is required for the characteristic in order to reproduce the widely varying compensation values there. For this reason, the input signals are quantized more finely, the smaller the acceleration is.

  Recommended values for

  – $AA_QEC_ACCEL_1 20 mm/s^2 (= 2% of $AA_QEC_ACCEL_3)
  – $AA_QEC_ACCEL_2 600 mm/s^2 (= 60% of $AA_QEC_ACCEL_3)
  – $AA_QEC_ACCEL_3 1000 mm/s^2 (maximum acceleration of the working range)

  The value of the parameter $AA_QEC_ACCEL_3 must be entered as appropriate to the requirements; i.e. the neural network only works and learns optimally in this range. If a higher acceleration is detected than the parameterized working area, the injection amplitude that was determined during the defined maximum acceleration of the working area is used. At high accelerations, this injection value is relatively constant.

  The recommended values must only be changed if the compensation is insufficient in these acceleration ranges. For further information see Section 2.6.5.

- **$AA_QEC_TIME_1** "Time constant for the neural QEC (decay time)"

  With this, the decay time of the compensation setpoint pulse is set if adaptation of the decay time is not used.

The optimum decay time must be ascertained manually using the circularity test at a working point in the mid acceleration range. The procedure is detailed in the section dealing with friction compensation (Section 2.5.3) (analogous to MD 32540: FRICT_COMP_TIME).

With the recommended value (15ms), it is possible to achieve good results.

Value range: $\geq$ 0; recommended value: 0.015s

If the decay time adaptation is active, then $AA_QEC_TIME_1 determines the filter time constant in the center of the operating range (i.e. with 0.5 * $AA_QEC_ACCEL_3).

- **$AA_QEC_TIME_2** "Compensation time constant for adaptation of the decay time of the correction values"

At a value of zero of less than or equal to $AA_QEC_TIME_1, no adaptation is performed.

The decay time is usually constant over the entire working range. In rare cases however, it can be advantageous to raise the decay time in the very small acceleration range, or to lower it at high accelerations. For further information see Section 2.6.5.

Value range: $\geq$ 0; recommended value: 0.015 s (identical to $AA_QEC_TIME_1)

- **$AA_QEC_MEAS_TIME_1 / _2 / _3**
  "Measurement duration for determining the error criterion in the acceleration range 1 / 2 / 3"

The measuring time is started, as soon as the criterion for injection of the compensation value is fulfilled (i.e. the set speed changes sign). The end of the measuring item is defined by the set parameter values.

Different measuring times are required for each characteristic range.

Recommended values for

– $AA_QEC_MEAS_TIME_1: 0.090 s (= 6 *$AA_QEC_TIME_1)

– $AA_QEC_MEAS_TIME_2: 0.045 s (= 3 * $AA_QEC_TIME_1)

– $AA_QEC_MEAS_TIME_3: 0.030 s (= 2 * $AA_QEC_TIME_1)

The recommended values must only be changed if the compensation is insufficient in these acceleration ranges or if $AA_QEC_TIME_1 is changed. For further information see Section 2.6.5.

Transfer of parameters

The QEC system variables are stored in the non-volatile user memory after the NC program is started where they remain unchanged until the memory is erased or re-formatted or until a new learning or relearning process takes place or until they are written by the NC program.

Before the learning cycle is called, all system variables must be assigned valid values for the learning process. For example, this can be done in a subprogram. After this NC program has run and a reset has been performed, the QEC data are active.

Characteristic data

The characteristic data determined during the learning process are stored as system variables in the user memory reserved for this purpose.

Format: **$AA_QEC**[**n**] Range of n: 0 to 1024

These values write the learned characteristic in internal formats and must therefore **not be changed**!

Quantization of characteristic

The quantization, and thus the resolution, of the characteristic is defined via the two quantities **fine quantization** ($AA_QEC_FINE_STEPS) and **coarse quantization** ($AA_QEC_COARSE_STEPS). The finer the resolution, the higher the memory requirement and the longer the duration of time required for the learning phase.

The number of memory locations required and the total number of quantization intervals is calculated by the formula:

Number of memory locations = $AA_QEC_FINE_STEPS * ($AA_QEC_COARSE_STEPS+1)

Up to 1025 memory locations per axis can be reserved. In this way, a sufficiently high resolution is achieved for high precision requirements.

The following 3 figures illustrate the meaning of the characteristic values for coarse and fine quantization, and their effect on the teach-in period, as a function of the parameter “Detailed learning active y/n”. Three cases are distinguished for better understanding.

Case 1: Coarse quantization > 1; fine quantization = 1
(special case; usually the fine quantization is in the region of eight):

In this case, the interpolation points of the characteristic are determined solely by coarse quantization (see figure below).

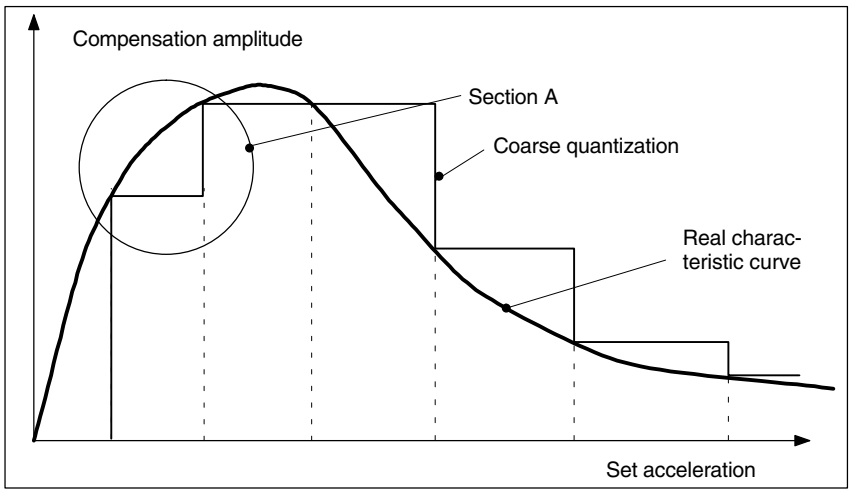


Fig. 3-17 Coarse quantization of characteristic

Case 2: Coarse quantization > 1; fine quantization > 1; “Detailed learning” is deactivated (this setting is the default);

In this case, discrete linear interpolation is used for fine quantization between the interpolation points of the coarse quantization.

The learning duration is identical with 1 because learning only occurs at the interpolation points of the coarse quantization.

The effect of fine quantization on a section of characteristic within a coarse quantization process is shown in the figure below (see also Section A in figure above).

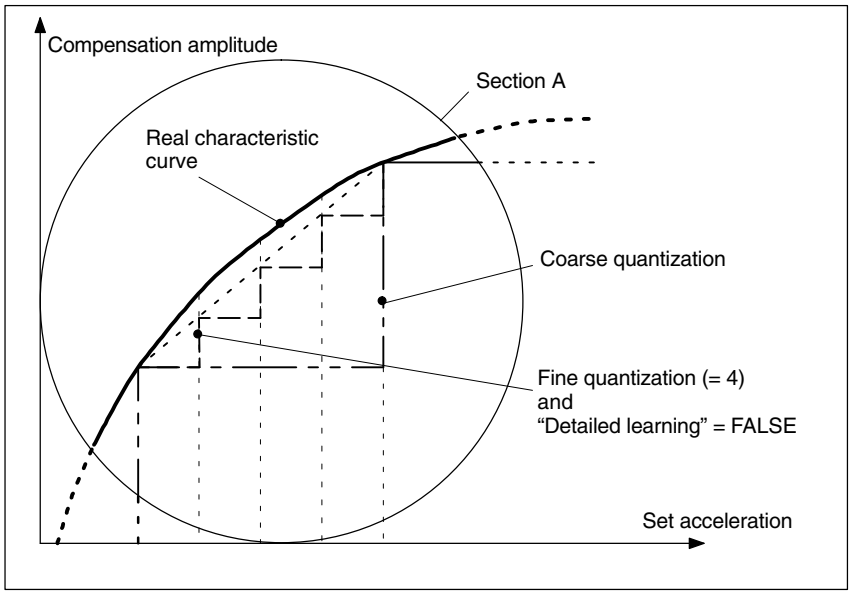


Fig. 3-18 Effect of fine quantization with “Detailed learning” inactive

Case 3: Coarse quantization > 1; fine quantization > 1; “Detailed learning” is activated (its use is only recommended for very high precision requirements):

With “Detailed learning”, learning occurs both at the interpolation points of the coarse quantization and of the fine quantization.

The learning duration is therefore much longer.

The figure below shows a severely fluctuating characteristic curve on which the effect of selecting and deselecting the “Detailed learning” function is clear.

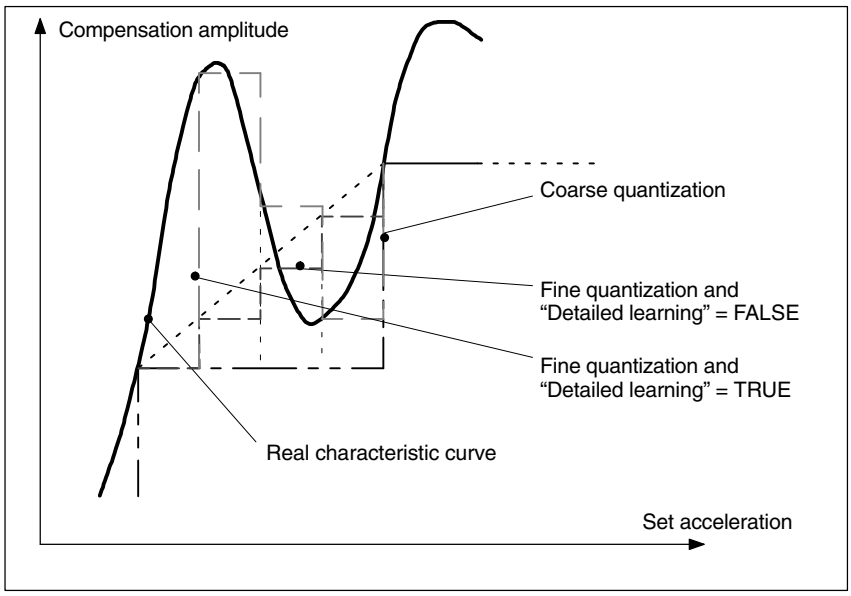


Fig. 3-19 Effect of fine quantization with “Detailed learning” active

2.6.3 Learning the neural network

Learning process sequence

A certain type of response is impressed upon the neural network during the learning phase. The relation between the input and output signals is learnt.

The learning process is controlled entirely by NC programs and is divided into the following areas:

1. Preset the QEC system variables for the learning process
2. Activate QEC system variables (by starting the NC program)
3. Parameterize the learning cycle
4. Start the learning cycle

The result is the learnt compensation characteristic that is stored in the non-volatile user memory.

The results achieved must be checked using the circularity test (Section 2.7).

Reference NC programs

In order to ease the task of the start-up engineer in starting up the QEC with neural networks, NC programs containing specimen routines for learning movements and assignments of QEC system variables (recommended values) are available.

These are the following reference NC programs:

- QECLRNP.SPF — Learning with POLY standards (Option “POLY” necessary)
- QECLRNC.SPF — Learning with circles
- QECDAT.MPF — Reference NC program for assigning system variables and for parameterization of the learning cycle
- QECSTART.MPF — Reference NC program that calls the learning cycle

These NC programs are contained on the diskette of the basic PLC program for the SINUMERIK 840 D.

Implementing the learning process solely via NC programs has the following advantages:

- Learning can be fully automatic without operator intervention.This is advantageous for series start-ups if the optimum learning parameters for a machine type have been found and the it only remains to determined or retrained the characteristic for each individual machine.
- Learning can be executed for several axes (up to 4) simultaneously.This reduces the learning phase for the machine considerably.
- The traverse movements can easily be adapted to special requirements.
- Start-up of the neural QEC is possible even where a simple MMC is used (e.g. MMC100) (exception: a circularity test on the MMC is only possible with MMC101–103; otherwise use an installation tool).

Learning motion

The axis traversing motions that must be executed to learn a specific response are generated by an NC program. Each learning motion of the sample learning cycle comprises a group of NC blocks with parabolic movements (ensuring that the axis traverses at the most constant possible setpoint speed after the zero crossing; see figure below) in which the axes oscillate at constant acceleration in each program section. The acceleration is decreased from group to group. In the figure below, NC blocks 2 to 3, 5 to 6 and 8 to 9 each form a group; the transitional movements to lower acceleration rates are programmed in blocks 1, 4, 7 and 10.

Note

So that the learning parameters act as preset, the feedrate override switch must be set to 100% during the learning phase.

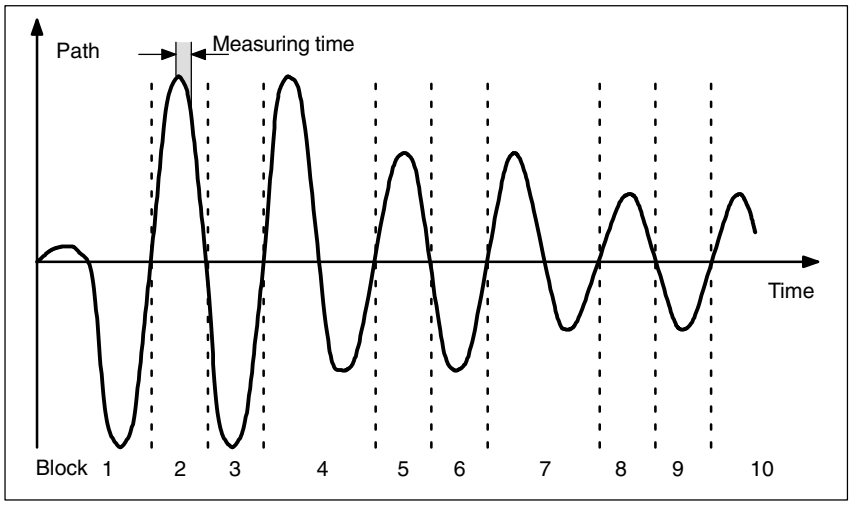


Fig. 3-20 Typical traverse motion of an axis when learning the QEC characteristic

Assignment of system variables

Before a learning cycle is called, all QEC system variables must be set to the values required for the learning process. The values recommended in the reference NC program must be checked and changed if necessary (see Section 2.6.2).

Learning ON / OFF

The actual learning process of the neural network is then activated in the reference NC program. This is done using the following high-level language command:

QECLRNON(axis name 1, ... 4) Activate learning (for the specified axes)

Only during this phase are the characteristics changed.

After the learning motions of the required axes have been completed, the learning process is deactivated for all axes. This is done with the high-level language command:

QECLRNOF Deactivate learning (simultaneously for all axes)

After power-on reset, end of program (M02/M30) or operator panel reset, the learning is also deactivated.

The current “Learning on / off” status is displayed in the service display “Axes” with “QEC learning active” (1 = active; 0 = inactive).

Learning cycle call

Once learning has been activated, the reference NC program calls the learning cycle by means of the following input parameters:

- **Number of axes** to which learning is to apply (up to four).

  Requirement: If more than one axis is to learn at the same time, all QEC system variables of the axes involved must have the same values. These values are monitored and an error message is output if they are not equal.

- **Names of the learning axes**
- **Initial number** (same for all axes)
  Value always 0 (setpoint branch)
- **Learning mode** (initial learning = 0; relearning = 1)

  0: Initial learning active. All values of the network are preset to 0 before learning.

  1: Relearning active: Learning continues with the values already learnt in the defined step width.

- **Detailed learning active** yes/no (TRUE/FALSE)

  FALSE: “Detailed learning” is not active. The characteristic is therefore learnt in the step width of the coarse quantization of the acceleration.

  TRUE: “Detailed learning” is active. The characteristic is therefore learnt in the step width of the fine quantization of the acceleration, i.e. with fine quantization of 10 steps per coarse step, determination of the characteristic takes ten times longer. This parameter must therefore only be used for extremely high precision requirements.

Note

If “Detailed learning” is selected, the number of learning passes can and must be reduced in order to reduce the learning duration (recommended range: between 1 and 5).

- **Number of learning passes**
  Default value = 15; range > 0

  The effect of this parameter depends on whether “Detailed learning active” is set or not.

  a) Detailed learning not active (= FALSE):
     The number of test motions (back and forth) is defined for each acceleration stage. The higher the number, the more accurate learning is, but the longer learning takes.

     With directional compensation ($AA_QEC_DIRECTION = TRUE), the parameterized number of test movements for every direction is generated.

  b) Detailed learning active (= TRUE):
     In this way, the number of complete passes from maximum to minimum acceleration and vice versa is activated with the fine step width.
     In other words, with a value of 1, all acceleration steps are passed through starting with the maximum value. For every acceleration stage, two test movements are generated if there is no directional compensation ($AA_QEC_DIRECTION = FALSE), otherwise four test movements are performed per acceleration stage.

  A reduction of the “Number of learning passes” can be made if data blocks for the machine type already exist (series machines) and these are to be used as a basis for further optimization.

- **Section-by-section learning active** yes/no (TRUE/FALSE)
  “Section-by-section learning” in certain acceleration ranges is especially interesting for “Detailed learning” e.g. in technologically important ranges of the machine. By defining the ranges appropriately it is possible to reduce the learning duration.

  Default = FALSE

- **Range boundaries for “Section-by-section learning”** (minimum acceleration, maximum acceleration); only relevant for “Section-by-section learning active”.

  Default value = 0; format: mm/s^2

- **Time taken for one test motion** (to and fro)

  Default value = 0.5; format: s (seconds)
  (corresponds to a frequency of 2 Hz)

Requirement

In the learning phase, the neural QEC requires a speed feedforward control (MD 32620: FFW_MODE=1; FFWON), but no jerk limitation (BRISK). The feedforward control must therefore be correctly parameterized and optimized. When the learning process is started a check is made to see whether the speed feedforward control is activated. If not, the learning process is cancelled and an error message is generated.

2.6.4 Start-up of neural QEC

General notes

This describes start-up of QEC with neural networks. As we have already mentioned, the compensation characteristics during the learning phase are determined automatically.

The axis involved must perform reversals with acceleration values constant section by section. The QEC system variables for parameterization of the neural network must also be preset to meet the requirements.

To simplify start-up as much as possible, NC programs are provided to serve as reference examples (see Section 2.6.3).

In the learning process, a distinction is made between “initial learning” (especially for first start-up) and “relearning” (especially for re-optimization of characteristics already learnt). The procedures of “initial learning” and “relearning” are described below.

If the compensation characteristics for the machine are to be learnt for the first time, we recommend use of the reference NC programs specified in Section 2.6.3.

“Initial learning” process

“Initial learning” –> cycle parameters “Learning mode” = 0

1. a) Activate QEC with neural networks for the required axes:

   MD 32490: FRICT_COMP_MODE = 2
   Note: QEC with neural networks is an option.

   b) Reserve memory space for the compensation points

   MD 38010: MM_QEC_MAX_POINTS
   If the required number is not yet known, a generous amount of memory must be reserved initially (see also item 12).

   c) Parameterize and optimize the speed feedforward control (required for the learning phase)

   d) Perform a hardware reset (because of the re-allocation of the non-volatile user memory).

2. Activate the QEC system variables:
   Adapt the reference NC program QECDAT.MPF for assigning the QEC system variables for all axes concerned (if necessary use the recommended values) and start the NC program. If error messages are output, correct the values and restart the NC program.

3. Create the NC program that moves the machine axes to the required positions and parameterizes and calls the reference learning cycle QECLRN.SPF (as in the example program QECSTART.MPF). The feedrate override switch must be set to 100% of the learning phase so that the parameters can take effect in accordance with the defaults.

4. Activate the learning phase by starting this NC program.
   The compensation characteristic is learnt for all the parameterized axes simultaneously. The learning duration depends on the specified learning parameters. If default values are used, it can take several minutes.
   The status of the axes concerned can be observed in the service display “axis” in the display “QEC learning active”.

5. Activation of the injection of the compensation values for the required axes: MD 32500: FRICT_COMP_ENABLE = 1.
6. Parameterize the trace for the circularity test in the menu “Circularity test measurement” (with MMC101–103 or installation tool).
   Parameter values for the reference NC program:
   Radius[mm]:
   Feedrate[mm/min].
   After this, enable the measuring function with the vertical softkey “start”.
7. Start the NC program with the test motion (circle).
   The position actual values are recorded during the circular movement and stored in the passive file system. After termination of data recording, the recorded contour is displayed as a diagram.
8. Check the quadrant transitions for the contour recorded.
9. Depending on the result, repeat items 2, 4, 7, 8 and 9 if necessary. It might be necessary to change certain QEC system variables first (see also Section 2.6.3).
10. The compensation characteristics must be saved as soon as the contour precision meets the requirements (see Section 2.6.3).
11. If necessary, the memory area previously reserved for the compensation values can be reduced to the memory actually required.

    Caution: When the setting in MD 38010: MM_QEC_MAX_POINTS is altered, the non-volatile user memory is automatically re-allocated on system power-on. All the user data in the non-volatile user memory are lost. These data must therefore be backed up first. After power-on of the control, the backed up characteristics must be loaded again.

“Relearning” process sequence

“Relearning” –> cycle parameter “Learning mode” = 1
The “Relearning” function allows characteristics that have already been learned to be re-optimized in a simple, automatic process. The values already in the user memory are taken as the basis.

The reference NC programs adapted to the machine (e.g. from “initial learning”) must be used in the learning phase for “relearning”. Generally, the previous values of the QEC system variables can still be used. Before the learning cycle is called, the parameter “learning mode” must be set to 1 (meaning “relearning”). It might also be used to reduce the “number of training passes”.

Sequence of operations for "Relearning"

The sequence of operations involved in the "Relearning" process is described below.

1. If characteristic values have not yet been stored in the user memory (RAM) (e.g. start-up of a series machine), the pre-optimized data block must be loaded (see Section 2.6.1).
2. Adapt the NC program that moves the machine axes to the required positions and parameterizes and calls the learning cycle. The parameters for the learning cycle (e.g. QECLRN.SPF) might have to be changed for "relearning".
   - Set "learning mode" = 1
   - Reduce the "number of learning passes" if necessary (e.g. to 5)
   - Activate "section-by-section learning" if necessary and define the associated range boundaries
3. Activate the learning phase by starting this NC program.
   The compensation characteristic is learnt for all the parameterized axes simultaneously.
4. Parameterize the trace for the circularity test in the menu "Circularity test measurement" (with MMC101–103 or installation tool). After this, enable the measuring function with the vertical softkey "start".
5. Start the NC program with the test motion for circularity test.
   The position actual values are recorded during the circular movement and stored in the passive file system. After termination of data recording, the recorded contour is displayed on the MMC.
6. Check the quadrant transitions for the contour recorded.
7. Depending on the result, repeat items 3, 4, 5 and 6 if necessary. It might be necessary to change certain QEC system variables first (see also Section 2.6.5).
8. The compensation characteristics must be saved as soon as the contour precision meets requirements (see Section 2.6.1).

2.6.5 Further optimization and intervention options

Optimization options

In cases where the results of the circularity test do not meet the required accuracy standards, the system can be further improved by selective changes to QEC system variables. Several ways of optimizing the neural QEC are explained here.

Alteration of coarse and fine quantization

As described in previous sections, input variables are quantized by means of the “Coarse quantization” and “Fine quantization” values.

A high value for the fine quantization causes a “similar” output signal to be obtained for adjacent intervals of the input signal, allowing, for example, measuring errors which occur only at a particular acceleration rate to be identified (see Fig. 3.16).

With a low fine quantization, highly fluctuating characteristics are reproduced better.

For the neural friction compensation, it is necessary to make use of the largest error tolerance by setting a high fine quantization ($AA_QEC_FINE_STEPS in the region of 5 to 10).

Direction-dependent compensation

Direction-dependent friction compensation must be used in cases where compensation is not applied equally on opposing quadrants when compensation values are being injected independently of direction (see figure below).

The directional injection is activated via the system variable $AA_QEC_DIRECTIONAL = TRUE.

Here, the following aspects must be observed:

- Since a characteristic is learned and stored for every direction of acceleration, double the memory space is required in the non-volatile user memory. MD 38010: MM_QEC_MAX_POINTS must be set accordingly.
- The number of learning passes must be raised because only every second passage occurs at the same location.
- If the characteristic resolution is the same, start-up takes longer.

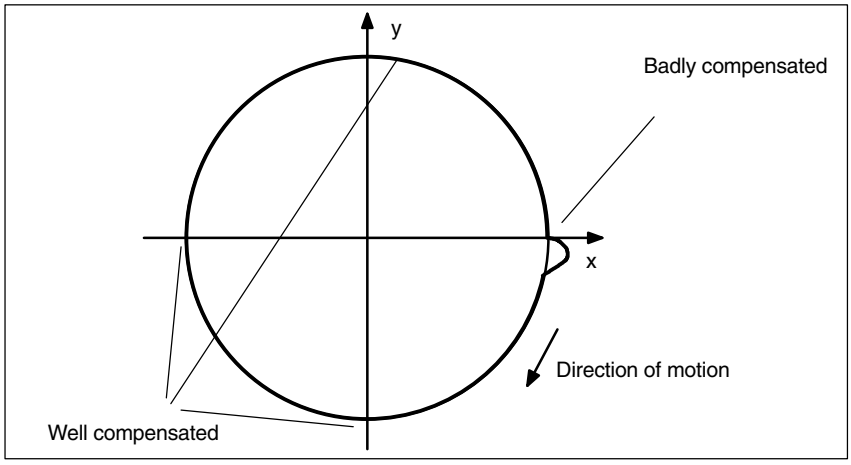


Fig. 3-21 Example of directional friction compensation (circularity test)

Modification of characteristic ranges

The acceleration characteristic is subdivided into three ranges. In the low acceleration range, an especially high resolution is required for the characteristic in order to reproduce the widely varying compensation values there. Therefore, the lower the acceleration rate, the finer the quantization of the input quantity (see figure below).

In the high acceleration range, there are only small changes in the compensation values so that a small resolution is perfectly sufficient.

The percentage settings recommended in Section 2.6.2 for $AA_QEC_ACCEL_1 (2% of $AA_QEC_ACCEL_3) and for $AA_QEC_ACCEL_2 (60% of $AA_QEC_ACCEL_3) are based on empirical values measured on machines with a maximum acceleration rate (= operating range) of up to approx. 1 m/s^2.

If the working range is significantly reduced, then the limit values for a_1 and a_2 must be set somewhat higher as a percentage of a_3. However, $AA_QEC_ACCEL_1 must not exceed the range of approx. 5% of the maximum acceleration. Useful boundaries for $AA_QEC_ACCEL_2 are approx. the values 40% to 75% of the maximum acceleration.

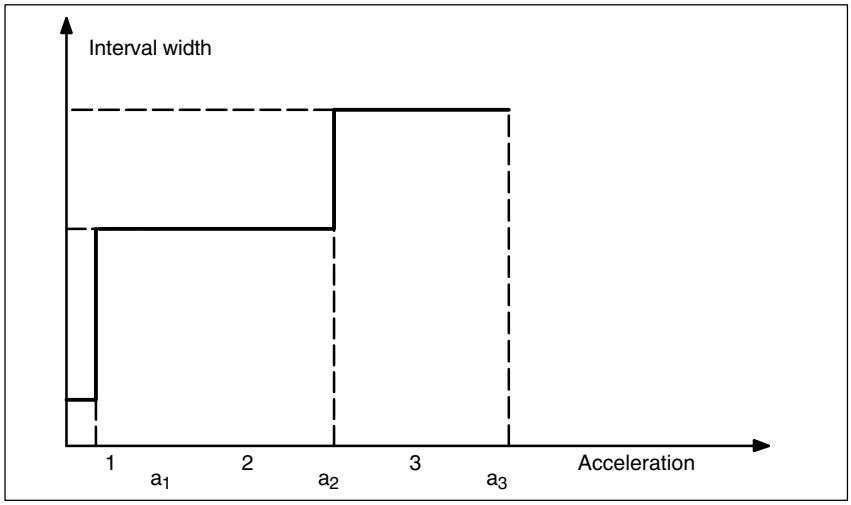


Fig. 3-22 Interval width in acceleration ranges

Adaptation of decay time

In special cases, it is possible to adapt the decay time of the compensation setpoint pulse in addition to the compensation amplitude.

If, for example, the circularity test reveals that in the low acceleration range (a_1) the quadrant transitions yield good compensation results but that radius deviations occur again immediately after this, it is possible to achieve an improvement by adapting the decay time.

The time constant without adaptation ($AA_QEC_TIME_1) is only valid in the mid acceleration range (50%).

The adaptation of the decay time for the compensation setpoint impulse according to the characteristic shown in the figure below is parameterized with system variable $AA_QEC_TIME_2 (for acceleration = 0). The adaptation is formed by these two points according to an e^{-x} function (see figure below).

The adaptation is performed under the following condition:
$AA_QEC_TIME_2 > $AA_QEC_TIME_1

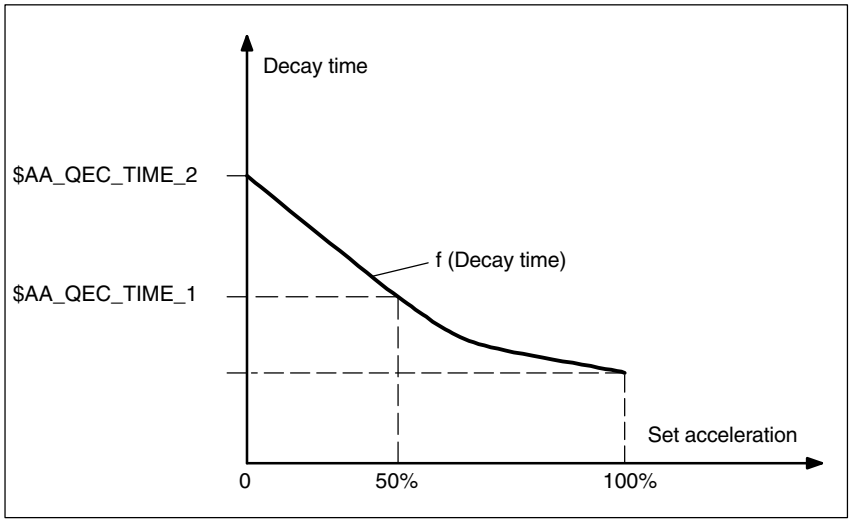


Fig. 3-23 Adaptation of the decay time

Alteration of error measuring time

During the learning phase for the neural network, the error measuring time determines the time window within which contour errors are monitored after a zero-speed passage.

Experience has shown that the error measuring time to be used for average acceleration rates (approx. 2 to 50mm/s^2) corresponds to three times the value of the decay time ($AA_QEC_MEAS_TIME_2 = 3 * $AA_QEC_ TIME_1).

In the very low and high acceleration ranges, the error measuring time must be adapted. This is done automatically according to the characteristic in the figure below. The error measurement duration for small accelerations is set to six times the value of the decay time ($AA_QEC_MEAS_TIME_1 = 6 * $AA_QEC_ TIME_1); double the decay time ($AA_QEC_MEAS_TIME_3 = 2 * $AA_QEC_ TIME_1) is taken as the error measurement time for larger accelerations.

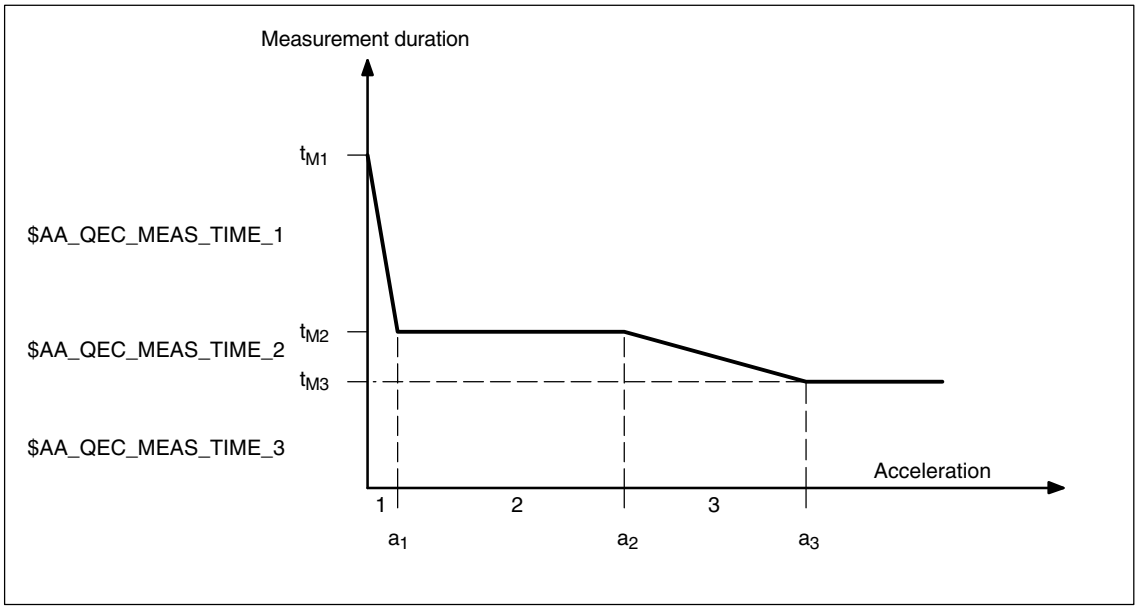


Fig. 3-24 Dependency of error measuring time on acceleration rate

In special cases, it might be necessary to reparameterize the error measuring times:

- Setting very extreme values for the compensation time constant of the QEC. Experience has shown that error measuring times of less than 10 ms and greater than 200 ms are not useful.
- Parameterization of the error measuring times with adaptation of the decay time of the compensation value.

  If the adaptation of the decay time of the compensation value is active (see above), the following rule of thumb is applicable to the parameterization of the error measuring time for acceleration range 1:

  $AA_QEC_MEAS_TIME_1 = 3 * $AA_QEC_ TIME_2

  Example:
  Decay time ($AA_QEC_TIME_1) = 10ms
  Adaptation of the decay time ($AA_QEC_TIME_2) = 30ms
  Following the rule of thumb given above, the error measuring time for acceleration range 1 is therefore:
  $AA_QEC_MEAS_TIME_1 = 3 * 30ms = 90ms
  Without adaptation of the decay time, it would only be
  $AA_QEC_MEAS_TIME_1 = 6 * 10ms = 60ms.

Overcompensation with short traversing motions

Practical experience has shown that the optimum friction compensation value calculated from the circularity test may result in overcompensation on the relevant axis if it executes very short axial positioning movements (e.g. on infeeds in the µm range).

To improve accuracy in such cases too, it is possible to reduce the compensation amplitude for short traversing motions.

This weighting factor programmed in MD 32580: FRICT_COMP_INC_FACTOR automatically takes effect when friction compensation is activated (conventional QEC or QEC with neural networks) acting on all positioning movements that are made within an interpolation cycle of the control.

The input range is between 0 and 100% of the calculated compensation value.

Control of learning process duration

As described in previous sections, the duration of the learning process is dependent on several parameters. It is mainly dependent on the following values:

- Coarse quantization ($AA_QEC_COARSE_STEPS)
- Measuring time for determining the error criterion ($AA_QEC_MEAS_TIME_1 up to $AA_QEC_MEAS_TIME_3)
- Number of learning passes
- Detailed learning active [yes/no]?
- Fine quantization ($AA_QEC_FINE_STEPS) (only if “detailed learning = yes” is selected)
- Directional compensation active [yes/no]? ($AA_QEC_DIRECTIONAL)
- Duration of reversing movement

The setting “Detailed learning active = yes” causes a significant increase in the time required for learning. It must therefore only be used where precision requirements are high. It is necessary to check whether these requirements only apply to certain acceleration ranges. If so, detailed learning only needs to be performed section by section (see “Section-by-section learning y/n?”). The number of learning passes must be reduced in any case.

If the reference NC programs mentioned above are used with the recommended parameter values, the following times have been determined for the learning process time:

- Detailed learning not active: approx. 6.5 min
- Detailed learning active: approx. 13 min

2.6.6 Quick start-up

Preparation for "Learning"

1. Determining the optimum friction compensation time constant (MD 32540 FRICT_COMP_TIME) with conventional friction compensation.
2. **Enter the following machine data without POWER ON:**

Machine data	Default	Change to	Meaning
MD 19330 NC-CODE_CONF_NAME_TAB[8]	0		Activate option "IPO_FUNKTION_MASK". **Only with learn program "Polynomial"! Bit 4 = 1**
MD 19300 COMP_MASK	0		Set option
MD 32490 FRIC_COMP_MODE	1	2	"Type of friction compensation" neural QEC
MD 32500 FRIC_COMP_ENABLE	0	0	"Friction compensation active" for learning **"OFF"**
MD 32580 FRIC_COMP_INC_FACTOR	0	0	"Weighting factor of friction compensation value for short traversing motions" (mm increments)
MD 38010 MM_QEC_MAX_POINTS	0	400	"Selection of values for QEC" = $AA_QEC_FINE_STEPS * ($AA_QEC_COARSE_STEPS + 1)
MD 32620 FFW_MODE	1	1	Speed feedforward control
MD 32610 VELO_FFW_WEIGHT	1	1	Injection 100%
MD 32630 FFW_ACTIVATION_MODE	1	0	Feedforward control ON continuously
MD 32810 EQUIV_SPEEDCTRL_TIME	0.004	Initial value t_pos + n_setSm.*	Adjust equivalent time constant n control loop

*) t_pos. ... Position control cycle (=basic system cycle * factor for position control cycle), n_setSm. ... speed setpoint smoothing (MD 1500 to 1521)

3. **Owing to re-allocation of memory (MD 38010), save the machine data with MMC 100:**
   "Services" "Data off" "Start-up data, NCK data" and if programmed, "LEC, measuring system error, sag and angularity error compensation tables" via PCIN.
   Execute a Power ON Reset and then read in the saved data with PCIN and "Data IN" (= series start-up).
   MMC 102:
   Save "Series startup" and if programmed, "LEC, measuring system error, sag and angularity error compensation tables".
   Execute a Power ON Reset and then read in the "Start-up" archive (saved data are loaded again).

4. **Copy the programs supplied on disk "MMC 100 TOOLBOX" into the NC (with archive!)**
   QECDAT.MPF
   QECSTART.MPF
   QECLRNP.SPF ("Polynomial" learning program) **or** QECLRNC.SPF ("Circle" learning program) is stored as QECLRN.SPF on the NC!
   With geometry axes, it is preferable to use the Circle learning program; for all other axes the Polynomial learning program only.

5. **Adapt the following programs:**
   – In the part program QECDAT
   if req. adapt friction compensation time constant (see Point 1)
   N1340 $AA_QEC_TIME_1[outNo,axNo] = 0.0xx
   N1040 def int numAxes = ... Enter the number of axes that are to learn.
   N1150 axisName[0] = ... Enter axis name of 1st axis.
   N1160 axisName[1] = ... Enter axis name of 1st axis.
   N1170 axisName[2] = ... Enter axis name of 1st axis.
   N1180 axisName[3] = ... Enter axis name of 1st axis.
   (AX1 .. AX8 or the machine or channel axis name can be used as an axis name for the "Circle" learning program. With "Polynomial" learning program, however, only the channel axis name)
   – In the part program QECSTART
   N1080 def int numAxes = ... Enter the number of axes that are to learn.
   N1310 axisName[0] = ... Enter axis name of 1st axis.
   N1320 axisName[1] = ... Enter axis name of 1st axis.
   N1330 axisName[2] = ... Enter axis name of 1st axis.
   N1340 axisName[3] = ... Enter axis name of 1st axis.
   (AX1 .. AX8 or the machine or channel axis name can be used as an axis name for the "Circle" learning program. In contrast, only the channel axis name may be used for the "Polynomial" learning program).

Executing "Learning" process

Start the following programs

- Select and start QECDAT.
  System variables are assigned.
- Select QECSTART, override 100% and start.
  The learning program runs for about 15 minutes involving approximately 30 cm traversing motions. If the message "REORG not possible" is displayed, it can be ignored. The message is displayed for about 10 seconds. It then disappears and the learning process continues with traversing motions.

Activate QEC

Machine data	Default	Change to	Meaning
MD 32500 FRIC_COMP_ENABLE	0	1	Switch on "Friction compensation active"

"Circularity test"

Use the "Circularity test" to check the result!

Save compensation data

Save the compensation data (QEC data are not saved with “SERIES START-UP”; can be selected in SW 4 and higher):
MMC 100:
Save with PCIN under SERVICES Data\Circle error compensation\All
MMC 102:
Save the file under SERVICES NCK \ NC Active Data \ Quadrant Error Co \ Quadrant error comp-complete.ini. This file contains all compensation values.
Note: Change the “displayed name length” to “20” in SERVICES “System settings” “for display” to ensure that the whole name is visible.

2.7 Circularity test

Function

One of the purposes of the circularity test is to check the contour accuracy obtained by the friction compensation function (conventional or neural QEC). It works by measuring the actual positions during a circular movement and displaying the deviations from the programmed radius as a diagram (especially at the quadrant transitions).

Procedure

The circular contour for the relevant axes is specified by an NC program. To simplify the circularity test as much as possible for the start-up engineer, an NC program is provided as a reference example for the circularity test motion (file QECTEST.MPF on the diskette with the basic PLC program). The start-up engineer must adapt this NC program to his application.

Several measurements must be made during the circularity test with different acceleration values to ascertain whether the learnt compensation characteristic (for neural QEC) or the defined compensation values (for conventional QEC) meet the requirements.

The circular movement can easily be made with different accelerations if you change the feedrate using the feedrate override switch without changing circular contour. The real feedrate must be taken into account in the measurement in the input field “feedrate”.

The circle radius chosen must be typical of machining operations on the machine (e.g. radius in the range 10 to 200 mm).

For the duration of the circular movement, the position actual values of the axes are recorded and stored in a “trace” in the passive file system. The circularity test is therefore purely a measuring function.

Parameterization of circularity test

The **axis names** or axis numbers with which the circle is to be traversed and for which the position actual values are to be recorded are selected in this menu. No check is made to ascertain whether the selected axes correspond to those programmed in the NC part program.

The parameter settings in the input fields “Radius” and “Feed” must correspond to the values from the part program that controls the circular motion of the axes, taking account of the feed override switch setting. No check is made to see whether the values in the part program (including feedrate override) and the input values match.

The “Measuring time” display field shows the measuring time calculated from the “Radius” and “Feed” values for recording the position actual values during the circular movement.

If only parts of the circle can be represented (i.e. measuring time too short) the measuring time can be increased in the menu by reducing the feed value. This also applies if the circularity test is started from the stationary condition.

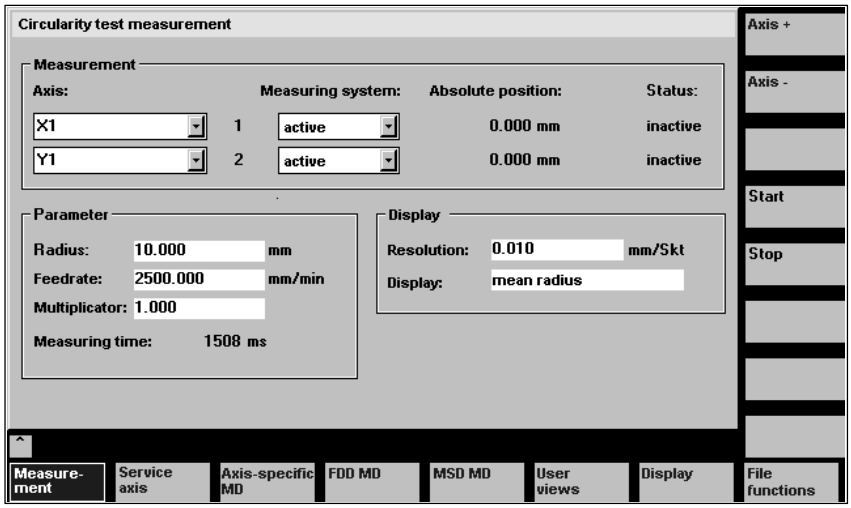


Fig. 3-25 Circularity test measurement menu

Mode of representation

The following parameter assignments for programming the mode of representation of measurement results can also be made:

- Display based on mean radius
- Display based on programmed radius
- Scaling of the diagram axes

If the measuring time calculated exceeds the time range that can be displayed from the trace buffers (maximum measuring time = position control cycle frequency * 2048), a coarser sampling rate is used for recording (n * position control cycle frequency), so that a complete circle can be displayed.

Start measurement

The operator must use an NC Start to start the part program in which the circular motion for the selected axes is stored (AUTOMATIC or MDA operating mode).

The measuring function is started with the vertical softkey **Start**.

The user can choose the operating sequence (NC start of the part program and starting measurement) to suit the application.

As soon as the circularity test is active for the specified axes, the message “Active” is output in the “Status” display field.

Stop measurement

The measurement can be interrupted at any time by means of softkey **Stop**. Any measurements that have not been fully recorded at that moment are displayed as completely as possible under softkey **Display**. There are no monitoring functions to check this.

For direct access to the required control parameters, the softkeys **Axis-specific MD, FDD-MD** and **MSD-MD** are provided. The required axis can be selected with the vertical softkeys **Axis+** and **Axis–**.

When you press the softkey **Service Axis** the “Service axis” display is shown. This cyclically displays the following service data for start-up of friction compensation:

- QEC learning active yes/no?
- Current position and speed actual values

Display

When the softkey **Display** is selected, the screen switches over to graphic representation of the recorded circle diagram.

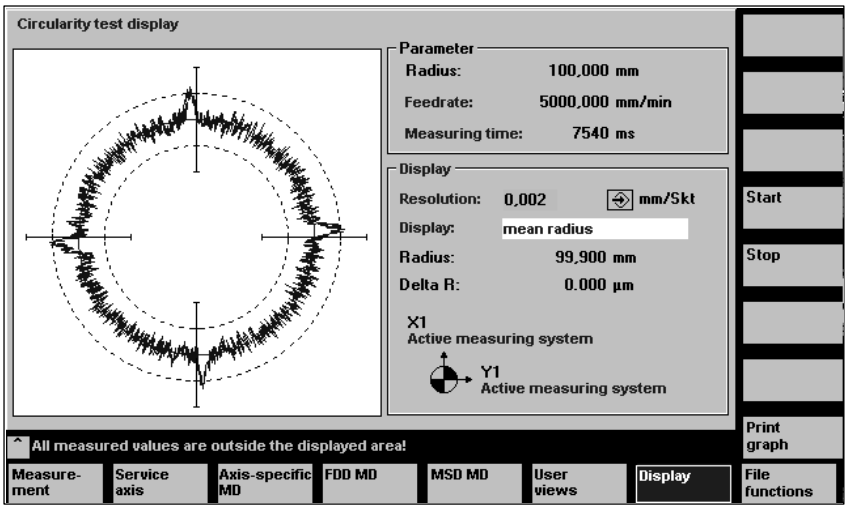


Fig. 3-26 Circularity test display menu

This display shows the measured progression of both position actual values as a circle with the set resolution.

The programmed radius, the programmed feedrate and the resulting measuring time are also displayed for documentation purposes (for subsequent storage of the measured circularity characteristic in a file).

The user can input a finer scaling for the diagram axes in the input field **Resolution**, to emphasize the quadrant transitions for example. To redisplay the diagram after having changed the resolution, press the softkey **Display**.

File functions

The displayed measurement results and the parameter settings can be stored as a file on the MMC by selection of softkey **File functions**.

Printer settings

The basic display for selecting a printer (Fig. 10-15) can be called by means of softkeys **MMC** \ **Printer selection**.

The toggle key is used to define whether the displayed graphic is to be output directly on the printer or transferred to a bit map file after softkey **Print graph** is selected.

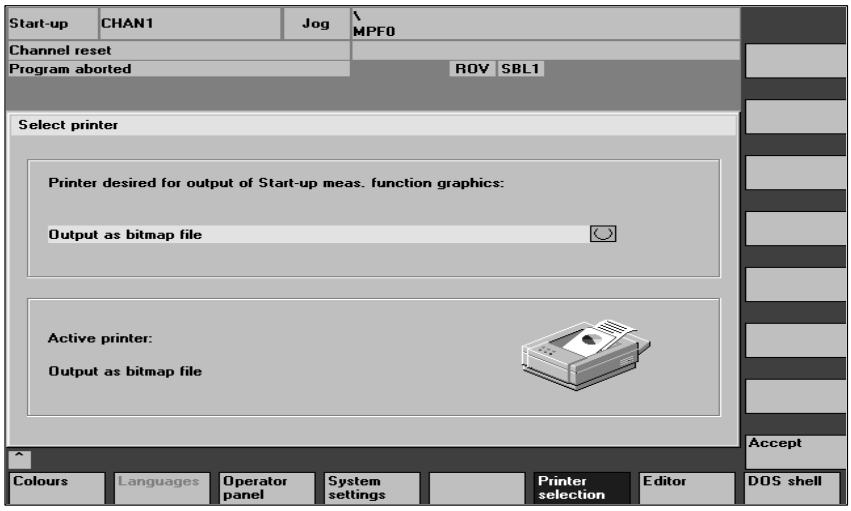


Fig. 3-27 Basic display for printer selection

Direct output on printer

The printer must be set up under MS-WINDOWS.

“Output on printer” is set in the selection field.
After selection of the softkey labeled **Print graph**, the displayed graph is output on the connected printer.

Output in bitmap file

The graphic is stored in a bitmap file (*.bmp).

“Output as bitmap file” is chosen in the printer settings selection field.
When the softkey labeled **Print graph** is selected, the screen form for assigning a file name appears in the “Circularity test display”. A new file name can be entered or an existing file name selected for overwriting in the drop-down list.

Softkey **Ok** is selected to store the file.
Softkey **Abort** is selected to return to the current graphic display.

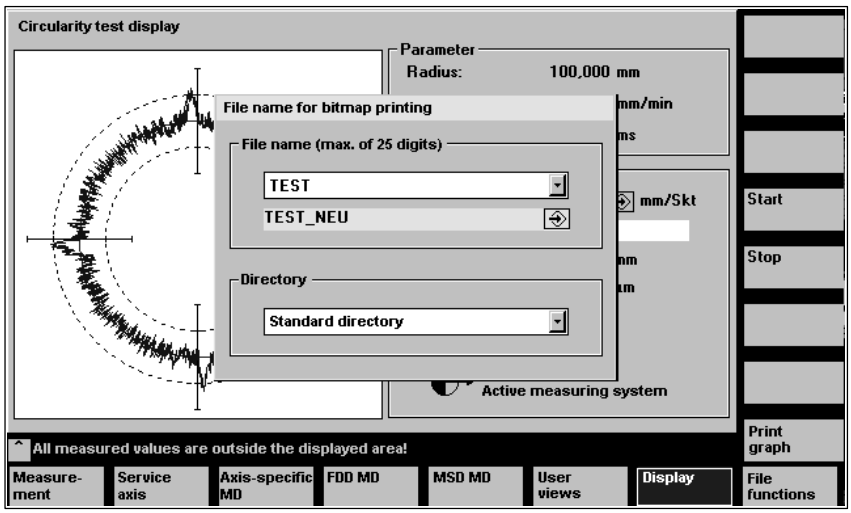


Fig. 3-28 Assignment of file name for output in a bit map file.

2.7.1 Neural quadrant error compensation, quick start-up

Quick start-up "Neural Quadrant Error Compensation" with parabolic/ circular movements on MMC 102/103/MMC 100

Preparation for "Learning"

The friction compensation time constant (MD 32540 FRICT_COMP_TIME) is calculated first by means of conventional friction compensation.

Table 2-2 Enter the following machine data **without** Power On

Machine data	De-fault	Change to	Meaning
MD 19330 IPO_FUNCTION_MASK	0	8	Activate "Polynomial interpolation" option. **For polynomial only! Bit 4 = 1**
MD 19300 COMP_MASK	0	8	Option "Neural QEC", bit 4 = 1
MD 32490 FRIC_COMP_MODE	1	2	"Type of friction compensation" neural QEC
MD 32500 FRIC_COMP_ENABLE	0	0	"Friction compensation active" for learning "OFF"
MD 32580 FRIC_COMP_INC_FACTOR	0	0	"Weighting factor of friction compensation value for short traverse motions" (µm increments)
MD 38010 MM_QEC_MAX_POINTS	0	400	"Selection of values for QEC" = $AA_QERC_FINE_STEPSA * ($AA_QEC_COARSE_STEPS + 1)
MD 32620 FFW_MODE	1	1	Speed feedforward control
MD 32610 VELO_FFW_WEIGHT	1	1	Injection 100%
MD 32630 FFW_ACTIVATION_MODE	1	0	Feedforward control ON continuously
MD 32810 EQUIV_SPEEDCTRL_TIME	0.004	t_pos.+ n_setSm. *)	Adjust equivalent time constant, n control loop

*) t_pos. ... Position control cycle (=basic system cycle * factor for position control cycle), n-setSm. ... speed setpoint filters (MD 1500 ... 1521)

Owing to re-allocation of memory (MD 38010), save the machine data with:

MMC 100: Save “Services”, “Data OUT”, “Start-up data, NCK data” and, if programmed, measuring system error and sag/angularity compensation tables via PCIN, execute a Power On-Reset and then read in the saved data with PCIN and “Data IN” (=series machine start-up).

MMC 102/103: Save “SERIES START-UP” and, if programmed, measuring system error and sag/angularity compensation tables, execute Power On-Reset and read in “Start-up” archive (saved data are reloaded).

Copy the programs supplied on diskette “MMC 100 TOOLBOX” into the NC (with archive!)

QECDAT.MPF
QECSTART.MPF
QECLERNP.SPF (“Polynomial” learning program) or QECLRNC.SPF (“Circle” learning program) is stored as QECLRN.SPF on the NC!

Adapt the following programs:

- **In part program QECDAT**

  N1040 def int numAxes=... Enter the number of axes to be learned

  N1150 axisName[0] Enter axis name of 1st axis.
  N1160 axisName[1] Enter axis name of 1st axis.
  N1170 axisName[2] Enter axis name of 1st axis.
  N1180 axisName[3] Enter axis name of 1st axis.
  (AX1 .. AX8 or the machine or channel axis name can be used as an axis name for the “Circle” learning program. In contrast, only the channel axis name may be used for the “Polynomial” learning program).

- **In part program QECSTART**

  N1080 def int numAxes=... Enter the number of axes to be learned

  N1310 axisName[0] Enter axis name of 1st axis.
  N1320 axisName[1] Enter axis name of 1st axis.
  N1330 axisName[2] Enter axis name of 1st axis.
  N1340 axisName[3] Enter axis name of 1st axis.
  (AX1 .. AX8 or the machine or channel axis name can be used as an axis name for the “Circle” learning program. In contrast, only the channel axis name may be used for the “Polynomial” learning program).

Execute LEARN process

Select and start QECDAT

System variables are assigned.

Select QECSTART: 100% override and start

The learning program runs for about 15 minutes involving approximately 30 cm traversing motions. The message “REORG not possible” can be ignored if it occurs. The message is displayed for about 10 seconds. It then disappears and the learning process continues with traversing motions.

Activate QEC

Machine data	Default	Change to	Meaning
MD 32500 FRIC_COMP_ENABLE	0	1	Switch on “Friction compensation active”

Use the “Circularity test” to check the result!

Save compensation data (QEC data are not included in back-up with “SERIES START-UP):

MMC 100: Save with PCIN under SERVICES\Data\Circle error compensation\All.

MMC 102/103: Save the file under SERVICES\NCK\NC Active Data\Quadrant Error Co\Quadrant error comp-complete.ini. This file contains all compensation values.

Note

Change the “displayed name length” to “20” in SERVICES “System settings” “for display” to ensure that the whole name is visible.

2.8 Drift compensation (for SINUMERIK FM-NC only)

Drift

As a result of the temperature-dependent drift in analog components, analog speed control loops must be driven by a low speed setpoint other than zero in order to reach standstill. The position controller can only generate this speed setpoint if a small following error builds up at its input even at zero velocity. The axis/spindle therefore slowly moves from its set position until the speed setpoint has become so large because of the following error that it is equivalent to the temperature drift.

Compensation

To avoid this static error, a small additional speed setpoint is injected. This comprises the following components (see figure below):

1. Basic drift value (MD 36720: DRIFT_VALUE)
   The value set in MD 36720: DRIFT_VALUE is always added to the speed setpoint. The basic drift value is always active. It is input as a percentage of the maximum manipulated variable.
2. Automatic drift compensation (MD 36700: DRIFT_ENABLE)
   With machine data MD 36700: DRIFT_ENABLE = 1 (automatic drift compensation) it is possible to active automatic drift compensation on **position-controlled axes**.
   The control determines the additional drift value still required to reduce the following error to zero (compensation criterion) while the axis is at zero velocity (interface signal “axis stop” is active).
   The total drift value is the sum of the basic drift value and the additional drift value.

   Automatic drift compensation for a position-controlled axis is performed under the following conditions:

   – The axis is at zero velocity

   – There is no traverse request for the axis

DRIFT_LIMIT

The additional drift value calculated during the automatic drift compensation process can be limited by MD 36710: DRIFT_LIMIT (drift limit value for automatic drift compensation). If the additional drift value exceeds the value entered in MD: DRIFT_LIMIT, alarm 25070 “Drift value too large” is signaled and the additional drift value is limited to this value. The limit is entered as a percentage of the maximum manipulated variable (100%).

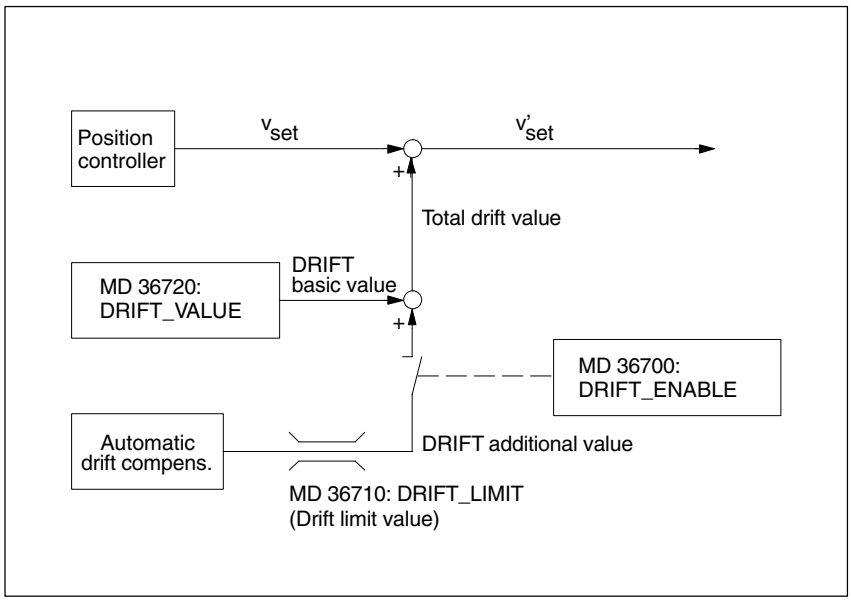


Fig. 3-29 Components of drift speed setpoint for position-controlled axes

Service display

The effect of the drift compensation can be monitored on the basis of the following error displayed in menu “Service display” in the **Diagnosis** operating area. If the axis or spindle is at zero velocity, the following error displayed should be zero.

Direct measuring system

The application of direct measuring systems combined with activation of “Automatic drift compensation” (MD 36700: DRIFT_ENABLE=1) causes the relevant axis to oscillate as a result of mechanical reversal errors.

The oscillation amplitude depends on the size of the backlash, the total servo gain and the concrete dynamic conditions (e.g. mass, vertical or horizontal axis).

2.9 Electronic weight compensation

Precondition

This function is available only for use in conjunction with SIMODRIVE 611D.

Note

The “Electronic weight compensation” functionality is not currently available for the combination SINUMERIK 840Di and SIMODRIVE 611 universal drive. The parameters required for the function cannot be transferred to the drive via the PROFIBUS–DP.

Response without-electronic weight compensation

The system responds as follows in the case of weight-bearing axes without programmed weight compensation:

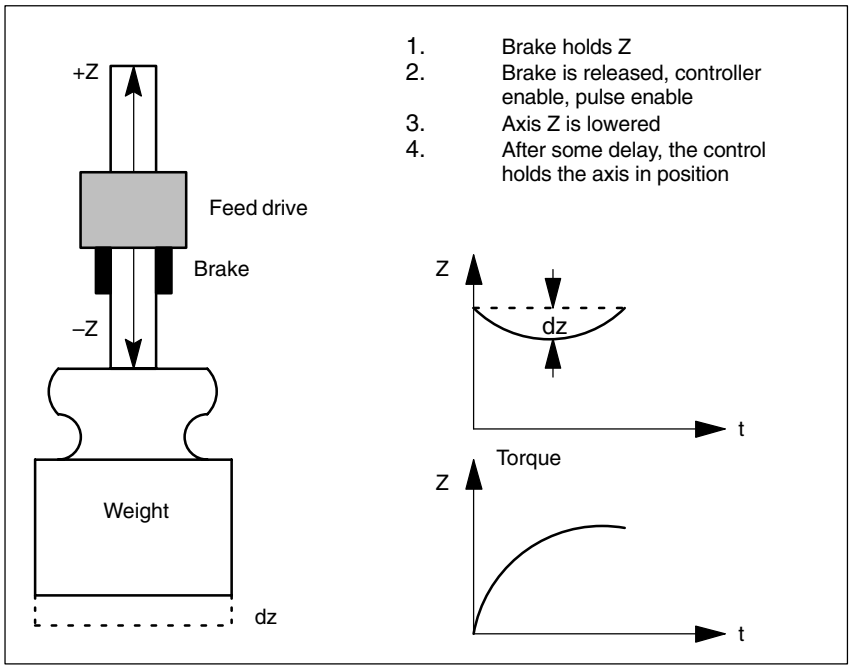


Fig. 3-30 Lowering of a vertical axis without electronic weight compensation

Lowering of the axis (Z) after release of the brake is not a desirable effect. The greater the reset time set with SIMODRIVE 611D-MD 1409: SPEEDCTRL_INTEGRATOR_TIME_1, the further the axis is lowered. Through activation of the electronic weight compensation function, it is possible to minimize the amount by which the axis is lowered.

Response with electronic weight compensation

The electronic weight compensation function prevents weight-bearing, vertical axes from sagging when the control is switched on. This switch-on process is illustrated in the figure below.

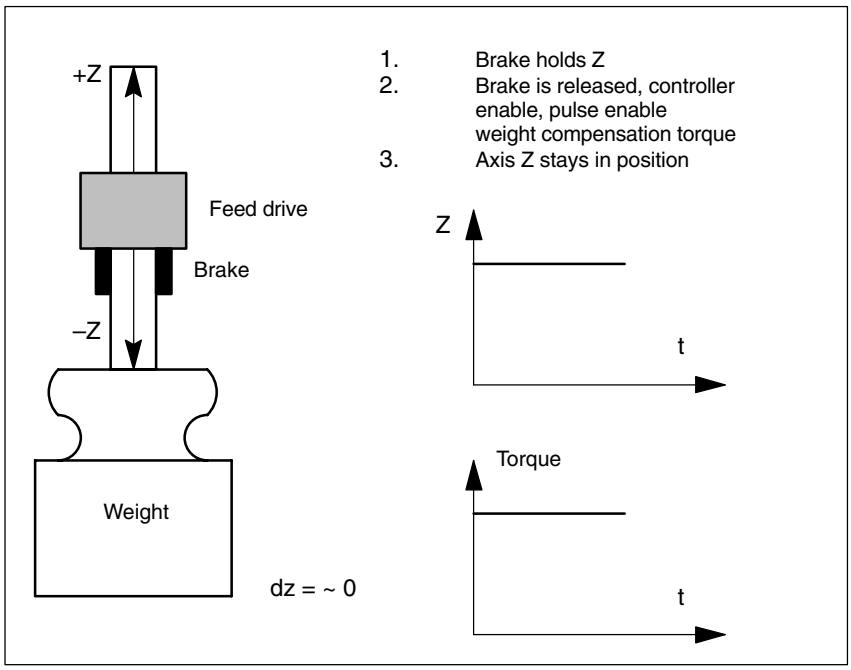


Fig. 3-31 Lowering of a vertical axis with electronic weight compensation

Note

This function is available only in conjunction with SIMODRIVE 611D.

Interaction with traverse against fixed stop

The “Electronic weight compensation” and “Traverse against fixed stop” functions may be used simultaneously, but the following special points should be noted in this respect:

The electronic weight compensation may **not** be used to offset the zero point for the fixed stop torque or fixed stop force as it is unsuitable for this purpose.

- If, for example, the axis requires 30% weight compensation in a case where 40% fixed stop torque is programmed in the same direction, then the actual torque with which the axis presses against the fixed stop only corresponds to 10% of rated torque.
- If 40% fixed stop torque is programmed in the other direction (in the opposite direction to weight compensation, i.e. in direction in which axis would drop) in the same situation described above, then the actual torque with which the axis presses against the fixed stop corresponds to 70% of rated torque.
- If the axis requires, for example, 30% weight compensation, then it is **not** possible to approach a fixed stop if less than 30% stop torque is programmed. This would limit the drive torque to such a degree that the control would not be able to keep the axis in position and it would drop.

These characteristics of the traverse against fixed stop function with vertical axes are determined by the available options for torque limitation in the drive. They are neither improved nor impaired by the weight compensation function.

Activation

The function is activated by setting axis-specific MD 32460: TORQUE_OFFSET to a value other than zero and becomes operative after the next RESET or POWER ON or selection of softkey “Set MD active”.

Deactivation

The function is deselected by setting axis-specific MD 32460: TORQUE_OFFSET to zero. The deselection takes effect after the next RESET or POWER ON or on selection of softkey “Set MD active”.

■

Notes

Supplementary Conditions 3

3.1 Availability

The individual compensation types are:

- Backlash compensation
- Leadscrew error and measuring system compensation
- Multi-dimensional beam sag compensation
- Manual quadrant error compensation
- Automatic quadrant error compensation (neural network)
- compensation
- Automatic drift compensation for analog speed setpoints
- Electronic weight compensation

"Backlash compensation" function

This function is available for

- SINUMERIK FM-NC with NCU 570, with SW 1 and higher
- SINUMERIK 840D with NCU 571/572/573, SW 1 and higher

"Leadscrew error and measuring system error compensation" function

This function is available for

- SINUMERIK FM-NC with NCU 570, with SW 1 and higher
- SINUMERIK 840D with NCU 571/572/573, SW 1 and higher

"Multi-dimensional beam sag compensation" function

This function is optional and available on

- SINUMERIK 840D with NCU 571/572/573, SW 2 and higher

The function is contained in the export version 840DE with restricted functionality; it is not contained in the FM-NC, 810DE (SW 3.1 and earlier).

- The function is available for the SINUMERIK 810DE in SW 3.2 and higher.

“Quadrant error compensation by operator input” function

This function is available for

- SINUMERIK FM-NC with NCU 570, with SW 1 and higher
- SINUMERIK 840D with NCU 571/572/573 with SW 1 and higher

“Automatic quadrant error compensation” function

This function is an option and available on

- SINUMERIK 840D with NCU 571/572/573, SW 2 and higher

“Temperature compensation” function

This function is an option and available on

- SINUMERIK FM-NC with NCU 570, with SW 1 and higher
- SINUMERIK 840D with NCU 571/572/573, SW 1 and higher

“Automatic drift compensation” function

This function is available for

- SINUMERIK FM-NC with NCU 570, with SW 1 and higher

“Electronic weight compensation” function

This function is available for

- SINUMERIK with NCU 571/572/573, SW 3 and higher, in conjunction with SIMODRIVE 611D.

■

Data Descriptions (MD, SD) 4

4.1 Description of machine data

4.1.1 General machine data

10082 MD number	**CTRLOUT_LEAD_TIME** Shift of setpoint transfer time		
Default value: 0.0	Min. input limit: 0.0	Max. input limit: 100.0	
Changes effective after Power On	Protection level: 2 / 7	Unit: %	
Data type: DOUBLE	Applies from SW version: 2		
Significance:	Lead time for output of the speed setpoints. The larger the value entered, the sooner the drive transfers the speed setpoints. • 0 % setpoints are transferred at the beginning of the next position control cycle. • 50 % setpoints are already transferred after execution of half of the position control cycle. A reasonable lead time can only be determined by measuring the maximum position control calculating time. MD 10083: CTRLOUT_LEAD_TIME_MAX suggests a value measured by the control. As this is a net value, it is advisable for the user to provide for a safety allowance of, for example, 5 %. If lead times that are too high are input, this can cause output of drive alarm 300506. The input value is rounded to the next speed controller pulse rate in the drive. If the speed controller pulse rates of the drives are different, changing the value will not necessarily lead to the same degree of controller improvement for all configured drives. **Note:** This MD is only important for axes with digital drives.		
Related to	MD 10083: CTRLOUT_LEAD_TIME_MAX		

10083 MD number	**CTRLOUT_LEAD_TIME_MAX** Maximum permissible setting for shift of setpoint transfer time		
Default value: 100.0	Min. input limit: 0.0	Max. input limit: 100.0	
Changes effective after NEW_CONF	Protection level: 2 / 7	Unit: %	
Data type: DOUBLE	Applies from SW version: 4		
Significance:	Maximum permissible lead time for output of the speed setpoints. MD 10083 represents a setting aid for MD 10082. The displayed value can be directly transferred to MD 10082, taking the safety allowance into account. The permissible lead time is determined from the maximum measured computing time required by the position controller. It decreases as the position controller's computing time requirements increase. By reducing the position control cycle via MD 10060 or 10050, you can reduce the permissible lead time. The lead time is measured during the entire operating life. The displayed value can only be increased by manual input. If the specified lead time is greater than the permissible one (e.g. 100 %), then it is automatically determined again. **Note:** This MD is only important for axes with digital drives.		
Related to	MD 10050: SYSCLOCK_CYCLE_TIME (basic system clock frequency) MD 10060: POSCTRL_SYSC LOCK_TIME_RATIO (factor for position control cycle) MD 10082: CTRLOUT_LEAD_TIME		

18342 MD number	**MM_CEC_MAX_ POINTS[t]** Maximum number of interpolation points for beam sag compensation [table t]		
Default value: 0	Min. input limit: 0	Max. input limit: 2000	
Changes effective after Power On	Protection level: 2 / 4	Unit: –	
Data type: DWORD	Applies from SW version: 2.1		
Significance:	For beam sag compensation, the number of required interpolation points must be defined for every compensation table [t]. with: [t] = Index of compensation table with ($0 \leq t \leq 2$ * maximum number of axes) i.e. t = 0: 1. compensation table t = 1: 2. compensation table etc. The necessary number can be calculated from the defined parameters as follows (see Section 2.3.3): $$\text{MM\_CEC\_MAX\_POINTS[t]} = \frac{\$\text{AN\_CEC\_MAX[t]} - \$\text{AN\_CEC\_MIN[t]}}{\$\text{AN\_CEC\_STEP[t]}} + 1$$ $AN_CEC_MIN [t] Initial position (system variable) $AN_CEC_MAX [t] End position (system variable) $AN_CEC_STEP [t] Distance between interpolation points (system variable) When selecting the number of interpolation points and the distance between them the resulting size of the compensation table and the resulting required memory capacity in the non-volatile user memory must be noted. 8 bytes are required for every compensation value (interpolation point). If the value 0 is entered, no memory is reserved for the table; i.e. the table does not exist and the function cannot therefore be activated.		
Special cases, errors, ...	**Caution!** When MD 18342: MM_CEC_MAX_POINTS[t] is changed the non-volatile NC user memory is automatically reallocated on power on. This deletes all the user data in the non-volatile user memory (e.g. drive and MMC machine data, tool offsets, part programs etc.).		
Related to	SD 41300: CEC_TABLE_ENABLE[t] Enable evaluation of beam sag compensation table [t]		
References	/FB/, S7, “Memory Configuration”		

4.1.2 Axis-specific machine data

32450 MD number	**BACKLASH[n]** Backlash		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective after NEW_CONF	Protection level: 2	Unit: mm or degrees	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	Backlash between the positive and the negative direction of travel. The compensation value input is • positive if the encoder leads the machine part (normal case) • negative if the encoder lags behind the machine part. If zero is entered backlash compensation is deactivated. Backlash compensation is always activated after reference point approach in all modes. The index [n] has the following coding: [encoder no.]: 0 or 1		
Special cases, errors, ...	If there is a second measuring system, a separate backlash value must be entered for this measuring system.		
Related to	MD: NUM_ENC (number of measuring systems) MD: ENC_CHANGE_TOL (maximum tolerance for position actual-value switchover)		

32452 MD number	**BACKLASH_FACTOR** Weighting factor for backlash		
Default value: 1.0	Min. input limit: 0.01	Max. input limit: 100.0	
Changes effective after NEW_CONF	Protection level: 2 / 7	Unit: –	
Data type: DOUBLE	Applies from SW version: 5		
Significance:	Weighting factor for backlash This machine data enables the backlash entered in MD 32450: BACKLASH to be changed as a function of a parameter set, e.g. in order to take account of gear-stage-specific backlash.		
Related to	MD 32450: BACKLASH[n]		

32460 MD number	**TORQUE_OFFSET** Additional torque for electronic weight compensation		
Default value: 0	Min. input limit: –100	Max. input limit: 100	
Changes effective after NEW_CONF	Protection level: 2 / 7	Unit: %	
Data type: DOUBLE	Applies from SW version: 3.1		
Significance:	The additional torque for the electronic weight compensation is entered in the % block of the static torque (calculated from MD1113 x MD1118). It is immediately effective when the current controller is activated. Vertical axes are thus prevented from sagging when the controller enabling signal is set, particularly when the speed controller reset time setting is high. 100% corresponds to the static torque of the axis drive. With the speed controller deactivated, a positive value would move the drive in a positive traversing direction (see also MD 32100: AX_MOTION_DIR for further details). If, therefore, the positive traversing direction is upwards (axis is raised), then a positive value must be entered for the weight compensation.Conversely, a positive traversing direction downwards would call for a negative value. MD is effective only for SIMODRIVE 611D drive systems.		
Special cases, errors, ...	See Interaction with “Traverse against fixed stop” function		
Related to			

32490 MD number	**FRICT_COMP_MODE** Friction compensation mode		
Default value: 0	Min. input limit: 0	Max. input limit: 2	
Changes effective after NEW_CONF	Protection level: 2/4	Unit: –	
Data type: BOOLEAN	Applies from SW version: 2.1		
Significance:	0: No friction compensation 1: Friction compensation with const. feedforward value or adaptive characteristic 2: Friction compensation with learnt characteristic via neural network		
Related to			

32500 MD number	**FRICT_COMP_ENABLE** Friction compensation active	
Default value: 0	Min. input limit: 0	Max. input limit: 1
Changes effective after NEW_CONF	Protection level: 2 / 4	Unit: –
Data type: BOOLEAN	Applies from SW version: 1.1	
Significance:	1: The axis is enabled for “friction compensation” and therefore injection of the friction compensation values. Quadrant errors on circular contours can be compensated with “friction compensation”. Axial machine data MD 32490: FRICT_COMP_MODE “friction compensation type” defines whether “friction compensation with constant injected value” or “quadrant error compensation with neural networks” is selected. In the case of neural networks, the machine data should first be set to “1” when a valid characteristic has been “learnt”. During the learning phase, the compensation values are injected independently of the contents of this machine data. 0: “Friction compensation” is not enabled for this axis. No friction compensation values are injected.	
Related to	MD 32490: FRICT_COMP_MODE MD 32510: FRICT_COMP_ADAPT_ENABLE MD 32520: FRICT_COMP_CONST_MAX MD 32540: FRICT_COMP_TIME MD 38010: MM_QEC_MAX_POINTS	Friction compensation type Friction compensation adaptation active Maximum friction compensation value Friction compensation time constant Number of interpolation points for quadrant error compensation with neural networks

32510 MD number	**FRICT_COMP_ADAPT_ENABLE [n]** Friction compensation adaptation active [setpoint branch]: 0		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEW_CONF	Protection level: 2	Unit: –	
Data type: BOOLEAN	Applies from SW version:		
Significance:	1: Friction compensation with amplitude adaptation is enabled for the axis.With friction compensation quadrant errors on circular contours can be compensated. Often, the injection amplitude of the friction compensation value is not constant over the entire acceleration range. In this case, for high accelerations a smaller compensation value must be injected than for small accelerations to achieve optimum friction compensation. The parameters of the adaptation curve (see Fig. 2-14) must be determined and entered in the machine data. 0: Friction compensation with amplitude adaptation must be enabled for the axis.		
MD irrelevant for	MD 32500: FRICT_COMP_ENABLE = 0 MD 32490: FRICT_COMP_MODE = 2 (neural QEC)		
Related to	MD 32500: FRICT_COMP_ENABLE MD 32520: FRICT_COMP_CONST_MAX MD 32530: FRICT_COMP_CONST_MIN MD 32550: FRICT_COMP_ACCEL1 MD 32560: FRICT_COMP_ACCEL2 MD 32570: FRICT_COMP_ACCEL3 MD 32540: FRICT_COMP_TIME	Friction compensation active Maximum friction compensation value Minimum friction compensation value Adaptation acceleration value 1 Adaptation acceleration value 2 Adaptation acceleration value 3 Friction compensation time constant	

32520 MD number	**FRICT_COMP_CONST_MAX [n]** Maximum friction compensation value [setpoint branch]: 0		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective after NEW_CONF	Protection level: 2	Unit: mm/min	
Data type: DOUBLE	Applies from SW version:		
Significance:	With machine data MD 32520: FRICT_COMP_CONST_MAX the magnitude of the (maximum) injection amplitude of the friction compensation value is defined. This value is injected over the entire acceleration range for friction compensation without adaptation. In the case of friction compensation with adaption, this value is merely applied in the acceleration range B2 of the adaptation characteristic (see Chapter 2, subsection “Conv. friction compensation”).		
MD irrelevant for	MD 32500: FRICT_COMP_ENABLE = 0 MD 32490: FRICT_COMP_MODE = 2 (neural QEC)		
Related to	MD 32500: FRICT_COMP_ENABLE MD 32510: FRICT_COMP_ADAPT_ENABLE MD 32530: FRICT_COMP_CONST_MIN MD 32550: FRICT_COMP_ACCEL1 MD 32560: FRICT_COMP_ACCEL2 MD 32570: FRICT_COMP_ACCEL3 MD 32540: FRICT_COMP_TIME	Friction compensation active Friction compensation adaptation active Minimum friction compensation value Adaptation acceleration value 1 Adaptation acceleration value 2 Adaptation acceleration value 3 Friction compensation time constant	

32530 MD number	**FRICT_COMP_CONST_MIN [n]** Minimum friction compensation value [setpoint branch]: 0		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective after NEW_CONF	Protection level: 2	Unit: mm/min	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	The minimum friction compensation value is needed only if “Friction compensation with adaptation” is active (MD 32510: FRICT_COMP_ADAPT_ENABLE = 1). The friction compensation amplitude entered in FRICT_COMP_CONST_MIN is applied in the acceleration range B4 ($a \geq a_3$) of the adaptation characteristic (see Section 2, subsection “Conv. friction compensation”).		
MD irrelevant for	MD 32510: FRICT_COMP_ADAPT_ENABLE = 0 MD 32490: FRICT_COMP_MODE = 2 (neural QEC)		
Special cases, ...	In exceptional cases, the value programmed for FRICT_COMP_CONST_MIN may even be higher than the setting for MD 32520: FRICT_COMP_CONST_MAX.		
Related to	MD 32500: FRICT_COMP_ENABLE MD 32510: FRICT_COMP_ADAPT_ENABLE MD 32520: FRICT_COMP_CONST_MAX MD 32550: FRICT_COMP_ACCEL1 MD 32560: FRICT_COMP_ACCEL2 MD 32570: FRICT_COMP_ACCEL3 MD 32540: FRICT_COMP_TIME	Friction compensation active Friction compensation adaptation active Maximum friction compensation value Adaptation acceleration value 1 Adaptation acceleration value 2 Adaptation acceleration value 3 Friction compensation time constant	

32540 MD number	**FRICT_COMP_TIME [n]** Friction compensation time constant [setpoint branch]: 0		
Default value: 0,015	Min. input limit: 0	Max. input limit: plus	
Changes effective after NEW_CONF	Protection level: 2	Unit: s	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	Time constant over which the friction compensation value is injected (decay time of the compensation setpoint pulse). Deviations at the quadrant transitions are not only influenced by the injection amplitude but also by a change in the friction compensation time constant (see Section 2.5.3)		
MD irrelevant for	MD 32500: FRICT_COMP_ENABLE = 0		
Related to	MD 32500: FRICT_COMP_ENABLE MD 32520: FRICT_COMP_CONST_MAX	Friction compensation active Maximum friction compensation value	

32550 MD number	**FRICT_COMP_ACCEL1 [n]** Adaptation acceleration value 1 [setpoint branch]: 0		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective after NEW_CONF	Protection level: 2	Unit: m/s^2	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	The adaptation acceleration value is only required if “Friction compensation with adaptation” is active. The adaptation acceleration values 1 to 3 are interpolation points for defining the adaptation curve. The adaptation curve is subdivided into four ranges in which different friction compensation values apply. Range B1 is defined by FRICT_COMP_ACCEL1 (a_1) (see Sect. 2, subsection “Conv. friction compensation”). For the injection amplitude within range B1 the following applies: $\Delta n = \Delta n_{max} * a / a_1$ for $a < a_1$		
MD irrelevant for	MD 32510: FRICT_COMP_ADAPT_ENABLE = 0 MD 32490: FRICT_COMP_MODE = 2 (neural QEC)		
Related to	MD 32500: FRICT_COMP_ENABLE	Friction compensation active	
	MD 32510: FRICT_COMP_ADAPT_ENABLE	Friction compensation adaptation active	
	MD 32520: FRICT_COMP_CONST_MAX	Maximum friction compensation value	
	MD 32530: FRICT_COMP_CONST_MIN	Minimum friction compensation value	
	MD 32560: FRICT_COMP_ACCEL2	Adaptation acceleration value 2	
	MD 32570: FRICT_COMP_ACCEL3	Adaptation acceleration value 3	
	MD 32540: FRICT_COMP_TIME	Friction compensation time constant	

32560 MD number	**FRICT_COMP_ACCEL2 [n]** Adaptation acceleration value 2 [setpoint branch]: 0		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective after NEW_CONF	Protection level: 2	Unit: m/s^2	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	The adaptation acceleration value is only required if “Friction compensation with adaptation” is active. The adaptation acceleration values 1 to 3 are interpolation points for defining the adaptation curve. The adaptation curve is subdivided into four ranges in which different friction compensation values apply. Range B2 is defined by MD 32550: FRICT_COMP_ACCEL1 (a_1) and FRICT_COMP_ACCEL2 (a_2) (see Sect. 2, subsection “Conv. friction compensation”). For the injection amplitude within range B2 the following applies: $\Delta n = \Delta n_{max}$ for $a_1 \leq a \leq a_2$		
MD irrelevant for	MD 32510: FRICT_COMP_ADAPT_ENABLE = 0 MD 32490: FRICT_COMP_MODE = 2 (neural QEC)		
Related to	MD 32500: FRICT_COMP_ENABLE	Friction compensation active	
	MD 32510: FRICT_COMP_ADAPT_ENABLE	Friction compensation adaptation active	
	MD 32520: FRICT_COMP_CONST_MAX	Maximum friction compensation value	
	MD 32530: FRICT_COMP_CONST_MIN	Minimum friction compensation value	
	MD 32550: FRICT_COMP_ACCEL1	Adaptation acceleration value 2	
	MD 32570: FRICT_COMP_ACCEL3	Adaptation acceleration value 3	
	MD 32540: FRICT_COMP_TIME	Friction compensation time constant	

32570 MD number	**FRICT_COMP_ACCEL3 [n]** Adaptation acceleration value 3 [setpoint branch]: 0		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective after NEW_CONF	Protection level: 2	Unit: m/s^2	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	The adaptation acceleration value is only required if “Friction compensation with adaptation” is active. The adaptation acceleration values 1 to 3 are interpolation points for defining the adaptation curve. The adaptation curve is subdivided into four ranges in which different friction compensation values apply. Range B3 is defined by MD 32560: FRICT_COMP_ACCEL2 (a_2) and FRICT_COMP_ACCEL3 (a_3) (see Sect. 2, subsection “Conv. friction compensation”). For the injection amplitude within range B3 the following applies: $\Delta n = \Delta n_{max} * (1 - (a - a_2) / (a_3 - a_2))$ for $a_2 < a < a_3$ Range B4 applies to acceleration values > a3. The following applies to the injection amplitude within range B3: $\Delta n = \Delta n_{min}$ for $a \geq a_3$		
MD irrelevant for	MD 32510: FRICT_COMP_ADAPT_ENABLE = 0 MD 32490: FRICT_COMP_MODE = 2 (neural QEC)		
Related to	MD 32500: FRICT_COMP_ENABLE — Friction compensation active MD 32510: FRICT_COMP_ADAPT_ENABLE — Friction compensation adaptation active MD 32520: FRICT_COMP_CONST_MAX — Maximum friction compensation value MD 32530: FRICT_COMP_CONST_MIN — Minimum friction compensation value MD 32550: FRICT_COMP_ACCEL1 — Adaptation acceleration value 2 MD 32560: FRICT_COMP_ACCEL2 — Adaptation acceleration value 3 MD 32540: FRICT_COMP_TIME — Friction compensation time constant		

32580 MD number	**FRICT_COMP_INC_FACTOR** Weighting factor of friction compensation value with short traversing movements		
Default value: 1	Min. input limit: 0	Max. input limit: 100	
Changes effective after Power On	Protection level: 2 / 4	Unit: %	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	The optimum friction compensation value determined by the circularity test can cause overcompensation of this axis if compensation is activated and axial positioning movements are short. In such cases, a better setting can be achieved by reducing the amplitude of the friction compensation value (conventional or quadrant error compensation with neural networks) and all positioning movements that are made within an interpolation cycle of the control. The factor that has to be entered can be determined empirically and can be different from axis to axis because of the different friction conditions. The input range is between 0 to 100% of the value determined by the circularity test. The default setting is 0; so that no compensation is performed for short traversing movements.		
Related to	MD 32500: FRICT_COMP_ENABLE — Friction compensation active		

32610 MD number	**VELO_FFW_WEIGHT** Feedforward control factor for speed feedforward control		
Default value: 1	Min. input limit: 0	Max. input limit: plus	
Changes effective after NEW_CONF	Protection level: 2	Unit: factor	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	In the case of speed feedforward control, a velocity setpoint is also applied directly to the input of the speed controller (see Section 2, subsection “Speed feedforward control”). This additional setpoint can be weighted with a factor (called feedforward control factor). To ensure that the speed feedforward control is set correctly, the equivalent time constant of the speed control loop must be determined precisely and entered in MD 32810: EQUIV_SPEEDCTRL_TIME. If the equivalent time constant of the speed control loop is defined exactly, the feedforward control factor has a value of approximately 1. In this case, the system deviation is roughly zero (check by looking at the service display in the operating area Diagnosis). If the feedforward control factor 0 is entered, feedforward control is deactivated. However, because the calculations are performed anyway, feedforward control must be deactivated with MD: FFW_MODE = 0.		
MD irrelevant for	MD 32620: FFW_MODE = 0 or 2		
Related to	MD 32620: FFW_MODE MD 32630: FFW_ACTIVATION_MODE MD 32810: EQUIV_SPEEDCTRL_TIME		

32620 MD number	**FFW_MODE** Feedforward control mode		
Default value: 1	Min. input limit: 0	Max. input limit: 4	
Changes effective after RESET	Protection level: 0/0	Unit: –	
Data type: byte	Applies from SW version: 4.3		
Significance:	FFW_MODE defines the feedforward control mode to be applied on an axis-specific basis. 0 = No feedforward control 1 = Speed feedforward control with PT1 symmetrization 2 = Torque feedforward control (only possible with SINUMERIK 840D) with PT1 symmetrization Extension of selection by **values 3 and 4 in SW 5.3 and later** The default setting is 1 to maintain compatibility to earlier software versions 3= Speed feedforward control with Tt symmetrization 4= Torque feedforward control (for 840D only) with Tt symmetrization FFWON and FFWOF are used to activate and deactivate the feedforward control for specific channels on all axes. To prevent the feedforward control from being affected by these instructions on individual axes, you can define that it is always activated or always deactivated in machine data FFW_ACTIVATION_MODE (see also the section on FFW_ACTIVATION_MODE). The global option data $ON_FFW_MODE_MASK can conceal torque feedforward control. If a feedforward control mode is selected (speed or torque feedforward control), it can be programmed additionally in MD 32630: FFW_ACTIVATION_MODE whether the feedforward control can be activated or deactivated by the part program. Torque feedforward control is an option that must be enabled.		
Application	In order to achieve excellent machining accuracy at high path velocities, contour inaccuracies due to following error can be eliminated using feedforward control.		
Related to	MD 32630: FFW_ACTIVATION_MODE MD 32610: VELO_FFW_WEIGHT MD 32650: AX_INERTIA		

32630 MD number	**FFW_ACTIVATION_MODE** Activate feedforward control from program		
Default value: 1	Min. input limit: 0	Max. input limit: 1	
Changes effective after RESET	Protection level: 0/0	Unit: –	
Data type: byte	Applies from SW version: 4.3		
Significance:	FFW_ACTIVATION_MODE can be set to define whether the feedforward control for this axis/spindle can be switched on and off in the part program. 0 = The feedforward control **cannot** be switched on or off by high-level language elements FFWON or FFWOF. The condition set in MD: FFW_MODE is always active for the axis/spindle. 1 = Feedforward control can be switched on/off with FFWON or FFWOF. The default setting is programmed in channel-specific data MD 20150: GCODE_RESET_VALUES. This setting is valid even before the first NC block has been executed. The last condition to be active remains active even after Reset (and therefore with JOG). Because the feedforward control for all axes of a channel is switched on/off with FFWON or FFWOF, MD:FFW_ACTIVATION_MODE should therefore have identical settings for axes that interpolate with each other.		
Related to	MD 32620: FFW_MODE MD 20150: GCODE_RESET_VALUES		
References	/PA1/, Programming Guide		

32640 MD number	**STIFFNESS_CONTROL_ENABLE** Activate dynamic stiffness control		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEW_CONF	Protection level: 2 / 7	Unit: –	
Data type:	Applies from SW version: 4.1		
Significance:	Activate dynamic stiffness control if bit is set. With active dynamic stiffness control, higher servo gain factors are possible (MD 32200: POSCTRL_GAIN). Due to the higher computing load in SIMODRIVE 611D, it may be necessary to adapt the settings of the scanning cycle (current/drive module cycle) in the 611D. For a single-axis drive module, the default setting (125 µs current, 125 µs speed controller cycle) is sufficient. The speed controller cycle might have to be increased (to 250 µs) for two-axis modules. Note: Currently, according to implementation in the drive, dynamic stiffness control is only possible with the motor measuring system.		
Related to			
References			

32650 MD number	**AX_INERTIA** Moment of inertia for torque feedforward control		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective after NEW_CONF	Protection level: 2	Unit: kgm^2	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	In the case of torque feedforward control, an additional current setpoint proportional to the torque is applied directly to the current controller input (see Section 2, subsection “Torque feedfoward control”). This value is formed using the acceleration and the moment of inertia. The equivalent time constant of the current control loop must be defined for this purpose and entered in MD 32800: EQUIV_CURRCTRL_TIME. The total moment of inertia of the axis (drive + load) must also be entered in AX_INERTIA (total moment of inertia referred to motor shaft according to data supplied by machine manufacturer). When AX_INERTIA and MD 32800: EQUIV_CURRCTRL_ TIME are set correctly, the following error is almost zero even during acceleration (check this by looking at the “following error” in the service display). The torque feedforward control is deactivated if AX_INERTIA is set to 0. However, because the calculations are performed anyway, torque feedforward control must always be deactivated with MD 32620 FFW_MODE = 0 or 1. Because of the direct current setpoint injection, torque feedforward control is only possible on digital drives (SIMODRIVE 611D).		
MD irrelevant for	SINUMERIK FM-NC or MD 32620: FFW_MODE = 0 or 1		
Application	Torque feedforward control is required to achieve high contour accuracy where the demands on the dynamics are great.		
Related to	MD 32620: FFW_MODE MD 32630: FFW_ACTIVATION_MODE MD 32800: EQUIV_CURRCTRL_TIME		

32652 MD number	**AX_MASS** Axis mass for torque feedforward control		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective after NEW_CONF	Protection level: 2 / 7	Unit: kg	
Data type: DOUBLE	Applies from SW version: 4.1		
Significance:	Mass of axis for torque feedforward control. This MD is used instead of AX_INERTIA on linear drives (DRIVE_TYPE=3).		
Related to			
References			

32700 MD number	**ENC_COMP_ENABLE[n]** LEC active [n]		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEW_CONF	Protection level: 2 / 7	Unit: –	
Data type: BOOLEAN	Applies from SW version: 1.1		
Significance:	1: LEC is activated for the axis/measuring system. With LEC, leadscrew errors and measuring system errors can be compensated. The function is only enabled if the measuring system has been referenced (IS: “Referenced/synchronized = 1”). Write protection function (compensation values) active. 0: Interpolatory compensation is not active for the axis/measuring system. Index [n] has the following coding: [encoder no.]: 0 or 1		
Related to	MD: MM_ENC_ COMP_ MAX_POINTS IS “Referenced/synchronized 1” IS “Referenced/synchronized 2”	Number of interpolation points for interpolatory compensation	

32710 MD number	**CEC_ENABLE** Enabling of beam sag compensation		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEW_CONF	Protection level: 2 / 4	Unit: –	
Data type: BOOLEAN	Applies from SW version: 2.0		
Significance:	1: "Beam sag compensation" if enabled for the compensation axis.With "beam sag compensation", inter-axis geometry errors (e.g. beam sag and angularity errors) can be compensated. The function is not enabled in the control until the following conditions have been fulfilled: • Option "Interpolatory compensation" is set • Associated compensation tables are available • Evaluation of the required compensation table is enabled (SD: CEC_TABLE_ENABLE[t] = 1) • The position measuring system required is referenced (IS "Referenced/synchronized" = 1). • Write protection function (compensation values) active. 0: "Beam sag compensation" is not enabled for the compensation axis.		
Related to	MD: MM_CEC_MAX_POINTS[t] SD: CEC_TABLE_ENABLE[t] IS "Referenced/synchronized 1 or 2"	Number of interpolation points for beam sag compensation Evaluation of beam sag compensation table t DB31–48, DBX60.4 or 60.5	

32711 MD number	**CEC_SCALING_SYSTEM_METRIC** Measuring system of beam sag compensation		
Default value: 1	Min. input limit: 0	Max. input limit: 1	
Changes effective after RESET	Protection level: 2 / 7	Unit: –	
Data type: BOOLEAN	Applies from SW version: 5		
Significance:	Compensation data are contained in: • MD 32711=0: inch system • MD 32711=1: metric system The measuring system can be configured for all beam sag compensation tables that affect the same axis. Hereby all position entries are interpolated together with the calculated total axial compensation value in the configured measuring system. External table conversions after the measuring system has been switched over are no longer necessary. Axial configuration of the measuring system is necessary, as only the total axial compensation value is referring unambiguously to a position, not the individual table contents that are calculated in relation to one another. **Note:** Only effective when MD 10260: CONVERT_SCALING_SYSTEM=1. (see /G2/)		
Related to	MD 10260: CONVERT_SCALING_SYSTEM	Basic system switchover active	

MD number	Measuring system of beam sag compensation		
32720 MD number	**CEC_MAX_SUM** Maximum compensation value for beam sag compensation		
Default value: 1	Min. input limit: 0	Max. input limit: 10	
Changes effective after NEW_CONF	Protection level: 2 / 4	Unit: Linear axis: mm Rotary axis: degrees	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	In beam sag compensation, the absolute magnitude of the total compensation value (sum of compensation values of all compensation relations) is monitored axially with machine data value CEC_MAX_SUM. If the determined total compensation value is larger than the maximum value, alarm 20124 is triggered. Program processing is not interrupted. The compensation value output as the additional setpoint is limited to the maximum value.		
MD irrelevant for	Leadscrew error compensation Backlash compensation Temperature compensation		
Related to	MD: CEC_ ENABLE SD: CEC_TABLE_ENABLE[t] IS “Referenced/synchronized 1 or 2”	Enable beam sag compensation Evaluation of beam sag compensation table t DB31–48, DBX60.4 or 60.5	

32730 MD number	**CEC_MAX_VELO** Maximum permissible change value for beam sag compensation		
Default value: 10	Min. input limit: 0	Max. input limit: 100	
Changes effective after NEW_CONF	Protection level: 2 / 4	Unit: %	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	In beam sag compensation, modification of the total compensation value (sum of the compensation values of all active compensation relations) is limited axially. The maximum change value is defined in this machine data as a percentage of MD 32000: MAX_AX_VELO (maximum axis velocity). If the change in the total compensation value is greater than the maximum value, alarm 20125 is output. Program processing is however continued. The path not covered because of the limitation is made up as soon as the compensation value is no longer subject to limitation.		
MD irrelevant for	Leadscrew error compensation Backlash compensation Temperature compensation		
Related to	MD: CEC_ ENABLE MD: MAX_AX_VELO SD: CEC_TABLE_ENABLE[t] IS “Referenced/synchronized 1 or 2”	Enable beam sag compensation Maximum axis velocity Evaluation of beam sag compensation table t DB31–48, DBX60.4 or 60.5	

32750 MD number	**TEMP_COMP_TYPE** Temperature compensation type		
Default value: 0	Min. input limit: 0	Max. input limit: 3	
Changes effective after Power On	Protection level: 2	Unit: Hex	
Data type: BYTE	Applies from SW version: 1.1		
Significance:	MD: TEMP_COMP_TYPE, the temperature compensation type effective for the machine axis is activated. A distinction is made between the following types: 0: No temperature compensation active 1: Position-**independent** compensation active (compensation value with SD: TEMP_COMP_ABS_VALUE) 2: Position-**dependent** compensation active (compensation value with SD: TEMP_COMP_SLOPE and SD: TEMP_COMP_REF_POSITION) 3: Position-dependent and position-independent temperature compensation active (compensation values with SD acc. to type 1 and type 2) Temperature compensation is an option that must be enabled.		
Related to	SD: TEMP_COMP_ABS_VALUE	Position-dependent temperature compensation value	
	SD: TEMP_COMP_REF_POSITION	Reference point for position-dependent temperature compensation	
	SD: TEMP_COMP_SLOPE	Gradient for position-dependent temperature compensation	
	MD: COMP_ADD_VELO_FACTOR	Velocity violation due to compensation	

32760 MD number	**COMP_ADD_VELO_FACTOR** Velocity violation due to compensation		
Default value: 0.01	Min. input limit: 0	Max. input limit: 0,1	
Changes effective after Power On	Protection level: 2	Unit: factor	
Data type: DOUBLE	Applies from SW version: 1.1		

Significance:

With axial MD: COMP_ADD_VELO_FACTOR the maximum distance that can be traversed because of temperature compensation in one IPO cycle is limited.
If the resulting temperature compensation value is above this maximum, it is traversed over several IPO cycles. There is no alarm.
The maximum compensation value per IPO cycle is a factor with reference to the maximum axis velocity (MD: MAX_AX_VELO).

With this machine data the maximum gradient of the temperature compensation $\tan\beta max$ is also limited.

Example of calculation of the maximum gradient $\tan\beta_{max}$:

1. Calculation of the interpolator closed-loop control (see Description of Functions Velocities, Setpoint/Actual-Value Systems, Closed-Loop Control (G2))
   Interpolator closed-loop control = basic system clock rate * factor for interpolation cycle
   Interpolator closed-loop control = MD: SYSCLOCK_CYCLE_TIME * MD: IPO_SYSCLOCK_TIME_RATIO
   Example: MD: SYSCLOCK_CYCLE_TIME = 0.004 [s]
   MD: IPO_SYSCLOCK_TIME_RATIO = 3
   ⇒ Interpolator closed-loop control = 0.004 * 3 = 0.012 [s]
2. Calculation of the maximum velocity increase because of a change made to the temperature compensation parameter Δv_{Tmax}
   Δv_{Tmax} = MD: MAX_AX_VELO * MD: COMP_ADD_VELO_FACTOR
   Example: MD: MAX_AX_VELO = 10 000 [mm/min]
   MD: COMP_ADD_VELO_FACTOR = 0.01
   ⇒ Δv_{Tmax} = 10 000 * 0.01 = 100 [mm/min]
3. Calculation of the traverse distances per interpolator cycle

$$S_1 \text{ (at } v_{max}) = 10\,000 \times \frac{0.012}{60} = 2.0 \text{ [mm]}$$

$$S_T \text{ (at } \Delta v_{Tmax}) = 100 \times \frac{0.012}{60} = 0.02 \text{ [mm]}$$

4. Calculation of $\tan\beta_{max}$

$$\tan\beta_{max} = \frac{S_T}{S_1} = \frac{0.02}{2} = 0.01$$ (corresponds to value for COMP_ADD_VELO_FACTOR)

   ⇒ β_{max} = arc tan 0.01 = 0.57 degrees
   With larger values of SD: TEMP_COMP_SLOPE the maximum gradient (here 0.57 degrees) for the position-dependent temperature compensation value is used internally. There is no alarm.

Note: Any additional velocity violation caused by temperature compensation must be taken into account when defining the limit value for velocity monitoring (MD: AX_VELO_LIMIT).

MD irrelevant for .....

TEMP_COMP_TYPE = 0, sag compensation, LEC, backlash compensation

Related to ....

MD: TEMP_COMP_TYPE	Temperature compensation type
SD: TEMP_COMP_ABS_VALUE	Position-independent temperature compensation
SD: TEMP_COMP_SLOPE	Gradient for position-dependent temperature compensation
MD: MAX_AX_VELO	Maximum axis velocity
MD: AX_VELO_LIMIT	Limit value for velocity monitoring
MD: IPO_SYSCLOCK_TIME_RATIO	Ratio basic system clock rate to IPO cycle
MD: SYSCLOCK_CYCLE_TIME	Basic system cycle

32800 MD number	**EQUIV_CURRCTRL_TIME[n]** Equivalent time constant of current control loop		
Default value: 0.0005	Min. input limit: 0	Max. input limit: plus	
Changes effective after NEW_CONF	Protection level: 2	Unit: s	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	This time constant must equal the equivalent time constant of the closed current control loop. It is used for parameterization of the torque feedforward control and for calculation of the dynamic following error model (contour monitoring). In order to set the torque feedforward control correctly, the equivalent time constant of the current control loop must be determined precisely by measuring the step response of the current control loop. With SIMODRIVE 611D the settling process can be displayed using the installation tool. The index [n] has the following coding: [control parameter block number]: 0 to 5 (**References:** /FB/, G2, “Velocities, Setpoint/Actual-Value Systems, Closed-Loop Control”)		
MD irrelevant for	SINUMERIK FM-NC		
Related to	MD: FFW_MODE MD: AX_INERTION MD: CONTOUR_TOL	Feedforward control type Moment of inertia for speed feedforward control Tolerance band contour monitoring	
References	/IAD/ Installation and Start-Up Guide /IAF/ Installation and Start-Up Guide		

32810 MD number	**EQUIV_SPEEDCTRL_TIME[n]** Equivalent time constant of speed control loop		
Default value: 0.004	Min. input limit: 0	Max. input limit: plus	
Changes effective after NEW_CONF	Protection level: 2	Unit: s	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	This time constant must equal the equivalent time constant of the closed speed control loop. It is used for parameterization of the speed feedforward control and for calculation of the dynamic following error model (contour monitoring). In order to set the speed feedforward control correctly, the equivalent time constant of the current control loop must be determined precisely by measuring the step response of the speed control loop. With SIMODRIVE 611D the settling process can be displayed using the installation tool. The index [n] has the following coding: [control parameter block number]: 0 to 5 **(References:** /FB/, G2, “Velocities, Setpoint/Actual-Value Systems, Closed-Loop Control”)		
Related to	MD: FFW_MODE MD: VELO_FFW_WEIGHT MD: CONTOUR_TOL	Feedforward control type Moment of inertia for speed feedforward control Tolerance band contour monitoring	
References	/IAF/, Installation and Start-Up Guide /IAD/, Installation and Start-Up Guide		

36700 MD number	**DRIFT_ENABLE** Automatic drift compensation		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEW_CONF	Protection level: 840D:0; FM-NC:2	Unit: –	
Data type: BOOLEAN	Applies from SW version: 1.1		
Significance:	MD: DRIFT_ENABLE activates automatic drift compensation. 1: Automatic drift compensation is active (only for position-controlled axes/spindles).With automatic drift compensation at zero speed, the control constantly determines the additional drift value required so that the value zero is reached for the following error (compensation criterion). The total drift value is therefore composed of the basic drift value (MD: DRIFT_VALUE) and the additional drift value (see Fig. 2-21). 0: Automatic drift compensation is not active. The drift value is only formed from the basic drift value (MD: DRIFT_VALUE).		
MD irrelevant for	840D or for axes/spindles which are not position-controlled		
Related to	MD: DRIFT_LIMIT MD: DRIFT_VALUE	Drift limit value for automatic drift compensation Drift basic value	

36710 MD number	**DRIFT_LIMIT** Drift limit value for automatic drift compensation		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective after NEW_CONF	Protection level: 840D: 0 FM-NC: 2	Unit: % of manipulated variable (e.g. 10 V ≐ 100%)	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	MD: DRIFT_LIMIT, the magnitude of the additional drift value determined during automatic drift compensation can be limited. If the additional drift value exceeds the value entered in MD: DRIFT_LIMIT, alarm 25070 “Drift value too large” is signalled and the additional drift value is limited to this value.		
MD irrelevant for	SINUMERIK 840D or MD: DRIFT_ENABLE = 0		
Related to	MD: DRIFT_ENABLE	Automatic drift compensation	

36720 MD number	**DRIFT_VALUE** Basic drift value		
Default value: 0	Min. input limit: 0	Max. input limit:	
Changes effective after NEW_CONF	Protection level: 840D:0; FM-NC:2	Unit: %	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	The basic drift value entered in MD: DRIFT_VALUE is always injected as an additional speed setpoint. The basic drift value is **always** active (independently of the MD: DRIFT_ENABLE)! While the automatic drift compensation only applies to position-controlled axes, the basic drift value is also active for speed-controlled axes/spindles.		
MD irrelevant for	SINUMERIK 840D		

38000 MD number	**MM_ENC_COMP_MAX_POINTS[n]** Number of interpolation points for LEC (SRAM)		
Default value: 0	Min. input limit: 0	Max. input limit: 5000	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	For leadscrew error compensation, the number of interpolation points required per measuring system must be defined. The required number can be calculated as follows using the defined parameters (see Section 2.3.2) $$\text{MD: MM\_ENC\_COMP\_MAX\_POINTS} = \frac{\$AA\_ENC\_COMP\_MAX - \$AA\_ENC\_COMP\_MIN}{\$AA\_ENC\_COMP\_STEP} + 1$$ $AA_ENC_COMP_MIN Initial position (system variable) $AA_ENC_COMP_MAX End position (system variable) $AA_ENC_COMP_STEP Distance between interpolation points (system variable) In selecting the number of interpolation points and the distance between them, it is important to take account of the resulting size of the compensation table and the required space in the backed-up NC user memory (SRAM). 8 bytes are required per compensation value (interpolation point). The index [n] has the following coding: [encoder no.]: 0 or 1		
Special cases, errors, ...	**Caution:** After any change in MD: MM_ENC_COMP_MAX_POINTS, the backed-up NC user memory is reallocated automatically on power up. All data in the backed-up NC user memory are then lost (e.g. part programs, tool offsets etc.). Alarm 6020 “Machine data changed – memory reallocated” is signalled. If reallocation of the NC user memory fails because the total memory capacity available is not sufficient, alarm 6000 “Memory allocation made with standard machine data” is signalled. In this case the NC user memory is allocated using the default values of the standard machine data. **References:** /FB/, S7, “Memory Configuration” /DA/, “Diagnostic Guide”		
Related to	MD: ENC_ COMP_ ENABLE[n] Interpolatory compensation active		
References	/FB/, S7, “Memory Configuration”		

38010 MD number	**MM_QEC_MAX_POINTS** Maximum number of compensation values for QEC with neural networks		
Default value: 0	Min. input limit: 0	Max. input limit: 1024	
Changes effective after Power On	Protection level: 2 / 4	Unit: –	
Data type: DWORD	Applies from SW version: 2.1		
Significance:	In quadrant error compensation with neural networks (QEC) the number of required compensation values must be entered for every axis to be compensated. The required number can be calculated as follows using the defined parameters (see Section 2.6.2): $MA_ MM_QEC_MAX_POINTS ≥ ($AA_QEC_COARSE_STEPS + 1) * $AA_QEC_FINE_STEPS $AA_QEC_COARSE_STEPS Coarse quantiz. of characteristic (system variable) $AA_QEC_FINE_STEPS Fine quantization of characteristic (system variable) For “direction-dependent” compensation the number must be greater than or equal to double value of this product. When selecting coarse or fine quantization, the resulting size of the compensation table and the memory required for it in the non-volatile user memory must be taken into account. 4 bytes are required for every compensation value. If the value 0 is entered, no memory is reserved for the table; i.e. the table does not exist and the function can therefore not be be activated.		
Special cases, errors, ...	**Caution!** If MD: MM_QEC_MAX_POINTS is altered, the non-volatile user memory is automatically re-allocated on system power-on. This deletes all the user data in the non-volatile user memory (e.g. drive and MMC machine data, tool offsets, part programs etc.). **Note:** Because the exact number of required interpolation points is not exactly known during the first installation of the function, a large number should be chosen initially. As soon as the characteristics are recorded and saved, the number can be reduced to the required size. After performing a power on again, the saved characteristics can be reloaded.		
References	/FB/, S7, “Memory Configuration”		

4.2 Description of setting data

41300 MD number	**CEC_TABLE_ENABLE** Enable evaluation of beam sag compensation table [t]		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective immediately	Protection level: 7	Unit: –	
Data type: BOOLEAN	Applies from SW version: 2.1		
Significance:	1: Evaluation of compensation table [t] is enabled. The compensation table defines, for example, the compensation relation (assignment of basis to compensation axis) with [t] = index of compensation table (see MD: MM_CEC_MAX_POINTS). In “beam sag compensation” the compensation axis can be influenced by several tables. SD:CEC_TABLE_ENABLE[t] can be altered by the NC part program or PLC user program to adapt the total compensation value of the machining application (e.g. switch over tables). The compensation is not enabled in the control until the following conditions have been fulfilled: • Option “Interpolatory compensation” is set • Assigned compensation tables exist • Beam sag compensation for compensation axis is activated (MD: $MA_CEC__ENABLE= 1) • The position measuring system required is referenced (IS “Referenced/synchronized” = 1). 0: Evaluation of the beam sag compensation table [t] is not enabled.		
Related to	MD: MM_CEC_MAX_POINTS[t] SD: CEC_TABLE_ENABLE[t] IS “Referenced/synchronized 1” IS “Referenced/synchronized 2”	Number of interpolation points for beam sag compensation Evaluation of beam sag compensation table t is enabled DB31–48, DBX60.4 DB31–48, DBX60.5	

41310 MD number	**CEC_TABLE_WEIGHT** Weighting factor for beam sag compensation table [t]		
Default value: 1.0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7	Unit: (factor)	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	The compensation value stored in the table [t] is multiplied by the weighting factor. When choosing the weighting factor, ensure that the resulting compensation value does not exceed the maximum value (MD: CEC_MAX_SUM). With [t] = index of compensation table (see MD: MM_CEC_MAX_POINTS) If, for example, the weight of the tools on the machine or workpiece to be machined differ greatly and affect the error curve by a change in amplitude, this can be corrected by changing the weighting factor. In beam sag compensation the weighting factor of the table can be altered for specific tools or workpieces by the PLC user program or the NC program by overwriting the setting data.If, however, the progression of the characteristic curve is greatly changed because of differing weights, different compensation tables must be used.		
Related to	SD: CEC_TABLE_ENABLE[t] MD: $MA_CEC_MAX_SUM	Enable evaluation of beam sag compensation table t Maximum compensation value for beam sag compensation	

43900 SD number	**TEMP_COMP_ABS_VALUE** Position-independent temperature compensation value		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: MMC-MD 9220	Unit: mm or degrees	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	With SD: TEMP_COMP_ABS_VALUE the **position-independent** temperature compensation value is defined (see Fig. 2-2). This value depends on the current temperature from the PLC (user program). As soon as position-independent temperature compensation has been activated (MD: TEMP_COMP_TYPE = 1 or 3), this additional compensation value is traversed by the machine axis.		
SD irrelevant for	MD: TEMP_COMP_TYPE = 0 or 2		
Related to	MD: TEMP_COMP_TYPE MD: COMP_ADD_VELO_FACTOR	Temperature compensation type Velocity violation caused by compensation	

43910 SD number	**TEMP_COMP_SLOPE** Gradient for position-dependent temperature compensation		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: MMC-MD 9220	Unit: mm or degrees	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	With **position-dependent** temperature compensation, the error curve of the temperature-dependent actual-value deviation can often be approximated by a straight line. This straight line is defined by a reference point P_0 and a gradient $\tan\beta$ (see Fig. 2-2). With SD: TEMP_COMP_SLOPE defines the gradient $\tan\beta$. This gradient can be changed by the PLC user program as a function of the current temperature. As soon as position-dependent temperature compensation is active (MD: TEMP_COMP_TYPE = 2 or 3), the axis traverses the compensation value calculated for the current actual position. MD: COMP_ADD_VELO_FACTOR limits the maximum gradient $\tan\beta_{max}$ of the error curve. This maximum gradient cannot be exceeded.		
SD irrelevant for	MD: TEMP_COMP_TYPE = 0 or 1		
Special cases, errors, ...	When TEMP_COMP_SLOPE is greater than $\tan\beta_{max}$, gradient $\tan\beta_{max}$ is used to calculate the position-dependent temperature compensation value internally. No alarm is output.		
Related to	MD: TEMP_COMP_TYPE SD: TEMP_COMP_REF_POSITION MD: COMP_ADD_VELO_FACTOR	Temperature compensation type Reference position for position-dependent temperature compensation Velocity violation caused by compensation	

43920 SD number	**TEMP_COMP_REF_POSITION** Reference position for position-dependent temperature compensation		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: MMC-MD 9220	Unit: mm or degrees	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	With **position-dependent** temperature compensation, the error curve of the temperature-dependent actual-value deviation can often be approximated by a straight line. This straight line is defined by a reference point P_0 and a gradient tanβ (see Fig. 2-2). With SD: TEMP_COMP_REF_POSITION the position of the reference point P0 is defined. This reference point position can be changed by the PLC user program as a function of the current temperature. As soon as position-dependent temperature compensation is active (MD: TEMP_COMP_TYPE = 2 or 3), the axis traverses the compensation value calculated for the current actual position.		
SD irrelevant for	MD: TEMP_COMP_TYPE = 0 or 1		
Related to	MD: TEMP_COMP_TYPE SD: TEMP_COMP_SLOPE	Temperature compensation type Gradient for position-dependent temperature compensation	

■

Signal Descriptions 5.

There are no separate signals for compensation.

Example 6.

– None –

Data Fields, Lists 7

7.1 Interface signals

DB number	Bit, byte	Name	Reference
Axis/spindle-specific			
31–48	60.4	Referenced/synchronized 1	R1
31–48	60.5	Referenced/synchronized 2	R1

7.2 Machine data

Number	Identifier	Name	Reference
General ($MN_...)			
10050	SYSCLOCK_CYCLE_TIME	Basic system cycle	G2
10070	IPO_SYSCLOCK_TIME_RATIO	Factor for interpolator cycle	G2
10082	CTRLOUT_LEAD_TIME	Shift of setpoint transfer time	
10083	CTRLOUT_LEAD_TIME_MAX	Maximum permissible setting for shift of setpoint transfer time	
18342	$MN_MM_CEC_MAX_POINTS[t]	Maximum number of interpolation points for the beam sag compensation	
Channel-specific ($MA_...)			
20150	GCODE_RESET_VALUES	Initial setting of G groups	K1
Axis-specific ($MC_ ...)			
32000	MAX_AX_VELO	Maximum axis velocity	G2
32200	POSCTRL_GAIN	Servo gain factor	G2
32450	BACKLASH[n]	Backlash	
32452	BACKLASH_FACTOR[n]	Weighting factor for backlash	
32460	TORQUE_OFFSET	Additional torque for electr. weight compen.	
32490	FRICT_COMP_MODE	Type of friction compensation	
32500	FRICT_COMP_ENABLE	Friction compensation active	
32510	FRICT_COMP_ADAPT_ENABLE [n]	Friction compensation adaptation active	
32520	FRICT_COMP_CONST_MAX [n]	Maximum friction compensation value	
32530	FRICT_COMP_CONST_MIN [n]	Minimum friction compensation value	
32540	FRICT_COMP_TIME [n]	Friction compensation time constant	
32550	FRICT_COMP_ACCEL1 [n]	Adaptation acceleration value 1	
32560	FRICT_COMP_ACCEL2 [n]	Adaptation acceleration value 2	
32570	FRICT_COMP_ACCEL3 [n]	Adaptation acceleration value 3	
32580	FRICT_COMP_INC_FACTOR	Weighting factor of friction compensation for short traversing movements	
32610	VELO_FFW_WEIGHT	Feedforward control factor for speed feedforward control	
32620	FFW_MODE	Feedforward control type	
32630	FFW_ACTIVATION_MODE	Activate feedforward control from program	
32640	STIFFNESS_CONTROL_ENABLE	Dynamic stiffness control	
32650	AX_INERTIA	Moment of inertia for torque feedforward control	
32652	AX_MASS	Mass of axis for torque feedforward control.	
32700	ENC_COMP_ENABLE	Interpolatory compensation active	
32710	CEC_ENABLE	Enable beam sag compensation	
32711	CEC_SCALING_SYSTEM_METRIC	Measuring system of beam sag compensation	
32720	CEC_MAX_SUM	Maximum compensation value for beam sag compensation	
32730	CEC_MAX_VELO	Max. value of change for beam sag comp.	
32750	TEMP_COMP_TYPE	Temperature compensation type	

Number	Identifier	Name	Reference
32760	COMP_ADD_VELO_FACTOR	Velocity violation through compensation	
32800	EQUIV_CURRCTRL_TIME[n]	Equivalent time constant of current control loop	
32810	EQUIV_SPEEDCTRL_TIME[n]	Equivalent time constant of speed control loop	
36200	AX_VELO_LIMIT	Limit value for velocity monitoring	A3
36400	CONTOUR_TOL	Tolerance band contour monitoring	A3
36500	ENC_CHANGE_TOL	Maximum tolerance for position actual-value switchover	G2
36700	DRIFT_ENABLE	Automatic drift compensation	
36710	DRIFT_LIMIT	Drift limit value for automatic drift compensation	
36720	DRIFT_VALUE	Drift basic value	
38000	MM_ENC_COMP_MAX_POINTS[n]	No. of interpolation points for interpolation compensation	
38010	MM_QEC_MAX_POINTS	Maximum number of offset values for QEC with neural networks	
SIMODRIVE 611D machine data ($MD_...)			
1004	CTRL_CONFIG	Configuration structure	IAD
1117	MOTOR_INERTIA	Motor moment of inertia	IAD

7.3 Setting data

Number	Identifier	Name	Reference
General ($MN_ ...)			
41300	CEC_TABLE_ENABLE	Enable evaluation of beam sag compensation table	
41310	CEC_TABLE_WEIGHT	Weighting factor for beam sag compensation table	
Axis-specific ($SA_...)			
43900	TEMP_COMP_ABS_VALUE	Position-independent temperature compensation value	
43910	TEMP_COMP_SLOPE	Lead angle for position-dependent temperature compensation	
43920	TEMP_COMP_REF_POSITION	Reference position for position-dependent temperature compensation	

7.4 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

■

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

Mode Groups, Channels, Axis Replacement (K5)

■

Notes

Brief Description 1

Mode groups

A mode group consists of a grouping or unit of machine axes, spindles and channels. A mode group can, in principle, be compared to an independent NC control (with several channels). A mode group contains all those channels that always have to operate in the same mode.

Note

In the standard case a mode group exists and is described in

References: /FB/, K1, “Mode Group, Channel, Program Operation Mode”

Channels

Each channel has its own functions for program decoding, block preparation and interpolation. A channel can process a part program independently.

Note

In the standard case a channel exists and is described in

References: /FB/, K1, “Mode Group, Channel, Program Operation Mode”

The processes in several channels of a mode group can be synchronized in the part programs.

Axis/spindle replacement

After control system Power On, an axis/spindle is assigned to a specific channel and can only be utilized in the channel to which it is assigned.

With the function “Axis/spindle replacement” it is possible to enable an axis/spindle and to allocate it to another channel, that means to replace the axis/spindle.

In SW 3 and higher, axis/spindle replacement can be activated both via the part program and via the PLC program.

■

Notes

Detailed Description 2

2.1 Mode groups

Mode groups

A mode group is an association of NC channels, axes and spindles that are grouped to form a machining unit.

A mode group contains all those channels that always have to operate in the same mode.

Any axis can be programmed in any channel of a certain mode group. A mode group therefore corresponds to an independent, multiple-channel NC.

Example

On large machine tools (machining centers), it may be necessary for a part program to be processed on one part of the machine while new workpieces to be machined need to be clamped and set up on another part. Such tasks usually require two independent NC controls.

With the mode group function, both tasks can be implemented on one NC control with two mode groups because a different mode can be set for each mode group (AUTOMATIC mode for the program processing, JOG for setting up a workpiece).

Mode group assignment

The configuration of a mode group defines the channels, geometry axes, machine axes and spindles which it is to contain.

A mode group consists of one or several channels which must not be assigned to any other mode group. Machine axes, geometry axes and special axes themselves are assigned to these channels. A machine axis can only be assigned to the channels of one mode group and can only traverse in this mode group.

A mode group is configured with the following data:

- Channel-specific MD 10010: ASSIGN_CHAN_TO_MODE_GROUP (channel valid in mode group)
- Configuration data of the channels

Note

For more information about the first mode group, please refer to

References: /FB/, K1, “Mode Group, Channel, Program Operation Mode”

2.2 Channels

Note

A description of the terms Channel, Channel Configuration, Channel States, Effects of Commands/Signals, etc. for the first channel can be found in

References: /FB/, K1, “Mode Group, Channel, Program Operation Mode”

For all other channels, this information applies, too.

2.2.1 Channel synchronization (program coordination)

General

Definition

As an example, double-slide machining operations or real-time processes can only be carried out if it is possible to synchronize processing in two channels. The channels affected shall perform certain processing procedures time-matched. To allow time-matched processing, the relevant channels must be joined to form a synchronization group (mode group).

The channel synchronization is programmed only via the NC language. The affected channels must be assigned to the **same mode group**.

Coordination

If several channels are involved in the machining of a workpiece, it may be necessary to synchronize program runs in the individual channels.There are special instructions (commands) for this program coordination. In each case, they are listed in one block.

Tabelle 2-1 Program coordination instructions

Instruction	Meaning
	SW 3
INIT(n,“identifier”,“q”)	Selection of a program for processing in a certain channel Acknowledgement mode: **n** (without) or **s** (synchronous) Name of the program with indication of the path Name of the channel: Values 1 to 4 possible
CLEAR (identifier)	Deletion of a program indicating the program identifier
START (n,n,n,)	Start of the programs selected in other channels Enumeration of the channel numbers: Values 1 to 4 possible
WAITM (Mnr, n, n, n, n)	Wait for mark number Mnr for program synchronization in the specified channels n (channel used can be indicated, but this is optional). The mark number must be identical in all channels. Numbers 0 to 9 are possible.
WAITE (n,n,n)	Waiting for the program end of the channels indicated (do not indicate program coordination channel)
	SW 4
SETM(Mnr1, Mnr2, ...Mnri)	Set wait marks Mnr1, Mnr2, ...Mnri for conditional wait with WAITMC() for the channel in which SETM() is issued. The channel thus declares to its partner channels that its wait characteristic is not/no longer fulfilled. The command can be activated in synchronized actions. Up to 10 marks (0 – 9) can be set using one command.
CLEARM(Mnr1, Mnr2, ...Mnri)	Delete wait marks Mnr1, Mnr2, ...Mnri for conditional wait with WAITMC() for the channel in which CLEARM() is issued. The channel thus declares to its partner channels that its wait characteristic is fulfilled. The command can be activated in synchronized actions. Up to 10 marks (0 – 9) can be deleted using one command.
WAITMC(Mnr, n1, n2, ...)	Conditional wait in continuous-path mode for the specified wait characteristic Mnr from the specified channels n1, n2, ... nk. The program coordination channel can be indicated, but this is optional. When processing continues after the wait marks from the other channels in the group have arrived, the wait marks of these channels are deleted.

Software version 3

Behavior

When a WAITM() call is reached, the axes in the current channel are decelerated and wait until the mark number specified in the call arrives from the other channels to be synchronized. The group is synchronized when the other channels are also decelerated as they reach their WAITM() command. The synchronized channels then continue operation.

Example of program coordination

```
Channel 1:
%100
N10 INIT(2,"_N_200_MPF","n")
N11 START(2)
.                    ; Processing in channel 1

N80 WAITM(1,1,2)     ; Waiting for WAIT mark 1 in channel 1 and in
                       channel 2
.                    ; Further processing in channel 1

N180 WAITM(2,1,2)    ; Waiting for WAIT mark 2 in channel 1 and in
                       channel 2
.                    ; Further processing in channel 1

N200 WAITE(2)        ; Waiting for program end of channel 2
N201 M30             ; Program end channel 1, total end
.
.

Channel 2:
%200
.                    ; Processing in channel 2
N70 WAITM(1,1,2)     ; Waiting for WAIT mark 1 in channel 1 and in
                       channel 2
.                    ; Further processing in channel 2
N270 WAITM(2,1,2)    ; Waiting for WAIT mark 2 in channel 1 and in
                       channel 2
.                    ; Further processing in channel 2

N400 M30             ; Program end of channel 2
.
.
```

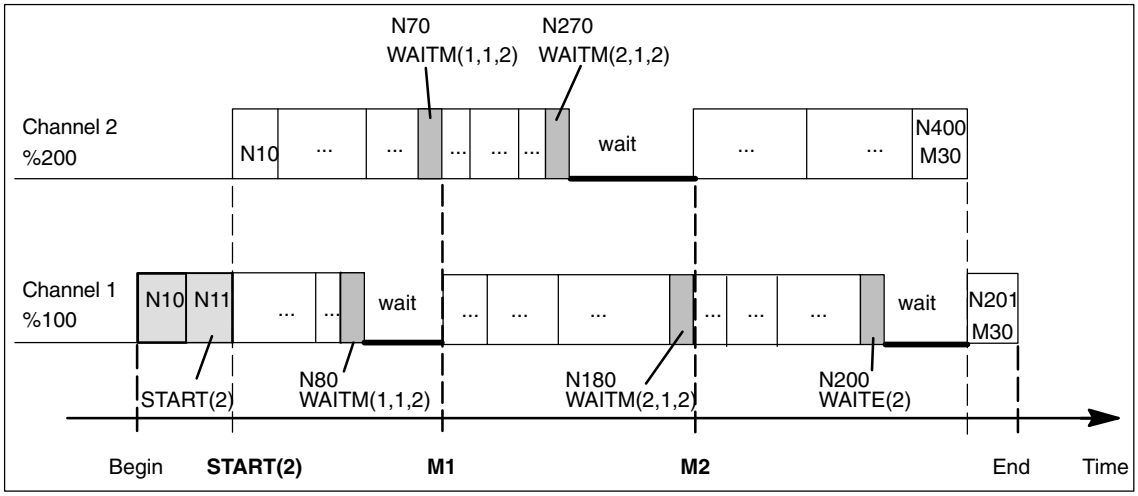


Fig. 2-1 Program runs illustrated by example of coordination with WAITM(), unconditional wait

References: /PA/, Programming Guide

Software version 4

Objective

Decelerating and waiting must take place only in cases where not all the channels to be coordinated have set their mark numbers for the purpose of synchronization. Conditional waiting.

The instants in time for generating wait marks and the conditional wait calls are decoupled.

For the purpose of inter-channel communication, marks may even be set when waiting and decelerating are not intended at all. No WAITMC() command. In this case, the channel marks settings remain valid after execution of RESET and NC Start.

Preconditions for conditional wait

To utilize conditional wait with WAITMC() and reduced wait times, the following conditions must be fulfilled:

- Continuous-path mode G64 must be set
- Look Ahead function must be active
- Exact stop (G60, G09) **must not** be set.
  If exact stop is selected, waiting with WAITMC() corresponds to waiting with WAITM() from SW level 3.

Response

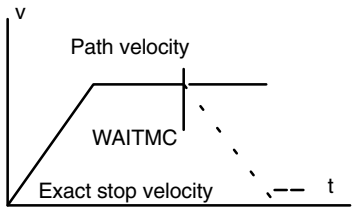
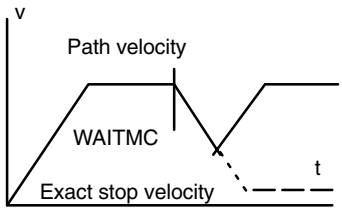
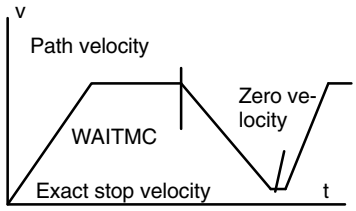
A) Starting with the motion block before the WAITMC() call, the wait marks of the other channels to be synchronized are checked. If these have all been supplied, then the channels continue to operate without deceleration in continuous-path mode. No wait. The path velocity remains unchanged.

B) If at least one wait mark from one of the channels to be synchronized is missing, then the axes start to decelerate from path velocity down to exact stop velocity. A check is now made in every interpolation cycle whether the missing wait marks from the channels to be coordinated have arrived. If they have, then the axes accelerate up to path velocity and continue to operate normally.

C) If the marks to be supplied by the channels to be synchronized have not arrived by the time exact stop velocity is reached, the machining operation is halted until the missing marks appear. When the last required mark appears, the axes are accelerated from standstill up to path velocity.

The following table shows the sequences of events for cases A) – C):

Tabelle 2-2 Deceleration response to conditional wait with WAITMC()

With WAITMC	Response	Velocity curve
A) Wait marks of all channels have already arrived	Continued operation with no deceleration	
B) All wait marks arrived during deceleration from path velocity down to exact stop velocity	Deceleration ceases immediately when last expected wait mark appears. The axes are accelerated back up to path velocity.	
C) The last wait mark does not arrive until exact stop velocity has been reached.	Brake down to exact stop velocity. When the last required mark appears, the axes are accelerated from exact stop velocity up to path velocity.	

Example of conditional wait in continuous-path mode

The example is schematic and shows only those commands that are relevant to the synchronization process.

Channel 1:

```
%100
N10   INIT(2, "_N_200_MPF","n")   ; Select partner program channel 2
N11   INIT(3, "_N_300_MPF","n")   ; Select partner program channel 3
N15   START(2, 3)                 ; Start programs in channels 2, 3
...                               ; Processing in channel 1
N20   WAITMC(7, 2, 3)             ; Wait conditionally for mark 7 from channels
                                  ; 2 and 3
...                               ; Processing continues in channel 1
N40   WAITMC(8, 2)                ; Wait conditionally for mark 8 from
                                  ; channel 2
...                               ; Processing continues in channel 1
N70   M30                         ; End of channel 1
```

Channel 2:

```
%200
N200                              ; Processing in channel 2
N210 SETM(7)                      ; Channel 2 sets wait mark 7
...                               ; Processing continues in channel 2
N250   SETM(8)                    ; Channel 2 sets wait mark 8
N260   M30                        ; End of channel 2
```

Channel 3:

```
%300
N300                                ; Processing in channel 3
...
N350   WHEN <condition> DO SETM(7)
                                    ; Set wait mark in a synchronized
                                    ; action
                                    ; Processing continues in channel 3
N360   M30                          ; End of channel 3
```

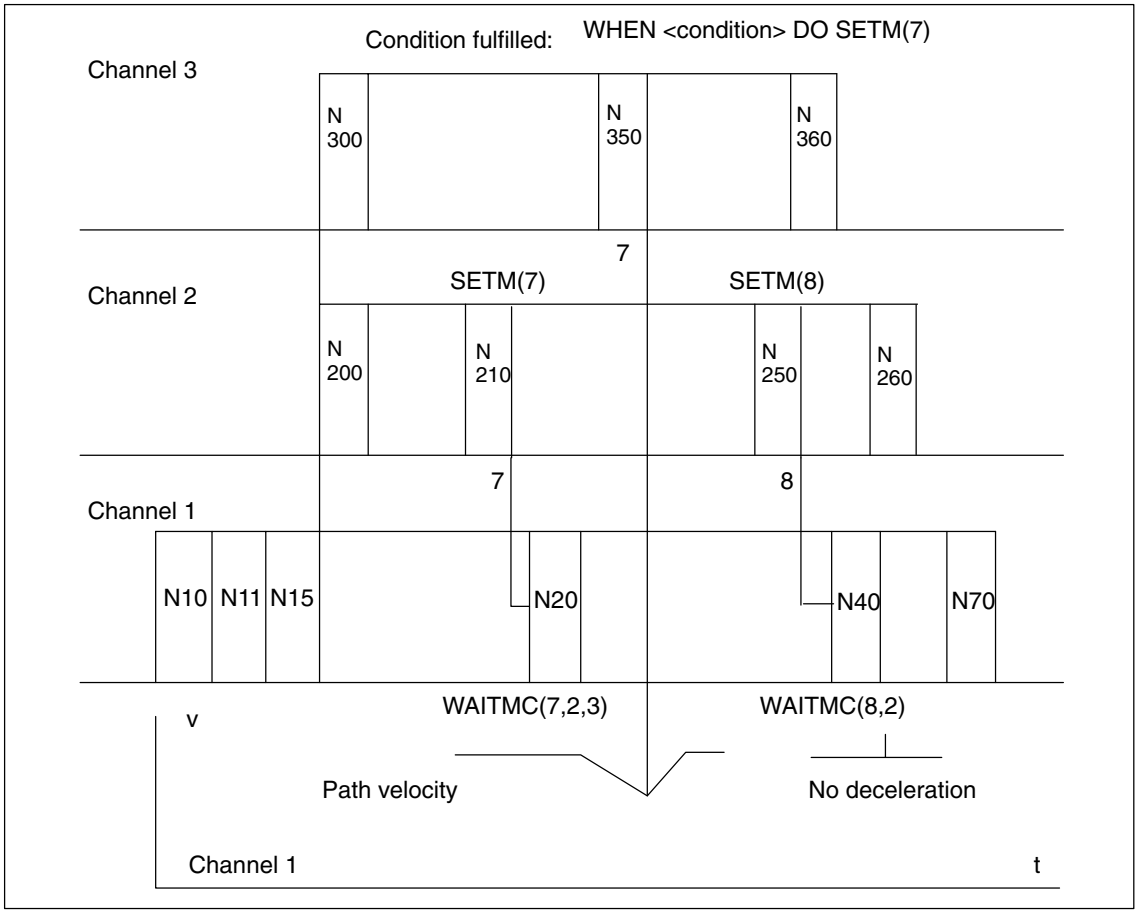


Fig. 2-2 Conditional wait involving three channels (schematic)

2.3 Axis/spindle replacement

2.3.1 Introduction

General

An axis/spindle is firmly allocated to a certain channel via the machine data. The axis/spindle can be used in this channel only.

Definition

The “Axis/spindle replacement” function allows an axis or spindle to be enabled and assigned to another channel, in other words, to be replaced.

Since the spindle function is subordinated to the axis function, only the term “Axis replacement” is used in the following.

Types of axes

According to the channel, we distinguish four types of axes: The reactions at axis change depend on the settings in MD 30552: AUTO_GET_TYPE.

Channel axis

A channel axis can be programmed in the part program and traversed in all modes.

PLC axis

A PLC axis can only be positioned via the PLC.

If a PLC axis is programmed in the part program

in case of MD AUTO_GET_TYPE = 0 an alarm will be output.

in case of MD AUTO_GET_TYPE = 1 an automatic GET will be generated.

in case of MD AUTO_GET_TYPE = 2 an automatic GETD will be generated.

Neutral axis

If a neutral axis is programmed in the part program

in case of MD AUTO_GET_TYPE = 0 an alarm will be output.

in case of MD AUTO_GET_TYPE = 1 an automatic GET will be generated.

in case of MD AUTO_GET_TYPE = 2 an automatic GETD will be generated.

Axis in another channel

This is actually not a proper type of axis. It is the internal state of a replaceable axis. If this happens to be active in another channel (as channel, PLC or neutral axis).

If an axis is programmed in another channel in the part program:

in case of MD AUTO_GET_TYPE = 0 an alarm will be output.

in case of MD AUTO_GET_TYPE = 1 an automatic GET will be generated.

in case of MD AUTO_GET_TYPE = 2 an automatic GETD will be generated.

Note

MD 20110: RESET_MODE_MASK and MD 20112: START_MODE_MASK control the behavior of the axis assignments in RESET, power-up and part program start. The settings for the channels between which axes are to be exchanged must be selected such that there are no incompatibilities in conjunction with MD 30552 : AUTO_GET_TYPE (alarms).
References: /FB/, K2, ... “Actual-Value System for Workpiece”

Display

The current type of axis and the current channel for this axis will be displayed in an axial PLC interface byte. See Section “Axis replacement by PLC”.

Preconditions

To allow an axis to be replaced, the following must be defined via
channel-specific MD 20070: AXCONF_MACHAX_USED
(machine axis number valid in channel)
and via
axis-specific MD 30550: AXCONF_ASSIGN_MASTER_CHAN
(initial setting of the channel for axis replacement):

1) In which channel can the axis be used and replaced?
2) To which channel shall the axis be allocated with POWER ON?

Example

With 6 axes and 2 channels, the 1st, 2nd, 3rd and 4th axis in channel 1 and the 5th and 6th axis in channel 2 shall be used. It shall be possible to replace the 1st axis, this shall be allocated to channel 2 after POWER ON.

The channel-specific MD must be allocated with:

CHANDATA(1)
AXCONF_MACHAX_USED=(**1**, 2, 3, 4, 0, 0, 0, 0)

CHANDATA(2)
AXCONF_MACHAX_USED=(5, 6, **1**, 0, 0, 0, 0, 0)

The axis-specific MD must be allocated with:

AXCONF_ASSIGN_MASTER_CHAN[AX1]=2

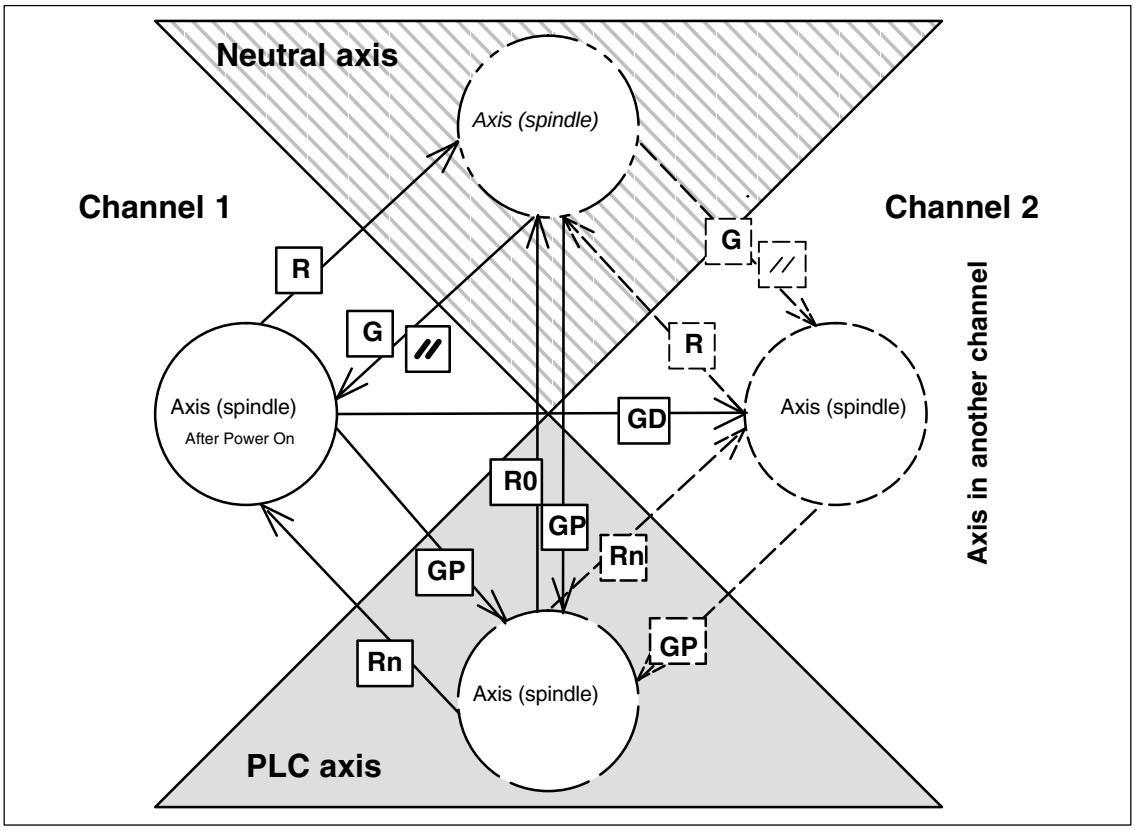
Note

If an axis is not valid in the channel selected, this is displayed by inversion of the axis name on the MMC.

2.3.2 Description

Possible transitions

The following diagram shows which axis replacement possibilities are available.



R	RELEASE(AX ...) from NC program	G	GET (AX ...) from NC program
//	RESET key		
R0	Release in neutral state via PLC	GP	GET via PLC
Rn	Release to a spec. channel via PLC	GD	GET directly from NC program

2.3.3 Axis transfer to neutral state (release)

RELEASE

Notation in part program:
RELEASE (axis name, axis name, SPI (spindle no.), ...)

Note

The axis name corresponds to the axis allocations in the system and is either

- AX1, AX2, AX3, ... **or**
- the name assigned in MD 10000: AXCONF_MACHAX_NAME_TAB.

With RELEASE (axis name, ...) a dedicated NC block will always be generated.

Exception: The axis is already in the neutral state.

The RELEASE command is interrupted if

- the prerequisites for axis replacement are not fulfilled
  (MD 20070: AXCONF_MACHAX_USED and
  MD 30550: AXCONF_ASSIGN_MASTER_CHAN)
- the axis is involved in a transformation
- the axis is within an axis network

Note

If the RELEASE command is applied to a gantry master axis, all following axes are released, too.

If ...	and ...	then ...
the axis is released, but not yet transferred with GET ...	... a RESET takes place via the operator panel ...	... the axis is allocated again to the last responsible channel.

2.3.4 Axis takeover

1a) Per command in the part program GET command

GET (axis name, axis name, SPI (spindle no.), ...)

Takeover of an axis is delayed if

- the axis is changing the measuring system
- servo disable is being processed for the axis (transition from control in follow-up/stop and vice versa)
- the axis/spindle disable is set
- the axis has not yet been enabled by the other channel with RELEASE
- interpolation for the axis has not yet been completed (except for a speed-controlled spindle)

With GET (axis name, ...) a separate NC block with search stop is always generated.

Exception:
- If the axis is already a channel axis, then no block is generated.
- If the axis is synchronous, (i.e. it has not been swapped to another channel in the meantime or received a signal from the PLC) no extra block is generated either.

1b) GETD command

An axis is fetched directly from another channel with **GETD** (GET Directly). That means that no suitable RELEASE must be programmed for this GETD in another channel. In addition, another channel communication must be created (e.g. wait marks), since the supplying channel is interrupted with GETD1. If the axis is a PLC axis, replacement is delayed until the PLC has enabled the axis.

Caution:

This programming command interrupts the program run in the channel in which the required axis is currently to be found! (REORG).

Exception: The axis is at the time in a neutral state.

Note

If ...	and ...	then ...
the GET command has been programmed, transfer is delayed ...	... a RESET takes place in the channel ...	... the channel does not try any longer to take over the axis.

An axis assumed with GET remains allocated to this channel even after a key RESET or program RESET. The axis can be replaced by programming RELEASE and GET again or will be assigned to the channel defined in MD 30550: AXCONF_ASSIGN_MASTER_CHAN after Power On.

---

2) Automatically through programming of axis name

Depending on the setting in MD 30552: AUTO_GET_TYPE, a GET or GETD command is automatically generated when a neutral axis is programmed again.

Example 1

```
N1  M3 S1000
N2  RELEASE (SPI(1))   ;=>Transition to neutral state
N3  S3000              ; New speed for released spindle
                       ; MD AUTO_GET_TYPE    =
                       ; 0 =>Alarm “Wrong axis type” is output
                       ; 1 => GET (SPI(1)) is generated.
                       ; 2 => GETD (SPI(1)) is generated.
```

Example 2

```
                          ; (axis 1 = X)
N1  RELEASE (AX1)         ;=>Transition to neutral state
N2  G04 F2
N3  G0 X100 Y100:         ; Motion of released axis
                          ; MD AUTO_GET_TYPE    =
                          ; 0 =>Alarm “Wrong axis type” is output
                          ; 1 => GET (AX1) is generated.
                          ; 2 => GETD (AX1) is generated.
```

Example 3

```
                          ; (axis 1 = X)
N1  RELEASE (AX1)         ;=>Transition to neutral state
N2  G04 F2
N3  POS (X) = 100:        ; Positioning the released axis:
                          ; MD AUTO_GET_TYPE    =
                          ; 0 => Alarm “Wrong axis type” is output
                          ; 1 => GET (AX1) is generated. *)
                          ; 2 => GETD (AX1) is generated. *)
```

*) If the axis is still synchronized, no dedicated block will be generated.

Note

If an automatic GETD is set, the following must be observed:

1. The channels may influence one another.
   (REORG if axis is taken away)
2. With simultaneous access of several channels to an axis it is not known which channel will have the axis at the end.

2.3.5 Examples of an axis replacement

Assumption

With 6 axes and 2 channels, the 1st, 2nd, 3rd and 4th axis in channel 1 and the 5th and 6th axis in channel 2 shall be used. It shall be possible to replace the 2nd axis between the channels and to allocate to channel 1 after POWER ON.

Task

The task is subdivided into the following areas:

- Machine data allocation so that the prerequisites for axis replacement are given.
- Programming of axis replacement between channel 1 and channel 2.

Fulfillment of preconditions

Assignment of channel-specific MD 20070:
AXCONF_MACHAX_USED[1]=(1, **2**, 3, 4, 0, 0, 0, 0)
AXCONF_MACHAX_USED[2]=(5, 6, **2**, 0, 0, 0, 0, 0)

Assignment of axis-specific MD 30550:
AXCONF_ASSIGN_MASTER_CHAN[AX2]=1

Program in channel 1

```
...
RELEASE (AX2)
            ; Release of axis AX2
INIT (2, "_N_MPF_DIR\_N_TAUSH2_MPF", "S")
            ; Selection of program TAUSH2 in channel 2
START (2)
            ; Start of program in channel 2
WAITM (1,1,2)
            ; Waiting for wait mark 1 in channels 1 and 2
...         ;
...         ; Further sequence of operations after axis
...         ;
...         ;
M30
```

Program TAUSH2 in channel 2

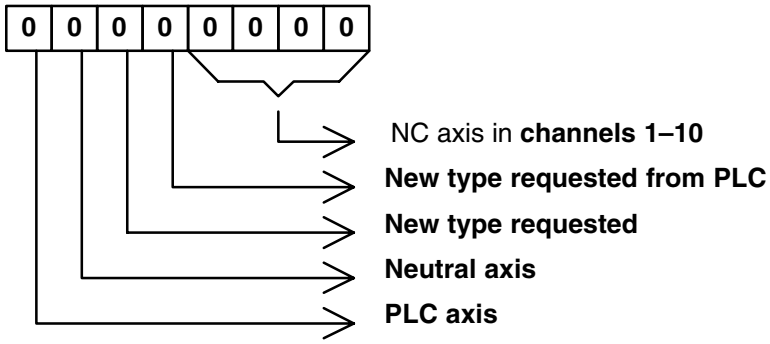
```
...
WAITM (1,1,2)
            ; Wait for wait mark 1 in channels 1 and 2
GET (AX2)
            ; Assumption of axis AX2
...         ;
...         ; Further sequence of operations after axis
...         ; replacement RELEASE (AX2)
            ; Release for further axis replacement
...         ;
M30
```

2.3.6 Axis replacement via PLC

- The type of an axis can be determined at any time via an interface byte (PLC axis, channel axis, neutral axis)

TYPE display

NCK => PLC (DB31– ..., DBB68)



Ex.1: Channel axis K2

0	0	0	0	0	0	1	0

Ex.2: PLC axis. Interpolation and interface processing in K1

1	0	0	0	0	0	0	1

Ex.3: Neutral axis. Interface processing in K3

0	1	0	0	0	0	1	1

	NCK=>PLC, DBB68	PLC=>NCK, DBB8
After Power On	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 0
RELEASE (K1)	0 1 1 0 0 0 0 1	
	0 1 0 0 0 0 0 1	
GET (K2)	0 0 1 0 0 0 1 0	
	0 0 0 0 0 0 1 0	

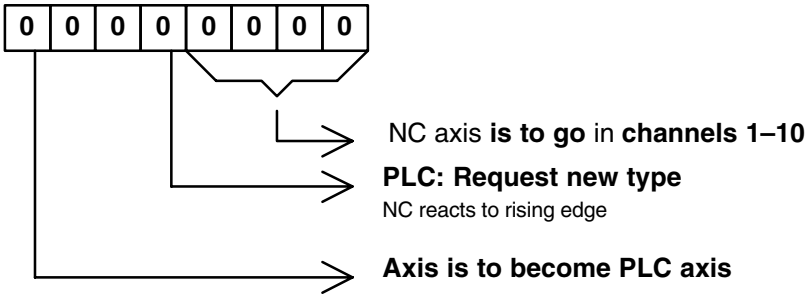
Time

Fig. 2-3 Changing an axis from K1 to K2 via part program.

- The PLC can request and traverse an axis at any time and in any operating mode.

Specifying TYPE

PLC => NCK (DB31– ..., DBB8)



In principle, the PLC must set the signal “Request new type”. It is deleted again after change. This also applies to a channel change with GET and RELEASE.

- The PLC can change an axis from one channel to another.

PLC axes and PLC spindles are traversed via special function modules in the basic PLC program.

FC15: POS_AX positioning of linear and rotary axes

FC16: PART_AX positioning of indexing axes

FC18: SpinCtrl spindle control

Examples

The following diagrams show the IS signal sequences for changing an NC axis to a PLC axis and transferring an NC axis to a neutral axis through the PLC.

	NCK=>PLC, DBB68	PLC=>NCK, DBB8
After Power On	0 0 0 0 0 0 0 **1**	0 0 0 0 0 0 0 0
New TYPE (PLC)		**1** 0 0 **1** 0 0 0 0
	0 0 **1** **1** 0 0 0 1	1 0 0 1 0 0 0 0
	0 0 1 1 0 0 0 1	1 0 0 **0** 0 0 0 0
	1 0 **0** **0** 0 0 0 1	1 0 0 0 0 0 0 0

Time

Fig. 2-4 Changing an NC axis to a PLC axis

	NCK=>PLC, DBB68	**PLC=>NCK, DBB8**
After Power On	0 0 0 0 0 0 0 **1**	0 0 0 0 0 0 0 0
New TYPE (PLC)		**1** 0 0 **1** 0 0 0 0
	0 0 **1** **1** 0 0 0 1	1 0 0 1 0 0 0 0
		1 0 0 **0** 0 0 0 0
	1 0 **0** **0** 0 0 0 1	1 0 0 0 0 0 0 0
New TYPE (PLC)		**0** 0 0 **1** 0 0 0 0
	1 0 **1** **1** 0 0 0 1	0 0 0 1 0 0 0 0
	0 **1** **0** **0** 0 0 0 1	0 0 0 **0** 0 0 0 0

Time

Fig. 2-5 Changing an NC axis to a neutral axis through the PLC.

■

Supplementary Conditions 3

"Mode group" function

There are up to 10 mode groups for the SINUMERIK 840D and one for the FM-NC.

Number of channels

Up to 10 channels are available on the SINUMERIK 840D and one on the FM-NC.

"Axis/spindle replacement" function

This function is available for

- SINUMERIK 840D with NCU 572/573, with SW 2 and higher

Change to the channel axis

If an axis is changed from PLC axis, neutral axis or axis in another channel to the axis type channel axis, a synchronization must take place.

With this synchronization,

- the current positions are assumed
- the current speed and gear stage is assumed with spindles.

It is therefore obligatory to perform a feed stop which interrupts the active path movement.

If the axis is transferred with GET, this transition is clearly defined by the part program.

If the axis is allocated by the PLC, the program section in which the change takes place is not clearly foreseeable.

(Except by a separate user-specific NC <–> PLC logics)

For this reason, the change to the channel axis is delayed in the following conditions:

- Path mode is active (G64+axes programmed)
- Thread cutting/tapping is active (G33/G331/G332)

Change from a channel axis

The change of a channel axis to a neutral axis or PLC axes cannot be performed during an active path operation.

With RELEASE this is caused by the fact that RELEASE must be located in a separate NC block.

If the PLC changes the axis type, a REORG is triggered internally. Therefore, the change with the listed program conditions is delayed.

Block search

During block search with calculation, all GET, GETD or RELEASE blocks are stored and output after the next NC Start.

Exception:

Blocks which are mutually exclusive are deleted.

Example:

N10	RELEASE (AX1)	**Blocks are deleted**
N40	GET (AX1)	”
N70	Destination	

■

Data Descriptions (MD, SD) 4

4.1 Axis/spindle-specific machine data

30550 MD number	**AXCONF_ASSIGN_MASTER_CHAN** Initial setting of channel for change of axis		
Default value: 0	Min. input limit: 0	Max. input limit: 2	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: BYTE	Applies from SW version: 2		
Significance:	Definition of the channel to which the axis is allocated after Power On		
Application	With the function “Axis/spindle replacement”, a machine axis must be allocated to a channel after power on. AXCONF_ASSIGN_MASTER_CHAN[AX2]=1 ⇒ Axis AX2 is assigned to channel 1 after power on.		
Related to	MD: AXCONF_MACHAX_USED		

30552 MD number	**AUTO_GET_TYPE** Definition for automatic GET		
Default value: 1	Min. input limit: 0	Max. input limit: 1	
Changes effective after Power On	Protection level: 2	Unit:	
Data type: BYTE	Applies from SW version: 3		
Significance:	0=No automatically created GET ⇒ Alarm in response to incorrect programming. 1=GET is output when GET is generated automatically. 2=GETD is output when GET is generated automatically.		
Related to ...			

■

Notes

Signal Descriptions

5

DB31, ... **DBB8** Data block	**Axis/spindle replacement** Signal(s) to channel (PLC ——> NCK)		
Edge evaluation: yes		Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The current axis type and currently active channel for this axis must be specified. Bit 0: A Assign NC axis/spindle to channel Bit 1: B ... Bit 2: C ... Bit 3: D Assign NC axis/spindle to channel Bit 4: Activation, assignment by positive signal edge Bit 5: – Bit 6: – Bit 7: Request PLC axis/spindle		
Signal state 0 or signal transition 1 ——> 0			
Related to ...	IS DB31, ... DBB68, “Axis/spindle replacement” MD 20070: AXCONF_ASSIGN_MASTER_USED (machine axis number valid in channel) MD 30550: AXCONF_ASSIGN_MASTER_CHAN (initial setting of channel for axis replacement)		
Special cases, errors, ...			

DB31, ... **DBB68** Data block	**Axis/spindle replacement** Signal(s) to channel (PLC ——> NCK)		
Edge evaluation: yes		Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The current axis type and currently active channel for this axis are displayed. Bit 0: A NC axis/spindle in channel Bit 1: B ... Bit 2: C ... Bit 3: D NC axis/spindle in channel Bit 4: New type requested by PLC Bit 5: New type requested (GETD) Bit 6: Neutral axis/spindle Bit 7: PLC axis/spindle		
Signal state 0 or signal transition 1 ——> 0			
Related to...	IS DB31, ... DBB68, “Axis/spindle replacement” MD 20070: AXCONF_ASSIGN_MASTER_USED (machine axis number valid in channel) MD 30550: AXCONF_ASSIGN_MASTER_CHAN (initial setting of channel for axis replacement)		
Special cases, errors, ...			

Notes

6.7 Example

None

Data Fields, Lists

Reference with the data

There is a reference for data which are not described in this Description of Functions (e.g. /K1/ means that the description can be found in Description of Functions K1).

7.1 General machine data

Number	Identifier	Name	Reference
General ($MN_ ...)			
10010	ASSIGN_CHAN_TO_MODE_GROUP[n]	Channel valid in mode group [channel no.]: 0, 1	K1

7.2 Channel machine data

7.2.1 Basic channel machine data

Number	Identifier	Name	Reference
Channel-specific ($MC_ ...)			
20000	CHAN_NAME	Channel name	K1
20050	AXCONF_GEOAX_ASSIGN_TAB[n]	Assignment between geometry axis and channel axis [GEO axis no.]: 0...2	K2
20060	AXCONF_GEOAX_NAME_TAB[n]	Geometry axis name in channel [GEO axis no.]: 0...2	K2

Number	Identifier	Name	Reference
Channel-specific ($MC_ ...)			
20070	AXCONF_MACHAX_USED[n]	Machine axis number valid in channel [channel axis no.]: 0...7	K2
20080	AXCONF_CHANAX_NAME_TAB[n]	Channel axis name in channel [channel axis no.]: 0...7	K2
20090	SPIND_DEF_MASTER_SPIND	Initial setting of master spindle in channel	S1
20100	DIAMETER_AX_DEF	Geometry axis with facing axis function	P1
20150	GCODE_RESET_VALUES[n]	Initial setting of G groups [G group no.]: 0...59	K1
20160	CUBIC_SPLINE_BLOCKS	No. of blocks with C spline	K1
20170	COMPRESS_BLOCK_PATH_LIMIT	Maximum traverse length of NC block with compression	K1
20200	CHFRND_MAXNUM_DUMMY_BLOCKS	Empty blocks with phase/radii	K1
20210	CUTCOM_CORNER_LIMIT	Maximum angle for compensation blocks with TRC	W1
20220	CUTCOM_MAX_DISC	Maximum value for DISC	W1
20230	CUTCOM_CURVE_INSERT_LIMIT	Maximum angle for calculation of intersection with TRC	W1
20240	CUTCOM_MAXNUM_CHECK_BLOCKS	Blocks for future contour calculation with TRC	W1
20250	CUTCOM_MAXNUM_DUMMY_BLOCKS	Block no. without traversing movement with TRC	W1
20270	CUTTING_EDGE_DEFAULT	Basic setting of tool cutting edge w/o programming	W1
20400	LOOKAH_USE_VELO_NEXT_BLOCK	Look Ahead on programmed following block velocity	B1
20430	LOOKAH_NUM_OVR_POINTS	No. of prepared override velocity characteristics with Look Ahead	B1
20440	LOOKAH_OVR_POINTS[n]	Prepared override velocity characteristics with lookahead [characteristic no.]: 0...1	B1
20500	CONST_VELO_MIN_TIME	Minimum time with constant velocity	B2
20600	MAX_PATH_JERK	Path-related maximum jerk	B2
20610	ADD_MOVE_ACCEL_RESERVE	Acceleration reserve for overlaid movements	K1
20650	THREAD_START_IS_HARD	Acceleration behavior of axis with thread cutting	K1
20700	REFP_NC_START_LOCK	NC start disable w/o reference point	R1
20750	ALLOW_GO_IN_G96	GO logics with G96	V1
20800	SPF_END_TO_VDI	Subprogram end to PLC	H2
21000	CIRCLE_ERROR_CONST	Circle end monitoring constant	K1
21010	CIRCLE_ERROR_FACTOR	Circle end monitoring factor	K1
21100	ORIENTATION_IS_EULER	Angle definition for orientation programming	F2
21110	X_AXIS_IN_OLD_X_Z_PLANE	Coordinate system with automatic frame definition	K2
21200	LIFTFAST_DIST	Distance traversed with fast lifting from contour	K1
21250	START_INDEX_R_PARAM	No. of first channel-specific R parameter	S7

7.2.2 Auxiliary function settings of channel

Number	Identifier	Name	Reference
Channel-specific ($MC_ ...)			
22000	AUXFU_ASSIGN_GROUP[n]	Auxiliary function group [aux. func. no. in channel]: 0...49	H2
22010	AUXFU_ASSIGN_TYPE[n]	Auxiliary function type [aux. func. no. in channel]: 0...49	H2
22020	AUXFU_ASSIGN_EXTENSION[n]	Auxiliary function extension [aux. func. no. in channel]: 0...49	H2
22030	AUXFU_ASSIGN_VALUE[n]	Auxiliary function value [aux. func. no. in channel]: 0...49	H2
22200	AUXFU_M_SYNC_TYPE	Output time of M functions	H2
22210	AUXFU_S_SYNC_TYPE	Output time of S functions	H2
22220	AUXFU_T_SYNC_TYPE	Output time of T functions	H2
22230	AUXFU_H_SYNC_TYPE	Output time of H functions	H2
22240	AUXFU_F_SYNC_TYPE	Output time of F functions	H2
22250	AUXFU_D_SYNC_TYPE	Output time of D functions	H2
22260	AUXFU_E_SYNC_TYPE (available soon)	Output time of E functions.	–
22300	AUXFU_AT_BLOCK_SEARCH_END	Auxiliary function output after block search run	H2
22400	S_VALUES_ACTIVE_AFTER_RESET	S function effective beyond RESET	S1
22410	F_VALUES_ACTIVE_AFTER_RESET	F functions effective beyond RESET	V1
22500	GCODE_OUTPUT_TO_PLC	G functions to PLC	K1
22550	TOOL_CHANGE_MODE	New tool offset with M function	W1
22560	TOOL_CHANGE_M_CODE	M function for tool change	W1

7.2.3 Transformation definitions in channel

Number	Identifier	Name	Reference
Channel-specific ($MC_ ...)			
24100	TRAFO_TYPE_1	Definition of transformation 1 in channel	F2
24110	TRAFO_AXES_IN_1[n]	Axis assignment for transformation [axis index]: 0...7	F2
24120	TRAFO_GEOAX_ASSIGN_TAB_1[n]	Assignment between GEO axis and channel axis for transformation 1[GEO axis no.]: 0...2	F2
24200	TRAFO_TYPE_2	Definition of transformation 2 in channel	F2
24210	TRAFO_AXES_IN_2[n]	Axis assignment for transformation 2 [axis index]: 0...7	F2
24220	TRAFO_GEOAX_ASSIGN_TAB_2[n]	Assignment between GEO axis and channel axis for transformation 2[GEO axis no.]: 0...2	F2
24300	TRAFO_TYPE_3	Definition of transformation 3 in channel	F2
24310	TRAFO_AXES_IN_3[n]	Axis assignment for transformation 3 [axis index]: 0...7	F2

Number	Identifier	Name	Reference
Channel-specific ($MC_ ...)			
24320	TRAFO_GEOAX_ASSIGN_TAB_3[n]	Assignment between GEO axis and channel axis for transformation 3 [GEO axis no.]: 0...2	F2
24400	TRAFO_TYPE_4	Definition of transformation 4 in channel	F2
24410	TRAFO_AXES_IN_4[n]	Axis assignment for transformation 4 [axis index]: 0...7	F2
24420	TRAFO_GEOAX_ASSIGN_TAB_4[n]	Assignment between GEO axis and channel axis for transformation 4 [GEO axis no.]: 0...2	F2
24430	TRAFO_TYPE_5	Definition of transformation 5 in channel	F2
24432	TRAFO_AXES_IN_5[n]	Axis assignment for transformation 5 [axis index]: 0...7	F2
24434	TRAFO_GEOAX_ASSIGN_TAB_5[n]	Assignment between GEO axis and channel axis for transformation 5 [GEO axis no.]: 0...2	F2, M1
24440	TRAFO_TYPE_6	Definition of transformation 6 in channel	F2
24442	TRAFO_AXES_IN_6[n]	Axis assignment for transformation 6 [axis index]: 0...7	F2
24444	TRAFO_GEOAX_ASSIGN_TAB_6[n]	Assignment between GEO axis and channel axis for transformation 6 [GEO axis no.]: 0...2	F2, M1
24450	TRAFO_TYPE_7	Definition of transformation 7 in channel	F2
24452	TRAFO_AXES_IN_7[n]	Axis assignment for transformation 7 [axis index]: 0...7	F2
24454	TRAFO_GEOAX_ASSIGN_TAB_7[n]	Assignment between GEO axis and channel axis for transformation 7 [GEO axis no.]: 0...2	F2, M1
24460	TRAFO_TYPE_8	Definition of transformation 8 in channel	F2
24462	TRAFO_AXES_IN_8[n]	Axis assignment for transformation 8 [axis index]: 0...7	F2
24464	TRAFO_GEOAX_ASSIGN_TAB_8[n]	GEO axis to channel axis assignment for transformation 8 [GEO axis no.]: 0...2	F2, M1
24500	TRAFO5_PART_OFFSET_1[n]	Offset vector of 5-axis transformation 1 [axis no.]: 0...2	F2
24510	TRAFO5_ROT_AX_OFFSET_1[n]	Position offset of rotary axes 1/2 for 5-axis transformation 1 [axis no.]: 0...1	F2
24520	TRAFO5_ROT_SIGN_IS_PLUS_1[n]	Sign of rotary axis 1/2 for 5-axis transformation 1 [axis no.]: 0...1	F2
24530	TRAFO5_NON_POLE_LIMIT_1	Definition of pole range for 5-axis transformation 1	F2
24540	TRAFO5_POLE_LIMIT_1	Limit angle tolerance with interpolation by pole for 5-axis transformation 1	F2
24550	TRAFO5_BASE_TOOL_1[n]	Vector of base tool with activation of 5-axis transformation 1 [axis no.]: 0...2	F2
24560	TRAFO5_JOINT_OFFSET_1[n]	Vector of kinematic offset for 5-axis transformation 1 [axis no.]: 0...2	F2
24600	TRAFO5_PART_OFFSET_2[n]	Offset vector of 5-axis transformation 2 [axis no.]: 0...2	F2
24610	TRAFO5_ROT_AX_OFFSET_2[n]	Position offset of rotary axes 1/2 for 5-axis transformation 2 [axis no.]: 0...1	F2

Number	Identifier	Name	Reference
Channel-specific ($MC_ ...)			
24620	TRAFO5_ROT_SIGN_IS_PLUS_2[n]	Sign of rotary axis 1/2 for 5-axis transformation 2 [axis no.]: 0...1	F2
24630	TRAFO5_NON_POLE_LIMIT_2	Definition of pole range for 5-axis transformation 2	F2
24640	TRAFO5_POLE_LIMIT_2	Limit angle tolerance with interpolation by pole for 5-axis transformation 2	F2
24650	TRAFO5_BASE_TOOL_2[n]	Vector of base tool with activation of 5-axis transformation 2 [axis no.]: 0...2	F2
24660	TRAFO5_JOINT_OFFSET_2[n]	Vector of kinematic offset for 5-axis transformation 2 [axis no.]: 0...2	F2

7.2.4 Channel-specific memory settings

Number	Identifier	Name	Reference
Channel-specific ($MC_ ...)			
25000	REORG_LOG_LIMIT	Percentage of IPO buffer for release of log file	S7
28000	MM_REORG_LOG_FILE_MEM	Memory size for REORG (DRAM)	S7
28010	MM_NUM_REORG_LUD_MODULES	No. of modules for local user variables with REORG (DRAM)	S7
28020	MM_NUM_LUD_NAMES_TOTAL	No. of local user variables (DRAM)	S7
28030	MM_NUM_LUD_NAMES_PER_PROG	No. of local user variables per program (DRAM)	S7
28040	MM_LUD_VALUES_MEM	Memory size for local user variables (DRAM)	S7
28050	MM_NUM_R_PARAM	No. of channel-specific R parameters (SRAM)	S7
28060	MM_IPO_BUFFER_SIZE	No. of NC blocks in IPO buffer (DRAM)	S7
28070	MM_NUM_BLOCKS_IN_PREP	No. of blocks for block preparation (DRAM)	S7
28080	MM_NUM_USER_FRAMES	No. of frames to be set (SRAM)	S7
28090	MM_NUM_CC_BLOCK_ELEMENTS	No. of block elements for compile cycles (DRAM)	S7
28100	MM_NUM_CC_BLOCK_USER_MEM	Size of block memory for compile cycles (DRAM)	S7
28500	MM_PREP_TASK_STACK_SIZE	Stack size of preparation task (DRAM)	S7
28510	MM_IPO_TASK_STACK_SIZE	Stack size of IPO task (DRAM)	S7

7.3 Axis/spindle-specific machine data

Number	Identifier	Name	Reference
Axis/spindle-specific($MA_ ...)			
30550	AXCONF_ASSIGN_MASTER_CHAN	Initial setting of channel for change of axis	
30552	AUTO_GET_TYPE	Definition of automatic GET	
30600	FIX_POINT_POS	Fixed value positions of axes with G75	K1
33100	COMPRESS_POS_TOL	Maximum deviation with compression	K1

7.4 Channel-specific setting data

Number	Identifier	Name	Reference
Channel-specific($SC_ ...)			
42000	THREAD_START_ANGLE	Start angle with thread	K1
42100	DRY_RUN_FEED	Test run feed	V1

7.5 Interface signals

7.5.1 Mode group signals

Description of interface signals

The mode group signals from PLC → NCK and from NCK → PLC are stored in data block 11 for the first mode group. The signals are displayed and described in

References: /FB/, K1, “Mode Group, Channel, Program Operation Mode”

7.5.2 Channel signals

Description of interface signals

The channel signals from PLC → NCK and from NCK → PLC are stored in data blocks 21, 22, ... for the first, second ... channels. The signals are displayed and described in

References: /FB/, K1, “Mode Group, Channel, Program Operation Mode”

7.6 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

■

Notes

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

FM-NC Local Bus (L1)

■

Notes

Brief Description

1

Overview

The SINUMERIK FM-NC has a local P bus. It is possible to connect modules belonging to the S7-300 system to this local P bus.

FM 354 as 5th axis

The FM-NC can be extended by one axis by connecting the FM 354 module (FM servo).

Implementing NCK I/Os

Apart from the FM 354, you can also connect digital and analog input/output modules to this local P bus. These modules make it possible to provide the FM-NC with digital inputs and outputs.

■

Notes

Detailed Description

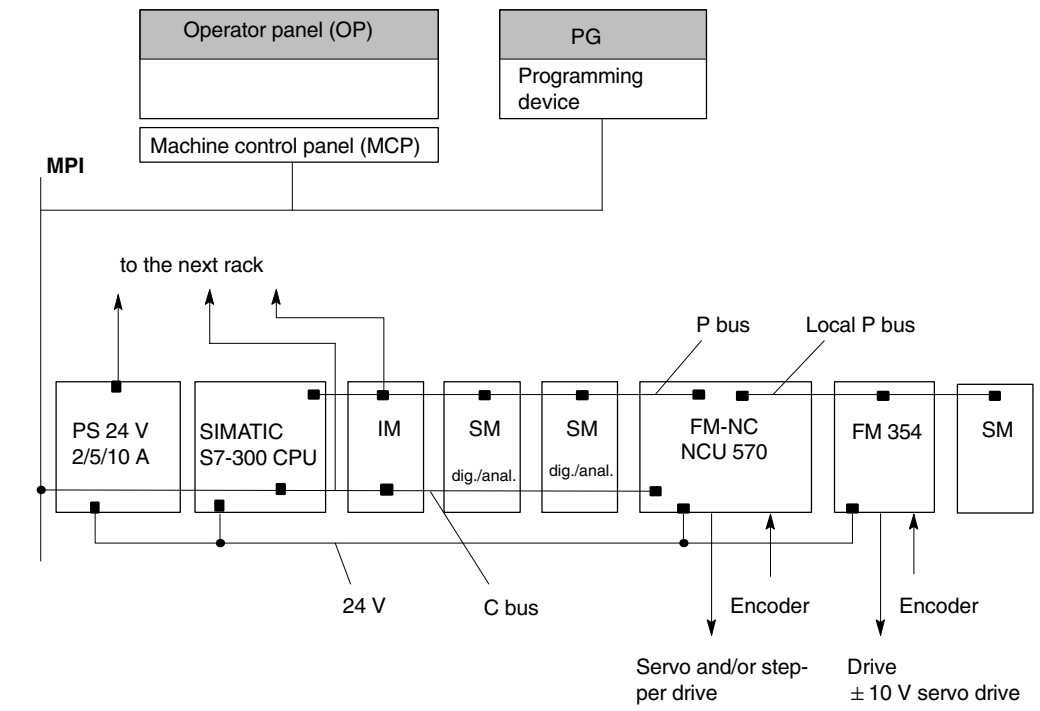
2

2.1 Configuration of local P bus

Overview

The purpose of the local P bus is to extend the functionality of the FM-NC module. An FM 354 (FM servo) and digital input/output modules (SMs) can be connected to this bus. The FM 354 increases the number of FM-NC axes from four to five while the SM modules provide the FM-NC with additional digital inputs and outputs.

Fig. 2-1 shows the local P bus of the FM-NC as an integral component of the overall S7-300 system



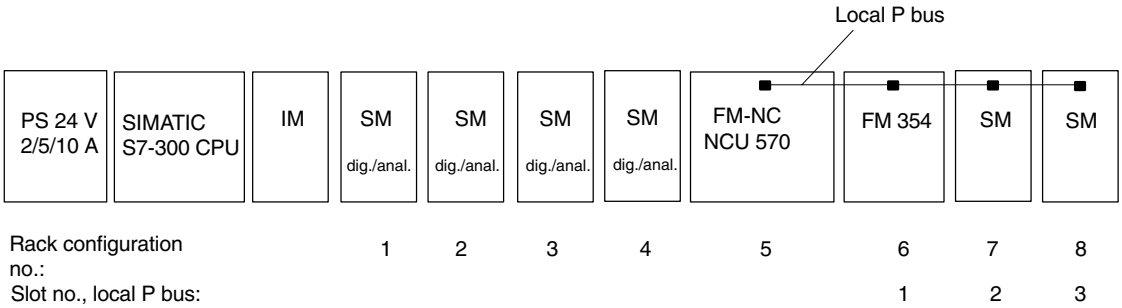
PS – Power supply
IM – Interface module
SM – Signal module (input/output module)
FM 354 – Function module as axis extension
CPU – PLC (Programmable Logic Controller) of FM-NC
MPI – Multi Point Interface (serial interface with multi-point capability)

Fig. 2-1 Local P bus of FM-NC in the S7-300 system

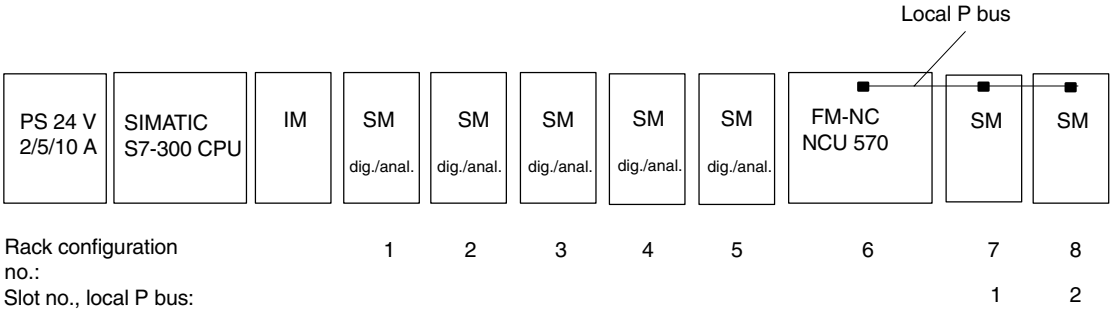
Configuration

The rack configuration determines how many modules can be connected to the local P bus. The maximum possible number the rack can hold is eight (without PLC and IM). The higher the number of modules connected to the P bus of the PLC, the lower the number which may be connected to the local P bus of the FM-NC (see Fig. 2-2).

Example 1: Rack configuration with FM 354 as 5th axis (max. rack configuration per rack tier)



Example 2: Rack configuration without FM 354 as 5th axis (max. rack configuration per rack tier)



Example 3: Rack configuration with FM 354 as 5th axis (rack can be extended by a maximum of three further modules)

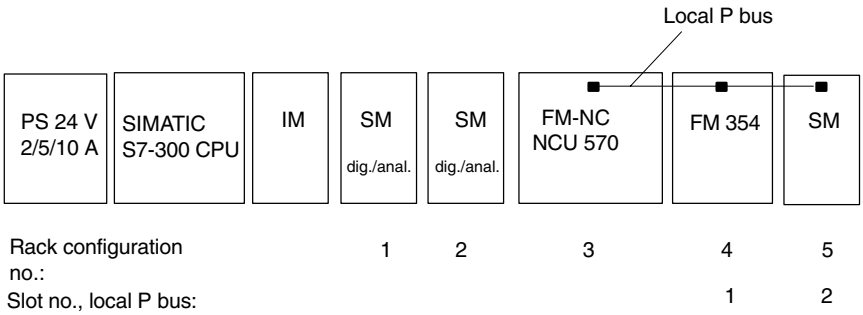


Fig. 2-2 Examples of configuration of S7-300 system rack

FM 354 as 5th axis

Only one FM 354 module (FM servo) may be connected to the local P bus as the 5th axis.

Inputs/outputs

It is possible to connect a maximum of 20 digital and analog inputs/outputs (nos. 9 to 28) to the local P bus. Nos. 1 to 8 are always simulation inputs/outputs (see Section 2.3).

Example:

Nos. 1 to 8	No hardware (simulation inputs)
Nos. 9 to 24	2 signal modules with 8 digital inputs each
Nos. 25 to 32	FM 354 with 4 digital hardware inputs (nos. 25 ... 28) and 4 simulation inputs (nos. 29...32)

Configuration

The configuration of the local P bus is detected by the PLC (CPU) during FM-NC power-up in exactly the same way as that of the P bus.

This rack configuration is displayed in the STEP7 main menu “S7 configuration” (read out SDB2). The “NCU” module must be selected.

In the menu:

Services

└──────── Parameterization

select submenu:

NC CPU Parameterization.

The fields “Process alarm generation” and “Local bus segment” must be selected in this menu and activated with OK (SDB100 is generated and transferred to the PLC). The local P bus is then activated.

2.2 FM 354 (FM servo) as 5th axis of the FM-NC

Overview

By connecting an FM 354 (FM servo) to the local P bus of the FM-NC, it is possible to increase the number of machine axes controlled by the FM-NC from four to five. All the parameter settings and calculations for the 5th axis are performed in the FM-NC.

The FM 354 acts as an actual-value input and as a setpoint output for the FM-NC.

Configuration

Please refer to Section 2.1 or SDB100 for “Configuration of 5th axis yes/no”. When the FM 354 (FM servo) is configured correctly as the 5th axis of the FM-NC, the yellow LED “DIAG” flashes according to the set basic clock rate.

Example

If machine data MD10050: SYSCLOCK_CYCLE_TIME is set to 0.006, then LED “DIAG” flashes in a 6-second cycle (3 s bright and 3 s dark).

Interfaces and front panel elements of FM 354 (FM servo)

Fig. 2-3 shows the FM 354 interfaces to the S7-300 system and the NC machine.

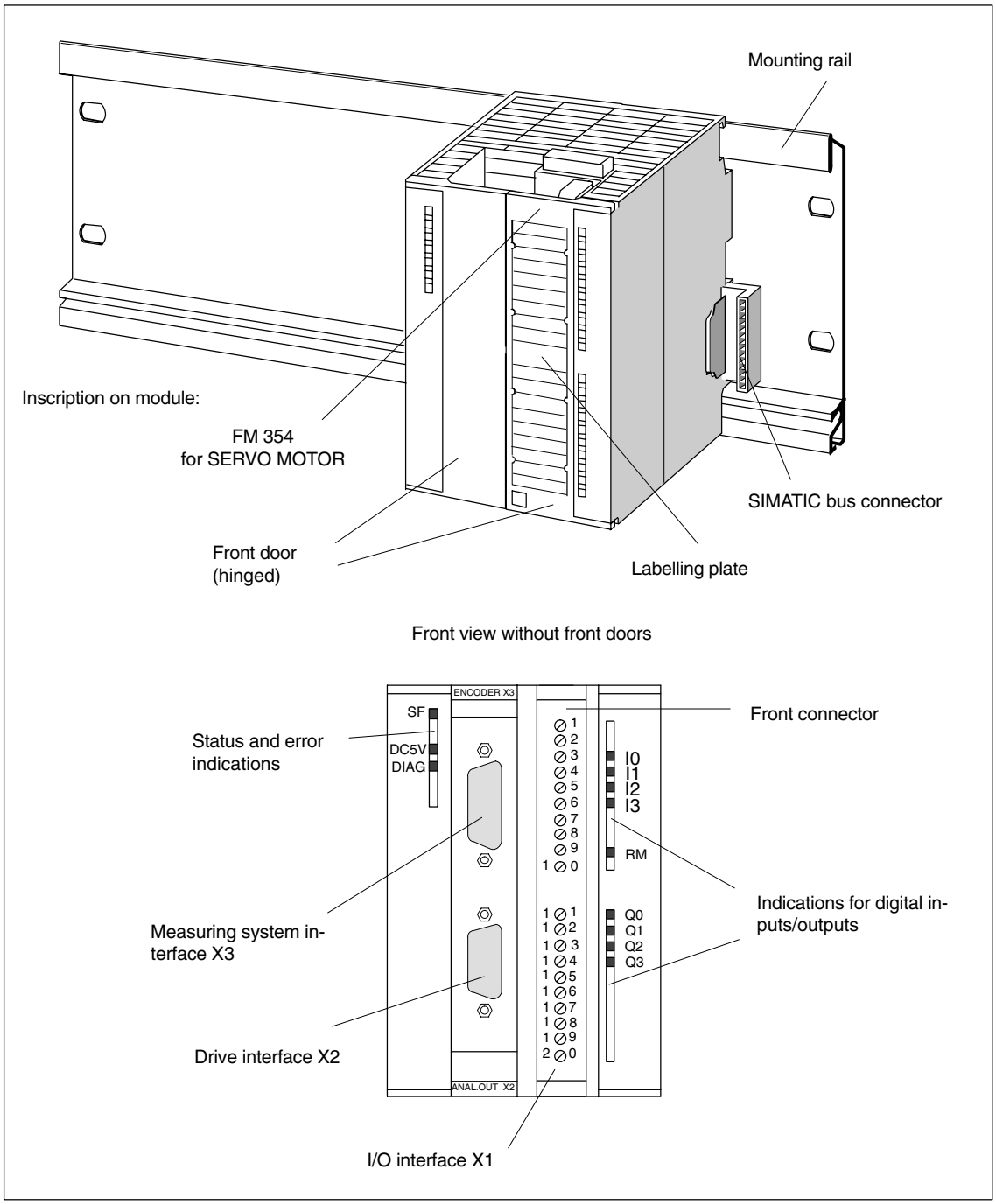


Fig. 2-3 Position of interfaces and front panel elements

Interfaces

- **SIMATIC interface**

  Rear panel connectors for connection of FM 354 to further control components via the S7-300 backplane bus (P and K buses)

- **Drive interface**

  9-pin sub D connector (X2) for connecting the drive unit

- **Measuring system interfaces**

  A position encoder can be connected to the 15-pin sub D socket (X3).

- **I/O interface**

  20-pin front panel connector (X1) for connecting the digital inputs/outputs and for wiring power supply

PIN 3...6	Digital input 1...4
PIN 11...14	Digital output 1...4
PIN 19/20	Load power supply 24 V/ground (L+, M)

Displays

- 3 LEDs for error and status displays
- 8 LEDs for digital inputs/outputs
- 1 LED for “Controller message” input (not used for FM-NC)

Machine data

The 5th axis (FM 354) need not be the 5th channel axis of the FM-NC. The 5th axis is parameterized in exactly the same way as the four other axes.

The setpoint output of the FM 354 (on the local P bus) is assigned to an axis by means of MD 30100: $MA_CTRLOUT_SEGMENT_NR[n] = 2 (local P bus identifier, see Section 6).

The actual-value input (measuring system) is assigned to the appropriate axis via MD 30210: $MA_ENC_SEGMENT_NR[n] = 2 (local P bus identifier, see Chapter 6).

!

Important

It is not permissible to mix the axes of the setpoint output and the actual-value input of the 5th axis (FM 354), i.e. the setpoint output and the actual-value input of the 5th axis may only be operated on the same channel axis (see Chapter 3).

2.3 Digital inputs/outputs on local P bus of FM-NC

Overview

Digital input and output modules of the SIMATIC S7-300 system (e.g. SM 321) can be connected to the local P bus of the FM-NC. It is possible to provide the FM-NC with digital inputs and outputs in this way.

For a detailed description of how to handle these inputs/outputs, please refer to

References: /FB/, A4, “Digital and Analog NCK I/Os”

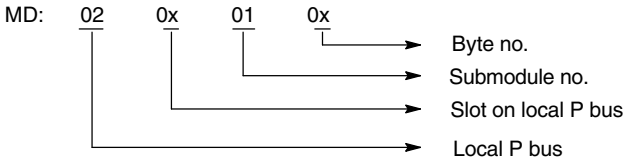
The following description contains only a few supplementary notes and references to special features of the local P bus.

Machine data for digital inputs/ outputs

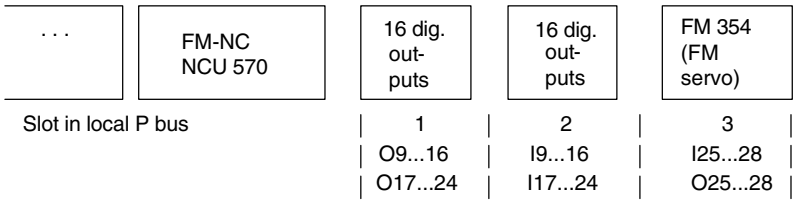
In addition to the configuration settings of the digital inputs/outputs in SDB100, the following machine data must also be noted:

- MD10366: $MN_HW_ASSIGN_DIG_FASTIN[0...3]
- MD10368: $MN_HW_ASSIGN_DIG_FASTOUT[0...3]

The following applies to these MDs:



Example: The local P bus of the FM-NC is structured as follows:



Consequently:

MD10366: ... IN[0]	=	02	**02**	01	**01**	I 9...16
MD10366: ... IN[1]	=	02	**02**	01	**02**	I17...24
MD10366: ... IN[2]	=	02	**03**	01	**01**	I25...28
MD10366: ... IN[3]	=	02	**00**	01	**00**	Not assigned
MD10368: ... OUT[0]	=	02	**01**	01	**01**	Q 9...16
MD10368: ... OUT[1]	=	02	**01**	01	**02**	Q17...24
MD10368: ... OUT[2]	=	02	**03**	01	**01**	Q25...28
MD10368: ... OUT[3]	=	02	**00**	01	**00**	Not assigned

!

Important

The FM 354 (FM servo) has four digital inputs/outputs which can be addressed via front connector X1. The remaining digital inputs/outputs of the byte (29...32) can be addressed as simulation inputs/outputs.

Setting the available digital inputs/outputs

The following MDs

MD 10350: $MN_FASTIO_DIG_NUM_INPUTS and

MD10360: $MN_FASTIO_DIG_NUM_OUTPUTS

are used to parameterize the available digital inputs/outputs for the part program (these take into account all inputs/outputs, simulation or hardware).

Possible setpoint	Meaning
1	1...8 NCK I/O without hardware (simulation)
2	9...16 ext. I/O of FM-NC (local P bus)
3	17...24 ext. I/O of FM-NC (local P bus)
4	25...32 ext. I/O of FM-NC (local P bus)

Reading and writing digital inputs/outputs via the part program

Read:

$A_IN[n] n = Number of input

Write:

$A_OUT[n] n = Number of output

Examples:

- R1 = $A_IN[9]

  The circuit state of input 9 is stored in R1.

- $A_OUT[9] = R1

  The circuit state (high or low) of R1 is output at output 9.

- $A_OUT[10] = $A_IN[11]

  The state of input 11 is output at output 10.

2.4 Analog inputs/outputs on local P bus of FM-NC)

Overview

Analog input/output modules of the SIMATIC S7-300 system (e.g. SM 335) can be connected to the local P bus of the FM-NC in addition to digital modules. For a detailed description of how to handle these input/outputs, please refer to

References: /FB/, A4, “Digital and Analog NCK I/O Devices”.

The parameter settings “CPU in Stop, retain last value” is not relevant for use on the local P bus of the FM-NC.

MDs for analog inputs/outputs

Apart from the configuration of the analog inputs/outputs in SDB100, the following machine data must also be noted:

- MD10362: $MN_HW_ASSIGN_ANA_FASTIN
- MD10364: $MN_HW_ASSIGN_ANA_FASTOUT

(See Section 2.3 for meaning of data.)

Setting the available analog inputs/outputs

The following MDs:

- MD10300: $MN_FASTIO_ANA_NUM_INPUTS and
- MD10310: $MN_FASTIO_ANA_NUM_OUTPUTS.

are used for parameterizing the available inputs and outputs that can be used for the part program.

Weighting factor

It is possible to adjust the analog NCK inputs/outputs to the various types of AD/DA converter hardware for reading in the part program by means of the following general machine data:

- MD10320: $MN_FASTIO_ANA_INPUT_WEIGHT
- MD10330: $MN_FASTIO_ANA_OUTPUT_WEIGHT

kann für die analoge NCK E/A eine Anpassung an die hardwaremäßig verschiedenen AD/DA-Wandler für das Lesen im Teileprogramm vorgenommen werden.

The following values can be set for the FM-NC:

INPUT:	Measuring range	0...2 V	→	MD10300 = 2370
	Measuring range	0...10 V	→	MD10300 = 11851 (default)
OUTPUT:	Output range	0...10 V	→	MD10310 = 11852 (default)
	Output range	± 10 V	→	MD10310 = 11851

■

Notes

Supplementary Conditions 3

System clock rate for FM-NC with five axes

The basic system clock rate of the FM-NC must be set to 8 ms if the FM-NC has five position-controlled axes, i.e. in contrast to the default setting (0.006), MD10050: $MN_SYSCLOCK_CYCLE_TIME must be set to 0.008.

Mixing axes of setpoint output with actual value input

The setpoint output and actual value input of an FM 354 (FM servo) acting as a 5th axis must be assigned to the same channel axis.

In other words, if the parameter setting of

MD 30100: $MA_CTRLOUT_SEGMENT_NR[0, AX3] is “2”,

then the parameter setting in

MD 30210: $MA_ENC_SEGMENT_NR[0, AX3] must also be “2”.

Number of analog inputs/outputs

The CPU computing time is required on the interpolation plane for processing the digital and analog NCK I/Os connected to the local P bus of the FM-NC. The number of active NCK I/O devices should be restricted to avoid overloading the system. For this reason, only one analog module may be activated on the local P bus for the FM-NC.

■

Notes

4 Data Descriptions (MD, SD)

4.1 Machine data for 5th axis

No special signals are required for the 5th axis of the FM-NC (except for those described in References: /FB/, G1, “Gantry Axes”).

The following machine data are relevant for parameterization of the 5th axis:

MD 30100: $MA_CTROUT_SEGMENT_NR[n]
MD 30110: $MA_CTROUT_MODULE_NR[n]
MD 30120: $MA_CTROUT_NR[n]
MD 30130: $MA_CTROUT_TYPE[n]
} for setpoint output

MD 30210: $MA_ENC_SEGMENT_NR[n]
MD 30220: $MA_ENC_MODULE_NR[n]
MD 30230: $MA_ENC_INPUT_NR[n]
MD 30240: $MA_ENC_TYPE[n]
} for actual value input

These machine data are described in

References: /FB/, G2, “Velocities, Setpoint/Actual Value Systems, Closed-Loop Control”

4.2 Machine data for digital inputs/outputs of FM-NC

The following machine data are relevant for parameterizing the inputs/outputs:

MD 10350: $MN_FASTIO_DIG_NUM_INPUTS
MD 10360: $MN_FASTIO_DIG_NUM_OUTPUTS
MD 10366: $MN_HW_ASSIGN_DIG_FASTIN[0...3]
MD 10368: $MN_HW_ASSIGN_DIG_FASTOUT[0...3]

These machine data are described in

References: /FB/, A4, “Digital and Analog NCK I/Os”

4.3 Machine data fur analog inputs/outputs of FM-NC

The following machine data are relevant for parameterizing the inputs/outputs:

MD10300: $MN_FASTIO_ANA_NUM_INPUTS
MD10310: $MN_FASTIO_ANA_NUM_OUTPUTS
MD10320: $MN_FASTIO_ANA_INPUT_WEIGHT
MD10330: $MN_FASTIO_ANA_OUTPUT_WEIGHT
MD10362: $MN_HW_ASSIGN_ANA_FASTIN
MD10364: $MN_HW_ASSIGN_ANA_FASTOUT

These machine data are described in

References: /FB/, A4, “Digital and Analog NCK I/Os”

■

Signal Descriptions 5

5.1 Signals for 5th axis

No special signals are required for the 5th axis of the FM-NC.

The axis signals are described in

References: /FB/, G2, “Velocities, Setpoint/Actual Value Systems, Closed-Loop Control”

5.2 Signals for digital and analog inputs/outputs of FM-NC

The signals for the digital inputs and outputs are described in

References: /FB/, A4, “Digital and Analog NCK I/Os”

■

Notes

Example 6

Parameterizing a 5th axis

The 3rd channel axis (machine axis name Z) must be given the actual value (SSI encoder) via the 5th axis (FM 354) and output the setpoint via the 5th axis (FM 354).

The following machine data must be parameterized for this purpose:

Actual value (SSI encoder):

MD 30210: $MA_ENC_SEGMENT_NR[0, AX3] = 2
MD 30220: $MA_ENC_MODULE_NR[0, AX3] = 1
MD 30220: $MA_ENC_INPUT_NR[0, AX3] = 1
MD 30220: $MA_ENC_TYPE[0, AX3] = 5 (SSI encoder)

Setpoint:

MD 30100: $MA_CTRLOUT_SEGMENT_NR[0, AX3] = 2
MD 30110: $MA_CTRLOUT_MODULE_NR[0, AX3] = 1
MD 30120: $MA_CTRLOUT_NR[0, AX3] = 1
MD 30130: $MA_CTRLOUT_TYPE[0, AX3] = 1

Data Fields, Lists 7

Data fields and lists are described in

References: /FB/, A4, “Digital and Analog NCK I/Os”

References: /FB/, G2, “Velocities, Setpoint/Actual Value Systems, Closed-Loop Control”

Notes

SINUMERIK 840D/810D/FM-NC Description of Functions, Extension Functions (FB2)

Kinematic Transformation (M1)

■

Notes

Brief Description

1

1.1 TRANSMIT

The functional range offered by TRANSMIT is as follows:

- Face-end machining on turned parts in the turning clamp
  - Holes
  - Contours
- A cartesian coordinate system can be used to program these machining operations.
- The control transforms the programmed traversing movements of the cartesian coordinate system into the traversing movements of the real machine axes (standard application):
  - Rotary axis (1)
  - Infeed axis perpendicular to axis of rotation (2)
  - Longitudinal axis in parallel to axis of rotation (3)
    Linear axes (2) and (3) are perpendicular to one another.
- Tool center offset relative to turning center is permissible.
- The velocity control makes allowance for the limitations defined for rotary motions.
- A path in the cartesian coordinate system must not pass through the turning center point (this restriction applies to SW 2 and 3).

Additional advantages from SW 4

- The tool center point path can pass through the turning center point of the rotary axis.
- The rotary axis does not need to be a modulo axis.

1.2 TRACYL

The functional scope of TRACYL (cylinder generated surface curve transformation) is as follows:

Machining of

- Longitudinal grooves on cylindrical objects,
- Transverse grooves on cylindrical objects,
- Arbitrary groove patterns on cylindrical objects.

The grooving path is programmed in relation to the developed, plane cylinder generated surface.

For machining purposes, the function supports lathes with

- X-C-Z kinematics and
- X-Y-Z-C kinematics.
- The control transforms the programmed traversing movements of the cylinder coordinate system into the traversing movements of the real machine axes (standard applications X-C-Z kinematics):
  - Rotary axis (1)
  - Infeed axis perpendicular to axis of rotation (2)
  - Longitudinal axis parallel to axis of rotation (3)

Note

Linear axes (2) and (3) are mutually perpendicular. The infeed axis (2) intersects the rotary axis. This constellation does not permit groove side compensation.

- Groove side compensation requires X-Y-Z-C kinematics with the following axes:
  - Rotary axis (1)
  - Infeed axis perpendicular to axis of rotation (2)
  - Longitudinal axis parallel to axis of rotation (3)
  - Longitudinal axis (4) to supplement (2) and (3) to obtain a right-hand cartesian coordinate system.

Note

Linear axes (2), (3) and (4) are perpendicular to one another. This constellation permits groove wall corrections.

- The velocity control makes allowance for the limitations defined for rotary motions.
  TRAANG (Inclined axis)

The “Inclined axis” function is provided for grinding applications.
Its functional scope is as follows:

- Machining with inclined infeed axis.
- A cartesian coordinate system can be used for programming purposes.
- The control transforms the programmed traversing movements of the cartesian coordinate system into the traversing movements of the real machine axes (standard application): Inclined infeed axis.

1.3 Chained transformations

Introduction

In SW 5 and higher two transformations can be chained such that the motion parts for the axes from the first transformation are input data for the chained second transformation. The motion parts from the second transformation act on the machine axes.

Chaining possibilities

- In SW 5 the chain may encompass **two** transformations.
- The **second** transformation must be **"Inclined axis"** (TRAANG).
- The first transformation can be:
  - Orientation transformations (TRAORI), incl. universal milling head
  - TRANSMIT
  - TRACYL
  - TRAANG

For details about chained transformations, please refer to Section 2.3, and for further information about other transformations to

References: /FB/, F2, "3 to 5-axis transformations".

1.4 Activating the transformation MD via part program/softkey (SW 5.2 and later)

SW 5.2 and later

Most existing machine data relevant to kinematic transformations are activated by POWER ON.

In SW 5.2 and later, transformation MD can also be activated via part program/softkey, i.e. the control system need not be booted.

Please refer to Section 2.4 for a detailed description.

■

Detailed Description

2

2.1 TRANSMIT

Note

The TRANSMIT transformation described below requires 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.
Only in this way can unambiguous assignments be made.

Task definition

Complete machining, see diagram.

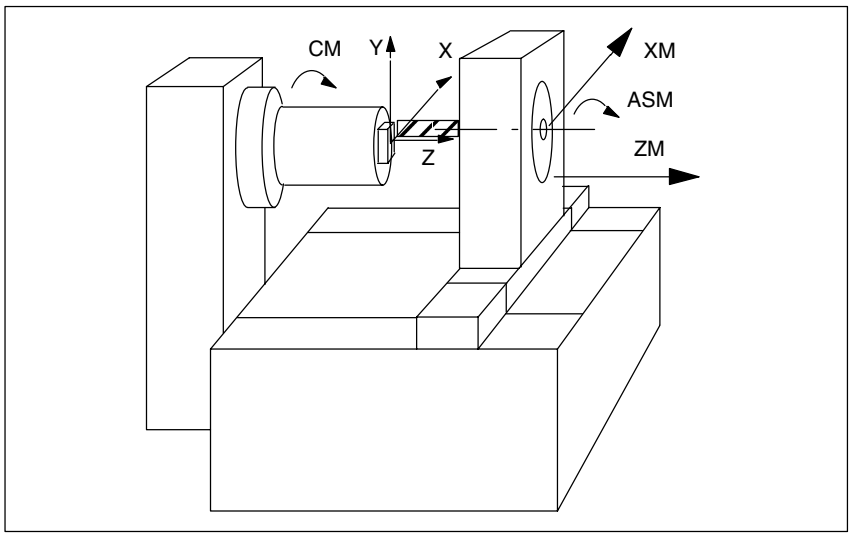


Fig. 2-1 End face machining of turned part

Legend:

CM	Rotary axis (main spindle)
ASM	Working spindle (milling cutter, drill)
X, Y, Z	Cartesian coordinate system for programming the face-end machining operation (origin in turning center point of face-end)
ZM	Machine axis (linear)
XM	Machine axis (linear)

2.1.1 Preconditions for TRANSMIT

Axis configuration

Before movements can be programmed in the Cartesian coordinate system (acc. to Fig. 2-1 X, Y, Z), the control system must be notified of the relationship between this coordinate system and the real machine axes (CM, XM, ZM, ASM):

- Assignment of names to geometry axes
- Assignment of geometry axes to channel axes
  - General situation (TRANSMIT not active)
  - TRANSMIT active
- Assignment of channel axes to machine axis numbers
- Identification of spindles
- Allocation of machine axis names

With the exception of the "– TRANSMIT active" point, the procedure is the same as for the normal axis configuration. If you already know the general steps, you need only read step "Assignment of geometry axes to channel axes" from the list of steps below.

References: /FB/, K2, "Coordinate Systems, Axis Types and Axis Configurations, Actual-Value System for Workpiece, External Zero Offset"

Number of transformations

Up to four (with Software Version 4 and higher: eight)transformation data blocks can be defined for each channel in the system. The machine data names of these transformations start with $MC_TRAFO .. and end with ..._n, "n" standing for a number between 1 and 8. The following sections include descriptions of these data:

$MC_TRAFO_TYPE_n
$MC_TRAFO_GEOAX_ASSIGN_TAB_n
$MC_TRAFO_AXES_IN_n.

Number of TRANSMIT structures

Two of the 8 permitted data structures for transformations in the channel may be assigned to the TRANSMIT function. They are characterized by the fact that the value assigned with $MC_TRAFO_TYPE_n is "256".
For these 2 TRANSMIT transformations, the following machine data must be set in a defined way:

$MC_TRANSMIT_ROT_AX_OFFSET_t
$MC_TRANSMIT_ROT_SIGN_IS_PLUS_t
$MC_TRANSMIT_BASE_TOOL_t
$MC_TRANSMIT_POLE_SIDE_FIX_t (SW 4 and higher)

In this case, "t" specifies the number of the declared TRANSMIT transformation (maximum of 2).

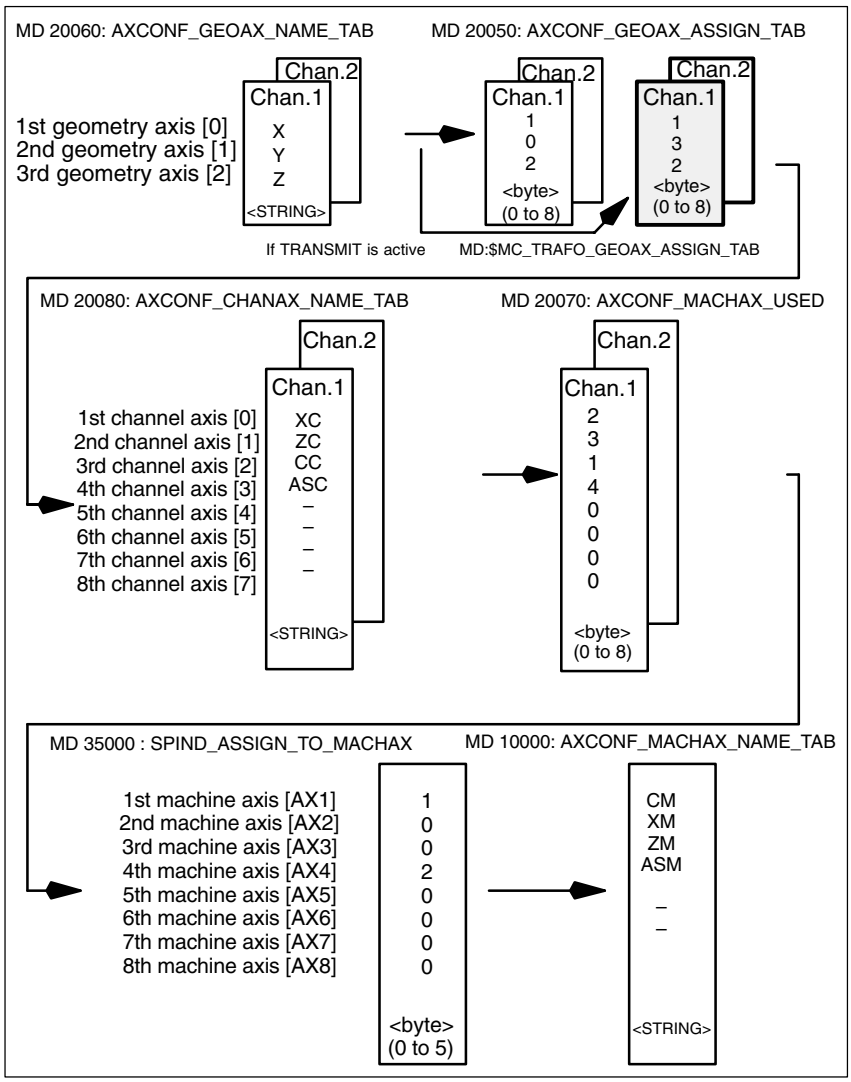


Fig. 2-2 Axis configuration for the example in Fig. 2-1.

The highlighted configuration in Fig. 2-2 applies when a TRANSMIT transformation is active.

Naming the geometry axes

According to the axis configuration overview shown above, the geometry axes to be involved in the TRANSMIT operation must be defined with:

$MC_AXCONF_GEOAX_NAME_TAB[0]="X"
" _TAB[1]="Y"
" _TAB[2]="Z"

(name selection according to Fig. 2-2, also corresponds to default setting).

Assignment of geometry axes to channel axes

These assignments are made according to whether or not TRANSMIT is active.

– TRANSMIT not active
  A Y axis is not available.

  $MC_AXCONF_GEOAX_ASSIGN_TAB[0]=1
  ” _TAB[1]=0
  ” _TAB[2]=2

– TRANSMIT active
  The Y axis can be addressed with the part program.
  $MC_TRAFO_GEOAX_ASSIGN_TAB_1[0]=1
  ” _TAB_1[1]=3
  ” _TAB_1[2]=2

  The Y axis is the third entry for the channel axes.

Entry of channel axes

Axes which do not belong to the Cartesian coordinate system are entered.

$MC_AXCONF_CHANAX_NAME_TAB[0]=“XC”
” [1]=“ZC”
” [2]=“CC”
” [3]=“ASC”

Assignment of channel axes to machine axes

With the cd of the channel axes as a reference, the machine axis number to which the channel axes have been assigned is transferred to the control system.

$MC_AXCONF_MACHAX_USED[0]=2
” [1]=3
” [2]=1
” [3]=4
” [4]=0

(entries corresponding to Fig. 2-2)

Identification of spindles

The user defines whether each machine axis is a spindle (value > 0: spindle number) or a path axis (value 0).

$MA_SPIND_ASSIGN_TO_MACHAX[0]=1
” [1]=0
” [2]=0
” [3]=2

Assignment of names to machine axes

With the cd of the machine axes as a reference, a machine axis name is transferred to the control system.

$MN_AXCONF_MACHAX_NAME_TAB[0]=“CM”
” [1]=“XM”
” [2]=“ZM”
” [3]=“ASM”

2.1.2 Settings specific to TRANSMIT

Type of transformation

The following paragraph describes how the transformation type is specified.

TRAFO_TYPE_n

The user must specify the transformation type for the transformation data blocks (maximum n = 8). A value of 256 must be set for TRANSMIT. Example:

$MC_TRAFO_TYPE_1=256

The setting must be made and the values activated with Power On before TRANSMIT or TRANSMIT(t) is called. “t” is the number of the declared TRANSMIT transformation.

Axis image

The following paragraph describes how the transformation axis image is specified.

TRAFO_AXES _IN_n

Three channel axis numbers must be specified for the transformation data block n:

$MC_TRAFO_AXES_IN_1[0]=	Channel axis number of axis perpendicular to rotary axis
$MC_TRAFO_AXES_IN_1[1]=	Channel axis number of rotary axis
$MC_TRAFO_AXES_IN_1[2]=	Channel axis number of axis parallel to Rotary axis

Example for the configuration in Fig. 2-1:
$MC_TRAFO_AXES_IN_1[0]=1
$MC_TRAFO_AXES_IN_1[1]=3
$MC_TRAFO_AXES_IN_1[2]=2

The setting must be made before TRANSMIT or TRANSMIT(t) is called. The axis numbers must refer to the channel axis sequences defined with $MC_TRAFO_GEOAX_ASSIGN_TAB_n.

Rotational position

The rotational position of the Cartesian coordinate system is specified by machine data as described in the following paragraph.

TRANSMIT _ROT_AX _OFFSET_t

The rotational position of the x-y plane of the Cartesian coordinate system in relation to the defined zero position of the rotary axis is specified with:

$MC_TRANSMIT_ROT_AX_OFFSET_t= ... ; degrees

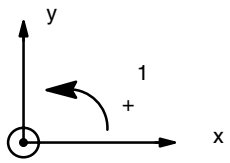
In this case, “t” is substituted by the number of the TRANSMIT transformations declared in the transformation data blocks
(t may be maximum 2).

Direction of rotation

The direction of rotation of the rotary axis is specified by machine data as described in the following paragraph.

TRANSMIT _ROT_SIGN _IS_PLUS_t

If the rotary axis rotates in an anti-clockwise direction on the x-y plane when viewed along the z axis, then the machine axis must be set to “1”, but otherwise to “0”.



$MC_TRANSMIT_ROT_SIGN_IS_PLUS_t=1

In this case, “t” is substituted by the number of the TRANSMIT transformations declared in the transformation data blocks (t must not be more than 2).

Position of tool zero

The position of the tool zero point is specified by machine data as described in the following paragraph.

TRANSMIT_ BASE_TOOL_t

Machine data $MC_TRANSMIT_BASE_TOOL_t is used to inform the control of the position of the tool zero point in relation to the origin of the coordinate system declared for TRANSMIT. The machine data has three components for the three axes of the Cartesian coordinate system.

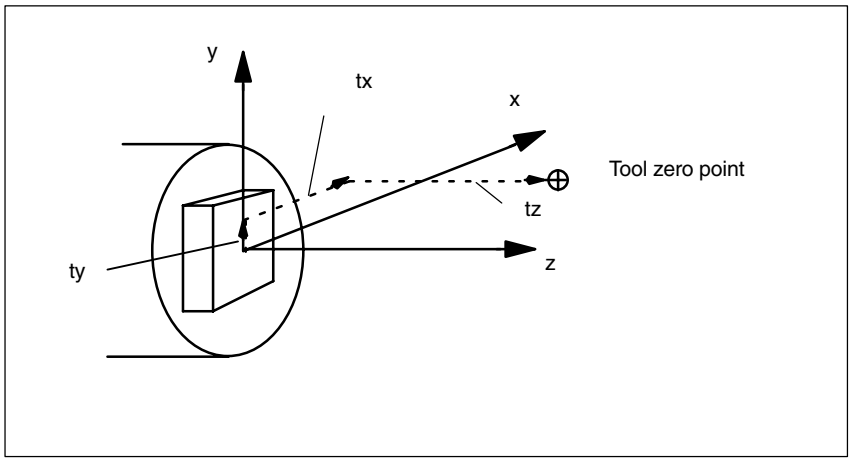


Fig. 2-3 Position of tool zero in relation to origin of the Cartesian coordinate system (see Fig. 2-1).

$MC_TRANSMIT_BASE_TOOL_t[0]=tx
" [1]=ty
" [2]=tz

In this case, t in front of the index value [ ] is replaced by the number of the TRANSMIT transformations declared in the transformation data blocks.
(t must not be more than 2).

Replaceable geometry axes

The PLC is informed when a geometry axis has been replaced using GEOAX( ) through the optional output of an M code that can be set in machine data.

- MD 22534: TRAFO_CHANGE_M_CODE

  Number of the M code that is output at the VDI interface in the case of transformation changeover.

Note

If this machine data is set to one of the values 0 to 6, 17, 30, then no M code is output.

References: /FB/, K2, “Coordinate Systems, Axis Types and Axis Configurations, Actual-Value System for Workpiece, External Zero Offset”

2.1.3 Activation of TRANSMIT

TRANSMIT

After the settings described above have been made, the TRANSMIT function can be activated:

TRANSMIT or
TRANSMIT(t)

The first declared TRANSMIT function is activated with TRANSMIT.
TRANSMIT(t) activates the t-th declared TRANSMIT function. t may not be more than 2.
When TRANSMIT is activated in SW 4 and later, the special processes for pole traversal, etc. according to 2.1.6 become available.

Between activation of the function and deactivation as described below, the traversing movements for the axes of the Cartesian coordinate system can be programmed.

2.1.4 Deactivation of TRANSMIT function

TRAFOOF

Keyword TRAFOOF deactivates an active transformation. When the transformation is deactivated, the base coordinate system is again identical to the machine coordinate system.
An active TRANSMIT transformation is likewise deactivated if one of the other transformations (e.g. TRACYL, TRAANG, TRAORI) is activated in the appropriate channel.

References: /FB/, F2, “3–5-Axis Transformation”.

2.1.5 Special system reactions with TRANSMIT

The transformation can be selected and deselected via part program or MDA.

Please note on selection

- An intermediate motion block is not inserted (phases/radii).
- A spline block sequence must be terminated.
- Tool radius compensation must be deselected.
- The frame which was active prior to TRANSMIT is deselected by the control. (Corresponds to “Reset programmed frame” G500).
- An active working area limitation is deselected by the control for the axes affected by the transformation (corresponds to programmed WALIMOF).
- Continuous path control and rounding are interrupted.
- DRF offsets in transformed axes must have been deleted by the operator.

Please note on deselection

- An intermediate motion block is not inserted (phases/radii).
- A spline block sequence must be terminated.
- Tool radius compensation must be deselected.
- The frame which was active prior to TRANSMIT is deselected by the control. (Corresponds to “Reset programmed frame” G500).
- An activated tool length compensation is incorporated into the transformation by the control.
- Continuous path control and rounding are interrupted.
- DRF offsets in transformed axes must have been deleted by the operator.
- Tool length compensation in the virtual axis (the Y axis in Fig. 2-1 is not implemented.

Restrictions imposed by active TRANSMIT

The restrictions listed below imposed by an activated TRANSMIT function must be noted.

Tool change

Tools may only be changed when the tool radius compensation function is deselected.

Frame

All instructions which refer exclusively to the base coordinate system are permissible (FRAME, tool radius compensation). Unlike the procedure for inactive transformation, however, a frame change with G91 (incremental dimension) is not specially treated. The increment to be traversed is evaluated in the workpiece coordinate system of the new frame – regardless of which frame was effective in the previous block.

Rotary axis

The rotary axis cannot be programmed because it is occupied by a geometry axis and cannot thus be programmed directly as a channel axis.

Pole

SW up to and including 3.x:
Movements through the pole (origin of Cartesian coordinate system) are disabled, i.e. a movement which traverses the pole is stopped in the pole followed by the output of an alarm. In the case of a cutter center offset, the movement is terminated accordingly at the end of the non-approachable area.
SW 4 and higher:
The options for pole traversal and machining operations close to the pole are described in the sections starting at 2.1.6.

Exceptions

Axes affected by the transformation cannot be used

- as a preset axis (alarm)
- to approach the fixed point (alarm)
- for referencing (alarm)

Note

TRANSMIT must be deselected before the tool radius compensation and length compensation are deselected.

Velocity control

Velocity monitoring with TRANSMIT is implemented as standard at the preprocessing stage. Monitoring and limitation in the main run are activated:

- In AUTOMATIC mode if a positioning or oscillation axis has been programmed which is included in the transformation via machine data $MC_TRAFO_AXES_IN_n index 0 or 1.
- On changeover to JOG mode.

The monitoring function is transferred from the main run back to the preprocessing routine if the axes relevant to the transformation process are operated as path axes.

The velocity monitoring function in preprocessing utilizes the machine better than the monitoring in the main run. Furthermore, the main run monitoring function deactivates the Look Ahead.

Interruption of part program

If part program processing is interrupted for JOG, then the following must be noted:

JOG

When JOG is selected, the conventional on-line velocity check is activated instead of the optimized velocity check provided in 2.1.6 SW version 4.

From AUTOMATIC to JOG

If part program processing is interrupted when the transformation is active followed by traversal in JOG mode, then the following must be noted when AUTOMATIC is selected again:

- The transformation is active in the approach block from the current position to the point of interruption. No monitoring for collisions takes place.

Warning

The operator is responsible for ensuring that the tool can be re-positioned without any difficulties.

In AUTOMATIC mode

The velocity-optimized velocity planning function (Software Version 4) remains active for as long as the axes relevant to the transformation are traversed in mutual synchronism as path axes. If an axis involved in the transformation is traversed as a positioning axis, the online velocity check remains active until the transformation is deactivated or until all axes involved in the transformation are operating as path axes again. The return to velocity-optimized operation according to 2.1.6 automatically initiates a STOPRE and synchronizes acyclic block preprocessing with the interpolation routine.

From start after reset

If part program processing is aborted with RESET and restarted with START, then the following must be noted:

- The remaining part program is traversed reproducibly only if all axes are traversed to a defined position by means of a linear block (G0 or G1) at the beginning of the part program. A tool which was active on RESET may no longer be taken into account by the control (settable via machine data).

Power On RESET

The system response after Power ON is determined by the settings stored in
MD 20110: RESET_MODE_MASK and
MD 20140: TRAFO_RESET_VALUE.

References: /FB/, K2, “Actual-Value System for Workpiece”

Reference point approach

Axes cannot be referenced when a transformation is active. Any active transformation is deselected by the control system during a referencing operation.

2.1.6 Machining with TRANSMIT using SW 4.x and higher

Introduction

The TRANSMIT transformation has a pole at the zero point of the TRANSMIT plane (example, see Fig.: 2-1, x = 0, Y = 0). The pole is located on the intersection between the radial linear axis and the rotary axis (X and CM). In the vicinity of the pole, small positional changes in the geometry axes generally result in large changes in position in the machine rotary axis. The only exceptions are linear motions into or through the pole.

With SW 4 and higher, a tool center point path through the pole does not cause the part program to be aborted. There are no restrictions with respect to programmable traversing commands or active tool radius compensations. Nevertheless, workpiece machining operations close to the pole are not recommended since these may require sharp feedrate reductions to prevent overloading of the rotary axis.

New features

Definition:
A pole is said to exist if the line described by the tool center point intersects the turning center of the rotary axis.

The following cases are examined:

- Under what conditions and by what methods the pole can be traversed
- The response in pole vicinity
- The response with respect to working area limitations
- Monitoring of rotary axis rotations over 360°.

Pole traversal

The pole can be traversed by two methods:

- Traversal along linear axis
- Traversal into pole with rotation of rotary axis in pole

Traversal along linear axis

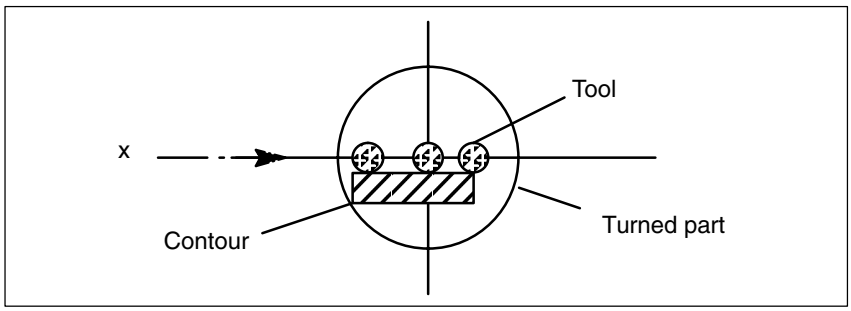


Fig. 2-4 Traversal of x axis through pole

Rotation in pole

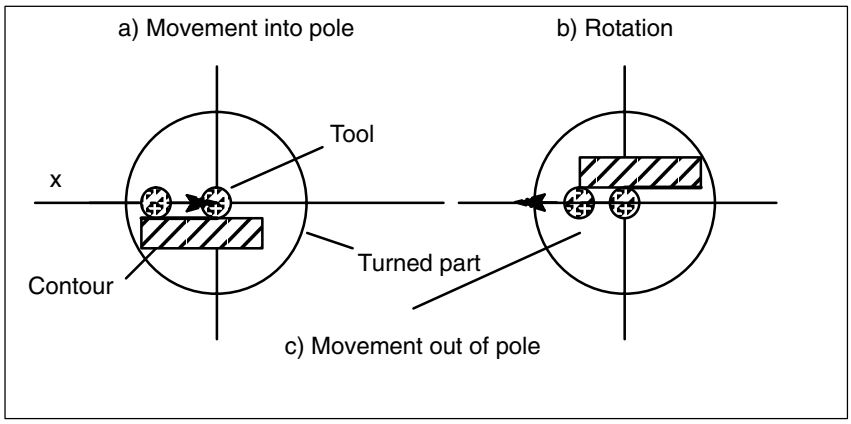


Fig. 2-5 Traversal of x axis into pole (a), rotation (b), exit from pole (c)

Selection of method

The method must be selected according to the capabilities of the machine and the requirements of the part to be machined. The method is selected by machine data:

MD 24911: TRANSMIT_POLE_SIDE_FIX_1
MD 24951: TRANSMIT_POLE_SIDE_FIX_2

The first MD applies to the first TRANSMIT transformation in the channel and the second MD correspondingly to the second TRANSMIT transformation.

Table 2-1

VALUE	Meaning
0	Pole traversal The tool center point path (linear axis) must traverse the pole on a continuous path.
1	Rotation around pole. The tool center point path must be restricted to the positive traversing range of the linear axis (in front of turning center).
2	Rotation around pole. The tool center point path must be restricted to the negative traversing range of the linear axis (behind turning center).

Special features relating to pole traversal

The method of pole traversal along the linear axis may be applied in the AUTOMATIC and JOG modes.

System response:

Table 2-2 Traversal of pole along the linear axis

Operating mode	Status	Reaction
AUTOMATIC	All axes involved in the transformation are moved synchronously. TRANSMIT active.	High-speed pole traversal
	Not all axes involved in the transformation are traversed synchronously (e.g. positioning axis). TRANSMIT not active.	Traversal of pole at creep speed
	An applied DRF (external zero offset) does not interfere with the operation. Servo errors may occur close to the pole during application of a DRF.	Abortion of machining operation, alarm
JOG	–	Traversal of pole at creep speed

Special features relating to rotation in pole

Precondition: This method is only effective in the AUTOMATIC mode.

MD 24911: TRANSMIT_POLE_SIDE_FIX_1 = 1 or 2
MD 24951: TRANSMIT_POLE_SIDE_FIX_2 = 1 or 2
Value: 1 Linear axis remains within positive traversing range
Value: 2 Linear axis remains within negative traversing range

In the case of a contour that would require the pole to be traversed along the tool center point path, the following three steps are taken to prevent the linear axis from traversing in ranges beyond the turning center:

Step	Action
1	Linear axis traverses into pole
2	Rotary axis turns through 180°, the other axes involved in the transformation remain stationary.
3	Execution of remaining block. The linear axis now exits from the pole again.

In JOG mode, the motion stops in the pole. In this mode, the axis may exit from the pole only along the path tangent on which it approached the pole. All other motion instructions would require a step change in the rotary axis position or a large machine motion in the cases of minimum motion instructions. They are rejected with alarm 21619.

Traversal close to pole

If a tool center point traverses past the pole, the control system automatically reduces the feedrate and path acceleration rate such that the settings of the machine axes (MD 32000: MAX_AX_VELO[AX*] and MD 32300: MAX_AX_ACCEL[AX*]) are not exceeded. The closer the path is to the pole, the greater the reduction in the feedrate.

Tool center point path with corner in pole

A tool center point path which includes a corner in the pole will not only cause a step change in axis velocities, but also a step change in the rotary axis position. These cannot be reduced by decelerating.

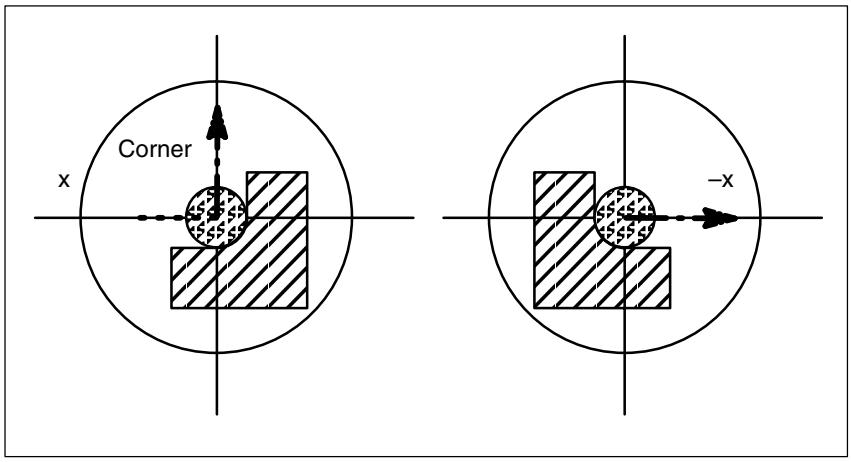


Fig. 2-6 Pole traversal

Preconditions:
AUTOMATIC mode, MD 24911: TRANSMIT_POLE_SIDE_FIX_1 = 0 or
MD 24951: TRANSMIT_POLE_SIDE_FIX_2 = 0

The control system inserts a traversing block at the step change point. This block generates the **smallest possible rotation** to allow machining of the contour to continue.

Corner without pole traversal

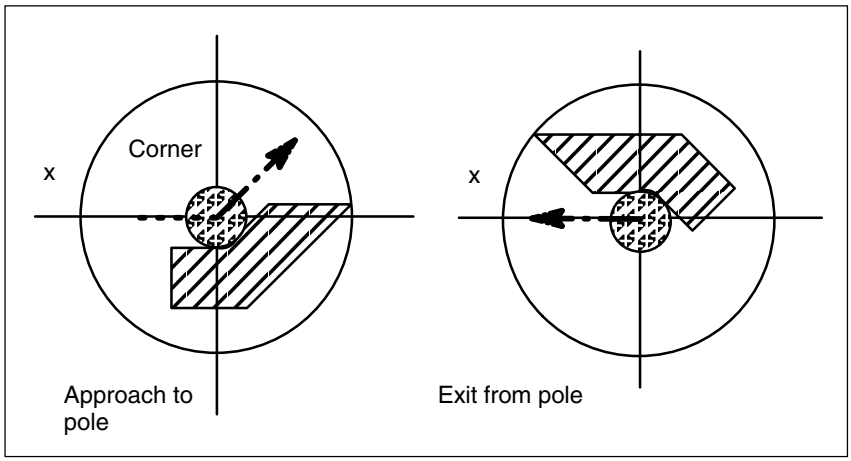


Fig. 2-7 Machining on one pole side

Preconditions:
AUTOMATIC mode,
MD 24911: TRANSMIT_POLE_SIDE_FIX_1 = 1 or 2 or
MD 24951: TRANSMIT_POLE_SIDE_FIX_2 = 1 or 2

The control system inserts a traversing block at the step change point. This block generates the **necessary rotation** so that machining of the contour can continue on the same side of the pole.

Transformation selection in pole

If the machining operation must continue from a position on the tool center path which corresponds to the pole of the activated transformation, then an exit from the pole is specified for the new transformation.

If MD 24911: TRANSMIT_POLE_SIDE_FIX_1= 0 or
MD 24951: TRANSMIT_POLE_SIDE_FIX_2= 0 is set (pole traversal), then the **smallest possible** rotation is generated at the beginning of the block that implements exit from the pole. Depending on this rotation, the axis then traverses either in front of or behind the turning center.

When MD 24911: TRANSMIT_POLE_SIDE_FIX_1= 1 or
MD 24951: TRANSMIT_POLE_SIDE_FIX_2= 1 machining continues **in front of** the turning center (linear axis in positive control range);
When MD 24911: TRANSMIT_POLE_SIDE_FIX_1= 2 or
MD 24951: TRANSMIT_POLE_SIDE_FIX_2= 2 machining is **behind** the turning center (linear axis in the negative control range).

Transformation selection outside pole

The control system moves the axes involved in the transformation without evaluating machine data $MC_TRANSMIT_POLE_SIDE_FIX_t. In this case, t = 1 stands for the first and t = 2 for the second TRANSMIT transformation in the channel.

2.1.7 Working area limitations

Initial state

When TRANSMIT is active, the pole is replaced by a working area limitation if the tool center point cannot be positioned in the turning center of the rotary axis involved in the transformation. This is the case when the axis perpendicular to the rotary axis (allowing for tool offset) is not positioned on the same radial plane as the rotary axis or if both axes are positioned mutually at an oblique angle. The distance between the two axes defines a cylindrical space in the BCS in which the tool cannot be positioned.

The illegal range cannot be protected by the software limit switch monitoring function since the traversing range of the machine axes is not affected.

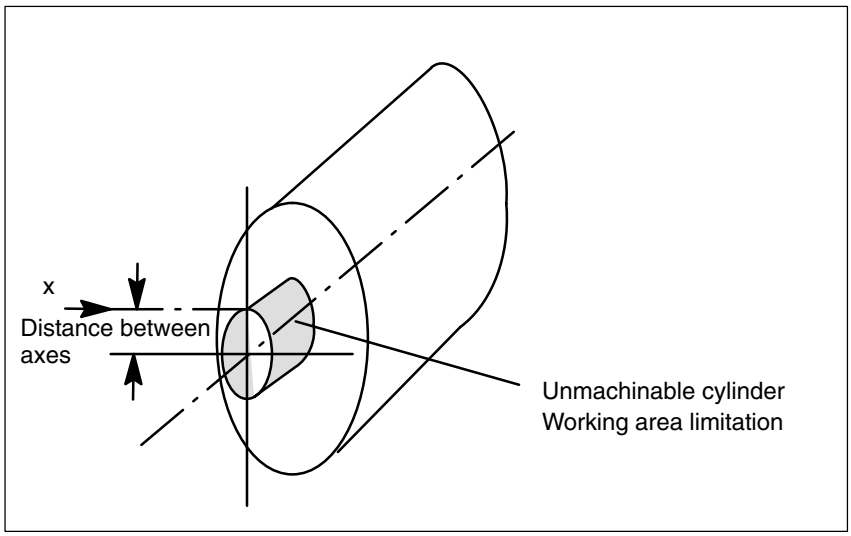


Fig. 2-8 Working area limitation based on offset linear axis

Traversal into working area limitation

Any motion that leads into the working area limitation is rejected with alarm 21619. Any corresponding part program block is not processed. The control system stops processing at the end of the preceding block.

If the motion cannot be foreseen promptly enough (JOG modes, positioning axes), then the control stops at the edge of the working area limitation.

Response close to working area limitation

If a tool center point path leads past the illegal range, the control automatically reduces the feedrate and path acceleration rate to ensure that the settings of the machine axes (MD 32000: MAX_AX_VELO[AX*] and MD 32300: MAX_AX_ACCEL[AX*]) are not exceeded. The closer the path is to the working area limitation, the greater the reduction in the feedrate may be.

2.1.8 Overlaid movements with TRANSMIT in SW 4

The control system cannot predict overlaid motions. However, these do not interfere with the function provided that they are very small (e.g. fine tool offset) in relation to the current distance from the pole (or from working area limitation). With respect to axes that are relevant for the transformation, the transformation monitors the overlaid motion and signals any critical quantity by alarm 21618. This alarm indicates that the block-related velocity planning function no longer adequately corresponds to the actual conditions on the machine. When the alarm is output, the conventional, non-optimized online velocity monitor is therefore activated. The preprocessing routine is re-synchronized with the main run by a REORG generated internally in the control.

Alarm 21618 should be avoided whenever possible since it indicates a state that can lead to axis overload and thus abortion of part program processing.

2.1.9 Monitoring of rotary axis rotations over 360°

Ambiguity of rotary axis positions

The positions of the rotary axis are ambiguous with respect to the number of rotations. The control breaks down blocks containing several rotations around the pole into sub-blocks.
This subdivision must be noted with respect to parallel actions (e.g. output of auxiliary functions, block-synchronized positioning axis motions) since the programmed block end is no longer relevant for synchronization, but the end of the first sub-block instead. See:

References: /FB/, H2, “Output of Auxiliary Functions to PLC”
/FB/, S5, “Synchronized Actions”

In single block mode, the control processes individual blocks explicitly. In other modes, the sub-blocks are traversed with Look Ahead like a single block. A limitation of the rotary axis setting range is monitored by the software limit switch monitoring function.

2.1.10 Supplementary Conditions

Look Ahead

All functions requiring Look Ahead (traversal through pole, Look Ahead) work satisfactorily only if the relevant axis motions can be calculated exactly in advance. With TRANSMIT, this applies to the rotary axis and the linear axis perpendicular to it. If one of these axes is the positioning axis, then the Look Ahead function is deactivated by alarm 10912 and the conventional online velocity check activated instead.

Selection of method

The **user is responsible** for making the optimum choice of “Traversal through pole” or “Rotation around pole”. The active prevention of axis traversal through the pole implemented in SW 2 and 3 has been eliminated in SW 4.

Several pole traversals

A block can traverse the pole any number of times (e.g. programming of a helix with several turns). The part program block is subdivided into a corresponding number of sub-blocks. Analogously, blocks which rotate several times around the pole are likewise divided into sub-blocks. The relevant restrictions applying in SW 2 and 3 have been eliminated in SW 4.

Rotary axis as modulo axis

The rotary axis can be a modulo rotary axis. However, this is not a mandatory requirement as was the case in SW 2 and 3. The relevant restrictions applying in SW 2 and 3 have been eliminated in SW 4.

Rotary axis as spindle

If the rotary axis without transformation is used as a spindle, it must be switched to position-controlled mode with SPOS before the transformation is selected.

REPOS

It is possible to reposition on the sub-blocks produced as a result of the extended TRANSMIT function in SW 4. In this case, the control uses the first sub-block that is closest to the repositioning point in the BCS.

Block search

In the case of block search with calculation, the block end point (of the last sub-block) is approached in cases where intermediate blocks have been generated as the result of the extended functionality in SW 4.

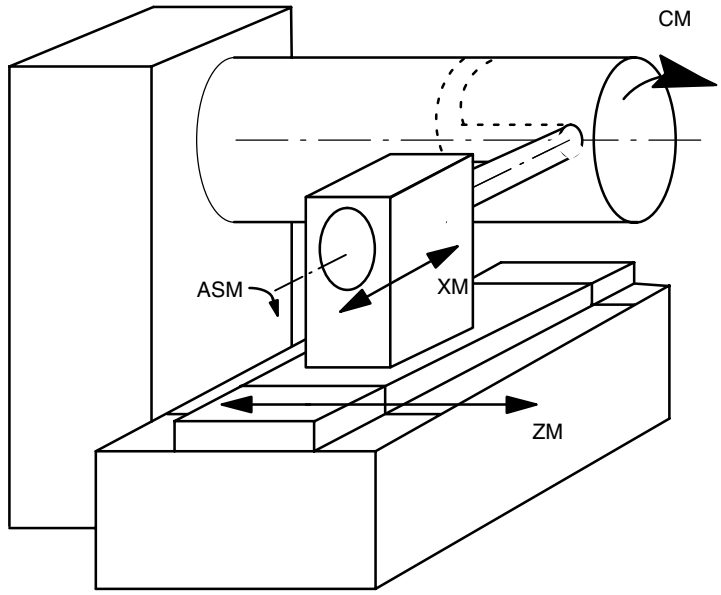
2.2 TRACYL

Note

The TRACYL transformation described below requires that unique 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.
Only in this way can unambiguous assignments be made.

Task definition

Groove machining, see diagram.



Legend:

XM	Infeed axis, perpendicular to rotary axis
ZM	Axis parallel to rotary axis
CM	Rotary axis
ASM	Working spindle

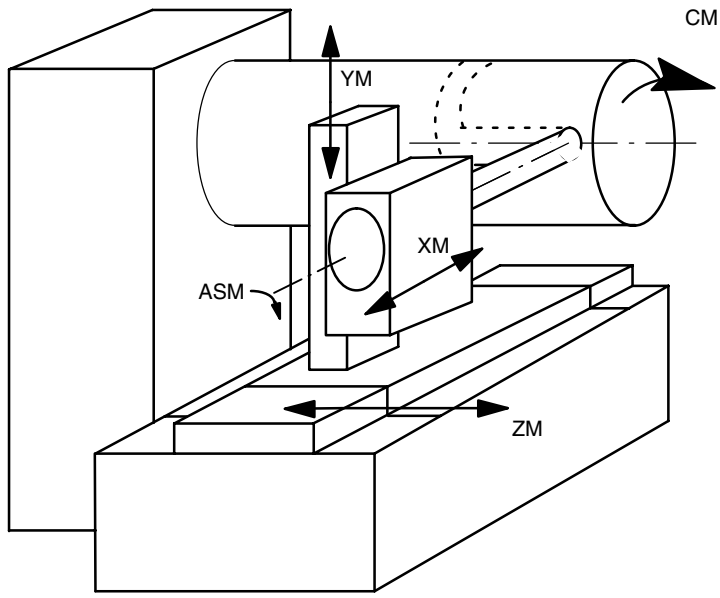
Fig. 2-9 Machining grooves on generated cylinder surface with X-C-Z kinematics

Axis configuration 1

The generated cylinder surface curve transformation allows a traversing path to be specified with respect to the generated surface of a cylinder coordinate system. The machine kinematics must correspond to the cylinder coordinate system. It must include one or two linear axes and a rotary axis. The two linear axes must be mutually perpendicular. The rotary axis must be aligned in parallel to one of the linear axes and intersect the second linear axis. In addition, the rotary axis must be co-linear to the cylinder coordinate system.

If there is only one linear axis (X), only grooves which are parallel to the periphery of the cylinder can be generated. In the case of two linear axes (X, Z), the groove pattern on the cylinder is optional. See Fig. 2-9.

Axis configuration 2



Legend:

XM	Infeed axis, perpendicular to rotary axis
YM	Supplementary axis
ZM	Axis parallel to rotary axis
CM	Rotary axis
ASM	Working spindle

Fig. 2-10 Machining grooves on generated cylinder surface with X-Y-Z-C kinematics

If a third linear axis is available (Fig. 2-10) which can produce a right-handed Cartesian coordinate system with the other two linear axes (axis configuration 1), then it is used to offset the tool **parallel to the programmed path** by means of tool radius compensation. thereby allowing grooves with rectangular traversing section to be generated.

Functionality

During transformation (both axis configurations), the full functionality of the control is available, both for processing from the NC program and in JOG mode (see 2.2.6).

Groove cross-section

In the case of axis configuration 1, longitudinal grooves along the rotary axis are subject to parallel limits only if the groove width corresponds exactly to the tool radius.

Grooves in parallel to the periphery (transverse grooves) are not parallel at the beginning and end.

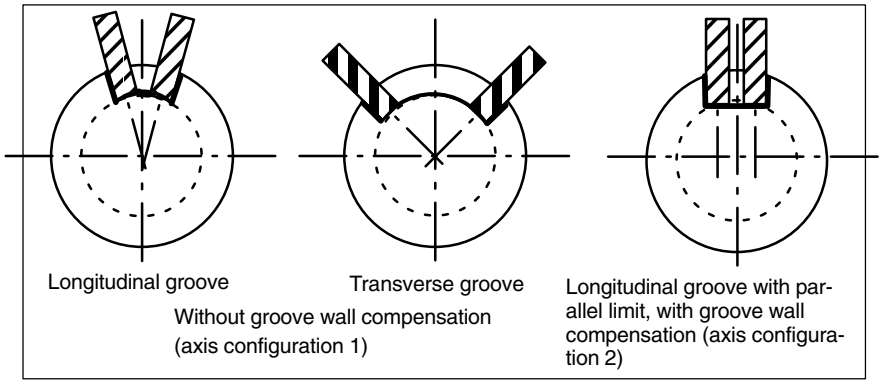


Fig. 2-11 Grooves with and without groove wall compensation

2.2.1 Preconditions for TRACYL

Number of transformations

Up to 8 transformation data blocks can be defined for each channel in the system. The machine data names of these transformations start with $MC_TRAFO .. and end with ..._n, “n” standing for a number between 1 and 8. The first machine data has the same meaning as described for TRANSMIT:

$MC_TRAFO_GEOAX_ASSIGN_TAB_n
$MC_TRAFO_TYPE_n
$MC_TRAFO_AXES_IN_n.

The special settings described below apply to $MC_TRAFO_TYPE_n and $MC_TRAFO_AXES_IN_n with respect to generated cylinder surface transformation (TRACYL).

Number of TRACYL structures

Two of the 8 permitted data structures for transformations may be assigned to the TRACYL function. They are characterized by the fact that the value assigned with $MC_TRAFO_TYPE_n is 512 or 513.
For these TRACYL transformations (maximum of 2), the following machine data must be set in a defined manner:

$MC_TRACYL_ROT_AX_OFFSET_t
$MC_TRACYL_ROT_SIGN_IS_PLUS_t
$MC_TRACYL_BASE_TOOL_t

In this case, t specifies the number of the declared TRACYL transformation (maximum of 2).

Axis configuration

The following overview shows the relationship between the axes of the machine illustrated in Fig. 2-10 and the relevant axis data.

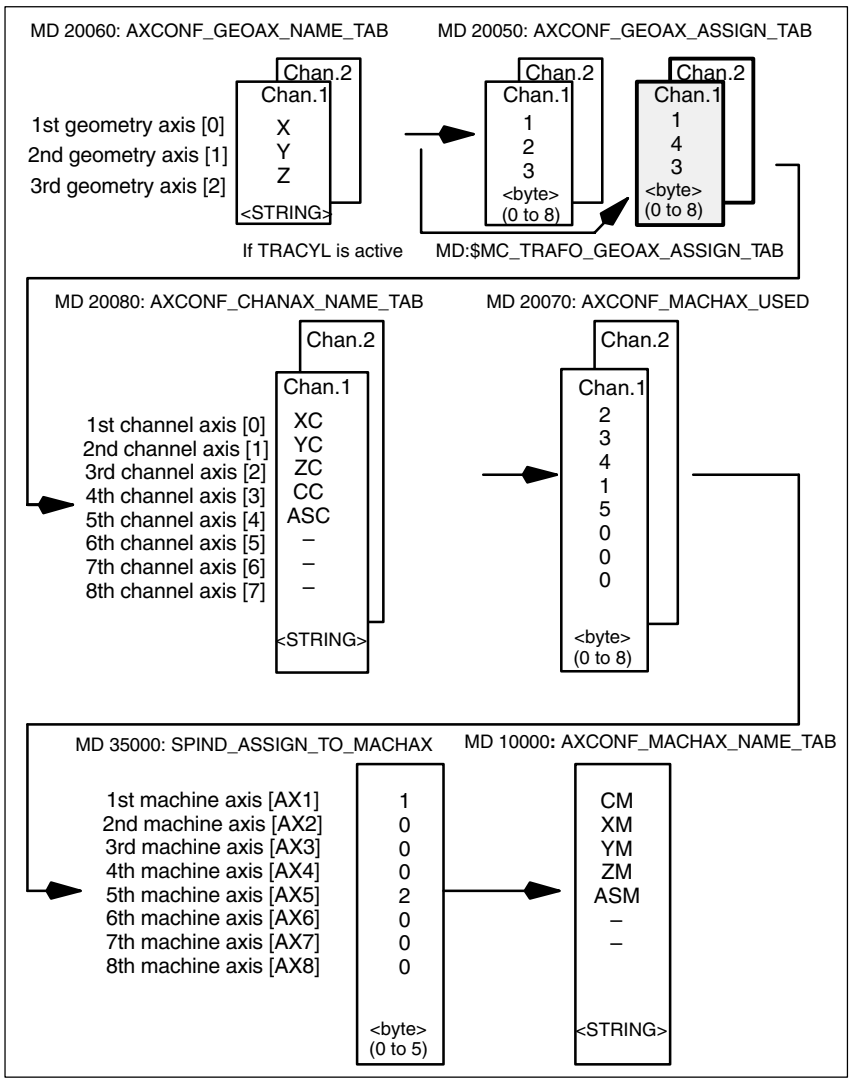


Fig. 2-12 Axis configuration for the example in Fig. 2-10

The highlighted configuration in Fig. 2-12 applies when TRACYL is active.

Naming the geometry axes

According to the above axis configuration overview, the geometry axes to be involved in the TRACYL operation must be defined with:

$MC_AXCONF_GEOAX_NAME_TAB[0]="X"
" _TAB[1]="Y"
" _TAB[2]="Z"

(name selection according to Fig. 2-12, also corresponds to default setting).

Assignment of geometry axes to channel axes

These assignments are made depending on whether or not TRACYL is active.

– TRACYL not active
  Normal traversal of Y axis.

  $MC_AXCONF_GEOAX_ASSIGN_TAB[0]=1
  ” _TAB[1]=2
  ” _TAB[2]=3

– TRACYL active
  The Y axis becomes the axis in the direction of the cylinder coordinate system in which the peripheral surface is generated.
  $MC_TRAFO_GEOAX_ASSIGN_TAB[0]=1
  ” _TAB[1]=4
  ” _TAB[2]=3

Entry of channel axes

Axes which do not belong to the Cartesian coordinate system are added.

$MC_AXCONF_CHANAX_NAME_TAB[0]=“XC”
” _TAB[1]=“YC”
” _TAB[2]=“ZC”
” _TAB[3]=“CC”
” _TAB[4]=“ASC”

Assignment of channel axes to machine axes

With the cd of the channel axes as a reference, the machine axis number to which the channel axes have been assigned is transferred to the control system.

$MC_AXCONF_MACHAX_USED[0]=2
” [1]=3
” [2]=4
” [3]=1
” [4]=5

(entries corresponding to Fig. 2-10)

Identification of spindles

The user defines whether each machine axis is a spindle (value > 0: spindle number) or a path axis (value 0).

$MA_SPIND_ASSIGN_TO_MACHAX[0]=1
” [1]=0
” [2]=0
” [3]=0
” [4]=2

Assignment of names to machine axes

With the cd of the machine axes as a reference, a machine axis name is transferred to the control.

$MN_AXCONF_MACHAX_NAME_TAB[0]=“CM”
” [1]=“XM”
” [2]=“YM”
” [3]=“ZM”
” [4]=“ASM”

2.2.2 TRACYL-specific settings

Type of transformation

The following paragraph describes how the transformation type is specified.

TRAFO_TYPE_n

The user must specify the transformation type for the transformation data blocks (maximum n = 8). For TRACYL, a value of 512 must be set for axis configuration 1 and a value of 513 for axis configuration 2. Example:

MD 24100: TRAFO_TYPE_1=512
The setting must be made and activated with Power ON before TRACYL(d,t) is called. “t” is the number of the declared TRACYL transformation.

Axis image

The following paragraph describes how the transformation axis image is specified.

TRAFO_AXES _IN_n

Three (or 4) channel axis numbers must be specified for the transformation data block n:

$MC_TRAFO_AXES_IN_1[0]=	Channel axis number of axis radial to rotary axis
$MC_TRAFO_AXES_IN_1[1]=	Channel axis number of rotary axis
$MC_TRAFO_AXES_IN_1[2]=	Channel axis number of axis parallel to rotary axis
$MC_TRAFO_AXES_IN_1[3]=	Channel axis number of additional axis, parallel to generated cylinder surface of cylinder and perpendicular to rotary axis (if axis configuration 2 is selected)

Example according to Fig. 2-10:
$MC_TRAFO_AXES_IN_1[0]=1
$MC_TRAFO_AXES_IN_1[1]=4
$MC_TRAFO_AXES_IN_1[2]=3
$MC_TRAFO_AXES_IN_1[3]=2

The setting must be made and activated with Power ON before TRACYL(d) or TRACYL(d,t) is called. The axis numbers must refer to the channel axis sequences defined with $MC_TRAFO_GEOAX_ASSIGN_TAB_n.

Rotational position

The rotational position of the axis on the cylinder peripheral surface perpendicular to the rotary axis must be defined as follows:

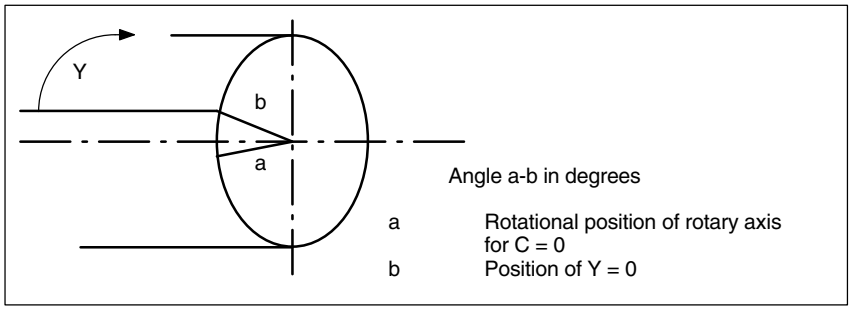


Fig. 2-13 Center of rotation of axis on generated cylinder surface

TRACYL _ROT_AX _OFFSET_t

The rotational position of the peripheral surface in relation to the defined zero position of the rotary axis is specified with:

$MC_TRACYL_ROT_AX_OFFSET_t= ... ; degrees

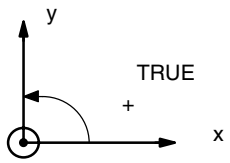
In this case, “t” is substituted by the number of the TRACYL transformations declared in the transformation data blocks (t must not be more than 2).

Direction of rotation

The direction of rotation of the rotary axis is specified by machine data as described in the following paragraph.

TRACYL _ROT_SIGN _IS_PLUS_t

If the direction of rotation of the rotary axis on the x-y plane is counter-clockwise when viewed against the z axis, then the machine data must be set to TRUE, otherwise to FALSE.



$MC_TRACYL_ROT_SIGN_IS_PLUS_t=TRUE

In this case, “t” is substituted by the number of the TRACYL transformations declared in the transformation data blocks (t must not be more than 2).

Switchable geometry axes

The PLC is informed when a geometry axis has been replaced using GEOAX( ) through the optional output of an M code that can be set in machine data.

- MD 22534: TRAFO_CHANGE_M_CODE

  Number of the M code that is output at the VDI interface in the case of transformation changeover.

Note

If this machine data is set to one of the values 0 to 6, 17, 30, then no M code is output.

References: /FB/, K2, “Coordinate Systems, Axis Types, Axis Configurations, Actual-Value System for Workpiece External Zero Offset”

Position of tool zero

The position of the tool zero point in relation to the origin of the Cartesian coordinate system is specified by machine data as described in the following paragraph.

TRACYL_ BASE_TOOL_t

Machine data $MC_TRACYL_BASE_TOOL_t is used to inform the control of the position of the tool zero point in relation to the origin of the cylinder coordinate system declared for TRACYL. The machine data has three components for the axes X, Y, Z of the machine coordinate system.

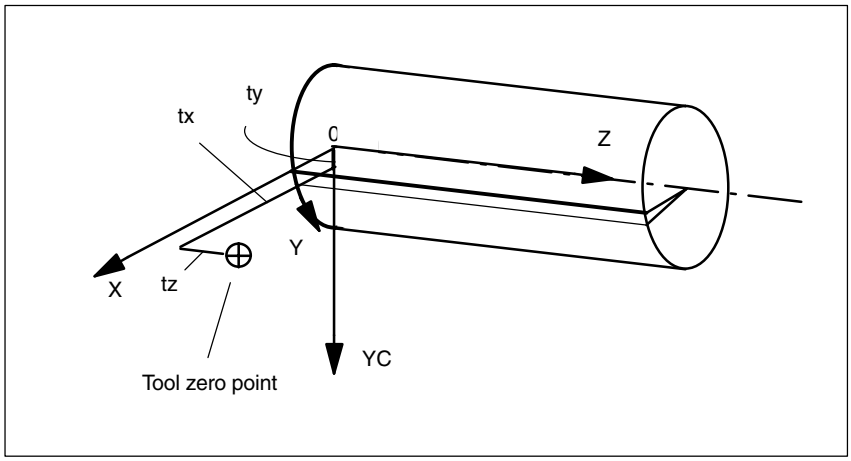


Fig. 2-14 Position of tool zero in relation to machine zero (see Fig. 2-10)

Example:
$MC_TRACYL_BASE_TOOL_t[0]=tx
" [1]=ty
" [2]=tz
In this case, “t” is substituted by the number of the TRACYL transformations declared in the transformation data blocks
(t must not be more than 2).

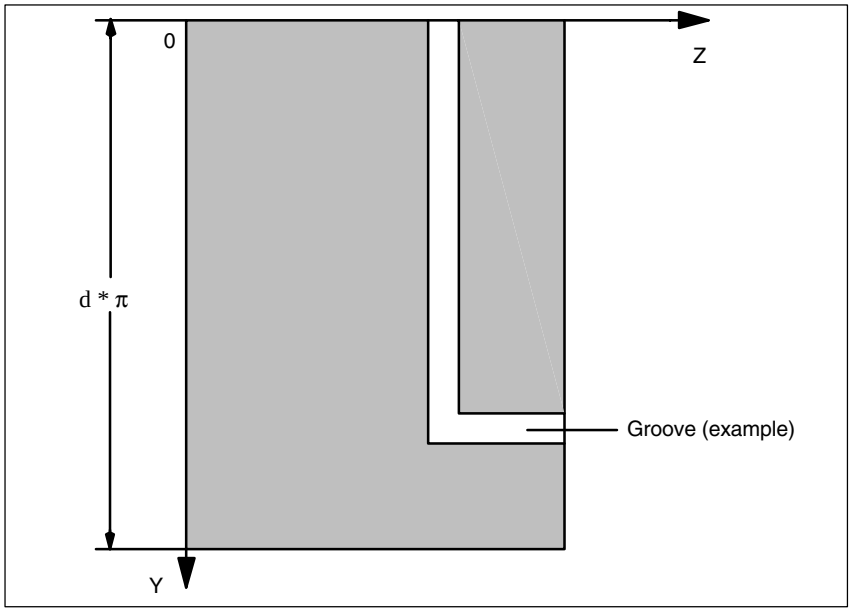


Fig. 2-15 Cylinder coordinate system for Fig. 2-14

2.2.3 Activation of TRACYL

TRACYL

After the settings described above have been made, the TRACYL function can be activated:

TRACYL(d) or
TRACYL(d,t)

The first declared TRACYL function is activated with TRACYL(d). TRACYL(d,t) activates the t-th declared TRACYL function. t must not be more than 2. The value “d” stands for the current diameter of the cylinder to be machined.

Between activation of the function and deactivation as described below, the traversing movements for the axes of the cylinder coordinate system can be programmed.

2.2.4 Deactivation of TRACYL function

TRAFOOF

Keyword TRAFOOF deactivates an active transformation. When the transformation is deactivated, the base coordinate system is again identical to the machine coordinate system.
An active TRACYL transformation is likewise deactivated if one of the other transformations(e.g. TRANSMIT, TRAANG, TRAORI) is activated in the appropriate channel.
References: /FB/, F2, “5-Axis Transformation”

2.2.5 Special system reactions with TRACYL

The transformation can be selected and deselected via part program or MDA.

Please note on selection

- An intermediate motion block is not inserted (phases/radii).
- A spline block sequence must be terminated.
- Tool radius compensation must be deselected.
- The frame which was active prior to TRACYL is deselected by the control (corresponds to “Reset programmed frame” G500).
- An active working area limitation is deselected by the control for the axes affected by the transformation (corresponds to programmed WALIMOF).
- Continuous path control and rounding are interrupted.
- DRF offsets must have been deleted by the operator.
- In the case of cylinder generated surface curve transformation with groove wall compensation (axis configuration 2, TRAFO_TYPE_n=513), the axis used for the correction (TRAFO_AXES_IN_n[3]) must be set to zero (y=0) so that the groove is machined in the center of the programmed groove center line.

Please note on deselection

The same points apply as for selection.

Restrictions imposed by active TRACYL

The restrictions listed below imposed by an activated TRACYL function must be noted:

Tool change

Tools may only be changed when the tool radius compensation function is deselected.

Frame

All instructions which refer exclusively to the base coordinate system are permissible (FRAME, tool radius compensation). Unlike the procedure for inactive transformation, however, a frame change with G91 (incremental dimension) is not specially treated. The increment to be traversed is evaluated in the workpiece coordinate system of the new frame – regardless of which frame was effective in the previous block.

Rotary axis

The rotary axis cannot be programmed because it is occupied by a geometry axis and cannot thus be programmed directly as a channel axis.

Axis utilization

The axes

– in the generated cylinder surface perpendicular to the rotary axis (Y) and

– additional axis (YC)

may not be used as a positioning or oscillation axis.

Exceptions

Axes affected by the transformation cannot be used

- as a preset axis (alarm)
- to approach the fixed point (alarm)
- for referencing (alarm)

Note

TRACYL must be deselected before the tool radius compensation and length compensation are deselected.

Interruption of part program

The following points must be noted with respect to interrupting part program processing in connection with TRACYL:

AUTOMATIC after JOG

If part program processing is interrupted when a transformation is active and JOG mode selected, then the following must be noted if AUTOMATIC mode is selected again:

- The transformation is active in the approach block from the current position to the point of interruption. No monitoring for collisions takes place.

Warning

The operator is responsible for ensuring that the tool can be re-positioned without any difficulties.

START after RESET

If part program processing is aborted with RESET and restarted with START, then the following must be noted:

- The remaining part program is traversed reproducibly only if all axes are traversed to a defined position by means of a linear block (G0 or G1) at the beginning of the part program. A tool which was active on RESET may no longer be taken into account by the control (settable via machine data).

2.2.6 JOG

Special features relating to JOG

When generated cylinder surface transformation with groove wall compensation ($MC_TRAFO_TYPE = 513) is active in JOG mode, it must be noted that the axes are traversed depending on the preceding status in AUTOMATIC. When groove wall compensation is active, the axes movement therefore differs from the situation when the correction function is deselected. The part program can therefore be continued (REPOS) after a part program interruption.

TRAANG (Inclined axis)

Note

The TRAANG transformation described below requires that unique 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.
Only in this way can unambiguous assignments be made.

Task definition

Grinding operations

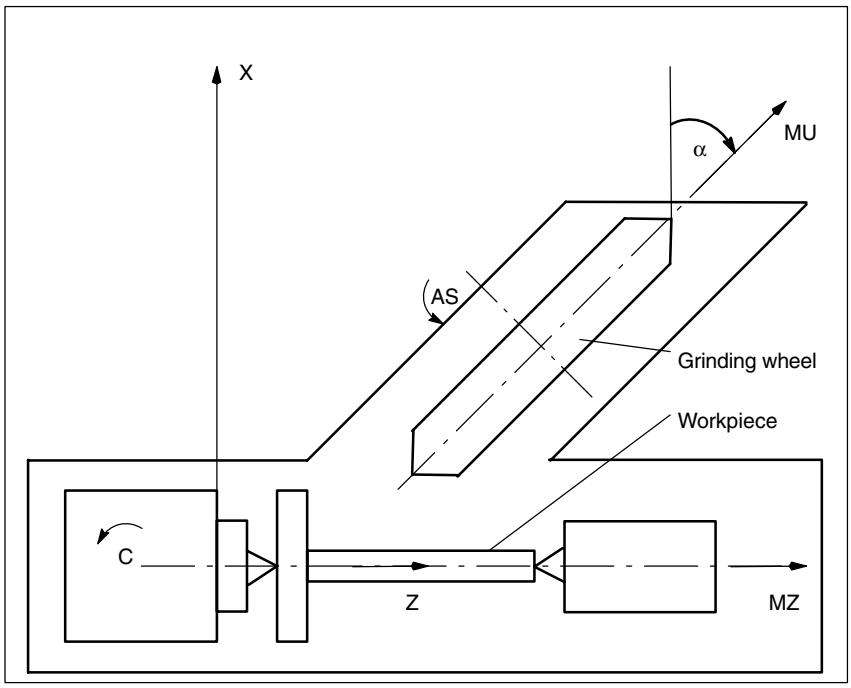


Fig. 2-16 Machine with inclined infeed axis

Legend:

X, Z	Cartesian coordinate system for programming
C	Rotary axis
AS	Working spindle
MZ	Machine axis (linear)
MU	Inclined axis

The following range of machining operations is available:

1. Longitudinal grinding
2. Face grinding
3. Grinding of a specific contour
4. Oblique plunge-cut grinding

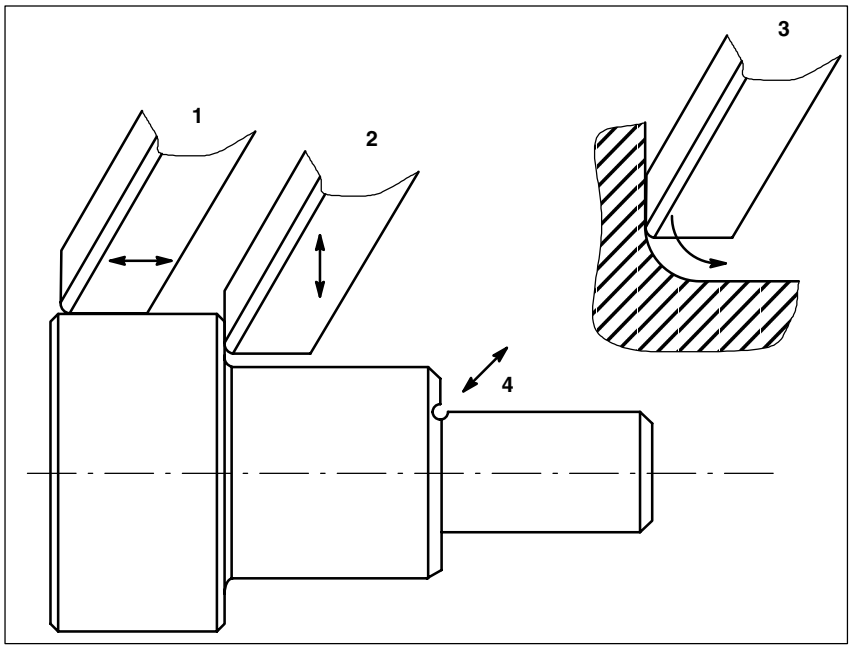


Fig. 2-17 Possible grinding operations

2.2.7 Preconditions for TRAANG (inclined axis)

Axis configuration

To be able to program in the Cartesian coordinate system (see Fig. 2-16: X, Y, Z), it is necessary to inform the control of the correlation between this coordinate system and the actually existing machine axes (MU, MZ):

- Assignment of names to geometry axes
- Assignment of geometry axes to channel axes
  - general situation (inclined axis not active)
  - inclined axis active
- Assignment of channel axes to machine axis numbers
- Identification of spindles
- Allocation of machine axis names

With the exception of the “– Inclined axis active” point, the procedure is the same as for the normal axis configuration.

References: /FB/, K2, “Coordinate Systems, Axis Types and Axis Configurations, Actual-Value System for Workpiece, External Zero Offset”

Number of transformations

Up to 8 transformation data blocks can be defined for each channel in the system. The machine data names of these transformations start with $MC_TRAFO .. and end with ..._n, “n” standing for a number between 1 and 8. The following sections include descriptions of these data:

$MC_TRAFO_GEOAX_ASSIGN_TAB_n
$MC_TRAFO_TYPE_n
$MC_TRAFO_AXES_IN_n

Number of inclined axes

Two of the 8 permitted data structures for transformations may be assigned to the inclined axis function. They are characterized by the fact that the value assigned with $MC_TRAFO_TYPE_n is “1024”.

Axis configuration

The axes of the grinding machine illustrated in Fig. 2-16 must be entered as follows in the machine data:

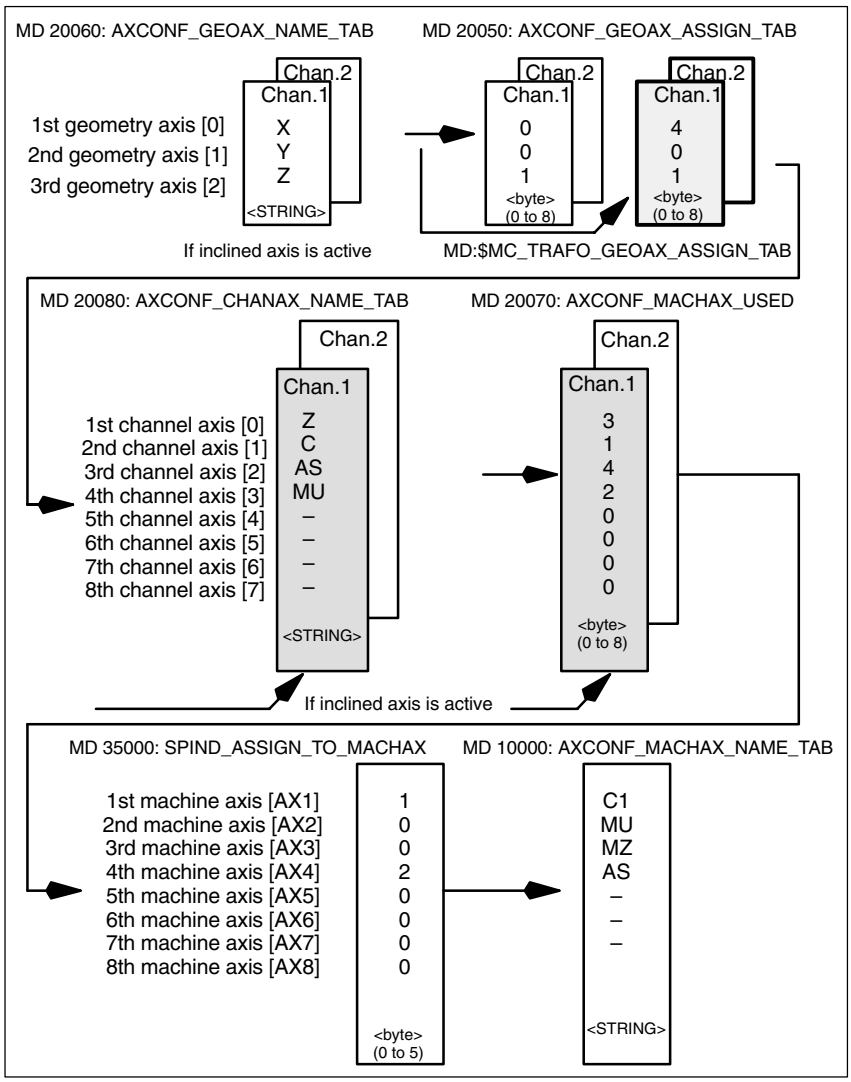


Fig. 2-18 Axis configuration for the example in Fig. 2-16

The highlighted configuration in Fig. 2-18 applies when TRAANG is active.

2.2.8 TRAANG-specific settings

Type of transformation

TRAFO_TYPE_n

The user must specify the transformation type in machine data $MC_TRAFO_TYPE_n for the transformation data blocks (maximum n = 8).
The value for an inclined axis is 1024:
MD 24100: TRAFO_TYPE_1=1024

Note

Changes to the machine data are activated after Power On.

Axis image

TRAFO_AXES _IN_n

Two channel axis numbers must be specified for the transformation data block n:

$MC_TRAFO_AXES_IN_1[0]=4	Channel axis number of inclined axis
$MC_TRAFO_AXES_IN_1[1]=1	Channel axis number of axis parallelto Z
$MC_TRAFO_AXES_IN_1[2]=0	Channel axis number not active

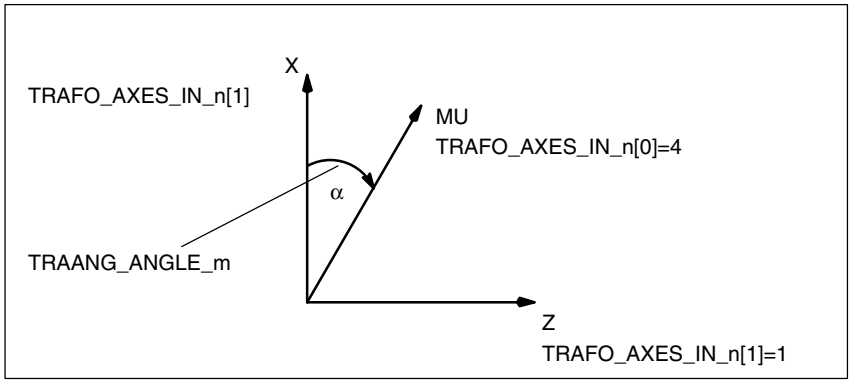


Fig. 2-19 Parameter TRAANG_ANGLE_m

Note

Changes to the machine data are activated after Power On.

Assignment of geometry axes to channel axes

Example:

$MC_TRAFO_TYPE_5 = 8192	Chaining
$MC_TRAFO_AXIS_IN_1[0..x]	
$MC_TRAFO_GEOAX_ASSIGN_TAB_5[0] =1	Definition of geo-axis assignment of transformer 1
$MC_TRAFO_GEOAX_ASSIGN_TAB_5[1] =6	Definition of geo-axis assignment of transformer 1
$MC_TRAFO_GEOAX_ASSIGN_TAB_5[2] =3	Definition of geo-axis assignment of transformer 1
$MC_TRACON_CHAIN_2[0] = 2	Input parameters in TRACON
$MC_TRACON_CHAIN_2[1] = 3	Input parameters in TRACON
$MC_TRACON_CHAIN_2[2] = 0	Input parameters in TRACON
$MC_TRACON_CHAIN_2[3] = 0	Input parameters in TRACON

Angle of inclined axis

TRAANG_ ANGLEm

Machine data $MC_TRAANG_ANGLE_m is used to inform the control of the angle which exists between a machine axis and the inclined axis in degrees.

$MC_TRAANG_ANGLE_m = Angle between a Cartesian axis and the associated inclined machine axis in degrees. The angle is counted positively in the clockwise direction (see Fig. 2-16, angle α).

In this case, m is substituted by the number of the TRAANG transformation declared in the transformation data blocks.m may be maximum 2.

Permissible angular range

The permissible angular range is:
–90° < TRAANG_ANGLE_m < 0°
0° < TRAANG_ANGLE_m < 90°

No transformation is required for 0°.
With +/ – 90°, the inclined axis is positioned parallel to the second linear axis.

Position of tool zero point

TRAANG_ BASE_TOOL_m

Machine data $MC_TRAANG_BASE_TOOL_m is used to inform the control of the position of the tool zero point in relation to the origin of the coordinate system declared for the inclined axis function. The machine data has three components for the 3 axes of the Cartesian coordinate system.

Zero is entered as default.
The corrections are not converted when the angle is changed.

Optimization of velocity control

The following machine data are used to optimize the velocity control in jog mode and in positioning and oscillation modes:

TRAANG_ PARALLEL_ VELO_RES_m

Machine data $MC_TRAANG_PARALLEL_VELO_RES_m is used to set the velocity reserve which is held ready on the parallel axis (see $MC_TRAFO_AXES_IN_n[1]) for the compensatory motion.

Value range: 0 ... 1

0 When value 0 is set, the control automatically determines the reserve: the axes are limited with equal priority(= default setting).

>0 When values of >0 are set, the reserve is fixed at

$MC_TRAANG_PARALLEL _VELO_RES_m	*	permissible machine axis velocity value of parallel axis.

The velocity characteristics of the vertical axis are determined by the control on the basis of the reserve.

TRAANG_ PARALLEL_ ACCEL_RES_m

Machine data $MC_TRAANG_PARALLEL_ACCEL_RES_m is used to set the axis acceleration reserve which is held ready on the parallel axis (see $MC_TRAFO_AXES_IN_n[1]) for the compensatory motion.

Value range: 0 ... 1

0 When value 0 is set, the control automatically determines the reserve: the axes are accelerated with equal priority(= default setting).

>0 With values of >0, the acceleration rate is fixed at

$MC_TRAANG_PARALLEL _ACCEL_RES_m	*	permissible machine axis velocity value of the parallel axis

The velocity characteristics of the vertical axis are determined by the control on the basis of the reserve.

Replaceable geometry axes

The PLC is informed when a geometry axis has been replaced using GEOAX( ) through the optional output of an M code that can be set in machine data.

- MD 22534: TRAFO_CHANGE_M_CODE

  Number of the M code that is output at the VDI interface in the case of transformation changeover.

Note

No M code is output if the machine data is set to one of the values 0 to 6, 17 or 30.

References: /FB/, K2, “Coordinate Systems, Axis Types, Axis Configurations, Actual-Value System for Workpiece External Zero Offset”

2.2.9 Activation of TRAANG

TRAANG(a)

After the settings described above have been made, the TRAANG function can be activated:

TRAANG(a) or
TRAANG(a,n)

The first declared “inclined axis” transformation is activated with TRAANG(a).
The angle of the inclined axis can be specified with “a”.
If “a” is omitted or zero entered, the transformation is activated with parameter settings of the previous selection.
On the first selection, the presettings according to the machine data apply.

TRAANG(a,n) activates the nth declared “inclined axis” transformation.
This form is required only if several transformations are activated in the channel.
n must not be more than 2.

Programming variants

TRAANG(a,1) == TRAANG(a,0) == TRAANG(a,) == TRAANG(a)

Between activation of the function and deactivation as described below, the traversing movements for the axes of the Cartesian coordinate system must be programmed.

2.2.10 Deactivation of TRAANG

TRAFOOF

Keyword TRAFOOF deactivates an active transformation. When the transformation is deactivated, the base coordinate system is again identical to the machine coordinate system.
An active TRAANG transformation is likewise deactivated if one of the other transformations (e.g. TRACYL, TRANSMIT, TRAORI) is activated in the appropriate channel.

References: /FB/, F2, “5-Axis Transformation”

2.2.11 Special system reactions with TRAANG

The transformation can be selected and deselected via part program or MDA.

Selection and deselection

- An intermediate motion block is not inserted (phases/radii).
- A spline block sequence must be terminated.
- Tool radius compensation must be deselected.
- The current frame is deselected by the control system (corresponds to programmed G500).
- An active working area limitation is deselected by the control for the axes affected by the transformation (corresponds to programmed WALIMOF).
- An activated tool length compensation is included in the transformation by the control.
- Continuous path control and rounding are interrupted.
- DRF offsets must have been deleted by the operator.
- All axes specified in machine data $MC_TRAFO_AXES_IN_n must be synchronized on a block-related basis (e.g. no traversing instruction with POSA...).

Restrictions

Tool change

Tools may only be changed when the tool radius compensation function is deselected.

Frame

All instructions which refer exclusively to the base coordinate system are permissible (FRAME, tool radius compensation). Unlike the procedure for inactive transformation, however, a frame change with G91 (incremental dimension) is not specially treated. The increment to be traversed is evaluated in the workpiece coordinate system of the new frame – regardless of which frame was effective in the previous block.

Exceptions

Axes affected by the transformation cannot be used

- as a preset axis (alarm)
- to approach the fixed point (alarm)
- for referencing (alarm)

Note

TRAANG must be deselected before the tool radius and length compensations.

Velocity control

The velocity monitoring function for TRAANG is implemented as standard during preprocessing.
Monitoring and limitation in the main run are activated:

- In AUTOMATIC mode if a positioning or oscillation axis has been programmed that is involved in the transformation.
- On changeover to JOG mode.

The monitoring function is transferred again from the main run to block preprocessing if the preprocessing is re-synchronized with the main run (currently, for example, on changeover from JOG to AUTOMATIC).

The velocity monitoring function in preprocessing utilizes the dynamic limitations of the machine better than the monitoring function in the main run.

This also applies to machines on which, with oblique machining operations,

2.3 Chained transformations

Introduction

SW 5 and higher supports chaining of the kinematic transformations described here:

- TRANSMIT
- TRACYL
- TRAANG (inclined axis)

as well as those described in

References: /FB/, F2, “3 to 5-axis transformations”

- Orientation transformations
- Universal milling head

with another transformation of the “Inclined axis” type.

Applications

The following is a selection from the range of possible chained transformations:

– Grinding contours that are programmed as a side line of a cylinder (TRACYL) using an inclined grinding wheel e.g. tool grinding.

– Finish cutting of a contour that is not round and was generated with TRANSMIT using inclined grinding wheel.

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:
MD 10000: AXCONF_MACHAX_NAME_TAB,
MD 20080: AXCONF_CHANAX_NAME_TAB,
MD 20060: AXCONF_GEOAX_NAME_TAB.
Only in this way can unambiguous assignments be made.

Axis configuration

The following configuration measures are necessary for a chained transformation:

- Assignment of names to geometry axes
- Assignment of names to channel axes
- Assignment of geometry axes to the channel axes
  – General case (no transformation active)
- Assignment of channel axes to machine axis numbers
- Identification of spindle, rotation, modulo for axes
- Allocation of machine axis names
- Transformation-specific settings (for each single transformation **and** for each **chained** transformation)
  – Transformation type
  – Axes included in the transformation
  – Assignment of geometry axes to the channel axes with active transformation
  – According to transformation, also
    rotational position of the coordinate system,
    direction of rotation,
    tool zero or original coordinate system
    angle of inclined axis, etc.

Number of transformations

Up to four (with SW 4 and higher: eight) transformation data blocks can be defined for each channel. The machine data names of these transformations start with $MC_TRAFO .. and end with ..._n, “n” standing for a number between 1 and 8.

Number of chained transformations

Of the maximum of 8 transformations for a channel, up to **two can be defined as chained** transformations.

Transformation sequence

When configuring the machine data, the data concerning the single transformations (that may also become part of chained transformations) must be specified before the data concerning the chained transformations.

Chaining sequence

With chained transformations the second transformation must be “inclined axis” (TRAANG).

Chaining direction

The BCS is the input for the first of the transformations to be chained; the MCS is the output for the second one.

Supplementary conditions

The supplementary conditions and special cases indicated in the individual transformation descriptions are also applicable for use in chained transformations.

2.3.1 Activating chained transformations

TRACON

A chained transformation is activated via:

TRACON(trf, par)

with:

trf — Number of the chained transformation:
0 or 1 for first/single chained transformation.
If nothing is programmed here, then this has the same meaning as specifying value 0 or 1, i.e. the first/single transformation is activated.
2 for the second chained transformation.
(Values not equal to 0 – 2 generate an error alarm).

par — One or more parameters separated by commas for transformations in the chain that require parameters. E.g. angle of inclined axis. If the parameters are not set the default settings or the parameter settings last used are applicable. The use of commas is to ensure that the specified parameters are evaluated in the required sequence, if default settings are to be valid for preceding

parameters. In particular, a comma is required before at least one parameter, even though it is not necessary to specify trf. For example:
TRACON(, 3.7).

If another transformation was previously activated, it is implicitly disabled by means of TRACON().

2.3.2 Switching off chained transformations

TRAFOOF

A chained transformation is switched off with TRAFOOF just like any other transformation.

2.3.3 Special characteristics of chained transformations

Tool data

A tool is always assigned to the first transformation in a chain. The subsequent transformation then behaves as if the active tool length were zero. Only the basic tool lengths set in the machine data (_BASE_TOOL_) are valid for the **first** transformation in the chain.

Example

Section 6.3 contains configuration examples for single transformations and the transformation chains created from them.

2.4 Activating the transformation MD via part program/softkey (SW 5.2 and later)

2.4.1 Functionality

SW 5.1 and earlier

Up to eight different transformations can be set in the control in SW version 5.1 and earlier. The transformation type is set in machine data $MC_TRAFO_TYPE_1 to $MC_TRAFO_TYPE_8.

For each transformation group (TRANSMIT, see Section 2.1), TRACYL (see Section 2.2), TRAANG (see Section 2.2.7) and chained transformations (see Section 2.3) there are two transformation data sets, i.e. no more than two transformations can be set from one group, even when the eight available transformations have not yet all been programmed.

SW 5.2 and later

Transformation MD can now be activated by means of a program command softkey (NEWCONFIG-capable), i.e. these can, for example, be written from the part program, thus altering the transformation configuration completely. The specified restrictions regarding the number of available transformations thus no longer apply.

Note

However, the number of transformation machine data sets is limited as in previous versions.

Properties

The machine data listed in Section 4.3 were activated by POWER ON in SW 5.1 and earlier. They are NEWCONFIG-capable in SW 5.2 and higher.
The protection level is now 7 / 7 (KEYSWITCH_0) which means that data can be modified from the NC program without any particular authorization.

Provided that no transformation is selected (activated) when a NEWCONFIG command is issued (regardless whether via the NEWCONF NC program command, the MMC or implicitly following Reset or end of program), the machine data listed above can be altered without restriction and then activated.

Of particular relevance is that new transformations can be configured or existing transformations replaced by one of a different type or deleted, since the modification options are not restricted to re-parameterization of existing transformations.

2.4.2 Supplementary Conditions

Modifying machine data

The machine data which affect an active transformation may not be altered; any attempt to do so will generate an alarm.

This generally applies to all machine data assigned to a transformation via the associated transformation data group. Machine data included in the group of an active transformation, but not currently in use, can be altered (although this would hardly be a meaningful thing to do). For example, it would be possible to change machine data $MC_TRAFO5_NUTATOR_AX_ANGLE_N for an active transformation with $MC_TRAFO_TYPE = 16 (5-axis transformation with rotatable tool and two mutually perpendicular rotary axes A and B) since this particular machine data is not involved in the transformation.

Please note that machine data $MC_X_AXIS_IN_OLD_X_Z_PLANE may not be altered for an active orientation transformation.

Note

In the case of a program interruption (Repos, deletion of distance to go, ASUBs, etc.), the control system requires a number of different blocks that have already been executed for the repositioning operation. The rule forbidding the machine data of an active transformation to be altered also refers to these blocks.

Example:
Two orientation transformations are set via machine data, e.g. $MC_TRAFO_TYPE_1 = 16, $MC_TRAFO_TYPE_2 = 18.
Let us assume that the second transformation is active when the NEWCONFIG command is executed. In this instance, only machine data which affect the first transformation may be altered, e.g. $MC_TRAFO5_PART_OFFSET_1, but not, for example, $MC_TRAFO5_BASE_TOOL_2 or $MC_X_AXIS_IN_OLD_X_Z_PLANE.

You could also set, for example, another transformation (TRANSMIT) with $MC_TRAFO_TYPE_3 = 256 and parameterize it with other machine data.

Defining geometry axes

Geometry axes must be defined in MD TRAFO_GEOAX_ASSIGN_TAB_X[n] or MD AXCONF_GEOAX_ASSIGN_TAB[n] **before** the control system powers up.

Changing the assignment

The assignment between a transformation data set and a transformation is determined by the sequence of entries in $MC_TRAFO_TYPE_X. The first transformation data set is assigned to the first entry in the table, the second data set to the second entry, etc. This assignment may (and can) not be altered for an active transformation.

Example:

Three transformations are set, two orientation transformations and one Transmit transformation, e.g.

```
$MC_TRAFO_TYPE_1 = 16      ; Orientation transformation,
                           ; 1st orientation transformer data set
$MC_TRAFO_TYPE_2 = 256     ; Transmit transformation
$MC_TRAFO_TYPE_3 = 18      ; Orientation transformation,
                           ; 2nd orientation transformer data set
```

The first data set for orientation transformations is assigned to the first transformation (equalling the first orientation transformation and the second transformation data set to the third transformation (equalling the second orientation transformation).

If the third transformation is active when the NEWCONFIG command is executed, it is not permissible to change the first transformation into a transformation of another group (e.g. TRACYL) since, in this case, the third transformation would then not become the second orientation transformation, but the first.

In the above example, however, it is permissible to set another orientation transformation for the first transformation (e.g. using $MC_TRAFO_TYPE_1 = 32) or a transformation from another group as the first transformation (e.g. using $MC_TRAFO_TYPE_1 = 1024, TRAANG), if the second transformation is changed into an orientation transformation at the same time, e.g. with $MC_TRAFO_TYPE_2 = 48.

2.4.3 Control response to Power On, mode change, Reset, block search, REPOS

Machine data $MC_RESET_MODE_MASK, $MC_START_MODE_MASK and $MC_TRAFO_RESET_VALUE can be programmed to select a transformation automatically in response to RESET (i.e. including end of program) and / or program start.

This may result in the generation of an alarm, for example, at the end or start of a program, if the machine data of an active transformation has been altered.

To avoid this problem when re-configuring transformations via an NC program, we therefore recommend that NC programs are structured as follows:

```
N 10  TRAFOOF()                          ; Deselect any active
                                         ; transformation
N20   $MC_TRAFO5_BASE_TOOL_1[0]=0        ; Write machine data
N30   $MC_TRAFO5_BASE_TOOL_1[0]=3        ;
N40   $MC_TRAFO5_BASE_TOOL_1[0]=200      ;
.
.
.
N130  NEWCONF                            ; Accept newly modified
                                         ; machine data
N140  M30
```

2.4.4 List of machine data affected

The machine data which can be made NEWCONFIG-capable are listed below.

All transformations

Machine data which are relevant for all transformations:

- $MC_TRAFO_TYPE_1 to $MC_TRAFO_TYPE_8
- $MC_TRAFO_AXES_IN_1 to $MC_TRAFO_AXES_IN_8
- $MC_TRAFO_GEOAX_ASSIGN_TAB_1 to $MC_TRAFO_GEOAX_ASSIGN_TAB_8

Orientation transformations

Machine data which are relevant for orientation transformations:

- $MC_TRAFO5_BASE_TOOL_1 and $MC_TRAFO5_BASE_TOOL_2
- $MC_TRAFO5_JOINT_OFFSET_1 and $MC_TRAFO5_JOINT_OFFSET_2
- $MC_TRAFO5_PART_OFFSET_1 and $MC_TRAFO5_PART_OFFSET_2
- $MC_TRAFO5_ROT_AX_OFFSET_1 and $MC_TRAFO5_ROT_AX_OFFSET_2
- $MC_TRAFO5_ROT_SIGN_IS_PLUS_1 and $MC_TRAFO5_ROT_SIGN_IS_PLUS_2
- $MC_TRAFO5_NON_POLE_LIMIT_1 and $MC_TRAFO5_NON_POLE_LIMIT_2
- $MC_TRAFO5_POLE_LIMIT_1 and $MC_TRAFO5_POLE_LIMIT_2
- $MC_TRAFO5_AXIS1_1 and $MC_TRAFO5_AXIS1_2
- $MC_TRAFO5_AXIS2_1 and $MC_TRAFO5_AXIS2_2
- $MC_TRAFO5_BASE_ORIENT_1 and $MC_TRAFO5_BASE_ORIENT_2
- $MC_TRAFO5_TOOL_ROT_AX_OFFSET_1 and $MC_TRAFO5_TOOL_ROT_AX_OFFSET_2
- $MC_TRAFO5_NUTATOR_AX_ANGLE_1 and $MC_TRAFO5_NUTATOR_AX_ANGLE_2
- $MC_TRAFO5_NUTATOR_VIRT_ORIAX_1 and $MC_TRAFO5_NUTATOR_VIRT_ORIAX_2

Transmit transformations

Machine data which are relevant for Transmit transformations:

- $MC_TRANSMIT_BASE_TOOL_1 and $MC_TRANSMIT_BASE_TOOL_2
- $MC_TRANSMIT_ROT_AX_OFFSET_1 and $MC_TRANSMIT_ROT_AX_OFFSET_2
- $MC_TRANSMIT_ROT_SIGN_IS_PLUS_1 and $MC_TRANSMIT_ROT_SIGN_IS_PLUS_2
- $MC_TRANSMIT_POLE_SIDE_FIX_1 and $MC_TRANSMIT_POLE_SIDE_FIX_2

Tracyl transformations

Machine data which are relevant for Tracyl transformations:

- $MC_TRACYL_BASE_TOOL_1 and $MC_TRACYL_BASE_TOOL_2
- $MC_TRACYL_ROT_AX_OFFSET_1 and $MC_TRACYL_ROT_AX_OFFSET_2
- $MC_TRACYL_ROT_SIGN_IS_PLUS_1 and $MC_TRACYL_ROT_SIGN_IS_PLUS_2

Inclined axis transformations

Machine data which are relevant for inclined axes:

- $MC_TRAANG_BASE_TOOL_1 and $MC_TRAANG_BASE_TOOL_2
- $MC_TRAANG_ANGLE_1 and $MC_TRAANG_ANGLE_2
- $MC_TRAANG_PARALLEL_VELO_RES_1 and $MC_TRAANG_PARALLEL_VELO_RES_2
- $MC_TRAANG_PARALLEL_ACCEL_RES_1 and $MC_TRAANG_PARALLEL_ACCEL_RES_2

Chained transformations

Machine data which are relevant for chained transformations:

- $MC_TRACON_CHAIN_1 and $MC_TRACON_CHAIN_2

Non-specific machine data

Machine data which are not specific to a particular transformation type. They are not uniquely assigned to a particular transformation data set or are relevant even when a transformation is not active:

- $MC_X_AXIS_IN_OLD_X_Z_PLANE
- $MC_MAX_LEAD_ANGLE
- $MC_MAX_TILT_ANGLE
- $MC_ORIENTATION_IS_EULER

■

Supplementary Conditions 3

3.1 TRANSMIT

Availability

The “TRANSMIT” function is an option with order number:
6FC5 251-0AB01-0AA0.

It is available from product version 2 onwards for:

- SINUMERIK FM-NC with NCU 570
- SINUMERIK 840D with NCU 571–573.
- SINUMERIK 810D

Pole traversal and optimized control response in pole vicinity are available with SW 4.1 and higher.

3.2 TRACYL (peripheral surface transformation)

Availability

The “TRACYL” function is an option with order number:
6FC5 251–0AB01–0AA0.

It is available in SW 2 and later for:

- SINUMERIK FM-NC with NCU 570
- SINUMERIK 840D with NCU 571–573.
- SINUMERIK 810D

3.3 TRAANG (inclined axis)

Availability

The “TRAANG” (inclined axis) function is an option with order number:
6FC5 251–0AB06–0AA0.

It is available in SW 2 and later for:

- SINUMERIK 840D with NCU 572–573.2.
- SINUMERIK 810D

3.4 Chained transformation (SW 5)

SW 5 and higher

Two transformations can be chained.

However, not just any transformation can be chained to another one.
In SW 5 the following restrictions apply:

- The **first** transformation in the chain must be:
  - an orientation transformation
    (3, 4, 5-axis transformation, universal milling head),
  - Transmit or
  - side line transformation or
  - inclined axis
- The **second** transformation must be an **inclined** axis transformation.
- Only two transformations may be chained.

It is permissible (e.g. for testing purposes) to enter only one transformation in the chain list.

■

Data Descriptions (MD, SD) 4

4.1 Channel-specific machine data

21110 **MD number**	**X_AXIS_IN_OLD_X_Z_PLANE** **Coordinate system with automatic FRAME definition**		
Default value: 1	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEWCONFIG (SW 5.2 or higher) Power On (up to SW 5.1)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: –	
Data type: BOOLEAN	Applies from SW version: 2		
Significance:	1 = With automatic definition of a frame (TOFRAME) whose Z direction is the same as the actual tool orientation, the new coordinate system is rotated additionally around the new Z axis with the result that the new X axis lies in the old Z/X plane. 0 = With automatic definition of a frame (TOFRAME) whose Z direction is the same as the actual tool orientation, the new coordinate system is not manipulated as it results out of the machine kinematics, i.e. the coordinate system rotates with the tool (orientation).		
MD irrelevant for	No orientation programming.		
Related to	MD 21100		
References	Programming Guide		

4.2 Transformation-specific machine data

22534 MD number	**TRAFO_CHANGE_M_CODE** M code for transformation changeover on geometry axes		
Default value: 0	Min. input limit: 0	Max. input limit: 99999999	
Changes effective after	Protection level: 2/7	Unit: –	
Data type: DWORD	Applies from SW version: 4.1		
Significance:	Number of M code that is output at the VDI interface in the case of a transformation changeover on the geometry axes. No M code is output if this MD is set to one of the values 0 to 6, 17 or 30. It is not monitored whether an M code created in this way will conflict with other functions.		

24100 MD number	**TRAFO_TYPE_1** Type of 1st transformation		
Default value: 0	Min. input limit: 0	Max. input limit: –	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit:	
Data type: DWORD	Applies from SW version: 2.0		
Significance:	This MD specifies for each channel which transformation is available as the first in the channel. Identifier for specifying axis sequence in the case of 5-axis transformation and transformation type for each of the permissible transformations 4 3 0 Transformation type | Axis sequence **Transformation type:** 0 No transformation 16 5-axis transformation with rotatable tool 32 5-axis transformation with rotatable tool 48 5-axis transformation with rotatable tool and workpiece **Axis sequence** for transformation types **16 – 48** 0 Axis sequence AB 1 Axis sequence AC 2 Axis sequence BY 3 Axis sequence BC 4 Axis sequence CA 5 Axis sequence CB 8 Generic 5-axis transformation 256 TRANSMIT transformation 512 TRACYL transformation 513 TRACYL transformation with X-Y-Z-C kinematics 1024 TRAANG transformation 2048 Centerless transformation 8192 Chained transformation When values are assigned to transformation types 16–48, the associated axis sequences are added. Axis sequences for transformation types 256 – 2048 are meaningless (no error message).		
MD irrelevant for	No transformations		
Application	$MC_TRAFO_TYP_1=20 ; (16+4)		
Related to	TRAFO_TYPE_2, TRAFO_TYPE_3, ... TRAFO_TYPE_8		
References	/FB/, F2, “5-Axis Transformation”		

24110 MD number	**TRAFO_AXES_IN_1[i]** Axis assignment for transformation 1 [axis index]: 0 ... [max. no. of channel axes]		
Default value: 1,2,3,4,5,0,0,0	Min. input limit: 1	Max. input limit: [max. no. of channel axes]	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit:	
Data type: byte	Applies from SW version: 2.0		
Significance:	Axis assignment at input of 1st transformation Index i assumes the values 0, 1 and 2 with TRANSMIT. The assignment for TRANSMIT is: $MC_TRAFO_AXES_IN_1[0]= Channel axis number of axis perpendicular to rotary axis $MC_TRAFO_AXES_IN_1[1]= Channel axis number of rotary axis $MC_TRAFO_AXES_IN_1[2]= Channel axis number of axis parallel to rotary axis The index entered at the nth position specifies which axis is mapped internally by the transformation on axis n.		
MD irrelevant for	No transformation		
Application	$MC_TRAFO_AXES_IN_1[0]=1		
Related to	TRAFO_AXES_IN_2, TRAFO_AXES_IN_3, ... TRAFO_AXES_IN_8		
References	/FB/, F2, “5-Axis Transformation”		

24120 MD number	**TRAFO_GEOAX_ASSIGN_TAB_1[i]** Assignment of geometry axes to channel axes with transformation 1 [geometry axis number]: 0 ... 2.		
Default value: 0,0,0	Min. input limit: 0	Max. input limit: [max. no. of channel axes]	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit:	
Data type: byte	Applies from SW version: 2.0		
Significance:	This MD specifies the channel axes on which the axes of the Cartesian coordinate system are mapped when transformation 1 is active. Index i assumes the values 0, 1, 2 with TRANSMIT. It refers to the first, second and third geometry axis.		
MD irrelevant for	No transformation		
Application	$MC_TRAFO_GEOAX_ASSIGN_TAB_1[0]= channel axis number		
Related to	$MC_AXCONF_GEOAX_ASSIGN_TAB, if no transformation is active.		
References	/FB/, K2, Coordinate Systems, Axis Types and Axis Configurations, Actual-Value System for Workpiece, External Zero Offset		

24200 / 24300 / 24400/24430/24440/ 24450/24460 MD number	**TRAFO_TYPE_2 / _3 / _4 / _5/ _6/ _7/ _8** Type of transformation		
Default value: 0	Min. input limit: 0	Max. input limit: –	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit:	
Data type: DWORD	Applies from SW version: 2.0		
Significance:	As for TRAFO_TYPE_1, but applies to transformation that is available as the second ... eighth transformation in the channel.		

24210/24310 / 24410/24432/24442/ 24452/24462 MD number	**TRAFO_AXES_IN_2[i] / _3[i] / _4[i]/ _5[i] / _6[i] / _7[i] / _8[i]** Axis assignment for transformation 2/3/4/5/6/7/8 [axis index]: 0 ... [max. no. of channel axes]		
Default value: 1,2,3,4,5,0,0,0	Min. input limit: 1	Max. input limit: [max. no. of channel axes]	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit:	
Data type: byte	Applies from SW version: 2.0		
Significance:	Axis assignment at input of 2nd to 8th transformation. Same meaning as for TRAFO_AXES_IN_1.		

24220/24320 / 24420/24434/24444/ 24454/24464 MD number	**TRAFO_GEOAX_ASSIGN_TAB_2[i] / _3[i] / _4[i]/_5[i] / _6[i] / _7[i]/ _8[i]** Assignment of geometry axes to channel axes with transformation 2/3/4/5/6/7/8 [geometry axis number]:0 ... 2.		
Default value: 0,0,0	Min. input limit: 0	Max. input limit:	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: [max. no. of channel axes]	
Data type: byte	Applies from SW version: 2.0		
Significance:	The channel axes on which the axes of the Cartesian coordinate system are to be allocated when transformation 2 to 8 is active are set in this MD. The function otherwise corresponds to TRAFO_GEOAX_ASSIGN_TAB_1.		

4.3 Function-specific machine data

4.3.1 TRANSMIT

24900 MD number	**TRANSMIT_ROT_AX_OFFSET_1** Position offset of rotary axis		
Default value: 0	Min. input limit: 0	Max. input limit: 360	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 2/4	Unit: degrees	
Data type: DOUBLE	Applies from SW version: 2.0		
Significance:	Specifies the offset of the rotary axis in degrees in relation to the zero position while TRANSMIT is active for the first declared TRANSMIT transformation for each channel.		
MD irrelevant for	No TRANSMIT active		
Application	$MC_TRANSMIT_ROT_AX_OFFSET_1=15.0		
Related to	TRANSMIT_ROT_AX_OFFSET_2		

24910 MD number	**TRANSMIT_ROT_SIGN_IS_PLUS_1** Sign of rotary axis 1/2		
Default value: 1	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 2/4	Unit: –	
Data type: BOOLEAN	Applies from SW version: 2.0		
Significance:	Specifies the sign which is applied to the rotary axis during the TRANSMIT transformation for the first declared TRANSMIT transformation for each channel.		
MD irrelevant for	No TRANSMIT active		
Application	$MC_TRANSMIT_ROT_SIGN_IS_PLUS_1=TRUE		
Related to	TRANSMIT_ROT_SIGN_IS_PLUS_2		

24911 MD number	**TRANSMIT_POLE_SIDE_FIX_1** Restriction of working range in front of/behind pole, 1st transformation		
Default value: 0	Min. input limit: 0	Max. input limit: 2	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 2/4	Unit:	
Data type: BYTE	Applies from SW version: 4.1		
Significance:	Restriction of working range in front of/behind pole or no restrictions, i.e. traversal through pole. The setpoints have the following meanings: 1: Working range of linear axis for positions >=0, (if tool length compensation parallel to linear axis equals 0) 2: Working range of linear axis for positions <=0, (if tool length compensation parallel to linear axis equals 0) 0: No restriction of working range. Traversal through pole.		

24920 MD number	**TRANSMIT_BASE_TOOL_1[i]** Vector of base tool on activation of transformation		
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)		Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: mm
Data type: DOUBLE		Applies from SW version: 2.0	
Significance:	MD specifies the distance of the tool zero point referred to the appropriate geometry axes for TRACYL active without a tool length compensation being selected. related to the geometry axes valid with TRANSMIT active and without tool length offset selected for the 2nd TRANSMIT transformation for each channel. Programmed length compensations are added to the base tool. Index i assumes values 0, 1, 2 for the 1st to 3rd geometry axis.		
MD irrelevant for	No TRANSMIT active		
Application	$MC_TRANSMIT_BASE_TOOL_1[0]=20.0		
Related to	$MC_TRANSMIT_BASE_TOOL_2		

24950 MD number	**TRANSMIT_ROT_AX_OFFSET_2** Position offset of rotary axis		
Default value: 0	Min. input limit: 0	Max. input limit: 360	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)		Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: degrees
Data type: DOUBLE		Applies from SW version: 2.0	
Significance:	Specifies the offset of the rotary axis in degrees in relation to the zero position while TRANSMIT is active for the second declared TRANSMIT transformation for each channel.		
MD irrelevant for	No TRANSMIT active		
Related to	TRANSMIT_ROT_AX_OFFSET_1		

24960 MD number	**TRANSMIT_ROT_SIGN_IS_PLUS_2** Sign of rotary axis 1/2		
Default value: 1	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)		Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: –
Data type: BOOLEAN		Applies from SW version: 2.0	
Significance:	Specifies the sign which is applied to the rotary axis during the TRANSMIT transformation for the second declared TRANSMIT transformation for each channel.		
MD irrelevant for	No TRANSMIT active		
Application	$MC_TRANSMIT_ROT_SIGN_IS_PLUS_1=TRUE		
Related to	TRANSMIT_ROT_SIGN_IS_PLUS_1		

24961 MD number	**TRANSMIT_POLE_SIDE_FIX_2** Restriction of working range in front of/behind pole, 2nd transformation		
Default value: 0	Min. input limit: 0	Max. input limit: 2	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)		Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: –
Data type: BYTE		Applies from SW version: 4.1	
Significance:	Restriction of working range in front of/behind pole or no restrictions, i.e. traversal through pole. The setpoints have the following meanings: 1: Working range of linear axis for positions >=0, (if tool length compensation parallel to linear axis equals 0) 2: Working range of linear axis for positions <=0, (if tool length compensation parallel to linear axis equals 0) 0: No restriction of working range. Traversal through pole.		

24970 MD number	**TRANSMIT_BASE_TOOL_2[i]** Vector of base tool on activation of transformation		
Default value: 0	Min. input limit: 0	Max. input limit:	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: mm	
Data type: DOUBLE	Applies from SW version: 2.0		
Significance:	Specifies the distance of the tool zero point related to the geometry axes valid with TRANSMIT active and without tool length offset selected for the 2nd TRANSMIT transformation for each channel. Programmed length compensations are added to the base tool. Index i assumes values 0, 1, 2 for the 1st to 3rd geometry axis.		
MD irrelevant for	No TRANSMIT active		
Application	$MC_TRANSMIT_BASE_TOOL_2[0]=tx		
Related to	$MC_TRANSMIT_BASE_TOOL_1		

4.3.2 TRACYL

24800 MD number	**TRACYL_ROT_AX_OFFSET_1** Offset of rotary axis for 1st TRACYL transformation		
Default value: 0	Min. input limit: [no limit]	Max. input limit: [no limit]	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: degrees	
Data type: DOUBLE	Applies from SW version: 2.0		
Significance:	Specifies the offset of the rotary axis in degrees in relation to the zero position while TRACYL is active for the first declared TRACYL transformation for each channel.		
MD irrelevant for	No TRACYL active		
Application	$MC_TRACYL_ROT_AX_OFFSET_1=15.0		
Related to	TRACYL_ROT_AX_OFFSET_2		

24810 MD number	**TRACYL_ROT_SIGN_IS_PLUS_1** Sign of rotary axis for 1st TRACYL transformation		
Default value: 1	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: –	
Data type: BOOLEAN	Applies from SW version: 2.0		
Significance:	Specifies the sign which is applied to the rotary axis during the TRACYL transformation for the first declared TRACYL transformation for each channel.		
MD irrelevant for	No TRACYL active		
Application	$MC_TRACYL_ROT_SIGN_IS_PLUS_1=TRUE		
Related to	TRACYL_ROT_SIGN_IS_PLUS_2		

24820 MD number	**TRACYL_BASE_TOOL_1[i]** Vector of base tool for 1st TRACYL transformation		
Default value: 0	Min. input limit: 0	Max. input limit:	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: mm	
Data type: DOUBLE	Applies from SW version:		
Significance:	MD specifies the distance of the tool zero point referred to the appropriate geometry axes for TRACYL active without a tool length compensation being selected. for the 1st TRACYL transformation for each channel. Programmed length compensations are added to the base tool. Index i assumes values 0, 1, 2 for the 1st to 3rd geometry axis.		
MD irrelevant for	No TRACYL active		
Application	$MC_TRACYL_BASE_TOOL_1[0]=tx		
Related to	$MC_TRACYL_BASE_TOOL_2		

24850 MD number	**TRACYL_ROT_AX_OFFSET_2** Offset of rotary axis for 2nd TRACYL transformation		
Default value: 0	Min. input limit: [no limit]	Max. input limit: [no limit]	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: degrees	
Data type: DOUBLE	Applies from SW version: 2.0		
Significance:	Specifies the distance of the tool zero point offset of the rotary axis in degrees in relation to the zero position while TRACYL is active for the second declared TRACYL transformation for each channel.		
MD irrelevant for	No TRACYL active		
Application	$MC_TRACYL_ROT_AX_OFFSET_2=15.0		
Related to	TRACYL_ROT_AX_OFFSET_1		

24860 MD number	**TRACYL_ROT_SIGN_IS_PLUS_2** Sign of rotary axis for 2nd TRACYL transformation		
Default value: 1	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: –	
Data type: BOOLEAN	Applies from SW version: 2.0		
Significance:	Specifies the distance of the tool zero point sign which is applied to the rotary axis during the TRACYL transformation for the second declared TRACYL transf. for each channel.		
MD irrelevant for	No TRACYL active		
Application	$MC_TRACYL_ROT_SIGN_IS_PLUS_2=TRUE		
Related to	TRACYL_ROT_SIGN_IS_PLUS_1		

24870 MD number	**TRACYL_BASE_TOOL_2[i]** Vector of base tool for 2nd TRACYL transformation		
Default value: 0	Min. input limit: 0	Max. input limit:	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: mm	
Data type: DOUBLE	Applies from SW version: 2.0		
Significance:	Specifies the distance of the tool zero point for the 1st TRACYL transformation for each channel. Programmed length compensations are added to the base tool. Index i assumes values 0, 1, 2 for the 1st to 3rd geometry axis.		
MD irrelevant for	No TRACYL active		
Application	$MC_TRACYL_BASE_TOOL_2[0]=tx		
Related to	$MC_TRACYL_BASE_TOOL_1		

4.3.3 TRAANG

24700 MD number	**TRAANG_ANGLE_1** Angle between Cartesian axis and real (inclined) axis for the first TRAANG transformation		
Default value: 0	Min. input limit: –90	Max. input limit: 90	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: degrees	
Data type: DOUBLE	Applies from SW version: 2.0		
Significance:	Specifies the angle of the inclined axis in degrees between the 1st machine axis and the 1st basic axis when TRAANG is active for the first declared TRAANG transformation of the channel. The angle is counted positively in clockwise direction.		
MD irrelevant for	No TRAANG active		
Application	$MC_TRAANG_ANGLE_1=15.0		
Related to	TRAANG_ANGLE_2		

24710 MD number	**TRAANG_BASE_TOOL_1[i]** Vector of base tool for first TRAANG transformation [axis no.]: 0 ... 2		
Default value: 0	Min. input limit: 0	Max. input limit: 2	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: mm	
Data type: DOUBLE	Applies from SW version: 2.0		
Significance:	MD specifies the distance of the tool zero point referred to the appropriate geometry axes for TRACYL active without a tool length compensation being selected. to the appropriate geometry axes for TRAANG active and without tool length compensation being selected. Distance is specified for the 2nd TRAANG transformation for each channel. Programmed length compensations are added to the base tool. Index i assumes values 0, 1, 2 for the 1st to 3rd geometry axis.		
MD irrelevant for			
Application	$MC_TRAANG_BASE_TOOL_1[0]=tx		
Related to	$MC_TRAANG_BASE_TOOL_2		

24720 MD number	**TRAANG_PARALLEL_VELO_RES_1**		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: –	
Data type: DOUBLE	Applies from SW version: 2.0		
Significance:	Specifies the velocity reserve for jog, positioning and oscillation movements which is held ready on the parallel axis (see $MC_TRAFO_AXES_IN_n[1]) for the compensatory movement; MD setting applies to the first TRAANG transformation for each channel.		
MD irrelevant for	No TRAANG active		
Application	$MC_TRAANG_PARALLEL_VELO_RES_1=0		
Related to	TRAANG_PARALLEL_VELO_RES_2		

24721 MD number	**TRAANG_PARALLEL_VELO_RES_2**		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: –	
Data type: DOUBLE	Applies from SW version: 2.0		
Significance:	Specifies the velocity reserve for jog, positioning and oscillation movements which is held ready on the parallel axis (see $MC_TRAFO_AXES_IN_n[1]) for the compensatory movement; MD setting applies to the second TRAANG transformation for each channel.		
Application	$MC_TRAANG_PARALLEL_VELO_RES_2=0		
Related to	$MC_TRAANG_PARALLEL_VELO_RES_1		

24750 MD number	**TRAANG_ANGLE_2** Angle between Cartesian axis and real (inclined) axis for second TRAANG transformation		
Default value: 0	Min. input limit: –90	Max. input limit: 90	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: degrees	
Data type: DOUBLE	Applies from SW version: 2.0		
Significance:	Specifies the angle of the inclined axis in degrees between the 1st machine axis and the 1st basic axis when TRAANG is active for the second declared TRAANG transformation of the channel. The angle is counted positively in clockwise direction.		
MD irrelevant for	No TRAANG active		
Application	$MC_TRAANG_ANGLE_1=15.0		
Related to	TRAANG_ANGLE_1		

24760 MD number	**TRAANG_BASE_TOOL_2[i]** Vector of base tool for second TRAANG transformation [axis no.]: 0 ... 2		
Default value: 0	Min. input limit: 0	Max. input limit: 2	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: mm	
Data type: DOUBLE	Applies from SW version: 2.0		
Significance:	Specifies the distance of the tool zero point to the appropriate geometry axes for TRAANG active and without tool length compensation being selected. Distance is specified for the 2nd TRAANG transformation for each channel. Programmed length compensations are added to the base tool. Index i assumes values 0, 1, 2 for the 1st to 3rd geometry axis.		
MD irrelevant for	No TRAANG active		
Application	$MC_TRAANG_BASE_TOOL_2[0]=tx		
Related to	$MC_TRAANG_BASE_TOOL_1		

24770 MD number	**TRAANG_PARALLEL_ACCEL_RES_1**		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: –	
Data type: DOUBLE	Applies from SW version: 2.0		
Significance:	Specifies the axis acceleration reserve for jog, positioning and oscillation movements which is held ready on the parallel axis (see $MC_TRAFO_AXES_IN_n[1]) for the compensatory movement; MD setting applies to the first TRAANG transformation for each channel.		
MD irrelevant for	No TRAANG active		
Application	$MC_TRAANG_PARALLEL_ACCEL_RES_1=0		
Related to	TRAANG_PARALLEL_ACCEL_RES_2		

24771 MD number	**TRAANG_PARALLEL_ACCEL_RES_2**		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: –	
Data type: DOUBLE	Applies from SW version: 2.0		
Significance:	Specifies the axis acceleration reserve for jog, positioning and oscillation movements which is held ready on the parallel axis (see $MC_TRAFO_AXES_IN_n[1]) for the compensatory movement; MD setting applies to the second TRAANG transformation for each channel.		
MD irrelevant for	No TRAANG active		
Application	$MC_TRAANG_PARALLEL_RES_2=0		
Related to	$MC_TRAANG_PARALLEL_RES_1		

4.3.4 MD for chained transformations

24995 MD number	**TRACON_CHAIN_1[n]** Transformation chain of the first chained transformation		
Default value: 0	Min. input limit: 0	Max. input limit: 8	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: –	
Data type: DWORD	Applies from SW version: 5		
Significance:	The MD is saved internally as a table. In the table the numbers of the transformations to be chained are specified in the same sequence as the transformations are to be implemented from the BCS to the MCS. n stands for the index of entries in the MD. Example: Optionally, a machine can be operated as a 5-axis machine or as a Transmit machine. A linear axis is not perpendicular to the other linear axes (inclined axis). 5 transformations must be set via machine data, e.g. TRAFO_TYPE_1 = 16 (5-axis trafo), first transformation TRAFO_TYPE_2 = 256 (Transmit), second transformation TRAFO_TYPE_3 = 1024 (Inclined axis), third transformation TRAFO_TYPE_4 = 8192 First chained transformation, fourth transformation TRAFO_TYPE_5 = 8192 Second chained transformation, fifth transformation If the 4th transformation is to be the chaining: 5-axis transformation / inclined axis and the 5th transformation is to be the chaining: Transmit / inclined axis, then TRACON_CHAIN_1 (1, 3, 0, 0) is entered in the first table and TRACON_CHAIN_2 (2, 3, 0, 0) in the second table. Detailed notation shown in the example in Section 6. Entry 0 means no transformation (a 3rd and 4th transformation cannot be chained in SW 5). The transformations can be assigned (TRAFO_TYPE_1 to TRAFO_TYPE_8) in any sequence. The chained transformations do not have to be the last ones. However, they must be behind all transformations that occur in a transformation chain. In the preceding example, this would mean that, for example, the position of the third and fourth transformation must not be swapped. It would be possible though to define a sixth transformation if it is not to be included in a chained transformation. However, not just any transformation can be chained to another one. In SW 5 the following restrictions apply: • The first transformation in the chain must be: – an orientation transformation (3, 4, 5-axis transformation, universal milling head), – Transmit or – side line transformation or – inclined axis • The second transformation must be an inclined axis transformation. • Only two transformations may be chained. It is permissible (e.g. for testing purposes) to enter only one transformation in the list.		
MD irrelevant for	TRAFOOF		
Application	Section 6		
Special cases, errors, ...	More than 2 transformations in the chain, 2nd transformation not TRAANG		
Related to	MD 24100: TRAFO_TYPE		
References	/FB/, F2, “3 to 5-axis transformation”		

24996 MD number	**TRACON_CHAIN_2[n]** Transformation chain of the second chained transformation		
Default value:	Min. input limit: 0	Max. input limit: 8	
Changes effective after NEWCONFIG (SW 5.2 and higher) Power On (SW 5.1 and earlier)	Protection level: 7 / 7 (SW 5.2 and later) 2 / 7 (SW 5.1 and earlier)	Unit: –	
Data type: DWORD	Applies from SW version: 5		
Significance:	Analogous to TRACON_CHAIN_1, but for the second chained transformation in the channel		
MD irrelevant for	TRAFOOF		
Application	Chapter 6		
Special cases, errors, ...	More than 2 transformations in the chain, 2nd transformation not TRAANG		
Related to	MD 24100: TRAFO_TYPE		
References	/FB/, F2, “3 to 5-axis transformation”		

■

Notes

Signal Descriptions

5

5.1 TRANSMIT

DB21, ... **DBX 33.6** Data block	**Transformation active** Signal(s) from NCK channel (NCK–>PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.2
Signal state 1 or signal transition 0 ——> 1	The NC command TRANSMIT, TRACYL, TRAANG or TRAORI is programmed in the part program. The corresponding block has been processed by the NC and a transformation is now active.	
Signal state 0 or signal transition 1 ——> 0	No transformation is active.	
References	/PA1/, Programming Guide /FB/, F2, “5-Axis Transformation”	

5.2 TRACYL

See 5.1

5.3 TRAANG

See 5.1

■

Notes

Example

6

6.1 TRANSMIT

The following programming example relates to the configuration illustrated in Fig. 6-1 and shows the sequence of main steps required to configure the axes and activate TRANSMIT.

; General axis configuration for turning

```
$MC_AXCONF_GEOAX_NAME_TAB[0] = “X”          ; Geometry axis
$MC_AXCONF_GEOAX_NAME_TAB[1] = “Y”          ; Geometry axis
$MC_AXCONF_GEOAX_NAME_TAB[2] =  “Z”         ; Geometry axis
$MC_AXCONF_GEOAX_ASSIGN_TAB[0] =1           ; X as channel axis 1
$MC_AXCONF_GEOAX_ASSIGN_TAB[1] = 0          ; Y not channel axis
$MC_AXCONF_GEOAX_ASSIGN_TAB[2] =  2         ; Z as channel axis 2
$MC_AXCONF_CHANAX_NAME_TAB[0] = “XC”;
$MC_AXCONF_CHANAX_NAME_TAB[1] = “ZC”;
$MC_AXCONF_CHANAX_NAME_TAB[2] =  “CC”;
$MC_AXCONF_CHANAX_NAME_TAB[3] =  “ASC”;
$MC_AXCONF_CHANAX_NAME_TAB[4] =  “”;
$MC_AXCONF_MACHAX_USED[0] = 2               ;XC as machine axis 2
$MC_AXCONF_MACHAX_USED[1] = 3               ;ZC as machine axis 3
$MC_AXCONF_MACHAX_USED[2] = 1               ;CC as machine axis 1
$MC_AXCONF_MACHAX_USED[3] = 4               ;ASC as machine axis 4
$MC_AXCONF_MACHAX_USED[3] = 0               ;empty
$MA_SPIND_ASSIGN_TO_MACHAX[AX1]= 1 ; C is spindle 1
$MA_SPIND_ASSIGN_TO_MACHAX[AX2]= 0 ; X is not spindle
$MA_SPIND_ASSIGN_TO_MACHAX[AX3]= 0 ; Z is not spindle
$MA_SPIND_ASSIGN_TO_MACHAX[AX4]= 2 ; AS is spindle 2
$MN_AXCONF_MACHAX_NAME_TAB[0]= “CM” ; 1st machine axis
$MN_AXCONF_MACHAX_NAME_TAB[1]= “XM” ; 2nd machine axis
$MN_AXCONF_MACHAX_NAME_TAB[2]= “ZM” ; 3rd machine axis
$MN_AXCONF_MACHAX_NAME_TAB[3]= “ASM” ; 4th machine axis
```

```
;Prepare for TRANSMIT (as first and only transformation)

$MA_ROT_IS_MODULO[3] = TRUE ; c as modulo axis
$MC_TRAFO_TYPE_1 = 256 ; TRANSMIT transformation
$MC_TRAFO_AXES_IN_1[0] = 1 ;Channel axis perpendicular to rotary axis
$MC_TRAFO_AXES_IN_1[1] =  3 ; Channel axis rotary axis
$MC_TRAFO_AXES_IN_1[2] =  2 ; Channel axis parallel to rotary axis
$MC_TRAFO_GEOAX_ASSIGN_TAB_1 [0] = 1 ; X 1st channel axis
$MC_TRAFO_GEOAX_ASSIGN_TAB_1 [1] = 3 ; Z 2nd channel axis
$MC_TRAFO_GEOAX_ASSIGN_TAB_1 [2] = 2 ; Y 3rd channel axis
$MC_TRANSMIT_ROT_AX_OFFSET_1 = 0.  ; Rotational position X-Y plane in
                                     rel. to zero of rotary axis
$MC_TRANSMIT_ROT_SIGN_IS_PLUS_1 = FALSE ; Rotary axis turns –
$MC_TRANSMIT_BASE_TOOL_1  [0] = 0.0 ; Tool distance in X
$MC_TRANSMIT_BASE_TOOL_1  [1] = 0.0 ; Tool distance in Y
$MC_TRANSMIT_BASE_TOOL_1  [2] = 0.0 ; Tool distance in Z


 ;  Activation of
TRANSMIT

 ;  Programming in X, Y, Z

 ;  Return to rotational operation
TRAFOOF

TRACYL
```

The following programming example relates to the configuration illustrated in Fig. 6-1 and shows the sequence of main steps required to configure the axes and activate TRACYL.

```
;General axis configuration for turning

$MC_AXCONF_GEOAX_NAME_TAB[0] = “X”          ; Geometry axis
$MC_AXCONF_GEOAX_NAME_TAB[1] = “Y”          ; Geometry axis
$MC_AXCONF_GEOAX_NAME_TAB[2] =  “Z”         ; Geometry axis
$MC_AXCONF_GEOAX_ASSIGN_TAB[0] = 1          ; X as channel axis 1
$MC_AXCONF_GEOAX_ASSIGN_TAB[1] = 2          ; Y not channel axis
$MC_AXCONF_GEOAX_ASSIGN_TAB[2] =  3         ; Z as channel axis 2
$MC_AXCONF_CHANAX_NAME_TAB[0] = “XC”;
$MC_AXCONF_CHANAX_NAME_TAB[1] = “YC”;
$MC_AXCONF_CHANAX_NAME_TAB[2] =  “ZC”;
$MC_AXCONF_CHANAX_NAME_TAB[3] =  “CC”;
$MC_AXCONF_CHANAX_NAME_TAB[4] =  “ASC”;
$MC_AXCONF_MACHAX_USED[0] = 2       ;X as machine axis 2
$MC_AXCONF_MACHAX_USED[1] = 3       ;Y as machine axis 3
$MC_AXCONF_MACHAX_USED[2] = 4       ;Z as machine axis 4
$MC_AXCONF_MACHAX_USED[3] = 1       ;C as machine axis 1
$MC_AXCONF_MACHAX_USED[4] = 5       ;AS as machine axis 5
$MA_SPIND_ASSIGN_TO_MACHAX[AX1]= 1 ; C is spindle 1
$MA_SPIND_ASSIGN_TO_MACHAX[AX2]= 0 ; X is not spindle
$MA_SPIND_ASSIGN_TO_MACHAX[AX3]= 0 ; Y is not spindle
$MA_SPIND_ASSIGN_TO_MACHAX[AX4]= 0 ; Z is not spindle
$MA_SPIND_ASSIGN_TO_MACHAX[AX5]= 2 ; AS is spindle 2
$MN_AXCONF_MACHAX_NAME_TAB[0]= >“CM” ; 1st machine axis
$MN_AXCONF_MACHAX_NAME_TAB[1]= “XM” ; 2nd machine axis
$MN_AXCONF_MACHAX_NAME_TAB[2]= “YM” ; 3rd machine axis
$MN_AXCONF_MACHAX_NAME_TAB[3]= “ZM” ; 4th machine axis
$MN_AXCONF_MACHAX_NAME_TAB[4]= “ASM” ; 5th machine axis
```

```
;Prepare for TRACYL (first and only transformation)

$MC_TRAFO_TYPE_1 = 513 ; TRACYL transformation with groove wall
                         compensation
$MC_TRAFO_AXES_IN_1[0] = 1 ; Channel axis radial to rotary axis
$MC_TRAFO_AXES_IN_1[1] = 4 ; Channel axis in cylinder generated
                         surface perpendicular to rotary axis
$MC_TRAFO_AXES_IN_1[2] =  3 ; Channel parallel to rotary axis
$MC_TRAFO_AXES_IN_1[3] =  2 ; Channel axis is add. axis for index [0]

$MC_TRAFO_GEOAX_ASSIGN_TAB_1 [0] = 1 ; X 1st channel axis
$MC_TRAFO_GEOAX_ASSIGN_TAB_1 [1] = 4 ; Y 2nd channel axis
$MC_TRAFO_GEOAX_ASSIGN_TAB_1 [2] = 3 ; Z 3rd channel axis

$MC_TRACYL_ROT_AX_OFFSET_1 = 0.  ; Rotational position X-Y plane
                                   in rel. to zero pos. of rotary axis
$MC_TRACYL_ROT_SIGN_IS_PLUS_1 = FALSE ; Rotary axis turns –
$MC_TRACYL_BASE_TOOL_1  [0] = 0.0 ; Tool distance in X
$MC_TRACYL_BASE_TOOL_1  [1] = 0.0 ; Tool distance in Y
$MC_TRACYL_BASE_TOOL_1  [2] = 0.0 ; Tool distance in Z


;  Activation of
TRACYL(40.0)

;  See below for programming in Y and Z

;  Return to rotational operation
TRAFOOF
```

Programming with groove wall compensation

(TRAFO_TYPE_n=513)

Contour

It is possible to produce a groove which is wider than the tool by using address OFFN to program the compensation direction (G41, G42) in relation to the programmed reference contour and the distance of the groove side wall from the reference contour (see Fig. 6-1).

Tool radius

The tool radius is automatically taken into account with respect to the groove side wall (see Fig. 6-1). The full functionality of the plane tool radius compensation is available (steady transition at outer and inner corners as well as solution of bottleneck problems).

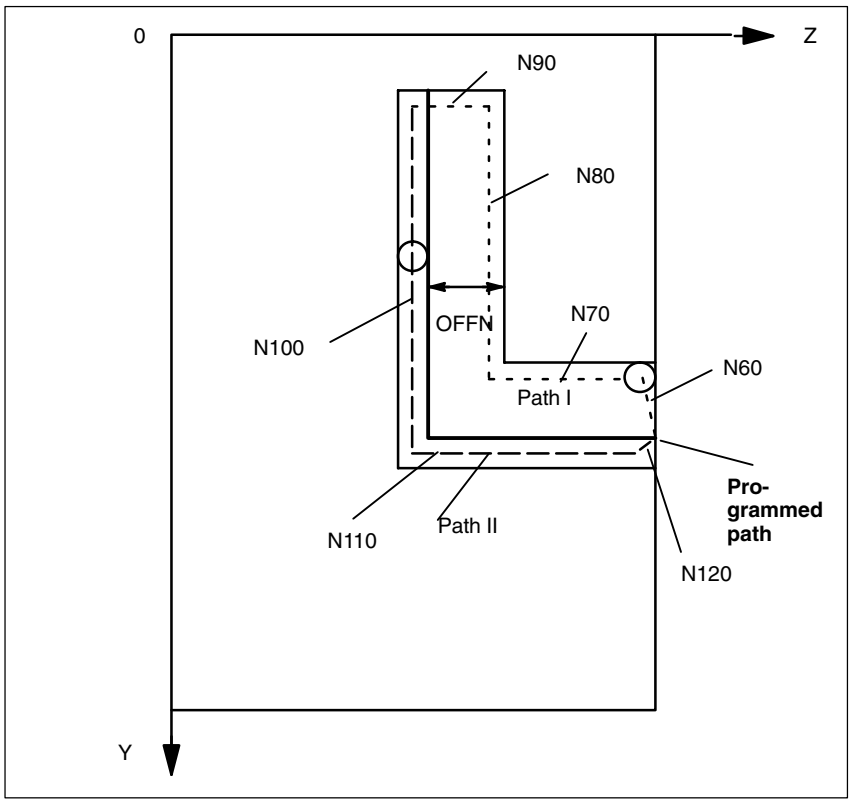


Fig. 6-1 Groove with wall compensation, cylinder coordinates (simplified sketch)

; Example program which leads the tool along path I across path II back to the
; initial position after transformation selection (machine data see Chapter 4,
; example X-Y-Z-C kinematics):

```
N1   SPOS=0;                Transfer of spindle to rotary axis mode
N5    G0 X25 Y0 Z105 CC=200 F5000 G64;
                            Positioning of machine above groove center
N10 TRACYL(40.) ;           Transformation selection with
                            ; reference diameter 40 mm
N20  G19 G90;               Machining plane is generated cylinder surface
N30   T1 D1;                Tool selection, can also be positioned before
                            ; TRACYL (..)
N40   G1 X20;               Infeed tool to groove base
N50   OFFN=12. ;            Define groove wall distance, must not be
                            ; in a separate line

; Approach groove wall
N60 G1 Z100 G42;            TRC selection to approach groove wall

; Machining of groove section path I
N70 G1 Z50;                 Groove section parallel to cylinder plane
N80 G1 Y10;                 Groove section parallel to periphery
```

```
                                ; Approach groove wall for path II
N90 OFFN=4 G42;                 Define groove wall distance and
                                ; TRC selection to approach groove wall

; Machine groove section path II
N100 G1 Y70;                    corresponds to CC=200 degrees
N110 G1 Z100;                   back to initial value

; Retract from groove wall
N120 G1 Z105 G40;               TRC deselection to retract from groove wall
N130 G0 X25;                    Retract from groove

N140 TRAFOOF;
N150 G0 X25 Y0 Z105 CC=200 D0;
                                Return to initial point and
                                ; deselect tool compensation
N160 M30
```

6.2 TRAANG (Inclined axis)

The following programming example relates to the configuration illustrated in Fig. 6-1 and shows the sequence of main steps required to configure the axes and activate TRAANG.

```
;General axis configuration for grinding

$MC_AXCONF_GEOAX_NAME_TAB[0] = “X”        ; Geometry axis
$MC_AXCONF_GEOAX_NAME_TAB[1] = “”         ; Geometry axis
$MC_AXCONF_GEOAX_NAME_TAB[2] =  “Z”       ; Geometry axis
$MC_AXCONF_GEOAX_ASSIGN_TAB[0] =0         ; X not channel axis
$MC_AXCONF_GEOAX_ASSIGN_TAB[1] = 0        ; Y not channel axis
$MC_AXCONF_GEOAX_ASSIGN_TAB[2] =  1       ; Z as channel axis 1
$MC_AXCONF_CHANAX_NAME_TAB[0] = “Z”;
$MC_AXCONF_CHANAX_NAME_TAB[1] = “C”;
$MC_AXCONF_CHANAX_NAME_TAB[2] =  “AS”;
$MC_AXCONF_CHANAX_NAME_TAB[3] =  “MU”
$MC_AXCONF_MACHAX_USED[0] = 3             ;Z as machine axis 3
$MC_AXCONF_MACHAX_USED[1] = 1             ;C as machine axis 1
$MC_AXCONF_MACHAX_USED[2] = 4             ;AS as machine axis 4
$MC_AXCONF_MACHAX_USED[3] = 2             ;MU as machine axis 2
$MC_AXCONF_MACHAX_USED[3] = 0             ;empty
$MC_AXCONF_MACHAX_USED[3] = 0             ;empty
$MC_SPIND_ASSIGN_TO_MACHAX[AX1]= 1 ; C is spindle 1
$MC_SPIND_ASSIGN_TO_MACHAX[AX2]= 0 ; X is not spindle
$MC_SPIND_ASSIGN_TO_MACHAX[AX3]= 0 ; Z is not spindle
$MC_SPIND_ASSIGN_TO_MACHAX[AX4]= 2 ; AS is spindle 2
$MC_AXCONF_MACHAX_NAME_TAB[0]= “C1” ; 1st machine axis
$MC_AXCONF_MACHAX_NAME_TAB[1]= “MU” ; 2nd machine axis
$MC_AXCONF_MACHAX_NAME_TAB[2]= “MZ” ; 3rd machine axis
$MC_AXCONF_MACHAX_NAME_TAB[3]= “AS” ; 4th machine axis

 ; Prepare for TRAANG (first and only transformation)
```

```
$MC_TRAFO_TYPE_1 = 1024 ; TRAANG transformation
$MC_TRAFO_AXES_IN_1[0] =  4 ; Channel axis “inclined axis”
$MC_TRAFO_AXES_IN_1[1] =  1 ; Channel axis parallel to axis Z
$MC_TRAFO_AXES_IN_1[2] =  0 ; Channel axis not active
$MC_TRAFO_GEOAX_ASSIGN_TAB_1 [0] = 4 ; X 1st channel axis
$MC_TRAFO_GEOAX_ASSIGN_TAB_1 [1] = 0 ; Y 2nd channel axis
$MC_TRAFO_GEOAX_ASSIGN_TAB_1 [2] = 1 ; Z 3rd channel axis
$MC_TRAANG_ANGLE_1 = 30.  ; Angle of inclined axis
$MC_TRAANG_BASE_TOOL_1  [0] = 0 ; Tool distance in X
$MC_TRAANG_BASE_TOOL_1  [1] = 0 ; Tool distance in Y
$MC_TRAANG_BASE_TOOL_1  [2] = 0 ; Tool distance in Z

TRAANG        ; Activation

              ; Programming in X, Y, Z

TRAFOOF       ; Return to turning mode
```

6.3 Chained transformations

Example

The following elements are determined in the next chapter:

- The general channel configuration
- Single transformations
- Chained transformations consisting of previously defined single transformations
- Activation of single transformations
- Activation of chained transformations

The examples include the following transformations:

1. 5-axis transformation with rotatable tool and axis sequence AB (trafo type 16)
2. Transmit (trafo type 256)
3. Inclined axis (trafo type 1024)
4. Chaining of the 1st and 3rd transformation (trafo type 8192)
5. Chaining of the 2nd and 3rd transformation (trafo type 8192)

General channel configuration

```
CHANDATA (1)    ; Channel data in channel 1
$MC_AXCONF_MACHAX_USED[0] = 1
$MC_AXCONF_MACHAX_USED[1] = 2
$MC_AXCONF_MACHAX_USED[2] = 3
$MC_AXCONF_MACHAX_USED[3] = 4
$MC_AXCONF_MACHAX_USED[4] = 5
$MC_AXCONF_MACHAX_USED[5] = 6
$MC_AXCONF_MACHAX_USED[6] = 7
$MC_AXCONF_MACHAX_USED[7] = 0

$MC_AXCONF_CHANAX_NAME_TAB[3]=“A”
$MC_AXCONF_CHANAX_NAME_TAB[4]=“B”
$MC_AXCONF_CHANAX_NAME_TAB[5]=“C”

$MA_IS_ROT_AX[ AX4 ] = TRUE
$MA_IS_ROT_AX[ AX5 ] = TRUE
$MA_IS_ROT_AX[ AX6 ] = TRUE
$MA_IS_ROT_AX[ AX7 ] = TRUE

$MA_SPIND_ASSIGN_TO_MACHAX[ AX5 ] = 0
$MA_SPIND_ASSIGN_TO_MACHAX[AX7] = 1
$MA_ROT_IS_MODULO[AX7] = TRUE
```

Single transformations

```
; 1st TRAORI
$MC_TRAFO_TYPE_1= 16          ; TRAORI: A-B kinematics
$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_AXES_IN_1[5]=0
$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_BASE_TOOL_1[0]=0
$MC_TRAFO5_BASE_TOOL_1[1]=0
$MC_TRAFO5_BASE_TOOL_1[2]=0

; 2nd TRANSMIT
$MC_TRAFO_TYPE_2 = 256  ;TRANSMIT
$MC_TRAFO_AXES_IN_2[0]=1
$MC_TRAFO_AXES_IN_2[1]=6
$MC_TRAFO_AXES_IN_2[2]=3
$MC_TRAFO_AXES_IN_2[3]=0
$MC_TRAFO_AXES_IN_2[4]=0
$MC_TRAFO_AXES_IN_2[5]=0
$MC_TRAFO_AXES_IN_2[6]=0
$MC_TRAFO_GEOAX_ASSIGN_TAB_2[0]=1
$MC_TRAFO_GEOAX_ASSIGN_TAB_2[1]=6
$MC_TRAFO_GEOAX_ASSIGN_TAB_2[2]=3

; 3rd TRAANG
$MC_TRAFO_TYPE_3 = 1024  ;TRAANG
$MC_TRAFO_AXES_IN_3[0] = 1
$MC_TRAFO_AXES_IN_3[1] = 3
$MC_TRAFO_AXES_IN_3[2] = 2
$MC_TRAFO_AXES_IN_3[3] = 0
$MC_TRAFO_AXES_IN_3[4] = 0
$MC_TRAFO_GEOAX_ASSIGN_TAB_3[0] = 1
$MC_TRAFO_GEOAX_ASSIGN_TAB_3[1] = 3
$MC_TRAFO_GEOAX_ASSIGN_TAB_3[2] = 2
$MC_TRAANG_ANGLE_1 = 45.
$MC_TRAANG_PARALLEL_VELO_RES_1 = 0.2
$MC_TRAANG_PARALLEL_ACCEL_RES_1 = 0.2
$MC_TRAANG_BASE_TOOL_1[0] = 0.0
$MC_TRAANG_BASE_TOOL_1[1] = 0.0
$MC_TRAANG_BASE_TOOL_1[2] = 0.0
```

Chained transformations

```
; 4th TRACON  (chaining TRAORI / TRAANG)
$MC_TRAFO_TYPE_4 = 8192

$MC_TRAFO_GEOAX_ASSIGN_TAB_4[0] = 2
$MC_TRAFO_GEOAX_ASSIGN_TAB_4[1] = 1
$MC_TRAFO_GEOAX_ASSIGN_TAB_4[2] = 3

$MC_TRACON_CHAIN_1[0] = 1
$MC_TRACON_CHAIN_1[1] = 3
$MC_TRACON_CHAIN_1[2] = 0

; 5th TRACON  (chaining TRANSMIT / TRAANG)

$MC_TRAFO_TYPE_5 = 8192

$MC_TRAFO_GEOAX_ASSIGN_TAB_5[0] = 1
$MC_TRAFO_GEOAX_ASSIGN_TAB_5[1] = 6
$MC_TRAFO_GEOAX_ASSIGN_TAB_5[2] = 3

$MC_TRACON_CHAIN_2[0] = 2
$MC_TRACON_CHAIN_2[1] = 3
$MC_TRACON_CHAIN_2[2] = 0
```

Part program (extracts)

Example for an NC program which uses the set transformations:

; Call single transformations

```
                          ; Tool definition
$TC_DP1[1,1]=120          ; Tool type
$TC_DP3[1,1] = 10         ; Tool length

n2 x0 y0 z0 a0 b0 f20000 t1 d1
n4 x20

n30 TRANSMIT                      ; Switch on Transmit
n40 x0 y20
n50 x-20 y0
n60 x0 y-20
n70 x20 y0
n80 TRAFOOF                       ; Switch off Transmit


n130 TRAANG(45.)          ; Switch on inclined axis transformation, parameter: angle 45°
n140 x0 y0 z20
n150 x–20 z0
n160 x0 z–20
n170 x20 z0
```

Note

The above examples assume that the angle of the “inclined axis” can be set on the machine and is set to 0 degrees when the single transformation is activated.

```
                                  ; 1st chained transformation
                                  ; TRAORI + TRAANG

n230 TRACON(1, 45.)         ; 1st of 2 chained transformations
                            ; The previously active transformation TRAANG is automatically deselected
                            ; The parameter for the inclined axis is 45°
n240 x10 y0 z0  a3=–1 C3 =1 oriwks
n250 x10 y20 b3 = 1 c3 = 1
...


                                  ; 2nd chained transformation
                                  ; TRANSMIT + TRAANG

n330 TRACON(2, 40.)         ; 2nd chained transformation
                            ; The parameter for the inclined axis is 40°
n335 x20 y0 z0
n340 x0 y20 z10
n350 x–20 y0 z0
n360 x0 y–20 z0
n370 x20 y0 z0
n380 TRAFOOF                          ; 2nd chained transformation
...

n1000 M30
```

6.4 Activating transformation MD via a part program (SW 5.2 and higher)

It would be permissible in the following example to re-configure (write) a machine data affecting the second transformation (e.g. $MC_TRAFO5_BASE_TOOL_2[2]) in block N90, since writing a machine data alone does not activate it. However, if the program remained otherwise unchanged, an alarm would occur in block N130, because an attempt would then be made to modify an active transformation.

Example program:

```
.
.
.
N40   TRAORI(2)                          ; Select 2nd orientation
                                         ; transformation
N50   X0 Y0 Z0 F20000 T1 T1
N60   A50 B50
N70   A0 B0
N80   X10
N90   $MC_TRAFO5_BASE_TOOL_1[2] =  50 ; Overwrite an MD
                                         ; of the 1st orientation
                                         ; Transformation
N100  A20
N110  X20
N120  X0
N130  NEWCONF                            ; Accept newly modified
                                         ; machine data
N140  TRAORI(1)                          ; Select 1st orientation
                                         ; transformation,
                                         ; MD becomes operative
N150  G19 X0 Y0 Z0
N160  A50 B50
N170  A0 B0
N180  TRAFOOF
N190  M30
```

■

Notes

Data Fields, Lists

7

7.1 TRANSMIT

7.1.1 Interface signals

DB number	Bit, byte	Name	Reference
Channel-specific			
21, ...	33.6	Transformation active	F2

7.1.2 Machine data

Number	Identifier	Name	Reference
Channel-specific ($MC_...)			
20110	RESET_MODE_MASK	Definition of basic control position after power-up and RESET/part program end	K2
20140	TRAFO_RESET_VALUE	Basic transformation position	K2
22534	TRAFO_CHANGE_M_CODE	M code for transformation changeover on geometry axes	
24100	TRAFO_TYPE_1	Definition of the 1st transformation in channel	F2
24110	TRAFO_AXES_IN_1	Axis assignment for the 1st transformation	F2
24120	TRAFO_GEOAX_ASSIGN_TAB_1	Geo-axis assignment for 1st transform.	F2
24200	TRAFO_TYPE_2	Definition of the 2nd transformation in channel	F2
24210	TRAFO_AXES_IN_2	Axis assignment for the 2nd transform.	F2
24220	TRAFO_GEOAX_ASSIGN_TAB_2	Geo-axis assignment for 2nd transform.	F2

Number	Identifier	Name	Reference
24300	TRAFO_TYPE_3	Definition of the 3rd transformation in channel	F2
24310	TRAFO_AXES_IN_3	Axis assignment for the 3rd transform.	F2
24320	TRAFO_GEOAX_ASSIGN_TAB_3	Geo-axis assignment for 3rd transform.	F2
24400	TRAFO_TYPE_4	Definition of the 4th transformation in channel	F2
24410	TRAFO_AXES_IN_4	Axis assignment for the 4th transformm	F2
24420	TRAFO_GEOAX_ASSIGN_TAB_4	Geo-axis assignment for 4th transform.	F2
24430	TRAFO_TYPE_5	Definition of the 5th transformation in channel	F2
24432	TRAFO_AXES_IN_5	Axis assignment for the 5th transform.	F2
24434	TRAFO_GEOAX_ASSIGN_TAB_5	Geo-axis assignment for 5th transform.	F2
24440	TRAFO_TYPE_6	Def. of the 6th transformation in channel	F2
24442	TRAFO_AXES_IN_6	Axis assignment for the 6th transform.	F2
24444	TRAFO_GEOAX_ASSIGN_TAB_6	Geo-axis assignment for 6th transform.	F2
24450	TRAFO_TYPE_7	Definition of the 7th transf. in channel	F2
24452	TRAFO_AXES_IN_7	Axis assignment for the 7th transform.	F2
24454	TRAFO_GEOAX_ASSIGN_TAB_7	Geo-axis assignment for 7th transform.	F2
24460	TRAFO_TYPE_8	Definition of the 8th transf. in channel	F2
24462	TRAFO_AXES_IN_8	Axis assignment for the 8th transform.	F2
24464	TRAFO_GEOAX_ASSIGN_TAB_8	Geo-axis assignment for 8th transform.	F2
24900	TRANSMIT_ROT_AX_OFFSET_1	Deviation of rotary axis from zero position in degrees (1st TRANSMIT)	
24910	TRANSMIT_ROT_SIGN_IS_PLUS_1	Sign of rotary axis for TRANSMIT (1st TRANSMIT)	
24911	TRANSMIT_POLE_SIDE_FIX_1	Limitation of working range in front of/behind pole, 1st transformation	
24920	TRANSMIT_BASE_TOOL_1	Distance of tool zero point from origin of geo-axes (1st TRANSMIT)	
24950	TRANSMIT_ROT_AX_OFFSET_2	Deviation of rotary axis from zero position in degrees (2nd TRANSMIT)	
24960	TRANSMIT_ROT_SIGN_IS_PLUS_2	Sign of rotary axis for TRANSMIT (2nd TRANSMIT)	
24961	TRANSMIT_POLE_SIDE_FIX_2	Limitation of working range in front of/behind pole, 2nd transformation	
24970	TRANSMIT_BASE_TOOL_2	Distance of tool zero point from origin of geo-axes (2nd TRANSMIT)	

7.1.3 Alarms

The alarms which may occur in conjunction with the TRANSMIT transformation are explained in the Diagnostic Guide and in the online help in systems with MMC 101/102/103

References: /DA/, Diagnostic Guide

7.2 TRACYL

7.2.1 Interface signals

DB number	Bit, byte	Name	Reference
Channel-specific			
21, ...	33.6	Transformation active	F2

7.2.2 Machine data

Number	Identifier	Name	Reference
Channel-specific ($MC_...)			
20110	RESET_MODE_MASK	Definition of basic control position after power-up and RESET/part program end	K2
20140	TRAFO_RESET_VALUE	Basic transformation position	K2
24100	TRAFO_TYPE_1	Definition of the 1st transformation in channel	F2
24110	TRAFO_AXES_IN_1	Axis assignment for the 1st transform.	F2
24120	TRAFO_GEOAX_ASSIGN_TAB_1	Geo-axis assignment for 1st transform.	F2
24200	TRAFO_TYPE_2	Definition of the 2nd transformation in channel	F2
24210	TRAFO_AXES_IN_2	Axis assignment for the 2nd transform.	F2
24220	TRAFO_GEOAX_ASSIGN_TAB_2	Geo-axis assignment for 2nd transform.	F2
24300	TRAFO_TYPE_3	Definition of the 3rd transformation in channel	F2
24310	TRAFO_AXES_IN_3	Axis assignment for the 3rd transform.	F2
24320	TRAFO_GEOAX_ASSIGN_TAB_3	Geo-axis assignment for 3rd transform.	F2
24400	TRAFO_TYPE_4	Definition of the 4th transformation in channel	F2
24410	TRAFO_AXES_IN_4	Axis assignment for the 4th transform.	F2
24420	TRAFO_GEOAX_ASSIGN_TAB_4	Geo-axis assignment for 4th transform.	F2
24430	TRAFO_TYPE_5	Definition of the 5th transformation in channel	F2
24432	TRAFO_AXES_IN_5	Axis assignment for the 5th transform.	F2
24434	TRAFO_GEOAX_ASSIGN_TAB_5	Geo-axis assignment for 5th transform.	F2
24440	TRAFO_TYPE_6	Definition of the 6th transformation in channel	F2
24442	TRAFO_AXES_IN_6	Axis assignment for the 6th transform.	F2
24444	TRAFO_GEOAX_ASSIGN_TAB_6	Geo-axis assignment for 6th transform.	F2
24450	TRAFO_TYPE_7	Definition of the 7th transf. in channel	F2
24452	TRAFO_AXES_IN_7	Axis assignment for the 7th transform.	F2

Number	Identifier	Name	Reference
24454	TRAFO_GEOAX_ASSIGN_TAB_7	Geo-axis assignment for 7th transform.	F2
24460	TRAFO_TYPE_8	Definition of the 8th transf. in channel	F2
24462	TRAFO_AXES_IN_8	Axis assignment for the 8th transform.	F2
24464	TRAFO_GEOAX_ASSIGN_TAB_8	Geo-axis assignment for 8th transformation	F2
24800	TRACYL_ROT_AX_OFFSET_1	Deviation of rotary axis from zero position in degrees (1st TRACYL)	
24810	TRACYL_ROT_SIGN_IS_PLUS_1	Sign of rotary axis for TRACYL (1st TRACYL)	
24820	TRACYL_BASE_TOOL_1	Distance of tool zero point from origin of geo-axes (1st TRACYL)	
24850	TRACYL_ROT_AX_OFFSET_2	Deviation of rotary axis from zero position in degrees (2nd TRACYL)	
24860	TRACYL_ROT_SIGN_IS_PLUS_2	Sign of rotary axis for TRACYL (2nd TRACYL)	
24870	TRACYL_BASE_TOOL_2	Distance of tool zero point from origin of geo-axes (2nd TRACYL)	
22534	TRAFO_CHANGE_M_CODE	M code for transformation changeover on geometry axes	

7.2.3 Alarms

The alarms which may occur in conjunction with the TRACYL transformation are explained in the Diagnostic Guide and in the online help in systems with MMC 101/102/103

References: /DA/, Diagnostic Guide

7.3 TRAANG (Inclined axis)

7.3.1 Interface signals

DB number	Bit, byte	Name	Reference
Channel-specific			
21, ...	33.6	Transformation active	F2

7.3.2 Machine data

Number	Identifier	Name	Reference
Channel-specific ($MC_...)			
20110	RESET_MODE_MASK	Definition of basic control position after power-up and RESET/part program end	K2
20140	TRAFO_RESET_VALUE	Basic transformation position	K2
22534	TRAFO_CHANGE_M_CODE	M code for transformation changeover on geometry axes	
24100	TRAFO_TYPE_1	Definition of the 1st transformation in channel	F2
24110	TRAFO_AXES_IN_1	Axis assignment for the 1st transform.	F2
24120	TRAFO_GEOAX_ASSIGN_TAB_1	Geo-axis assignment for 1st transform.	F2
24200	TRAFO_TYPE_2	Definition of the 2nd transformation in channel	F2
24210	TRAFO_AXES_IN_2	Axis assignment for the 2nd transform.	F2
24220	TRAFO_GEOAX_ASSIGN_TAB_2	Geo-axis assignment for 2nd transform.	F2
24300	TRAFO_TYPE_3	Definition of the 3rd transformation in channel	F2
24310	TRAFO_AXES_IN_3	Axis assignment for the 3rd transform.	F2
24320	TRAFO_GEOAX_ASSIGN_TAB_3	Geo-axis assignment for 3rd transform.	F2
24400	TRAFO_TYPE_4	Definition of the 4th transformation in channel	F2
24410	TRAFO_AXES_IN_4	Axis assignment for the 4th transform.	F2
24420	TRAFO_GEOAX_ASSIGN_TAB_4	Geo-axis assignment for 4th transform.	F2
24430	TRAFO_TYPE_5	Definition of the 5th transformation in channel	F2
24432	TRAFO_AXES_IN_5	Axis assignment for the 5th transform.	F2
24434	TRAFO_GEOAX_ASSIGN_TAB_5	Geo-axis assignment for 5th transform.	F2
24440	TRAFO_TYPE_6	Definition of the 6th transf. in channel	F2
24442	TRAFO_AXES_IN_6	Axis assignment for the 6th transform.	F2
24444	TRAFO_GEOAX_ASSIGN_TAB_6	Geo-axis assignment for 6th transform.	F2

Number	Identifier	Name	Reference
24450	TRAFO_TYPE_7	Definition of the 7th transformation in channel	F2
24452	TRAFO_AXES_IN_7	Axis assignment for the 7th transform.	F2
24454	TRAFO_GEOAX_ASSIGN_TAB_7	Geo-axis assignment for 7th transform.	F2
24460	TRAFO_TYPE_8	Definition of the 8th transformation in channel	F2
24462	TRAFO_AXES_IN_8	Axis assignment for the 8th transform.	F2
24464	TRAFO_GEOAX_ASSIGN_TAB_8	Geo-axis assignment for 8th transform.	F2
24700	TRAANG_ANGLE_1	Angle of inclined axis in degrees (1st TRAANG)	
24710	TRAANG_BASE_TOOL_1	Distance of tool zero point from origin of geo-axes (1st TRAANG)	
24720	TRAANG_PARALLEL_VELO_RES_1	Velocity reserve of parallel axis for compensatory motion (1st TRAANG)	
24721	TRAANG_PARALLEL_VELO_RES_2	Velocity reserve of parallel axis for compensatory motion (2nd TRAANG)	
24750	TRAANG_ANGLE_2	Angle of inclined axis in degrees (2nd TRAANG)	
24760	TRAANG_BASE_TOOL_2	Distance of tool zero point from origin of geo-axes (2nd TRAANG)	
24770	TRAANG_PARALLEL_ACCEL_RES_1	Axis acceleration reserve of parallel axis for compensatory motion (1st TRAANG)	
24771	TRAANG_PARALLEL_ACCEL_RES_2	Axis acceleration reserve of parallel axis for compensatory motion (2nd TRAANG)	

7.3.3 Alarms

The alarms which may occur in conjunction with the TRAANG transformation are explained in the Diagnostic Guide and in the online help in systems with MMC 101/102/103.

References: /DA/, Diagnostic Guide

7.4 TRACON (chained transformations)

Number	Identifier	Name	Reference
Channel-specific ($MC_...)			
24995	TRACON_CHAIN_1	Transformation chain of the first chained transformation	
24996	TRACON_CHAIN_2	Transformation chain of the second chained transformation	

7.5 Non transformation-specific machine data

Number	Identifier	Name	Reference
Channel-specific ($MC_...)			
21090	X_AXIS_IN_OLD_X_Z_PLANE	Coordinate system with automatic frame definition	
21090	MAX_LEAD_ANGLE	Maximum permissible lead angle for orientation programming	
21092	MAX_TILT_ANGLE	Maximum permissible side angle for orientation programming	
21100	ORIENTATION_IS_EULER	Angle definition for orientation programming	F2

■

Notes

SINUMERIK 840D/810D/FM-NC Description of Functions, Extension Functions (FB2)

Measurements (M5)

■

Notes

Brief Description

1

Channel-specific measuring

Channel-specific measuring is available with SW 1 and higher.
A trigger event which initiates the measuring operation and defines a corresponding measuring mode is programmed in a part program block. The instructions apply to all axes programmed in this particular block.

Axial measuring

Axial measuring is available with SW 4.1 and higher.
A trigger event which initiates a measuring operation is programmed in a part program block. A measuring mode for the measurement is defined together with the axis in which the measurements must be taken.

Measuring cycles

A description of how to handle measuring cycles can be found in

References: /FB III/, Measuring Cycles (M4)

■

Notes

Detailed Description

2

2.1 Hardware requirements

2.1.1 Suitable probes

General

In order to sense the dimensions of tool and workpiece, a touch trigger probe that outputs a constant signal (not a pulse) on deflection is required.

The sensor must operate virtually bounce-free. Most sensors can be adjusted mechanically to ensure that they operate in this manner.

Different types of probe supplied by a variety of manufacturers are available on the market. Probes are therefore divided into three groups according to the number of directions in which they can be deflected (see figure below).

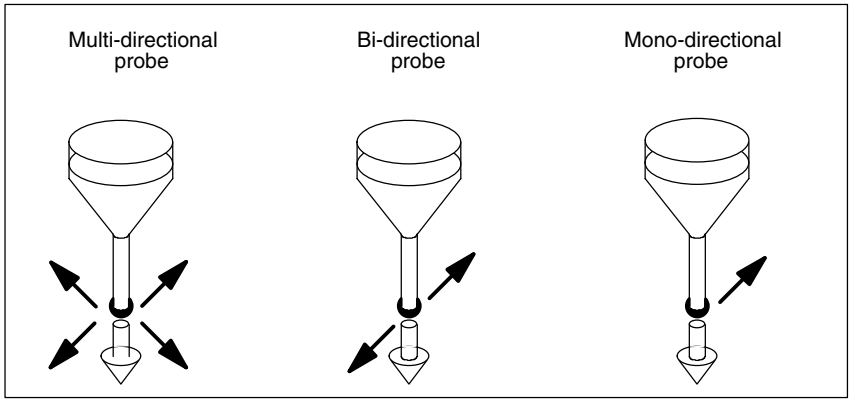


Fig. 2-1 Probe types

Table 2-1 Assignment between probe type and application

Probe type	Turning machines		Milling and machining centers
	Tool measurements	Workpiece measurements	Workpiece measurements
Multi-directional	X	X	X
Bi-directional	_	X	X
Mono-directional	_	_	X

Bi-directional probes must be used on turning machines for workpiece measurements, whereas a mono-probe can also be used for this purpose on milling and machining centers.

Multi-directional probe (3D)

This probe type can be used unconditionally for measuring tool and workpiece dimensions.

Bi-directional probe

This probe type is applied in the same way as a mono-probe in milling and machining centers. Bi-directional probes can be used to take workpiece measurements on turning machines.

Mono-directional probe

This probe type can be used subject to some restrictions to take workpiece measurements on milling and machining centers.

Spindle position with mono-probe

To be able to use this probe type on milling and machining centers, it must be possible to position the spindle with NC function SPOS and to transfer the switching signal from the probe over 360° to the receiver station (on machine stator).

The probe must be mechanically aligned in the spindle such that it can take measurements in the following directions when the spindle is positioned at 0 degrees.

Table 2-2 Spindle positions for alignment of probe

	Measurements at 0 degrees spindle position
X-Y plane G17	Positive X direction
Z-X plane G18	Positive Z direction
Y-Z plane G19	Positive Y direction

The measurement takes longer with a mono-probe as the spindle needs to be positioned several times with SPOS in the measuring cycle.

2.1.2 Connection of probe

Connection to 840D, 810D

The probe is connected to the SINUMERIK 840D or 810D system via the I/O device interface X121 located on the front plate of the NCU module.

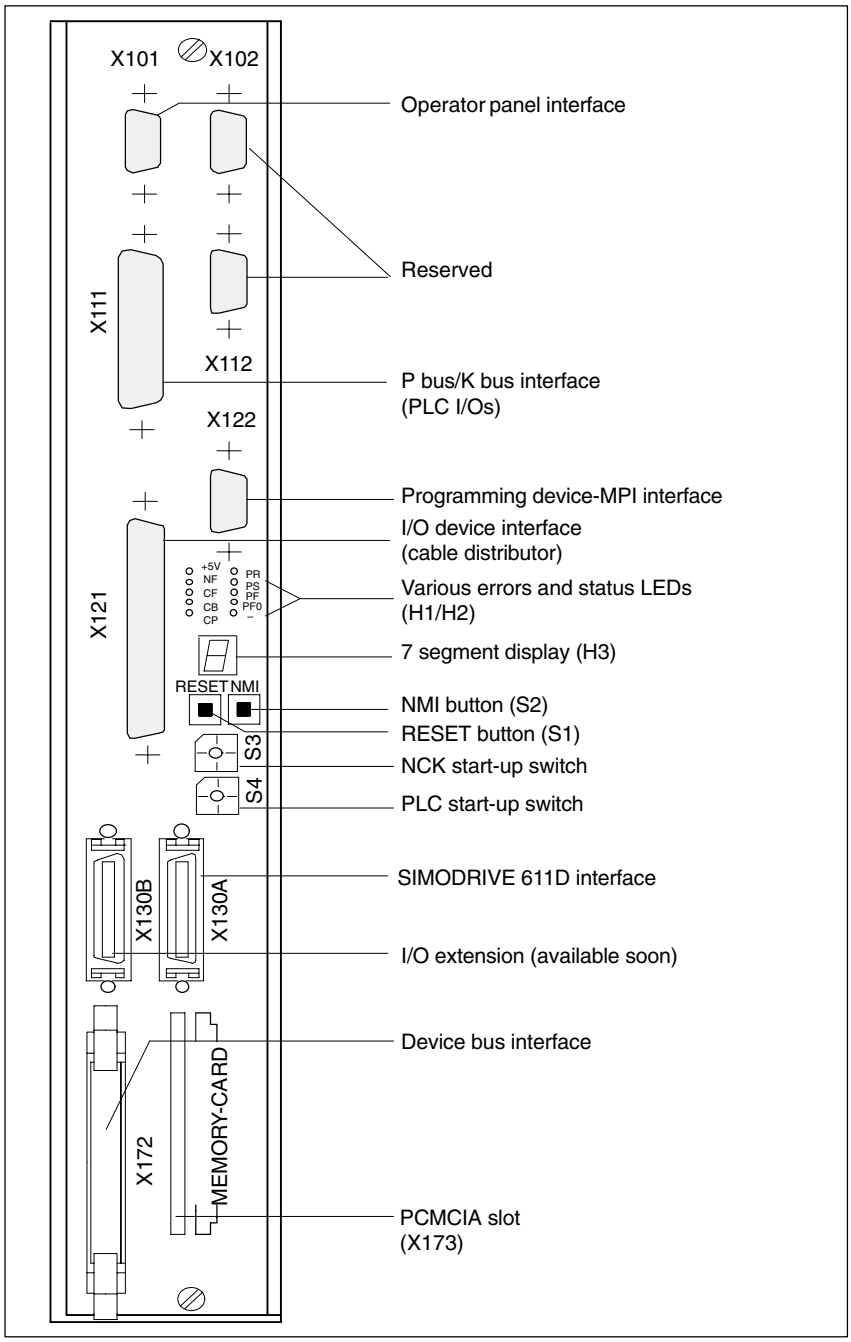


Fig. 2-2 Interfaces, operating and display elements on NCU module

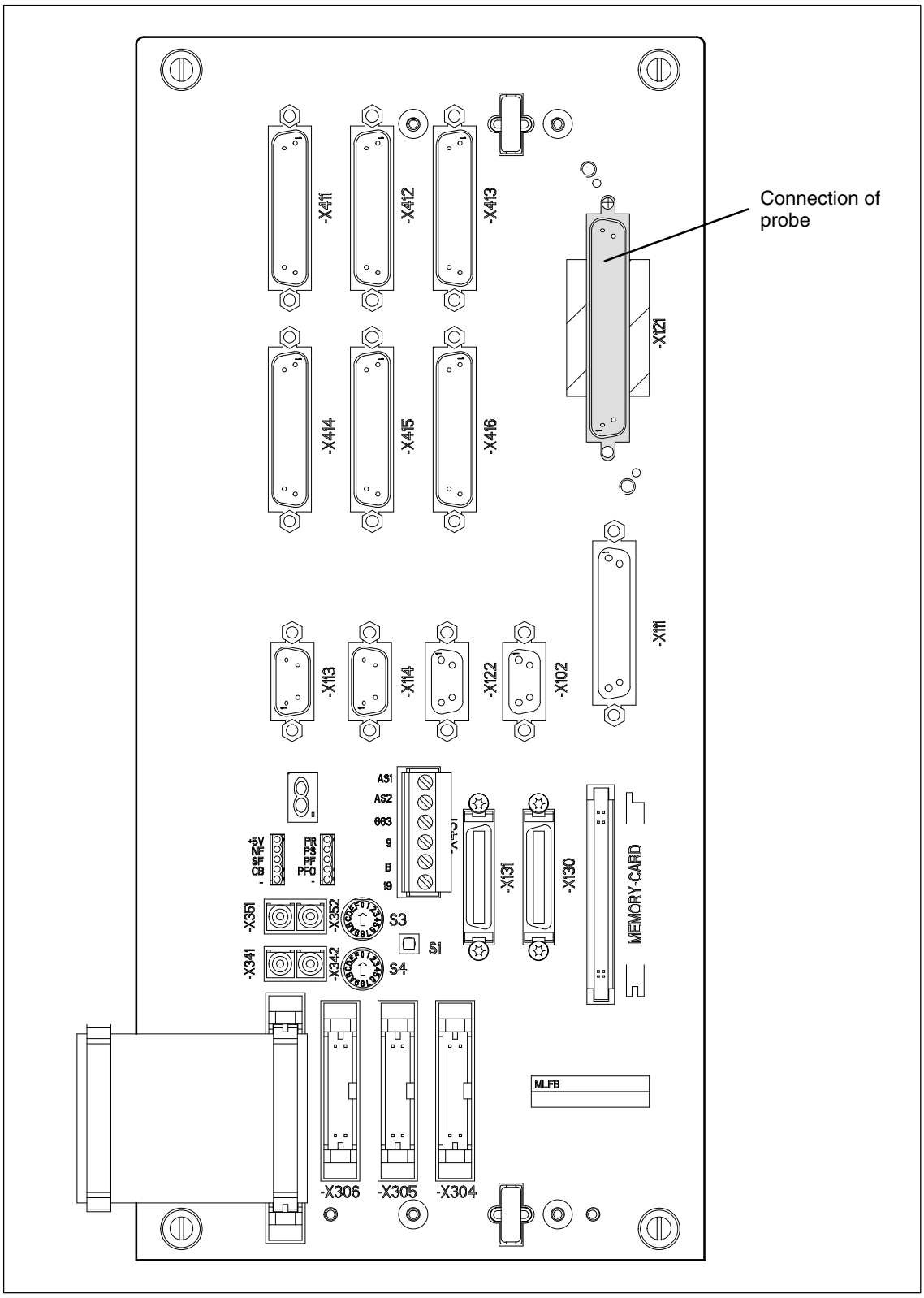


Fig. 2-3 Interfaces, operating and display elements on SINUMERIK 810D

Interface

The interface connection for a probe is made via the

- **I/O device interface**

  37-pin D-sub plug connector (X121), a **maximum of 2** probes can be connected;

The 24 V load power supply for the binary inputs is also connected to X121.

Table 2-3 Extract from PIN assignment table for front connector X121

	PIN		Designation
X121			External power supply
	1	M24EXT	External ground
	2	M24EXT	External ground
	...	...	...
			Connection for probe 1
	9	MEPUS 0	Measuring pulse signal input
	10	MEPUC 0	Measuring pulse common input
	...	...	...
			External power supply
	20	P24EXT	P 24 V external
	21	P24EXT	P 24 V external
	...	...	...
			Connection of probe 2
	28	MEPUS 1	Measuring pulse signal input
	29	MEPUC 1	Measuring pulse common input
	...	...	...

The interfaces (e.g. pin assignments) are illustrated and described in detail in

References: /PHD/, Hardware Planning Guide

The cable distributor is the same as that used on the SINUMERIK 840C.

Connection to FM-NC NCU 570.2

The figure below shows the interface on the FM-NC for connecting a probe.

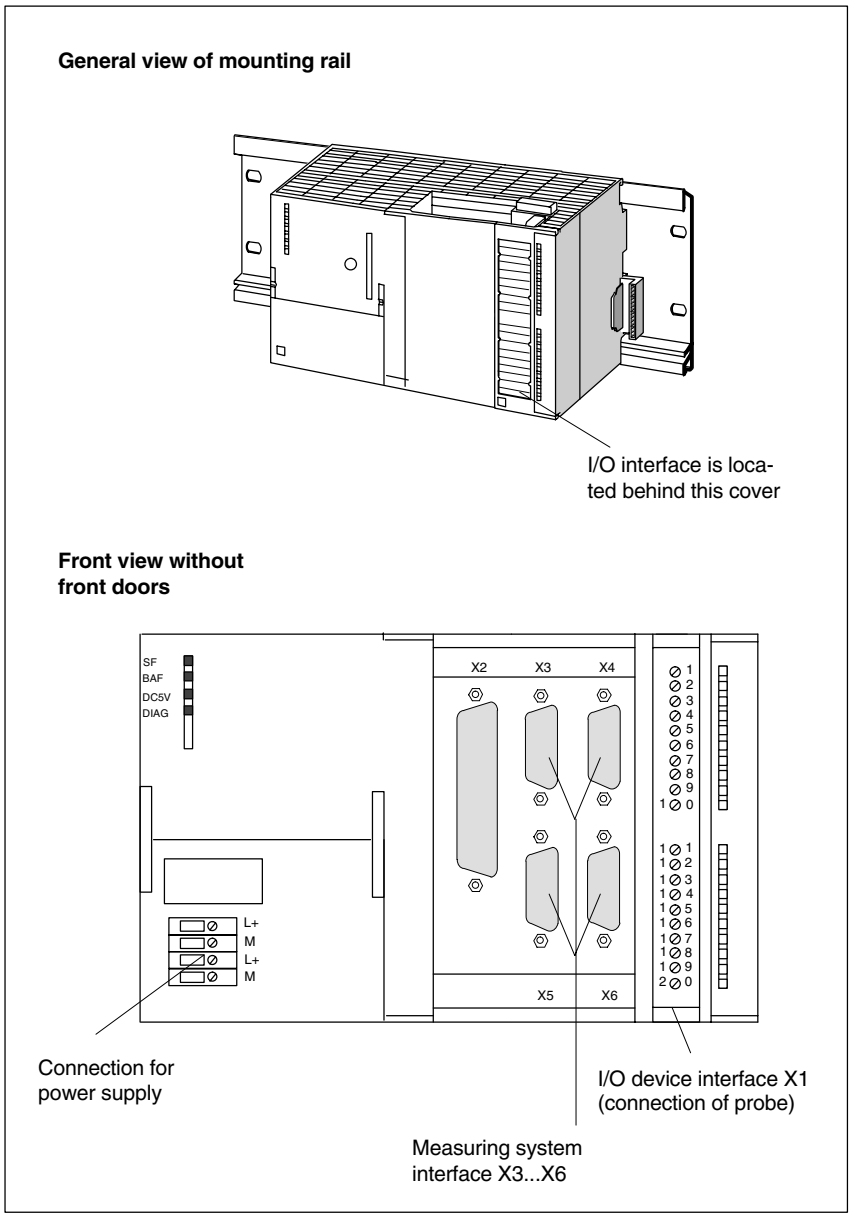


Fig. 2-4 Block diagram showing probe connection to FM-NC

Interfaces

The interface connection for a probe is made via interface:

- **I/O device interface**

  20-pin front connector (X1) for connecting handwheels (maximum 2) and high-speed inputs including probes and for wiring NC-READY relay

Table 2-4 Extract from PIN assignment table for front connector X1

	PIN	MD 30120 CTRLOUT_NR	Designation
X1: Connection of handwheels and I/O devices, 20-pin front connector			
X1	...	...	...
	17	–	Digit. input 3/measuring pulse input 1 (DE3/MEPU1)
	18	–	Digit. input 3/measuring pulse input 2 (DE3/MEPU2)
	20	–	M24EXT external ground

- **Power supply connection**

  4-pin screw terminal block (X10) for connecting 24 V load power supply

For further details, please refer to the “Installation and Start-Up Guide SINUMERIK FM-NC”.

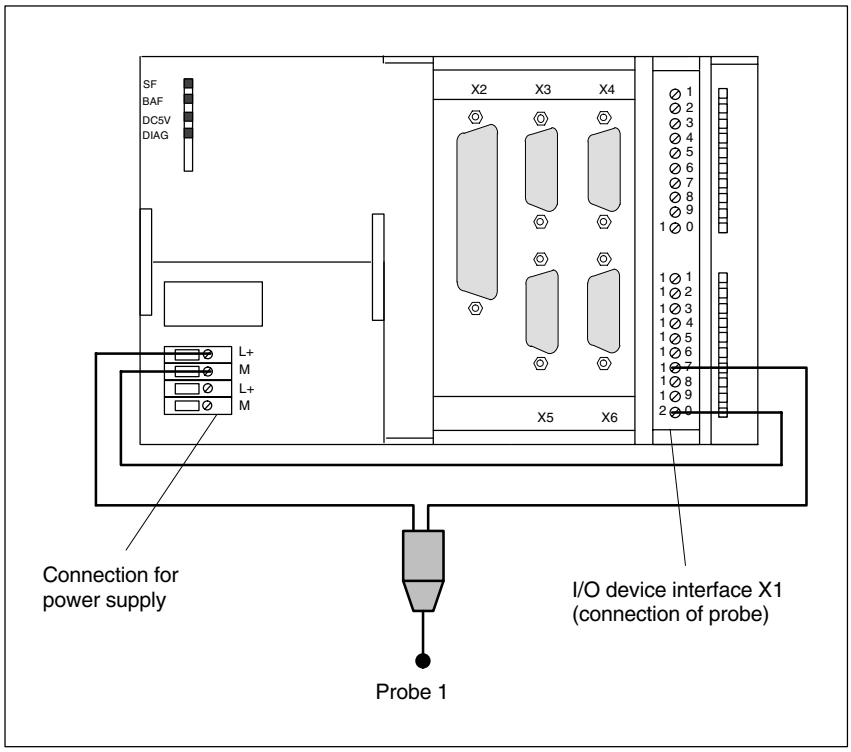


Fig. 2-5 Example of probe interface connection on FM-NC (NCU 570.2), probe 1

2.2 Channel-specific measurements

2.2.1 Software requirements

NC software version

Channel-specific measuring functionality is available for SW 1 and higher.

MMC software version

The “Measurement result display” and “Parameter assignment via input dialog” functions require an MMC SW of 3.2 or higher.

2.2.2 Measuring mode

MEAS and MEAW measurement commands

The measuring operation is activated from the part program. A trigger event and a measuring mode are programmed.
Two different measuring modes are available:

- MEAS: Measurement with deletion of distance-to-go

  Example:
  N10 G01 F300 X300 Z200 MEAS=–2
  Trigger event is the falling edge (–) of the second probe (2).

- MEAW: Measurement without deletion of distance-to-go

  Example:
  N20 G01 F300 X300 Y100 MEAW=1
  Trigger event is the rising edge of the first probe (1).

The measuring job is aborted with RESET or when the program advances to a new block.

Note

If a GEO axis is programmed in a measuring block, then the measured values are stored for all current GEO axes.

If an axis which is taking part in a transformation is programmed in a measuring block, then the measured values of all of the axes taking part in this transformation are stored.

Probe status

As of SW 4, it is possible to scan the probe status directly in the part program and in synchronized actions.

$A_PROBE[n] with n=probe
$A_PROBE[n]==1: probe deflected
$A_PROBE[n]==0: probe not deflected

2.2.3 Measurement results

Read measurement results in PP

The results of the measurement commands are stored in system data of the NCK and can be read via system variables in the part program.

- **System variable $AC_MEA[<No>]**
  Scan status signal of measurement job.
  <No.> stands for probe (1 or 2)

  The variable is deleted at the beginning of a measurement. As soon as the probe fulfills the activation criterion (rising or falling edge), the variable is set. Execution of the measurement job can thus be checked in the part program.

- **System variable $AA_MM[<Axis>]**
  Access to measurement result in machine coordinate system.
  Read in part program and in synchronized actions.
  <Axis> stands for the name of the measurement axis (X, Y, ...).

- **System variable $AA_MW[axis]**
  Access to measurement result in workpiece coordinate system.
  Read in part program and in synchronized actions.
  <Axis> stands for the name of the measurement axis (X, Y, ...).

References: /PAZ/, Programming Guide

PLC service display

The probe is tested functionally by means of an NC program.

The measuring signal can be checked at the end of the program in the diagnostic menu “PLC status”.

Table 2-5 Status display for measuring signal

	Status display
Probe 1 deflected	DB10 DB B107.0
Probe 2 deflected	DB10 DB B107.1

The current measuring status of the axis is displayed by means of the interface signal DB(31–48) DBX62.3.

Bit 3=1: Measurement active
Bit 3=0: Measurement not active

Note

This signal can be displayed for all measuring functions and also read in synchronized actions with $AA_MEAACT[“axis”].
References /FB2/, S5, Synchronized Actions

2.3 Axial measurement (optional)

A measuring operation can be initiated from both the part program and synchronized actions. A measuring mode, the encoder and up to four trigger events are programmed, the trigger events comprising the probe number (1 or 2) and the activation criterion (rising/falling signal edge).

If the measured values are to be stored from encoder 1 and 2 for each trigger event, then only two trigger events can be programmed.

2.3.1 Software requirements

NC software version

Axial measuring functionality is available as an option for SW 4 and higher.

MMC software version

The “Measurement result display” and “Parameter assignment via input dialog” functions require an MMC SW of 3.2 or higher.

2.3.2 Supplementary conditions

Operating mode changeover

Measurement job from part program
A measurement job activated by a part program is not affected by a changeover in operating mode. However, it is deleted immediately the program advances to a new block.
RESET aborts measurement jobs.

Measurement job from synchronized actions
A measurement job activated by a modal synchronized action is not affected by a changeover in operating mode. The job is modally active beyond block limits.

Block search

Measurement job from part program
The job is not started. No measurement check-back signals are supplied.

Measurement job from synchronized actions
Modal measurement jobs are not activated until the programmed conditions are fulfilled.

REPOS

Measurement job from part program
If a measurement job is currently in progress, it is aborted and restarted again after the REPOS block. If the job had already been completed, it is not started again.

Measurement job from synchronized actions
Activated measurement jobs remain unaffected.

2.3.3 Measuring mode

The measuring mode specifies whether trigger events must be activated in parallel or sequentially in ascending sequence and defines the number of measurements to be taken.

Measuring mode 1

The user can program up to 4 different trigger events in the same position control cycle.
The measuring signal edges are evaluated in chronological sequence.

- Up to 2 probes with 2 measuring signal edges each can be programmed for each measurement job. If 2 encoders are used, the number of programmed trigger events is halved.
- Where six-axis modules are installed, measuring mode 1 is imaged on measuring mode 2 internally in the control.
- The traversing velocity must be lower or equal to the shortest distance between 2 identical trigger events in each position control clock cycle.

Note

With this mode, the compensation value which was present when the last measuring signal edge was received is calculated for all measured values.

Measuring mode 2

The user can program up to 4 different trigger events one after the other in the configured sequence.Evaluation of measuring signal edges is activated for one trigger event at a time and takes place in the programmed sequence.

- Trigger events are detected only in the programmed sequence.
- The traversing velocity must be lower or equal to the shortest distance between 2 trigger events in each position control cycle.

Note

The measurement does not work with simulated axes!

Probe status

As of SW 4, it is possible to scan the probe status directly in the part program and in synchronized actions.

$A_PROBE[n] with n=probe
$A_PROBE[n]==1: probe deflected
$A_PROBE[n]==0: probe not deflected

2.3.4 Programming

Programming

Axial measurements can be programmed with and without deletion of distance-to-go.
MEASA with deletion of distance-to-go
MEAWA without deletion of distance-to-go

MEASA[axis] = (mode, trigger event 1, trigger event 2,
trigger event 3, trigger event 4)

Description of parameters:

- Axis : Channel axis name (X, Y, ...)
- Mode : Ones decade
  0 = Abort measurement job (e.g. for synchronized actions)
  1 = Up to 4 trigger events that can be activated simultaneously
  2 = Up to 4 trigger events that can be activated sequentially
  Error output if the first trigger event is already active
  3 = Up to 4 trigger events that can be activated sequentially
  NONE Error output if the first trigger event is already active, alarms 21700/21703 are suppressed

  Tens decade (= encoder selection)
  0/not set = Use active measuring system
  1 ; 1st measuring system
  2 ; 2nd measuring system (If installed. Otherwise the first measuring system is used, no alarm is output)
  3 ; 1st and 2 measuring system

  If the measurement is taken using two measuring systems, a maximum of two trigger events may be programmed. The measured values of both encoders are recorded for each of the two trigger events.
- Trigger event 1 = rising edge of probe 1
  –1 = falling edge of probe 1
  2 = rising edge of probe 2
  –2 = falling edge of probe 2

Note

MEASA and MEAWA can be programmed in one block.

MEASA cannot be programmed in synchronized actions.

The axes for which MEASA has been programmed are not decelerated until all programmed trigger events have arrived.

Measurement jobs started from a part program are aborted by RESET or when the program advances to a new block.

If MEASA/MEAWA are programmed in the same block as MEAS/MEAW this is refused with alarm 21701.

If a geometry axis is used in a measurement, the measured values are only prepared in the workpiece coordinate system if all geometry axes are programmed with the same measurement task. If a geometry axis is missing from the measurement task, the measured value is only stored in the machine coordinate system and alarm 21702 is output. The same applies to axes involved in a transformation.
If the measurement must start on the probe signal edge (with the position of the probe unknown at the instant measurement commences), the customer must evaluate the probe in the part program. By scanning the probe status, it is generally possible to ensure that the next probe signal edge (positive or negative) detected in the hardware will initiate the measurement job.

```
if $A_PROBE [1] =1          ; Probe deflected ?
MEAC [X] = (1,1,–1,1)       ; Starts on the first detected negative edge.
else
MEAC [X] =(1,1,1,–1)        ; Starts on the first detected positive edge.
endif.
```

The alarms are described in the online help or in
References /DA/, Diagnostic Guide

2.3.5 Measurement results

Measurement results

The results of the measurement command are stored in NCK system data and can be read in the part program by means of system variables.

- **System variable $AC_MEA[]**
  Scan status signal of measurement job.
  <No.> stands for probe (1 or 2)

  The variable is deleted at the beginning of a measurement. As soon as the probe fulfills the activation criterion (rising or falling edge), the variable is set. Execution of the measurement job can thus be checked in the part program.

- **System variable $AA_MM1[axis]** to **$AA_MM4[axis]**
  Access to measurement result of trigger signal in machine coordinate system. Read in part program and in synchronized actions.
  <Axis> stands for the name of the measurement axis (X, Y, ...).

- **System variable $AA_MW1[axis]** to **$AA_MW4[axis]**
  Access to measurement result of trigger signal in machine coordinate system. Read in part program and in synchronized actions.
  <Axis> stands for the name of the measurement axis (X, Y, ...).

Programming

If two measuring systems are used to take the measurement, a maximum of two trigger events may be programmed. The measured values of both encoders are recorded for each of the two trigger events.

One trigger event
$AA_MM1[axis] = trigger event 1, measured value from encoder 1
$AA_MM2[axis] = trigger event 1, measured value from encoder 2

Two trigger events
$AA_MM1[axis] = trigger event 1, measured value from encoder 1
$AA_MM2[axis] = trigger event 1, measured value from encoder 2
$AA_MM3[axis] = trigger event 2, measured value from encoder 1
$AA_MM4[axis] = trigger event 2, measured value from encoder 2

PLC service display

The probe is functionally tested by an NC program.

The measuring signal can be checked at the end of the program in the diagnostic menu “PLC status”.

Table 2-6 Status display for measuring signal

	Status display
Probe 1 deflected	DB10 DB B107.0
Probe 2 deflected	DB10 DB B107.1

References: /PAZ/, Programming Guide Cycles
/BNM/, User’s Guide Measuring Cycles

2.3.6 Continuous measurements (cyclic measurements)

All measurements are written to a previously defined FIFO variable. The number of measured values is defined in machine data.

- Correct operation of the function can be guaranteed only with an IPO/position control cycle ratio of $\leq$ 8 : 1.
- The contents of the FIFO memory can be read only once. When measurement results are used more than once, the read-out values must be buffered in the user data.

MEAC

Continuous, axial measurements without deletion of distance-to-go

MEAC[axis] = (mode, measurement memory, trigger event 1,
trigger event 2, trigger event 3,
trigger event 4)

Description of parameters:

- Axis : Channel axis name (X, Y, ...)
- Mode : Ones decade
  0 = Abort measurement job (for synchronized actions)
  1 = Up to 4 trigger events that can be simultaneously activated (a maximum of 4 signals can be triggered simultaneously in one position controller cycle, but the correct order must be observed)
  2 = Up to 4 trigger events that can be activated sequentially (only **one** signal can be triggered per position controller

  Tens decade (= encoder selection)
  0/not enabled = active measuring system
  1 = 1. measuring system
  2 = 2. measuring system (if installed, the first measuring system is otherwise used, no alarm is generated)
  3 = 1. and 2 measuring system

  If the measurement is taken using two measuring systems, a maximum of two trigger events may be programmed.
- Measurement memory: Number of FIFO
- Trigger event 1 = rising edge of probe 1
  –1 = falling edge of probe 1
  2 = rising edge of probe 2
  –2 = falling edge of probe 2

The axial measurement values are available in the machine coordinate system (MCS). They are written to a FIFO variable defined by the user, e.g. $AC_FIFO1. When two probes are configured to take the measurement, the measured values from the second probe are stored separately in the following FIFO.
The number of measured values is limited by MD 28264: LEN_AC_FIFO. Variables $AC_MEA and $AA_MM are therefore irrelevant.

The values can be read from the FIFO both in the part program and from synchronized actions.

The measurement is active until

- MEAC[“axis”]=(0) is programmed
- a FIFO is full
- RESET is pressed or end of program M02/M30 is detected

Endless measuring

In order to implement endless measuring, FIFO values must be read cyclically from the part program. The frequency at which measured values are read from the FIFO memory and processed must correspond to the write rate of the NC. The number of valid entries can be read in a FIFO variable.
In order to achieve a defined number of measured values, the measuring function must be explicitly deselected by the program.

FIFO variable

For definition of FIFO variables, see
References: /FB2/, S5, Synchronized Actions

2.4 Measurement accuracy and functional testing

2.4.1 Measurement accuracy

Accuracy

The propagation time of the measuring signal is determined by the hardware used. The delay times for a SIMODRIVE 611D installation are between 3.625µ ... 9.625µ plus the reaction time of the probe.

The measurement uncertainty is calculated as follows:
Measurement uncertainty = measuring signal propagation time x traversing velocity

The permissible traversing velocities depend on the number of programmed measuring signal edges and the ratio between the IPO clock cycle and position control cycle.
The correctness of measurement results can be guaranteed only in the case of traversing velocities at which no more than 1 identical and no more than 4 different trigger signals arrive in each position control cycle.

2.4.2 Functional testing of probe

Example of functional test

```
%_N_TEST_PROBE_MPF
;$PATH=/_N_MPF_DIR
;Test program for probe interface
N05   DEF INT MTSIGNAL                  ;Flag for trigger status

N10   DEF INT ME_NR=1                   ;Measuring input number
N20   DEF REAL MEASVALUE_IN_X
N30   G17 T1 D1                         ;Preselect tool
                                        ;offset for probe
N40   _ANF: G0 G90 X0 F150              ;Start position and
                                        ;measuring velocity
N50   MEAS=ME_NR G1 X100                ;Measurement at measuring
                                        ;input 1 in X axis
N60   STOPRE
N70   MTSIGNAL=$AC_MEA[1]               ;Read software switching signal
                                        ;at 1st measuring input
N80   IF MTSIGNAL == 0 GOTOF _FEHL1     ;Evaluate signal
N90   MEASVALUE_IN_X=$AA_MW[X]          ;Read measured value
                                        ;into workpiece dimensions
N95   M0
N100  M02
N110  _FEHL1: MSG (“Probe is not switching!”)
N120  M0
N130  M02
```

■

Notes

Supplementary Conditions 3

Axial measurement functionality is available with SW package 4 and higher.

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

Note

The Measurement function is currently available only on request for the SINUMERIK 840Di.

■

Notes

Data Descriptions (MD, SD) 4

13200 MD number	**MEAS_PROBE_LOW_ACTIVE** Switching characteristics of probe		
Default value: FALSE **When using measurement cycles: FALSE**	Min. input limit: FALSE	Max. input limit: TRUE	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: BOOLEAN	Applies from SW version: SW2.2		
Significance:	Value 0:(default setting) non-deflected state 0 V deflected state 24 V Value 1 non-deflected state 24 V deflected state 0 V		

28264 MD number	**LEN_AC_FIFO** Length of FIFO variables $AC_FIFO ...		
Default value: 0	Min. input limit: 0	Max. input limit: 10000	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies from SW version: SW4.1		
Significance:	Length of FIFO variables $AC_FIFO1 to $AC_FIFO10. All FIFO variables have the same length.		

■

Notes

Signal Descriptions

5

DB31, ... **DBX62.3** Data block	**Measuring status** Signal(s) from axis/spindle (NCK !→ PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 4
Signal state 1 or signal transition 0 ——> 1	The “Measuring” function is active. This signal is used during measuring and displays the current measuring status of the axis.	
Signal state 0 or signal transition 1 ——> 0	The “Measuring” function is not active.	

DB10, ... **DBX107.0 and 107.1** Data block	**Probe actuated** Signal(s) from axis/spindle (drive → PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	Probe 1 or 2 is actuated.	
Signal state 0 or signal transition 1 ——> 0	Probe 1 or 2 is not actuated.	
References	/PHD/, “NCU 571 - 573 Manual” /PHF/, “NCU 570 Manual”	
Note	With SW 3.2 and earlier, the signal is active only while the NC block containing the measuring operation is being processed.	

■

Notes

Example

6

6.1 Measuring mode 1

Measurement with 1 encoder

– Single measurement
– 1 probe
– Trigger signals are the rising and falling edges
– Actual value from current encoder

```
N2    MEASA[X] = (1, 1, –1) G01 X100 F100
N3    STOPRE
N4    IF $AC_MEA[1]==FALSE gotof END
N5    R10=$AA_MM1[X]
N6    R11=$AA_MM2[X]
N7    END:
```

Measurement with 2 encoders

– Single measurement
– 1 probe
– Trigger signals are the rising and falling edges
– Actual values from 2 encoders

```
N2    MEASA[X]=(31, 1, –1) G01 X100 F100
N3    STOPRE
N4    IF $AC_MEA[1]==FALSE gotof END
N5    R10=$AA_MM1[X]
N6    R11=$AA_MM2[X]
N7    R12=$AA_MM3[X]
N8    R13=$AA_MM4[X]
N9    END:
```

6.2 Measuring mode 2

– 2 probes
– Trigger signals are the rising and falling edges
– Actual value from current encoder

```
N2    MEASA[X] = (2, 1, –1, 2, –2) G01 X100 F100
N3    STOPRE
N4    IF $AC_MEA[1]==FALSE gotof PROBE2
N5    R10=$AA_MM1[X]
N6    R11=$AA_MM2[X]
N7    PROBE2
N8    IF $AC_MEA[2]==FALSE gotof END
N9    R12=$AA_MM3[X]
N10   R13=$AA_MM4[X]
N11   END:
```

6.3 Continuous measurement

6.3.1 Cont. measurement on completion of prog. traversing movement

– The measurement is taken in measuring mode 1
– Measurement with 100 values
– 1 probe
– Trigger signal is the falling edge
– Actual value from current encoder

```
N1    DEF REAL MEASVALUE[100]
N2    DEF INT INDEX=0
N3    MEAC[x]=(1, 1, –1) G01 X1000 F100
N4    MEAC[X]=(0)                          ;Abort
N5    R1=$AC_FIFO1[4]                      ;No. of measured values
N6    FOR INDEX=0 TO R1
N7    MEASVALUE[INDEX]=$AC_FIFO1[0]        ;Read out measured values
N8    ENDFOR:
```

6.3.2 Continuous measurements with deletion of distance-to-go

– Delete distance-to-go after last measurement
– The measurement is taken in measuring mode 1
– Measurement with 100 values
– 1 probe
– Trigger signal is the falling edge
– Actual value from current encoder

```
N1   DEF INT NUMBER=100
N2   DEF REAL MEASVALUE[NUMBER]
N3   DEF INT INDEX=0
N4   WHEN $AC_FIFO1[4]==NUMBER DO DELDTG (X) MEAC[X] =(0)
N5   MEAC[X]=(1, 1, –1) G01 X1000 F100        ;Start measurement
N6   R1=$AC_FIFO1[4]                          ;No. of measured values
N7   FOR INDEX=0 TO R1
N8   MEASVALUE[INDEX]=$AC_FIFO1[0]            ;Read out measured
                                              values
N9   ENDFOR:
```

6.3.3 Continuous measurements modally over several blocks

– The measurement is taken in measuring mode 1
– Measurement with 100 values
– 1 probe
– Trigger signal is the falling edge
– Actual value from current encoder

```
N1   DEF INT NUMBER=100
N2   DEF REAL MEASVALUE[NUMBER]
N3   DEF INT INDEX=0
N4   ID=1 MEAC[X]=(1, 1, –1)                  ;Start measurement
N5   ID=2 WHEN $AC_FIFO1[4]==NUMBER DO MEAC[X]=(0) CANCEL(2)
N6   G01 X1000 Y100
N7   X100 Y100
N8   R1=$AC_FIFO1[4]                          ;Number of measured
                                              values
N9   FOR INDEX=0 TO R1
N10  MEASVALUE[INDEX]=$AC_FIFO1[0]            ;Read out measured
                                              values
N11  ENDFOR:
```

6.4 Functional test and repeat accuracy

Functional test

```
%_N_TEST_PROBE_MPF
;$PATH=/_N_MPF_DIR
;Test program for probe interface
N05   DEF INT MTSIGNAL                  ;Flag for trigger status

N10   DEF INT ME_NR=1                   ;Measuring input number
N20   DEF REAL MEASVALUE_IN_X
N30   G17 T1 D1                         ;Preselect tool
                                        ;offset for probe
N40   _ANF: G0 G90 X0 F150              ;Start position and
                                        ;measuring velocity
N50   MEAS=ME_NR G1 X100                ;Measurement at measuring
                                        ;input 1 in X axis
N60   STOPRE
N70   MTSIGNAL=$AC_MEA[1]               ;Read software switching signal
                                        ;at 1st measuring input
N80   IF MTSIGNAL == 0 GOTOF _FEHL1     ;Evaluate signal
N90   MEASVALUE_IN_X=$AA_MW[X]          ;Read measured value
                                        ;into workpiece dimensions
N95   M0
N100  M02
N110  _FEHL1: MSG (“Probe is not switching!”)
N120  M0
N130  M02
```

Repeat accuracy

This program allows the measuring scatter (repeat accuracy) of the entire measuring system (machine-probe-signal transmission to NC) to be calculated.

In the example, 10 measurements are taken in the X axis and the measured value recorded in the workpiece coordinates.

It is therefore possible to determine the so-called “random” dimensional deviations which are not subject to any trend.

```
%_N_TEST_GENAU_MPF;
$PATH=/_N_MPF_DIR
N05   DEF INT SIGNAL, II                  ;Variable definition
N10   DEF REAL MEASVALUE_IN_X[10]
N15   G17 T1 D1                           ;Initial conditions,
                                          ;Preselect tool
                                          ;offset for probe
N20   _ANF: G0 X0 F150   ←                ;Preposition in measurement axis
N25   MEAS=+1 G1 X100    ←                ;Measurement at 1st measuring
                                          ;input with switching signal not
                                           deflected
                                          ;deflected in the X axis
N30   STOPRE             ←                ;Stop decoding for subsequent
                                          ;evaluation of result
N35   SIGNAL= $AC_MEA[1]                  ;Read software switching signal at
                                          ;1st measuring input
N37   IF SIGNAL == 0 GOTOF_FEHL1          ;Check switching signal
N40   MESSWERT_IN_X[II]=$AA_MW[X];Read measured value into
                                           workpiece coordinates
N50   II=II+1
N60   IF II<10 GOTOB_ANF                  ;Repeat 10 times
N65   M0
N70   M02
N80   _FEHL1: MSG (“Probe is not switching”)
N90   M0
N95   M02
```

After the parameter display (user-defined variables) have been selected, the measurement results can be read in field MEASVALUE_IN_X[10] provided that the program is still being processed.

■

Notes

Data Fields, Lists

7

Number	Identifier	Name	Reference
General ($MN_...)			
13200	MEAS_PROBE_LOW_ACTIVE	Switching characteristics of probe	
Channel-specific ($MC_...)			
28264	MM_LEN_AC_FIFO	Length of FIFO variables $AC_FIFO ...	

■

Notes

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

Software Cams, Position Switching Signals (N3)

■

Notes

Brief Description 1

The “Software cams” function allows position-dependent cam signals to be output to the PLC as well as to the NCK I/Os (in position control cycle).

The position values at which the signal outputs are set can be defined and altered via setting data.

32 software cam pairs are available. These can be used, for example:

- As reversing signals for hydraulically controlled oscillation axes.
- As limit switches

Note

Software cams can be applied for linear axes and modulo rotary axes.

■

Notes

Detailed Description 2

2.1 General, applications

General

The “Software cams” function (see Chapter 3) generates position-dependent switching signals for axes that supply an actual position value (machine axes) and for simulated axes. The cam signals can be output to the PLC as well as to the NCK I/Os.

The cam positions at which signal outputs are set can be defined and altered via setting data. The setting data can be read and written via MMC, PLC and part program.

Activation

The “Software cams” function can be activated and used in all operating modes. The function remains active in the event of RESET or EMERGENCY STOP.

Applications

Examples of cam signal applications are as follows:

- To activate protection zones
- To initiate additional movements as a function of position
- As reversing signals for hydraulically controlled oscillation axes.

Axis types

Software cams can be used on linear and modulo rotary axes that are defined as machine axes.

Cam range/cam pair

Cams are always assigned in pairs to axes. A pair consists of a plus and a minus cam. 32 cam pairs are available.

The plus and minus cams each simulate a mechanical cam which is actuated at a defined point (cam position) in a specific approach direction when the axis reaches the cam position.

Cam ranges are assigned to the plus and minus cams as follows:

Cam range plus:	All positions $\geq$ plus cam
Cam range minus:	All positions $\leq$ minus cam

2.2 Cam signals and cam positions

2.2.1 Generation of cam signals

Linear axes

The switching edges of the cam signals are generated as a function of the axis traversing direction:

- The minus cam signal switches from 1 to 0 when the axis traverses the minus cam in the positive axis direction.
- The plus cam signal switches from 0 to 1 when the axis traverses the plug cam in the positive direction.

Both cam signals can be output to the PLC and to the NCK I/Os. Separate output of the plus and minus cam signals makes it easy to detect whether the axis is within or outside the plus or minus cam range.

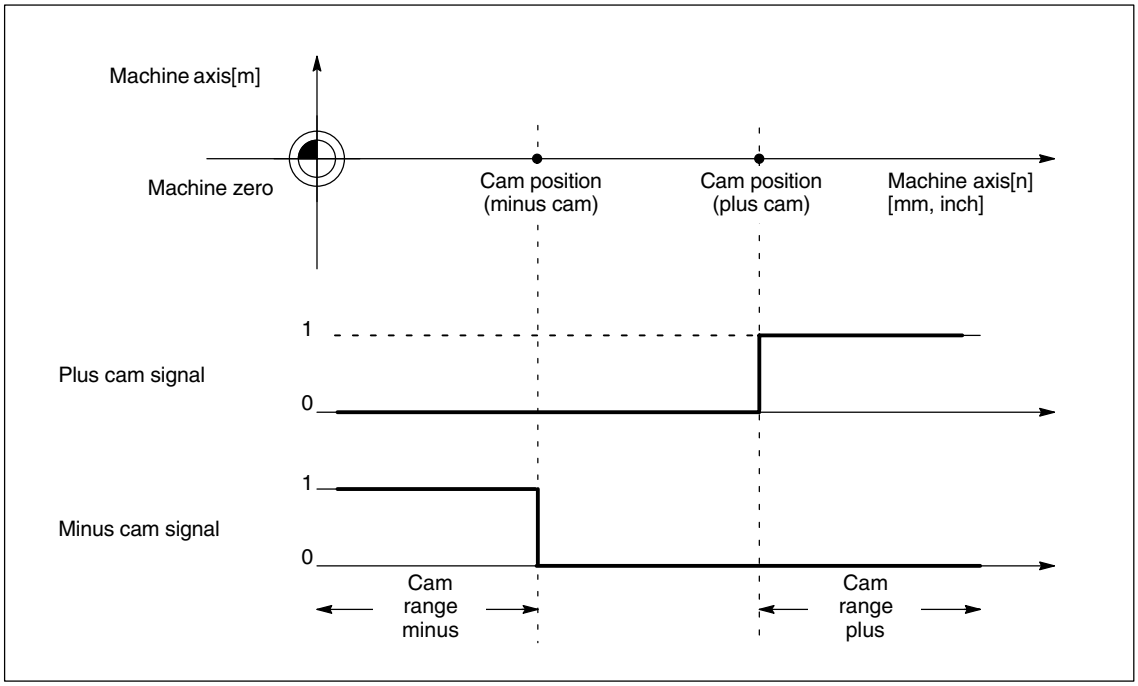


Fig. 2-1 Software cams for linear axis (minus cam < plus cam)

Note

Position switching signals:

If the axis is positioned exactly on the cam, plus or minus, the defined output “flickers”. If it traverses by one increment, the output is clearly zero or one.

“Flickering” of the actual position as it is evaluated causes the signals to flicker in this manner.

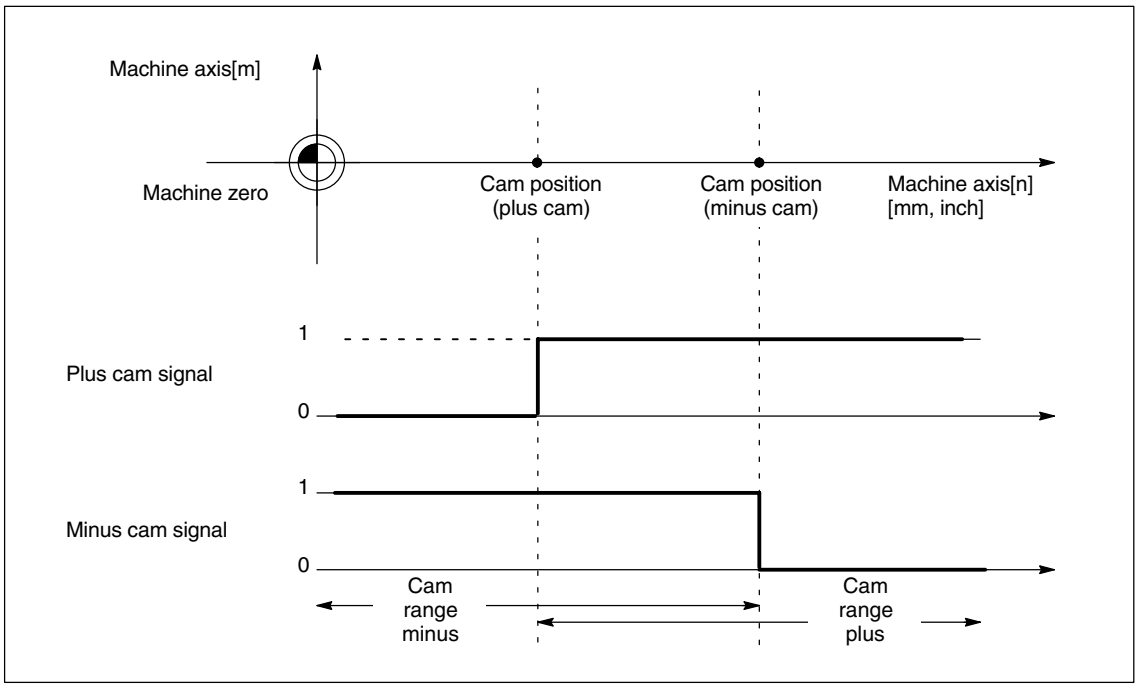


Fig. 2-2 Software cams for linear axis (plus cam < minus cam)

Modulo rotary axes

The switching edges of the cam signals are generated as a function of the rotary axis traversing direction:

- The plus cam signal switches from 0 to 1 when the axis traverses the minus cam in a positive axis direction and from 1 back to 0 when it traverses the plus cam.
- The minus cam signal changes level in response to every positive edge of the plus cam signal.

Note

The plus cam response applies under the following **condition**:
Plus cam – minus cam < 180 degrees

If this condition is not fulfilled or if the minus cam is set to a higher value than the plus cam, then the plus cam signal response is inverted. The response of the minus cam signal remains unchanged.

The signal change of the minus cam makes it possible to detect traversal of the cam even if the cam range is set so small that the PLC cannot detect it reliably.

Both cam signals can be output to the PLC and to the NCK I/Os. Separate output of the plus and minus cam signals makes it easy to detect whether the axis is within or outside the plus or minus cam range.

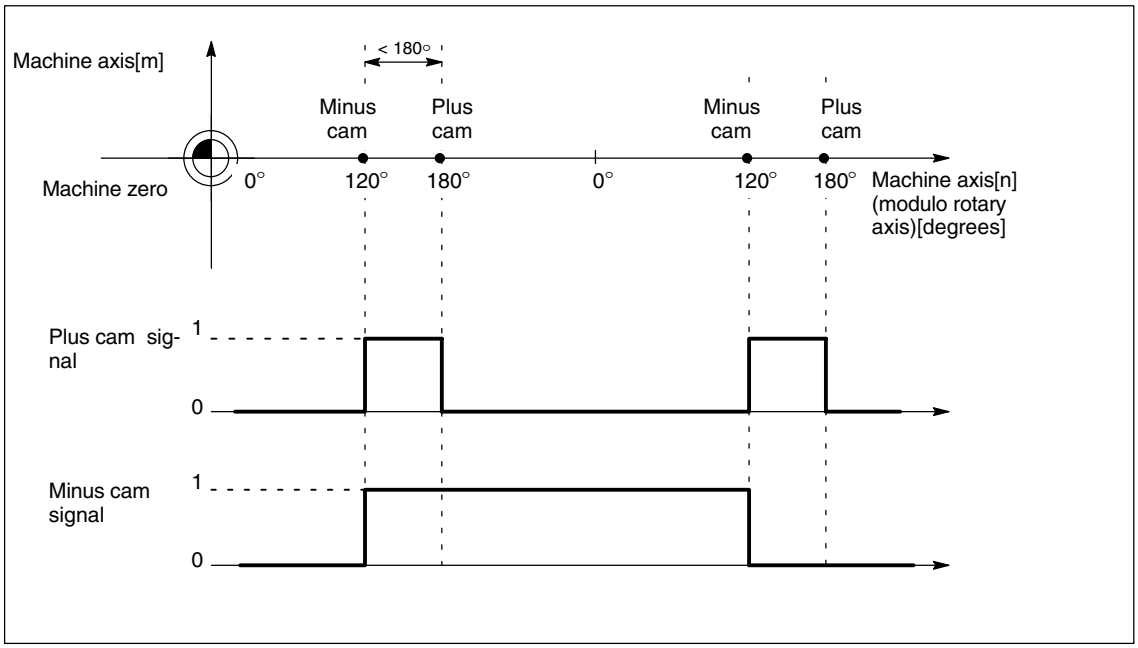


Fig. 2-3 Software cams for modulo rotary axis (plus cam – minus cam < 180 degrees)

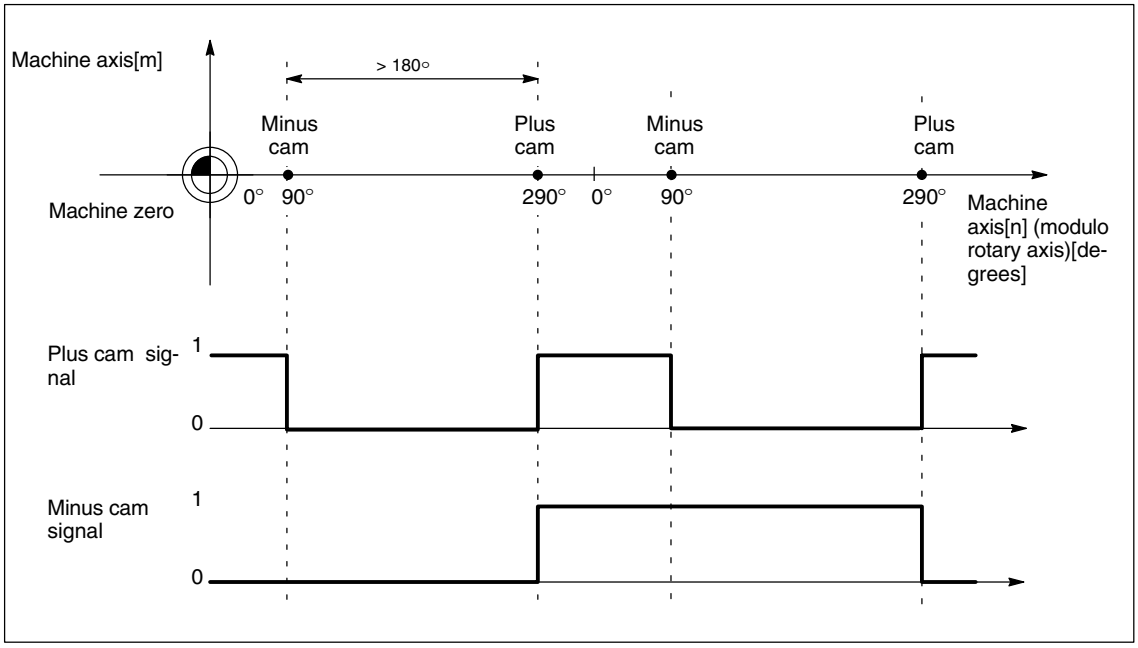


Fig. 2-4 Software cams for modulo rotary axis (plus cam – minus cam > 180 degrees)

Note

The shortest clock cycle is the servo cycle (MN_SW_CAM_TIMER FASTOUT_MASK = "HO ", i.e. not a timer-triggered output). The clock frequency for a rotational velocity of 1 rev/ 40 ms and a servo cycle of 2 ms is 18 degrees/ 2 ms, resulting in an IPO cycle of 54 degrees/ 6ms. The software cams are activated in the IPO cycle and the cam signals output in the servo cycle.
The cams do not therefore switch reliably with a distance of 10 degrees between the plus and minus cams.
With a combined fast output, only the plus cam signal is ever output, i.e. the minus cam acting as a signal edge is not used at the same time. There are two variants of this method:
Cam distance < 180 degrees:

Cam = 1 between minus and plus cams, otherwise 0

Cam distance >180 degrees:

Cam = 0 between minus and plus cams, otherwise 1

In this instance, therefore, the cam signal should switch to 1 only between 20 and 30 degrees. This cannot be switched (servo cycle too long).
The software cams can only be used up to a certain speed for modulo rotary axes depending on IPO cycle, servo cycle and cam distance. Reliable switching cannot be guaranteed at higher speeds.

2.2.2 Cam positions

Setting cam positions

The cam positions of the plus and minus cams are defined via the following general setting data:

SD 41500: SW_CAM_MINUS_POS_TAB_1[n] Position of minus cams 1–8
SD 41501: SW_CAM_PLUS_POS_TAB_1[n] Position of plus cams 1–8

SD 41502: SW_CAM_MINUS_POS_TAB_2[n] Position of minus cams 9–16
SD 41503: SW_CAM_PLUS_POS_TAB_2[n] Position of plus cams 9–16

In addition, from SW 4.1:

SD 41504: SW_CAM_MINUS_POS_TAB_3[n] Position of minus cams 17–24
SD 41505: SW_CAM_PLUS_POS_TAB_3[n] Position of plus cams 17–24

SD 41506: SW_CAM_MINUS_POS_TAB_4[n] Position of minus cams 25–32
SD 41507: SW_CAM_PLUS_POS_TAB_4[n] Position of plus cams 25–32

Note

Owing to the grouping of cam pairs (eight in each group), it is possible to assign different access authorization levels (e.g. for machine-related and workpiece-related cam positions).

The positions are entered in the machine coordinate system. No check is made with respect to the maximum traversing range.

Measuring system metric/inch

From SW 5 and MD 10260: CONVERT_SCALING_SYSTEM=1 (see /G2/) the cam positions no longer refer to the basic system that is set but to the measuring system set in MD 10270: POS_TAB_SCALING_SYSTEM.

MD 10270: POS_TAB_SCALING_SYSTEM=0: metric
MD 10270: POS_TAB_SCALING_SYSTEM=1: inch

The MD 10270 thus defines the measuring system for position specifications from SD 41500 to SD 41507.

A switchover with G70/G71 or G700/G710 has no effect.

Sensing of cam positions

To set the cam signals, the **actual position** of the axes is compared to the **cam position**.

Writing/reading of cam positions

The setting data can be accessed for reading and writing via the MMC, PLC and part program.
Data accessing from the part program is not synchronized with block processing. Synchronization can only be achieved by means of a programmed block preprocessing stop (STOPRE command).

It is possible to read and write the cam positions with FB 2 and FB 3 in the PLC user program.

Axis/cam assignment

An assignment between a cam pair and a machine axis is made via the general MD: SW_CAM_ASSIGN_TAB[n] (assignment of software cams to machine axes).

Note

Changes to an axis assignment take effect after the next NCK power-up.

Cam pairs to which no axis is assigned are not active.

A cam pair can only be assigned to one machine axis at a time.

Several cam pairs can be defined for one machine axis.

2.2.3 Lead/delay times (dynamic cam)

Times

To compensate for any delays, it is possible to assign two lead or delay times with additive action to each minus and plus cam for the cam signal output.

The two lead or delay times are entered in a machine data and a setting data.

Note

The input of negative time values causes a delay in output of cam signals.

Input in machine data

The first lead or delay time is entered in the following general machine data:

MD 10460: SW_CAM_MINUS_LEAD_TIME[n]
Lead or delay time on minus cams
MD 10461: SW_CAM_PLUS_LEAD_TIME[n]
Lead or delay time on plus cams

For example, the following entries can be made in these machine data:

- Constant internal delay times between actual-value sensing and cam signal output (e.g. as determined by an oscilloscope) or
- Constant external delay times.

Input in setting data

The second lead or delay time is entered in the following general setting data:

SD 41520: SW_CAM_MINUS_TIME_TAB_1[n]
Lead or delay time on minus cams 1–8
SD 41521: SW_CAM_PLUS_TIME_TAB_1[n]
Lead or delay time on plus cams 1–8

SD 41522: SW_CAM_MINUS_TIME_TAB_2[n]
Lead or delay time on minus cams 9–16
SD 41523: SW_CAM_PLUS_TIME_TAB_2[n]
Lead or delay time on plus cams 9–16

SD 41524: SW_CAM_MINUS_TIME_TAB_3[n]
Lead or delay time on minus cams 17–24
SD 41525: SW_CAM_PLUS_TIME_TAB_3[n]
Lead or delay time on plus cams 17–24

SD 41526: SW_CAM_MINUS_TIME_TAB_4[n]
Lead or delay time on minus cams 25–32
SD 41527: SW_CAM_PLUS_TIME_TAB_4[n]
Lead or delay time on plus cams 25–32

Delay times which may change during machining must, for example, be entered in these setting data.

2.3 Output of cam signals

The cam status, i.e. cam signals, can be output to the PLCas well as to the NCK I/Os.

Activation of cam signal output

Axis-specific **IS "Cam activation" (DB31-62, DBX2.0)** is used to activate the output of cam signals of an axis.

Check-back signal to PLC

Axis-specific **IS "Cams active" (DB31-62, DBX62.0)** is sent to the PLC to indicate successful activation of all the cams of an axis.

Note

Activation can also be linked to other conditions (e.g. axis referenced, RESET effective) by the PLC user.

2.3.1 Output of cam signals to PLC

The status of the cam signals for all machine axes with activated software cams is output to the PLC.

The status is output in the IPO cycle and is transferred to the PLC asynchronously.

Minus cam signals

The status of the minus cam signals is entered in the general **IS "Minus cam signals 1 to 32" (DB10, DBX110.0 to 113.7)**.

Plus cam signals

The status of the plus cam signals is entered in the general **IS "Plus cam signals 1 to 32" (DB10, DBX114.0 to 117.7)**.

Note

If ...	then ...
no measuring system is selected **or** IS "Cam activation" (DB31-62, DBX2.0) = "0"	the **following IS's** are set to "0": – Minus cam signals 1–32 (DB10, DBX110.0–113.7) – Plus cam signals 1–32 (DB10, DBX114.0–117.7) – Cam active (DB31–62, DBX62.0)

2.3.2 Output of cam signals to NCK I/Os

The cam signals are output to the NCK I/Os in the position control cycle.

The 4 on-board outputs on the NCU and a total of 32 optional external NCK outputs are available as the digital outputs of the NCK I/Os.

References: /FB/, A4, “Digital and Analog NCK I/Os”

Hardware assignment

8 pairs of cams in each case are assigned to the hardware bytes used in the two general machine data

MD 10470: SW_CAM_ASSIGN_FASTOUT_1 Hardware assignment for output of cams 1–8 to the NCK I/Os

MD 10471: SW_CAM_ASSIGN_FASTOUT_2 Hardware assignment for output of cams 9–16 to the NCK I/Os

MD 10472: SW_CAM_ASSIGN_FASTOUT_3 Hardware assignment for output of cams 17–24 to the NCK I/Os

MD 10473: SW_CAM_ASSIGN_FASTOUT_4 Hardware assignment for output of cams 25–32 to the NCK I/Os

---

Note

It is possible to define one HW byte for the output of 8 minus cam signals and one HW byte for the output of 8 plus cam signals in each machine data.

In addition, the output of the cam signals can be inverted with the two machine data.

If the 2nd byte is not specified (= “0”), then the 8 cams are output as an AND operation of the minus and plus cam signals via the 1st HW byte using the 1st inversion screen form.

Status query in part program

The status of the HW outputs can be read in the part program with main run variable **$A_OUT[n]** (n = no. of output bit).

2.3.3 Higher-precision cam signal output

Switching accuracy

The cam signals are output to the NCK I/Os in the position control cycle. Owing to the time grid of the position control cycle, the switching accuracy of the cam signals is limited as a function of velocity.

In this case: Delta pos = V_{act} * position control cycle

Parameters

Delta pos: Switching accuracy (governed by position control cycle)
V_{act}: Current axis velocity

Example

V_{act} = 20 m/min, Pos. contr. cycle = 4 ms Delta pos = 1.33 mm
V_{act} = 2000 rev/min, Pos. contr. cycle = 2 ms Delta pos = 24 degrees

Timer-controlled output

A significantly higher degree of accuracy can be achieved by outputting the cam signals independently of the position control cycle using a timer interrupt.

Timer-controlled output to the 4 NCU on-board outputs can be selected for 4 cam pairs in general MD: SW_CAM_TIMER_FASTOUT_MASK (screen form for the output of cam signals via timer interrupts to NCU).

In this case, the minus and plus signals of a cam pair are EXCLUSIVE ORed for output as one signal.

Note

This function works independently of the assignment set in
MD 10470: SW_CAM_ASSIGN_FASTOUT_1 or
MD 10471: SW_CAM_ASSIGN_FASTOUT_2 or
MD 10472: SW_CAM_ASSIGN_FASTOUT_3 or
MD 10473: SW_CAM_ASSIGN_FASTOUT_4.

The on-board byte may not be used more than once at any one time.

Restriction

The following applies to the mutual position of the cam positions:

Only one signal is output on a timer-controlled basis per IPO cycle. If there are signal changes for more than one cam pair in an IPO cycle, then the signals are output on a priority bases:
The cam pair with the lowest number (1 ... 32) determines the instant at which all pending signals are output, i.e. the signal change of the other cam pairs takes place at the same instant in time.

■

Notes

Supplementary Conditions 3

Availability of function "Software cams, position switching signals"

This function is an option and available for

- SINUMERIK 840D with NCU 572/573, with SW 2 and higher
- SINUMERIK FM-NC with NCU 570.2, with SW 3.2 and higher
- SINUMERIK 810D, SW 3.2 and higher

Data Descriptions (MD, SD) 4

4.1 General machine data

10450 MD number	**SW_CAM_ASSIGN_TAB[n]** Assignment of software cams to machine axes		
Default value: 0	Min. input limit: 0	Max. input limit: 8 or 31	
Changes effective after Power On	Protection level: 2/4	Unit: –	
Data type: BYTE	Applies from SW version: 2.1 or 4.1		
Significance:	This machine data allows one machine axis to be assigned to each of the 16 possible cam pairs (comprising one minus and one plus cam). When a "0" is entered, the appropriate cam is not processed. The cam signal output is activated via the axial IS "Cam activation" (DB31-48, DBX2.0). Index [n] of the machine data addresses the cam pair: n = 0, 1, ... , 31 corresponds to cam pair 1, 2, ... , 32		
Application	Cam pair 1 must be assigned to machine axis 2 and cam pair 3 to machine axis 4. Cam pair 2 is not to be assigned to any axis. ⇒MD: SW_CAM_ASSIGN_TAB[0]= 3 MD: SW_CAM_ASSIGN_TAB[1]= 0 MD: SW_CAM_ASSIGN_TAB[2]= 4		
Related to	IS "Cam activation" (DB31-48, DBX2.0)		

10460 MD number	**SW_CAM_MINUS_LEAD_TIME[n]** Lead or delay time on minus cams 1–16		
Default value: 0.0	Min. input limit: ***	Max. input limit: ***	
Changes effective after Power On	Protection level: 2/4	Unit: s	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	A lead or delay time can be assigned to each **minus cam 1–16** in this machine data to compensate for delay times. The switching edge of the associated cam signal is advanced or delayed by the time value entered. Positive value: ⇒ Lead time Negative value: ⇒ Delay time Index [n] of the machine data addresses the cam pair: n = 0, 1, ... , 15 corresponds to cam pair 1, 2, ... , 16 This machine data is added to setting data SW_CAM_MINUS_TIME_TAB_1[n] and SW_CAM_MINUS_TIME_TAB_2[n].		
Related to	SD: SW_CAM_MINUS_TIME_TAB_1[n] (lead or delay time on minus cams 1–8) SD: SW_CAM_MINUS_TIME_TAB_2[n] (lead or delay time on minus cams 9–16)		

10461 MD number	**SW_CAM_PLUS_LEAD_TIME[n]** Lead or delay time on plus cams 1–16		
Default value: 0.0	Min. input limit: ***	Max. input limit: ***	
Changes effective after Power On	Protection level: 2/4	Unit: s	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	A lead or delay time can be assigned to each **plus cam 1–16** in this machine data to compensate for delay times. The switching edge of the associated cam signal is advanced or delayed by the time value entered. Positive value: ⇒ Lead time Negative value: ⇒ Delay time Index [n] of the machine data addresses the cam pair: n = 0, 1, ... , 15 corresponds to cam pair 1, 2, ... , 16 This machine data is added to setting data SW_CAM_PLUS_TIME_TAB_1[n] and SW_CAM_PLUS_TIME_TAB_2[n].		
Related to	SD: SW_CAM_PLUS_TIME_TAB_1[n] (lead or delay time on plus cams 1–8) SD: SW_CAM_PLUS_TIME_TAB_2[n] (lead or delay time on plus cams 9–16)		

10470 MD number	**SW_CAM_ASSIGN_FASTOUT_1** Hardware assignment for output of cams 1–8 to NCK I/Os		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective after Power On	Protection level: 2/4	Unit: HEX	
Data type: DWORD	Applies from SW version: 2.1		
Significance:	The cam signal status can be output to the NCK I/Os as well as to the PLC. The hardware assignment of the minus and plus cam signals to the digital output bytes used can be made in this machine data for **cam pairs 1–8**. In addition, the assigned output signals can be inverted with this machine data. The MD is coded as follows: Bit 0–7: Number of 1st HW byte used with digital outputs Bit 8–15: Number of 2nd HW byte used with digital outputs Bit 16–23: Inverting screen form for writing 1st HW byte used Bit 24–31: Inverting screen form for writing 2nd HW byte used Bit=0: Do not invert Bit=1: Invert If both HW bytes are specified, the 1st byte contains the minus cam signals and the 2nd byte the plus cam signals. If the 2nd byte is not specified (= “0”), then the 8 cams are output as an AND operation of the minus and plus cam signals via the 1st HW byte using the 1st inversion screen form. The status of the non-inverted output signal for linear axes and for rotary axes with “plus cam – minus cam < 180 degrees” is: “1“ between minus and plus cams “0“ outside this range The status of the non-inverted output signal for rotary axes with “plus cam – minus cam ≥ 180 degrees” is: “0” between minus and plus cams “1” outside this range The following must be specified as the byte address for the digital outputs: 1: for on-board byte 2–5: for external bytes		
Application	The minus cam signals must be output via the on-board byte. The plus cam signals must be output via byte address 3 on the NCU terminal block. The following must also be inverted: Minus cam signal 2, 4, 5 (corresponds to bits 1, 3, 4 of 1st HW byte) Plus cam signal 1, 3, 4 (corresponds to bits 0, 2, 3 or 2nd HW byte) ⇒ MD: SW_CAM_ASSIGN_FASTOUT_1=‘H0D1A0301’		

10471 MD number	**SW_CAM_ASSIGN_FASTOUT_2** Hardware assignment for output of cams 9–16 to NCK I/Os		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective after Power On	Protection level: 2/4	Unit: HEX	
Data type: DWORD	Applies from SW version: 2.1		
Significance:	The cam signal status can be output to the NCK I/Os as well as to the PLC. The hardware assignment of the minus and plus cam signals to the digital output bytes used can be made in this machine data for **cam pairs 9–16**. In addition, the assigned output signals can be inverted with this machine data. The MD is coded as follows: Bit 0–7: Number of 1st HW byte used with digital outputs Bit 8–15: Number of 2nd HW byte used with digital outputs Bit 16–23: Inverting screen form for writing 1st HW byte used Bit 24–31: Inverting screen form for writing 2nd HW byte used Bit=0: Do not invert Bit=1: Invert If both HW bytes are specified, the 1st byte contains the minus cam signals and the 2nd byte the plus cam signals. If the 2nd byte is not specified (= “0”), then the 8 cams are output as an AND operation of the minus and plus cam signals via the 1st HW byte using the 1st inversion screen form. The status of the non-inverted output signal for linear axes and for rotary axes with “plus cam – minus cam < 180 degrees” is: “1“ between minus and plus cams “0“ outside this range The status of the non-inverted output signal for rotary axes with “plus cam – minus cam ≥ 180 degrees” is: “0” between minus and plus cams “1” outside this range The following must be specified as the byte address for the digital outputs: 1: for on-board byte 2–5: for external bytes		
Application	see MD: SW_CAM_ASSIGN_FASTOUT_1		

10472 MD number	**SW_CAM_ASSIGN_FASTOUT_3** Hardware assignment for output of cams 17–24 to NCK I/Os		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective after Power On	Protection level: 2/7	Unit: HEX	
Data type: DWORD	Applies from SW version: 4.1		
Significance:	The cam signal status can be output to the NCK I/Os as well as to the PLC. The hardware assignment of the minus and plus cam signals to the digital output bytes used can be made in this machine data for **cam pairs 17–24**. In addition, the assigned output signals can be inverted with this machine data. The MD is coded as follows: Bit 0–7: Number of 1st HW byte used with digital outputs Bit 8–15: Number of 2nd HW byte used with digital outputs Bit 16–23: Inverting screen form for writing 1st HW byte used Bit 24–31: Inverting screen form for writing 2nd HW byte used Bit=0: Do not invert Bit=1: Invert If both HW bytes are specified, the 1st byte contains the minus cam signals and the 2nd byte the plus cam signals. If the 2nd byte is not specified (= “0”), then the 8 cams are output as an AND operation of the minus and plus cam signals via the 1st HW byte using the 1st inversion screen form. The status of the non-inverted output signal for linear axes and for rotary axes with “plus cam – minus cam < 180 degrees” is: “1“ between minus and plus cams “0“ outside this range The status of the non-inverted output signal for rotary axes with “plus cam – minus cam ≥ 180 degrees” is: “0” between minus and plus cams “1” outside this range The following must be specified as the byte address for the digital outputs: 1: for on-board byte 2–5: for external bytes		
Application	see MD: SW_CAM_ASSIGN_FASTOUT_1		

10473 MD number	**SW_CAM_ASSIGN_FASTOUT_4** Hardware assignment for output of cams 25–32 to NCK I/Os		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective after Power On	Protection level: 2/7	Unit: HEX	
Data type: DWORD	Applies from SW version: 4.1		
Significance:	The cam signal status can be output to the NCK I/Os as well as to the PLC. The hardware assignment of the minus and plus cam signals to the digital output bytes used can be made in this machine data for **cam pairs 25–32**. In addition, the assigned output signals can be inverted with this machine data. The MD is coded as follows: Bit 0–7: Number of 1st HW byte used with digital outputs Bit 8–15: Number of 2nd HW byte used with digital outputs Bit 16–23: Inverting screen form for writing 1st HW byte used Bit 24–31: Inverting screen form for writing 2nd HW byte used Bit=0: Do not invert Bit=1: Invert If both HW bytes are specified, the 1st byte contains the minus cam signals and the 2nd byte the plus cam signals. If the 2nd byte is not specified (= “0”), then the 8 cams are output as an AND operation of the minus and plus cam signals via the 1st HW byte using the 1st inversion screen form. The status of the non-inverted output signal for linear axes and for rotary axes with “plus cam – minus cam < 180 degrees” is: “1“ between minus and plus cams “0“ outside this range The status of the non-inverted output signal for rotary axes with “plus cam – minus cam ≥ 180 degrees” is: “0” between minus and plus cams “1” outside this range The following must be specified as the byte address for the digital outputs: 1: for on-board byte 2–5: for external bytes		
Application	see MD: SW_CAM_ASSIGN_FASTOUT_1		

10480 MD number	**SW_CAM_TIMER_FASTOUT_MASK** Screen form for output of cam signals via timer interrupts to NCU		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective after Power On	Protection level: 2/4	Unit: HEX	
Data type: DWORD	Applies from SW version: 2.1		
Significance:	A timer-controlled output to the 4 on-board outputs of the NCK I/Os can be selected in this machine data for 4 cam pairs. In this case, the minus and plus signals of a cam pair are EXCLUSIVE ORed for output as one signal. Meaning for set bit: Associated cam (minus and plus cam signals EXCLUSIVE ORed) is output via a timer interrupt at one of the 4 on-board outputs of the NCU. In this case, the cam pair signals are assigned in ascending sequence to on-board outputs 1–4. This function works independently of the assignment set in MD: SW_CAM_ASSIGN_FASTOUT_1 or MD: SW_CAM_ASSIGN_FASTOUT_2. **Note:** The on-board byte may not be used several times. If there is more than one signal change in the IPO cycle for the cam pairs specified in the MD, then the cam pair with the lowest number determines the instant of output. The other signal changes take place at the same time.		
Application	The signals of cam pairs 2, 5 and 7 must be output on a timer-controlled basis: ⇒ MD: SW_CAM_TIMER_FASTOUT_MASK= ‘H52’ ⇒ Signal for cam pair 2 to on-board output 1 of NCK Signal for cam pair 5 to on-board output 2 of NCK Signal for cam pair 7 to on-board output 3 of NCK		

4.2 General setting data

41500 SD number	**SW_CAM_MINUS_POS_TAB_1[n]** Position of minus cams 1–8		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: mm, degrees	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	The cam position of **minus cams 1–8** is entered in this setting data. The positions are entered in the machine coordinate system. The response when the cam positions are overtraveled by linear and modulo rotary axes is described in Section 2.2.1. Index [n] of the setting data addresses the cam pair: n = 0, 1, ... , 7 corresponds to cam pair 1, 2, ... , 8		

41501 SD number	**SW_CAM_PLUS_POS_TAB_1[n]** Position of plus cams 1–8		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: mm, degrees	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	The cam position of **plus cams 1–8** is entered in this setting data. The positions are entered in the machine coordinate system. The response when the cam positions are overtraveled by linear and modulo rotary axes is described in Section 2.2.1. Index [n] of the setting data addresses the cam pair: n = 0, 1, ... , 7 corresponds to cam pair 1, 2, ... , 8		

41502 SD number	**SW_CAM_MINUS_POS_TAB_2[n]** Position of minus cams 9–16		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: mm, degrees	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	The cam position of **minus cams 9–16** is entered in this setting data. The positions are entered in the machine coordinate system. The response when the cam positions are overtraveled by linear and modulo rotary axes is described in Section 2.2.1. Index [n] of the setting data addresses the cam pair: n = 8, 9, ... , 15 corresponds to cam pair 9, 10, ... , 16		

41503 SD number	**SW_CAM_PLUS_POS_TAB_2[n]** Position of plus cams 9–16		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: mm, degrees	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	The cam position of **plus cams 9–16** is entered in this setting data. The positions are entered in the machine coordinate system. The response when the cam positions are overtraveled by linear and modulo rotary axes is described in Section 2.2.1. Index [n] of the setting data addresses the cam pair: n = 8, 9, ... , 15 corresponds to cam pair 9, 10, ... , 16		

41504 SD number	**SW_CAM_MINUS_POS_TAB_3[n]** Position of minus cams 17–24		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: mm, degrees	
Data type: DOUBLE	Applies from SW version: 4.1		
Significance:	The cam position of **minus cams 17–24** is entered in this setting data. The positions are entered in the machine coordinate system. The response when the cam positions are overtraveled by linear and modulo rotary axes is described in Section 2.2.1. Index [n] of the setting data addresses the cam pair: n = 0, 1, ... , 7 corresponds to cam pair 17, 18, ... , 24		

41505 SD number	**SW_CAM_PLUS_POS_TAB_3[n]** Position of plus cams 17–24		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: mm, degrees	
Data type: DOUBLE	Applies from SW version: 4.1		
Significance:	The cam position of **plus cams 17–24** is entered in this setting data. The positions are entered in the machine coordinate system. The response when the cam positions are overtraveled by linear and modulo rotary axes is described in Section 2.2.1. Index [n] of the setting data addresses the cam pair: n = 0, 1, ... , 7 corresponds to cam pair 17, 18, ... , 24		

41506 SD number	**SW_CAM_MINUS_POS_TAB_4[n]** Position of minus cams 25–32		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: mm, degrees	
Data type: DOUBLE	Applies from SW version: 4.1		
Significance:	The cam position of **plus cams 25–32** is entered in this setting data. The positions are entered in the machine coordinate system. The response when the cam positions are overtraveled by linear and modulo rotary axes is described in Section 2.2.1. Index [n] of the setting data addresses the cam pair: n = 8, 9, ... , 15 corresponds to cam pair 25, 26, ... , 32		

41507 SD number	**SW_CAM_PLUS_POS_TAB_4[n]** Position of plus cams 25–32		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: mm, degrees	
Data type: DOUBLE	Applies from SW version: 4.1		
Significance:	The cam position of **plus cams 25–32** is entered in this setting data. The positions are entered in the machine coordinate system. The response when the cam positions are overtraveled by linear and modulo rotary axes is described in Section 2.2.1. Index [n] of the setting data addresses the cam pair: n = 8, 9, ... , 15 corresponds to cam pair 25, 26, ... , 32		

41520 SD number	**SW_CAM_MINUS_TIME_TAB_1[n]** Lead or delay time on minus cams 1–8		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: s	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	A lead or delay time can be assigned to each **minus cam 1–8** in this setting data to compensate for lead or delay times. The switching edge of the associated cam signal is advanced or delayed by the time value entered. Positive value: Lead time Negative value: Delay time Index [n] of the setting data addresses the cam pair: n = 0, 1, ... , 7 corresponds to cam pair 1, 2, ... , 8 This setting data is added to MD: SW_CAM_MINUS_LEAD_TIME[n].		
Related to	MD: SW_CAM_MINUS_LEAD_TIME[n] (lead or delay time on minus cams 1–16)		

41521 SD number	**SW_CAM_PLUS_TIME_TAB_1[n]** Lead or delay time on plus cams 1–8		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: s	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	A lead or delay time can be assigned to each **plus cam 1–8** in this setting data to compensate for lead or delay times. The switching edge of the associated cam signal is advanced or delayed by the time value entered. Positive value: Lead time Negative value: Delay time Index [n] of the setting data addresses the cam pair: n = 0, 1, ... , 7 corresponds to cam pair 1, 2, ... , 8 This setting data is added to MD: SW_CAM_PLUS_LEAD_TIME[n].		
Related to	MD: SW_CAM_PLUS_LEAD_TIME[n] (lead or delay time on plus cams 1–16)		

41522 SD number	**SW_CAM_MINUS_TIME_TAB_2[n]** Lead or delay time on minus cams 9–16		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: s	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	A lead or delay time can be assigned to each **minus cam 9–16** in this setting data to compensate for delay times. The switching edge of the associated cam signal is advanced or delayed by the time value entered. Positive value: Lead time Negative value: Delay time Index [n] of the setting data addresses the cam pair: n = 8, 9, ... , 15 corresponds to cam pair 9, 10, ... , 16 This setting data is added to MD: SW_CAM_MINUS_LEAD_TIME[n+8].		
Related to	MD: SW_CAM_MINUS_LEAD_TIME[n] (lead or delay time on minus cams 1–16)		

41523 SD number	**SW_CAM_PLUS_TIME_TAB_2[n]** Lead or delay time on plus cams 9–16		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: s	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	A lead or delay time can be assigned to each **plus cam 9–16** in this setting data to compensate for delay times. The switching edge of the associated cam signal is advanced or delayed by the time value entered. Positive value: Lead time Negative value: Delay time Index [n] of the setting data addresses the cam pair: n = 8, 9, ... , 15 corresponds to cam pair 9, 10, ... , 16 This setting data is added to MD: SW_CAM_PLUS_LEAD_TIME[n+8].		
Related to	MD: SW_CAM_PLUS_LEAD_TIME[n] (lead or delay time on plus cams 1–16)		

41524 SD number	**SW_CAM_MINUS_TIME_TAB_3[n]** Lead or delay time on minus cams 17–24		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: s	
Data type: DOUBLE	Applies from SW version: 4.1		
Significance:	A lead or delay time can be assigned to each **minus cam 17–24** in this setting data to compensate for lead or delay times. The switching edge of the associated cam signal is advanced or delayed by the time value entered. Positive value: Lead time Negative value: Delay time Index [n] of the setting data addresses the cam pair: n = 0, 1, ... , 7 corresponds to cam pair 17, 18, ... , 24 This setting data is added to MD: SW_CAM_MINUS_LEAD_TIME[n].		
Related to	MD: SW_CAM_MINUS_LEAD_TIME[n] (lead or delay time on minus cams 1–16)		

41525 SD number	**SW_CAM_PLUS_TIME_TAB_3[n]** Lead or delay time on plus cams 17–24		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: s	
Data type: DOUBLE	Applies from SW version: 4.1		
Significance:	A lead or delay time can be assigned to each **plus cam 17–24** in this setting data to compensate for delay times. The switching edge of the associated cam signal is advanced or delayed by the time value entered. Positive value: Lead time Negative value: Delay time Index [n] of the setting data addresses the cam pair: n = 0, 1, ... , 7 corresponds to cam pair 17, 18, ... , 24 This setting data is added to MD: SW_CAM_PLUS_LEAD_TIME[n].		
Related to	MD: SW_CAM_PLUS_LEAD_TIME[n] (lead or delay time on plus cams 1–16)		

41526 SD number	**SW_CAM_MINUS_TIME_TAB_4[n]** Lead or delay time on minus cams 25–32		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: s	
Data type: DOUBLE	Applies from SW version: 4.1		
Significance:	A lead or delay time can be assigned to each **minus cam 25–32** in this setting data to compensate for delay times. The switching edge of the associated cam signal is advanced or delayed by the time value entered. Positive value: Lead time Negative value: Delay time Index [n] of the setting data addresses the cam pair: n = 8, 9, ... , 15 corresponds to cam pair 25, 26, ... , 32 This setting data is added to MD: SW_CAM_MINUS_LEAD_TIME[n+8].		
Related to	MD: SW_CAM_MINUS_LEAD_TIME[n] (lead or delay time on minus cams 1–16)		

41527 SD number	**SW_CAM_PLUS_TIME_TAB_4[n]** Lead or delay time on plus cams 25–32		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective immediately	Protection level: 7/7	Unit: s	
Data type: DOUBLE	Applies from SW version: 4.1		
Significance:	A lead or delay time can be assigned to each **plus cam 25–32** in this setting data to compensate for lead or delay times. The switching edge of the associated cam signal is advanced or delayed by the time value entered. Positive value: Lead time Negative value: Delay time Index [n] of the setting data addresses the cam pair: n = 8, 9, ... , 15 corresponds to cam pair 25, 26, ... , 32 This setting data is added to MD: SW_CAM_PLUS_LEAD_TIME[n+8].		
Related to	MD: SW_CAM_PLUS_LEAD_TIME[n] (lead or delay time on plus cams 1–16)		

■

Notes

Signal Description

5

5.1 Signal overview

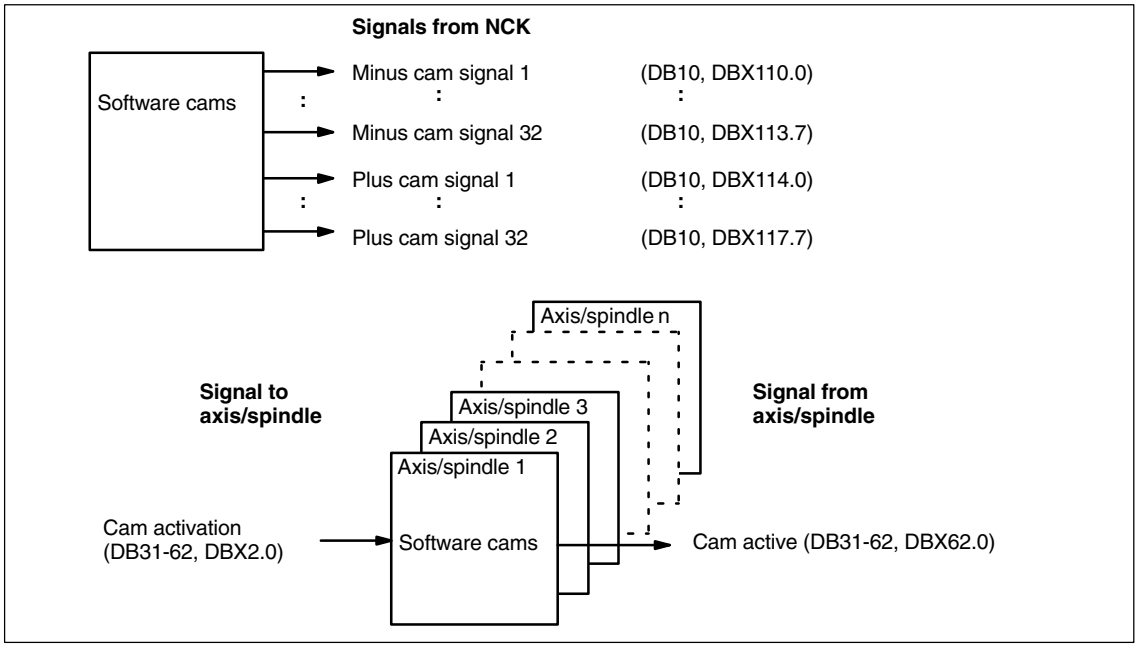


Fig. 5-1 PLC interface signals for “Software cams, position switching signals”

5.2 General signals

5.2.1 Signals from NCK

DB10 **DBX110.0-113.7** Data block	**Minus cam signals 1-32** Signal(s) from NCK (NCK → PLC)		
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1	
Signal state 1 or signal transition 0 ——> 1	The switching edge of the minus cam signal 1–32 is generated as a function of the traversing direction of the (rotary) axis and transferred to the PLC interface in the IPO cycle. **Linear axis:** – The minus cam signal switches from 0 to 1 if the axis overtravels the minus cam in the negative axis direction. **Modulo rotary axis:** – The minus cam signal changes level in response to every positive edge of the plus cam signal.		
Signal state 0 or signal transition 1 ——> 0	**Linear axis:** – The minus cam signal switches from 1 to 0 if the axis overtravels the minus cam in the positive axis direction. **Modulo rotary axis:** – The minus cam signal changes level in response to every positive edge of the plus cam signal.		

DB10 **DBX114.0–117.7** Data block	**Plus cam signals 1–32** Signal(s) from NCK (NCK → PLC)		
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1	
Signal state 1 or signal transition 0 ——> 1	The switching edge of the plus cam signal 1–32 is generated as a function of the traversing direction of the (rotary) axis and transferred to the PLC interface in the IPO cycle. **Linear axis:** – The plus cam signal switches from 0 to 1 if the axis overtravels the plus cam in the positive axis direction. **Modulo rotary axis:** – The plus cam signal switches from 0 to 1 when the minus cam is overtraveled in the positive axis direction. The described response of the plus cam applies under the condition: Plus cam – minus cam < 180 degrees If this condition is not fulfilled or if the minus cam is set to a greater value than the plus cam, then the response of the plus cam signal is inverted. The response of the minus cam signal remains unchanged.		
Signal state 0 or signal transition 1 ——> 0	**Linear axis:** – The plus cam signal switches from 1 to 0 if the axis overtravels the plus cam in the negative direction. **Modulo rotary axis:** – The plus cam signal switches from 1 back to 0 if the plus cam is overtraveled in the positive axis direction. The described response of the plus cam applies under the condition: Plus cam – minus cam < 180 degrees If this condition is not fulfilled or if the minus cam is set to a greater value than the plus cam, then the response of the plus cam signal is inverted. The response of the minus cam signal remains unchanged.		

5.3 Axis/spindle-specific signals

5.3.1 Signals to axis/spindle

DB 31–62 **DBX2.0** Data block	**Cam activation** Signal(s) to axis/spindle (PLC → NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	Output of the minus and plus cam signals of an axis to the general PLC interface is activated. The activation takes effect immediately after processing of IS “Cam activation” in the NCK.	
Signal state 0 or signal transition 1 ——> 0	The minus and plus cam signals of an axis are **not** output to the general PLC interface.	
Related to	IS “Minus cam signal 1–32” (DB10, DBX110.0–113.7) IS “Plus cam signal 1–32” (DB10, DBX114.0–117.7)	

5.3.2 Signals from axis/spindle

DB 31–62 **DBX62.0** Data block	**Cams active** Signal(s) from axis/spindle (NCK → PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	All cams of the axis selected via IS “Cam activation” (DB31–48, DBX2.0) have been activated successfully.	
Signal state 0 or signal transition 1 ——> 0	The cams of the axis are **not** activated.	
Related to	IS “Cam activation” (DB31–62, DBX2.0) IS “Minus cam signal 1–32” (DB10, DBX110.0–113.7) IS “Plus cam signal 1–32” (DB10, DBX114.0–117.7)	

■

Notes

6 Example

– None –

7 Data Fields, Lists

7.1 Interface signals

DB number	Bit, byte	Name	Reference
General			
10	110.0 ... 110.7	Minus cam signal 1...8	
10	111.0 ... 111.7	Minus cam signal 9...16	
10	112.0 ... 112.7	Minus cam signal 17...24	
10	113.0 ... 113.7	Minus cam signal 25...32	
10	114.0 ... 114.7	Plus cam signal 1...8	
10	115.0 ... 115.7	Plus cam signal 9...16	
10	116.0 ... 116.7	Plus cam signal 17...24	
10	117.0 ... 117.7	Plus cam signal 25...32	
Axis-specific			
31–62	2.0	Cam activation	
31–62	62.0	Cam active	

7.2 Machine data

Number	Identifier	Name	Reference
General ($MN_ ...)			
10260	CONVERT_SCALING_SYSTEM	Basic system switchover active	G2
10270	POS_TAB_SCALING_SYSTEM	Measuring system of position tables	T1
10450	SW_CAM_ASSIGN_TAB[n]	Assignment software cams to machine axes	

General ($MN_ ...)			
10460	SW_CAM_MINUS_LEAD_TIME[n]	Lead or delay time on minus cams 1–16	
10461	SW_CAM_PLUS_LEAD_TIME[n]	Lead or delay time on plus cams 1–16	
10470	SW_CAM_ASSIGN_FASTOUT_1	Hardware assignment for output of cams 1–8 to NCK I/Os	
10471	SW_CAM_ASSIGN_FASTOUT_2	Hardware assignment for output of cams 9–16 to NCK I/Os	
10472	SW_CAM_ASSIGN_FASTOUT_3	Hardware assignment for output of cams 17–24 to NCK I/Os	
10473	SW_CAM_ASSIGN_FASTOUT_4	Hardware assignment for output of cams 25–32 to NCK I/Os	
10480	SW_CAM_TIMER_FASTOUT_MASK	Screen form for output of cam signals via timer interrupts on NCU	

7.3 Setting data

Number	Identifier	Name	Reference
General ($MN_ ...)			
41500	SW_CAM_MINUS_POS_TAB_1[n]	Position of minus cams 1–8	
41501	SW_CAM_PLUS_POS_TAB_1[n]	Position of plus cams 1–8	
41502	SW_CAM_MINUS_POS_TAB_2[n]	Position of minus cams 9–16	
41503	SW_CAM_PLUS_POS_TAB_2[n]	Position of plus cams 9–16	
41504	SW_CAM_MINUS_POS_TAB_3[n]	Position of minus cams 17–24	
41505	SW_CAM_PLUS_POS_TAB_3[n]	Position of plus cams 17–24	
41506	SW_CAM_MINUS_POS_TAB_4[n]	Position of minus cams 25–32	
41507	SW_CAM_PLUS_POS_TAB_4[n]	Position of plus cams 25–32	
41520	SW_CAM_MINUS_TIME_TAB_1[n]	Lead or delay time on minus cams 1–8	
41521	SW_CAM_PLUS_TIME_TAB_1[n]	Lead or delay time on plus cams 1–8	
41522	SW_CAM_MINUS_TIME_TAB_2[n]	Lead or delay time on minus cams 9–16	
41523	SW_CAM_PLUS_TIME_TAB_2[n]	Lead or delay time on plus cams 9–16	
41524	SW_CAM_MINUS_TIME_TAB_3[n]	Lead or delay time on minus cams 17–24	
41525	SW_CAM_PLUS_TIME_TAB_3[n]	Lead or delay time on plus cams 17–24	
41526	SW_CAM_MINUS_TIME_TAB_4[n]	Lead or delay time on minus cams 25–32	
41527	SW_CAM_PLUS_TIME_TAB_4[n]	Lead or delay time on plus cams 25–32	

7.4 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

■

Notes

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

Punching and Nibbling (N4)

■

Brief Description 1

The functions specific to punching and nibbling operations comprise the following:

- Stroke control
- Automatic path segmentation
- Rotatable punch and die
- Clamp protection

including their activation and deactivation via language commands.

Notes

Detailed Description 2

2.1 Stroke control

The stroke control is used in the actual machining of the workpiece. The punch is activated via an NC output signal when the position is reached. The punching unit acknowledges its punching motion with an input signal to the NC. No axis may move within this time period. Repositioning takes place after the punching operation.

High-speed signals

"High-speed signals" are used for direct communication between the NC and punching unit. Combined with the punch, they allow a large number of holes to be punched per minute since the punch positioning times are interpreted as machining delays.

PLC signals

PLC interface signals are used for non-time-critical functions such as enabling and monitoring.

2.1.1 High-speed signals

High-speed signals are used to synchronize the NC and punching unit. On the one hand, they are applied via a high-speed output to ensure that the punch stroke is not initiated until the metal sheet is stationary. On the other, they are applied via a high-speed input to ensure that the sheet remains stationary while the punch is active. The high-speed digital inputs and outputs on the control are used to operate the punching unit.
The following diagram shows the signal sequence.

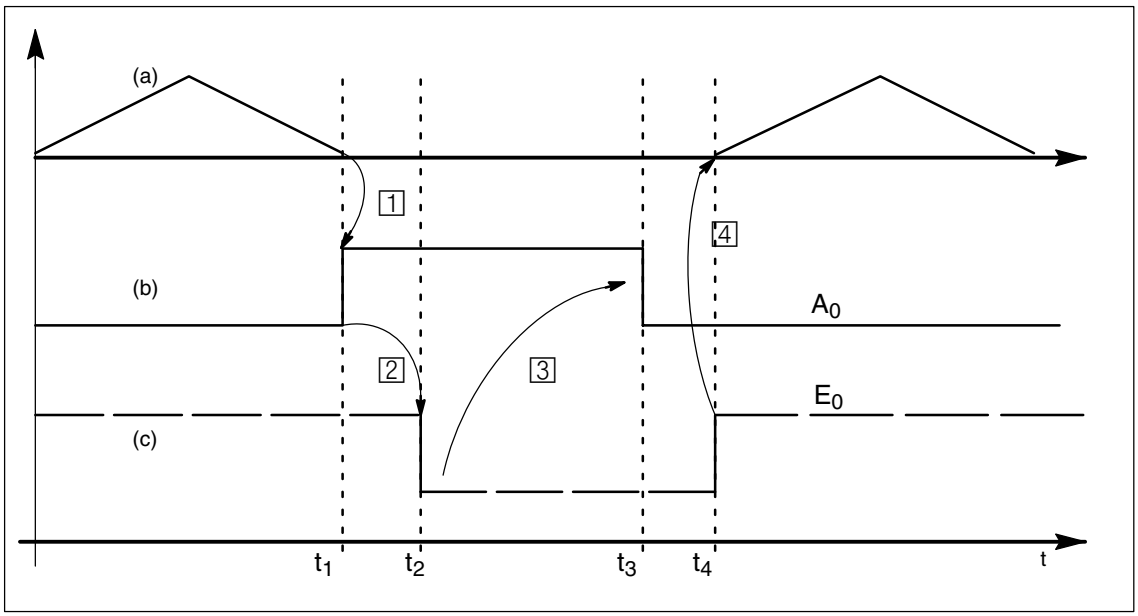


Fig. 2-1 Signal chart

Note

The diagram illustrates the following:

(a) Axis motion of the machine as a v(t) function
(b) “Stroke activation” signal
(c) “Stroke active” signal

The “Stroke active” signal is high-active for reasons relating to open-circuit monitoring.

The chronological sequence of events for punching and nibbling is controlled by the two signals A_0 and E_0.
A_0 is set by the NCK and is identical to stroke initiation.
E_0 defines the status of the punching unit and is identical to the “Stroke active” signal.
The signal states characterize and define times t_1 to t_4 in the following way:

t_1

The motion of the workpiece (metal sheet) in relation to the punch is completed at instant t_1. Depending on the criterion defined for stroke activation (see following section “Criteria for stroke initiation”), high-speed output A_0 is set for punch initiation 1.

t_2

The punching unit signals a punch movement via high-speed input E_0 at instant t_2. This is triggered by signal A0 2.

For safety reasons, signal E_0 is high-active (in the case of an open circuit, “Stroke active” is always set and the axes do not move).

The “Stroke active” signal is not reset again until the tool has moved away from the metal sheet (t_4).

t_3

The NC reacts to the “Stroke active” signal at instant t_3 by cancelling the “Stroke activation” signal 3. From this point in time onwards, the NC is in a waiting state. It simply waits for cancellation of the “Stroke active” signal so it can initiate the next axis motion. The next stroke can be initiated only after signal A_0 has disappeared.

t_4

The punching operation is complete at instant t_4, i.e. the punch has exited from the metal sheet again. The NC reacts to a signal transition in signal E_0 by starting an axis motion 4. The reaction of the NC to a signal edge change 4 is described in the section headed “Axis start after punching” below.

Note

The stroke time is determined by the period $\Delta t_h = t_4 - t_1$. Reaction times at instant t_4 between the signal transition of E_0 and the start of the axis motion must also be added.

2.1.2 Criteria for stroke initiation

Initiate stroke

The stroke initiation signal must not be set before it is assured that the axes have reached zero speed. In this way, it can be guaranteed that no relative motion between the punching tool and the metal sheet is taking place on the machining plane while punching is in progress.

The following diagram shows the various criteria that can be applied to stroke initiation.

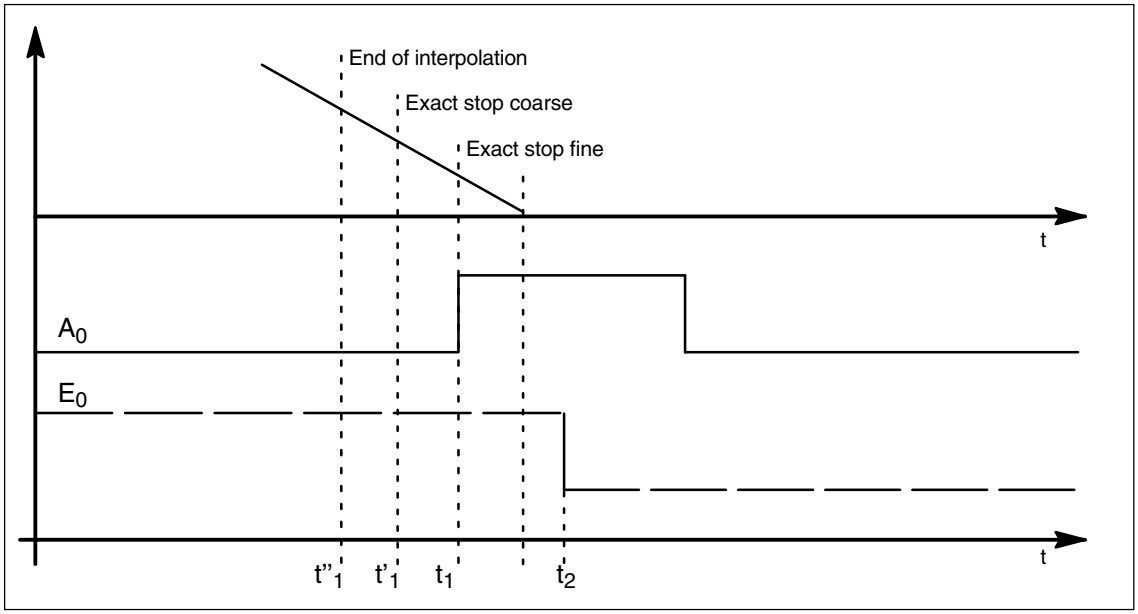


Fig. 2-2 Signal chart: Criteria for stroke initiation

The time interval between t_1 and t_2 is determined by the reaction of the punching unit to setting of output A_0. This cannot be altered, but can be utilized as a lead time for minimizing deadtimes. The diagram above shows the default setting with which the output is set when the “Exact stop fine window” is reached (default setting of G group 12 G601). The punch initiation times t''_1 and t'_1 are programmed by means of G602 and G603 (see table below).

If	then	Description
G603 is programmed,	stop the interpolation	The interpolation reaches the block end. In this case, the axes continue to move until the overtravel has been traversed, i.e. the signal is output at an appreciable interval before the axes have reached zero speed (see t''_1).
G602 is programmed,	reach the coarse in-position window	The signal is output once the axes have reached the coarse in-position window. If this criterion is selected for stroke initiation output, then the instant of stroke initiation can be varied through the size of interpolation window (see t'_1).
G601 is programmed,	reach the fine in-position window	In this case, it can always be ensured that the machine will have reached a standstill at the instant of punching provided that the axis data are set well. However, this variant also guarantees a maximum deadtime (see t_1).
Note: The initial setting of the G group with G601, G602 and G603 (G group 12) is defined via MD: GCODE_RESET_VALUES[11] (G601 is the default setting)		

G603

Depending on velocity and machine dynamics, approximately 3–5 interpolation cycles are processed at the end of interpolation before the axes reach zero speed.

In combination with machine data 26018: NIBBLE_PRE_START_TIME, it is possible to delay, and therefore optimize, the instant between reaching the end of interpolation and setting the high-speed output for “Stroke ON”.

Apart from MD 26018: NIBBLE_PRE_START_TIME, SD 42402 NIB_PUNCH_PRE_START_TIME is also available. This can be altered from the part program and thus adapted to the punching process depending on the processing status of the part program.

The following delay times apply depending on the value programmed for the setting data:

MD 26018 = 0 → SD 42402 is active.
MD 26018 ≠ 0 → MD 26018 is active.

If the “Punching with dwell time, PDELAYON” is active, then the dwell time programmed in connection with this function is active. Both MD 26018 and SD 42402 are inoperative.

2.1.3 Axis start after punching

The start of an axis motion after stroke initiation is controlled via input signal “Stroke ON”.

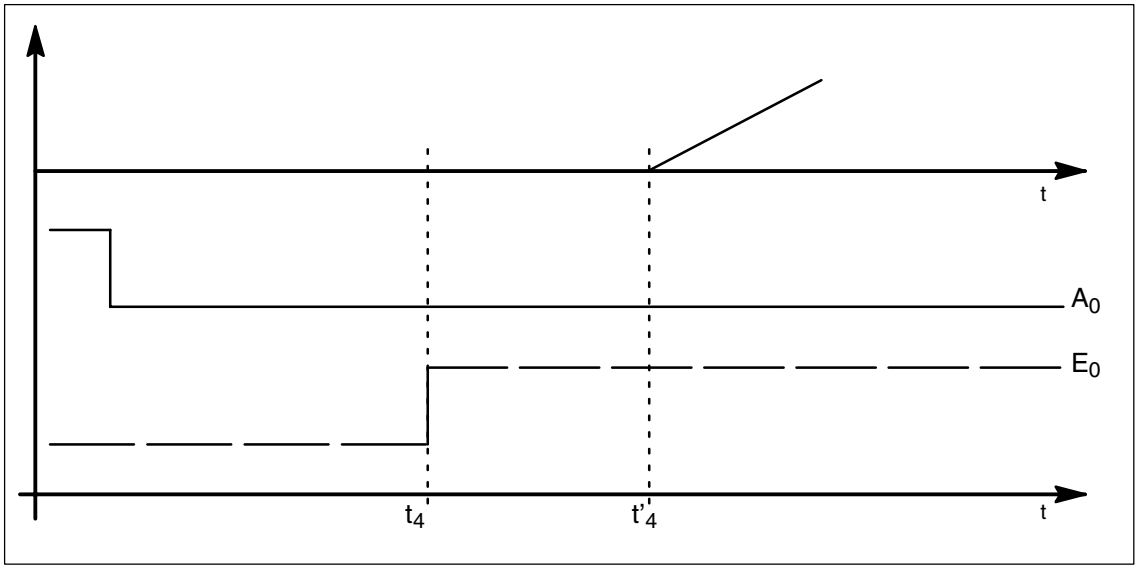


Fig. 2-3 Signal chart: Axis start after punching

In this case, the time interval between t_4 and t'_4 acts as a switching-time-dependent reaction time. It is determined by the interpolation sampling time and the programmed punching/nibbling mode.

PON/SON

When the punching unit is controlled by means of PON/SON, the maximum delay time is calculated as $|t'_4 - t_4|$ = 3 x interpolation cycle.

PONS/SONS

If the punch is controlled by means of PONS/SONS, then the delay time is determined by $|t'_4 - t_4| \leq$ 3 x position control cycle.
(Precondition: Stroke time $(t_4 - t_2)$ > 4 interpolation cycles).

2.1.4 PLC signals specific to punching and nibbling

In addition to the signals used for direct stroke control, channel-specific PLC interface signals are also available. These are used both to control the punching process and to display operational states.

The “No stroke enable” signal prevents the NC from initiating any punching operation. The NC waits until the enable signal is available before continuing the part program. The “Stroke suppression” signal allows the part program to be processed without initiating a punching operation (dry run). With active path segmentation, the axes traverse in “Stop and go” mode. The “Delayed stroke” signal activates a delayed stroke output such as that programmable with PDELAYON. The “Manual stroke initiation” signal allows the operate the initiate a punching process (controlled via PLC), even when the part program is not being processed. This signal is acknowledged by the “Acknowledge manual stroke initiation” signal.

Note

The signals from/to channel are described in Chapter 5 and are listed in Chapter 7.

2.1.5 Punching and nibbling-specific reactions to standard PLC signals

“Feed stop” interface signal

In the case of a “Feed stop” signal, the NC reacts as follows with respect to the stroke control:
If the signal is detected before instant t_1, then stroke initiation is suppressed. The next stroke is not initiated until the next start or until the “Feed stop” signal has been cancelled. Machining is then continued as if there had been no interruption. If the signal is detected at instant t_1, then the current stroke is completed and the NC then rests in the state characterized by t_4. To allow it to respond in this manner, time monitoring of the “Stroke active” and “Stroke initiation” signals is dispensed with.

2.1.6 Signal monitoring

Owing to ageing of the punch hydraulics, overshooting of the punch may cause the “Stroke active” signal to oscillate at the end of a stroke. In such cases, an alarm may be generated depending on machine data 26020: NIBBLE_SIGNAL_CHECK (alarm 22054 “distorted punch signal”).

Reset response

In the case of an NC reset, the “Stroke initiation” signal is cancelled immediately even if the acknowledgement via the high-speed input has not arrived. A currently activated stroke cannot be suppressed.

2.2 Activation and deactivation

2.2.1 Language commands

Punching and nibbling functions are activated and deactivated via configurable language commands. These replace the special M functions that were used in earlier systems.

References: /PA/, Programming Guide

Groups

The language commands are arranged in groups as follows:

Group 35
The actual punching and nibbling functions are activated and deactivated by means of the following language commands:

PON = Punching ON
SON = Nibbling ON
PONS = Punching ON, activation in position controller
SONS = Nibbling ON, activation in position controller
SPOF = Punching/nibbling OFF

Group 36
This group includes the commands which have only a preparatory character and which determine the real nature of the punching function. These language commands are as follows:

PDELAYON = Punching with delay ON
PDELAYOF = Punching with delay OFF

Since the PLC normally needs to perform some preliminary tasks with respect to these preparatory functions, they are programmed before the activating commands.

Group 38
This group contains the commands for switching over to a second punch interface. It can be used, for example, for a second punching unit or set of hammer shears. A second I/O pair which can be used for punching functionality is defined via machine data.

SPIF1 = First interface is active
SPIF2 = Second interface is active

Note

Only one function at a time can be active within a G code group (similar, for example, to the various interpolation modes G0, G1, G2, G3, etc., which are also mutually exclusive).

SPOF

Punching and nibbling OFF
The SPOF function terminates all punching and nibbling functions. In this state, the NCK responds neither to the “Stroke active” signal nor to the PLC signals specific to punching and nibbling functions.

If SPOF is programmed together with a travel command in one block (and in all further blocks if punching/nibbling is not activated with SON or PON), the machine approaches the programmed position without the initiation of a punching operation. SPOF deselects SON, SONS, PON and PONS and is equivalent to a Reset state.

Programming example:

```
:
:
N20   G90   X100        SON   Activate punching
N25   X50   SPOF              Deactivate punching,
:                             Positioning without stroke initiation
:
```

SON

Nibbling ON

SON activates the nibbling function and deselects the other functions in G group 35 (e.g. PON).
In contrast to punching, the first stroke is made at the start point of the block with the activating command, i.e. before the first machine motion. SON has a modal action, i.e. it remains active until either SPOF or PON is programmed or until the program end is reached. The stroke initiation is suppressed in blocks without traversing information relating to the axes designated as punching or nibbling axes (typically those in the active plane). If a stroke still needs to be initiated, then one of the punching/nibbling axes must be programmed with a 0 traversing path. If the first block with SON is a block without traversing information of the type mentioned, then only one stroke takes place in this block since the start and end points are identical.

Programming example:

```
:
:
N70   X50   SPOF              Positioning without punch initiation
N80   X100  SON               Activate nibbling, initiate a stroke
:                             before motion (X=50) and at end of
:                             programmed motion (X=100)
```

SONS

Nibbling ON (in position control cycle)

SONS acts in the same way as SON. The function is activated in the position control cycle, thus allowing time-optimized stroke initiation and an increase in the punching rate per minute.

PON

Punching ON

PON activates the punching function and deactivates SON.Like SON, PON also has a modal action.
In contrast to SON, however, a stroke is not executed until the end of the block or, in the case of automatic path segmentation, at the end of a path segment. PON has an identical action to SON in the case of blocks which contain no traversing information.

Programming example:

```
:
:
N100  Y30   SPOF              Positioning without punch initiation
N110  Y100  PON               Activate punching, initiate punch
:                             at end of positioning operation (Y=100)
:
```

PONS

Punching ON (in position controller)
PONS acts in the same way as PON. For explanation, please refer to SONS.

PDELAYON

Punching with delay ON
PDELAYON is a preparatory function. This means that PDELAYON is generally programmed before PON. The punch stroke is output with a delay when the programmed end position is reached. The delay time can be defined in seconds by setting data 42400: PUNCH_DWELLTIME. If the defined value cannot be divided as an integer into the interpolation clock cycle, then it is rounded to the next divisible integer value. The function has a modal action.

PDELAYOF

Punching with delay OFF
PDELAYOF deactivates punching with delay function, i.e. the punching process continues normally. PDELAYON and PDELAYOF form a G code group.

Programming example:

```
:
:
N170  PDELAYON     X100     SPOF   Positioning without stroke initiation,
:                                  activate delayed punch initiation
:
N180  X800  PON                    Activate punching. When end
:                                  position is reached, a punch
:                                  stroke is output with a delay.
N190  PDELAYOF     X700            Deactivate delayed punching,
:                                  activate normal
:                                  punch initiation. End of
:                                  programmed motion.
```

SPIF1

Activation of first punch interface

SPIF1 activates the first punch interface, i.e. the stroke is controlled via the first pair of high-speed I/Os (see machine data 26004: NIBBLE_PUNCH_OUTMASK, MD 26006: NIBBLE_PUNCH_INMASK). The first punch interface is always active after a reset or control system power up. If only one interface is used, then it need not be programmed.

SPIF2

Activation of second punch interface

SPIF2 activates the second punch interface, i.e. the stroke is controlled via the second pair of high-speed I/Os (see machine data 26004: NIBBLE_PUNCH_OUTMASK, MD 26006: NIBBLE_PUNCH_INMASK).

Programming example:

```
:
:
N170  SPIF1        X100    PON   At the end of the block
 :                               a stroke is initiated at the
 :                               first high-speed output. The “Stroke
 :                               active” signal is monitored
 :                               at the first input.
N180  X800  SPIF2                The second stroke is initiated at
 :                               the second high-speed output.
 :                               The “Stroke active” signal is
 :                               monitored at the second input.
N190  SPIF1        X700          The first interface is used to
 :                               control all further strokes.
```

2.2.2 Compatibility with earlier systems

Use of M functions

As in earlier versions, macro technology allows special M functions to be used instead of language commands (compatibility).
The following assignments corresponding to those used in earlier systems apply:

M20, M23 ≐ SPOF
M22 ≐ SON
M25 ≐ PON
M26 ≐ PDELAYON

Note

The M functions can be configured via machine data. As regards the assignments between the M functions and language commands, it must be noted that the M functions are divided into auxiliary function groups.

Examples

DEFINE M20 AS SPOF or DEFINE M20 AS SPOF M=20	Punching/nibbling OFF Punching with auxiliary function output
DEFINE M20 AS SPOF PDELAYOF	Punching/nibbling OFF and delayed punching OFF
DEFINE M22 AS SON or DEFINE M22 AS SON M=22	Nibbling ON Nibbling ON with auxiliary function output
DEFINE M25 AS PON or DEFINE M25 AS PON M=25	Punching ON Punching ON with auxiliary function output
DEFINE M26 AS PDELAYON or DEFINE M26 AS PDELAYON M=26	Delayed punching Punching and auxiliary function output

Programming example:

```
:
:
N100  X100  M20                 Positioning without punch initiation
N110  X120  M22                 Activate nibbling, stroke initiation before
:                               and after motion
N120  X150  Y150        M25     Activate punching,
:                               stroke initiation at end of motion.
:
```

2.3 Automatic path segmentation

One of the following two methods can be applied to automatically segment a programmed traversing path:

- Path segmentation with maximum path feed value programmed via language command SPP and
- Path segmentation with a number of segments programmed via language command SPN

Both functions generate sub-blocks independently. In earlier systems, language command SPP <number> corresponds to E <number> and SPN <number> to H <number>. Since addresses E and H now represent auxiliary functions, language commands SPP and SPN are used to avoid conflicts. The new procedure is therefore not compatible with those implemented in earlier systems. Both language commands (SPP and SPN) can be configured.

Note

The values programmed with SPP are either mm or inch settings depending on the initial setting (analogous to axes).

The automatic path segmentation function ensures that the path is divided into equidistant sections with linear and circle interpolation.

When the program is interrupted and automatic path segmentation is active (SPP/SPN), the contour can be reentered only at the beginning of the segmented block.

SPP

- The automatic path segmentation function SPP divides the programmed traversing path into sections of equal size as a function of the programmed feed path.
- Path segmentation is active only when SON or PON is active. (Exception: MD 26014: PUNCH_PATH_SPLITTING = 1).
- SPP is modally active, i.e. the programmed feed value remains valid until it is programmed again, but it can be suppressed on a block-by-block (non-modal) basis by means of SPN.
- The path segments are rounded off by the control system if required so that a total programmed distance can be divided into an integral number of path sections.
- The feed value unit is either mm/stroke or inch/stroke (depending on axis settings).
- If the programmed SPP value is greater than the traversing distance, then the axis is positioned on the programmed end position without path segmentation.
- SPP = 0, reset or program end delete the programmed SPP value. SPP is not deleted when punching/nibbling is deactivated.

SPN

- The automatic path segmentation function SPN divides the traversing path into the programmed number of path segments.
- SPN is active non-modally and is activated if SON or PON has already been activated.
  (Exception: MD 26014: PUNCH_PATH_SPLITTING = 1).
- Any previously programmed SPP value is suppressed for the block containing SPN, but is re-activated again in the following blocks.

Supplementary conditions

- Path segmentation is active for linear and circular interpolation.
  The interpolation mode remains valid, i.e. circles are traversed in the case of circular interpolation.
- If a block contains both SPN (number of strokes) and SPP (stroke path), then the “number of blocks” is activated in the current block while the “stroke path” is activated in all blocks that follow.
- Path segmentation is active only in conjunction with punching or nibbling functions.
  (Exception: MD PUNCH_PATH_SPLITTING = 1).
- Any programmed auxiliary functions are output before, during the first or after the last sub-block.
- In the case of blocks without traversing information, the same rules apply to programmed SPP and SPN commands as defined for SON and PON, i.e. a stroke is initiated only if an axis motion has been programmed.

2.3.1 Operating characteristics with path axes

All axes defined and programmed via machine data PUNCHNIB_AXIS_MASK (26010) are traversed along path sections of identical size with SPP and SPN until the programmed end point is reached. This also applies to rotatable tool axes if programmed. The response can be adjusted for single axes (see below).

Example of SPP

```
N1   G01    X0     Y0        SPOF   Position without punch initiation
N2   X75    SPP=25           SON    Nibble at feed value 25 mm;
:                                   Initiate punch before first
:                                   motion and after each path segment.
N3   Y10                            Position with reduced SPP value
:                                   because traversing distance <
:                                   SPP value and initiate punch after
:                                   motion.
N4   X0                             Reposition with initiation of punch
:                                   after every path segment.
```

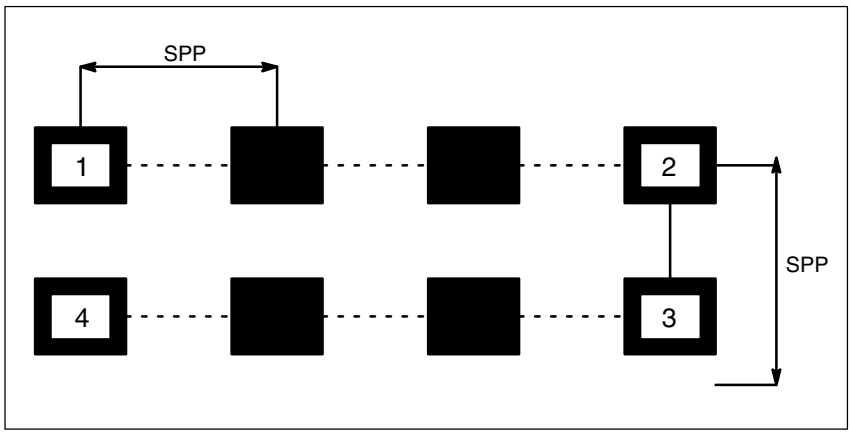


Fig. 2-4

If the programmed path segmentation is not an integral multiple of the total path, then the feed path is reduced (see following diagram).

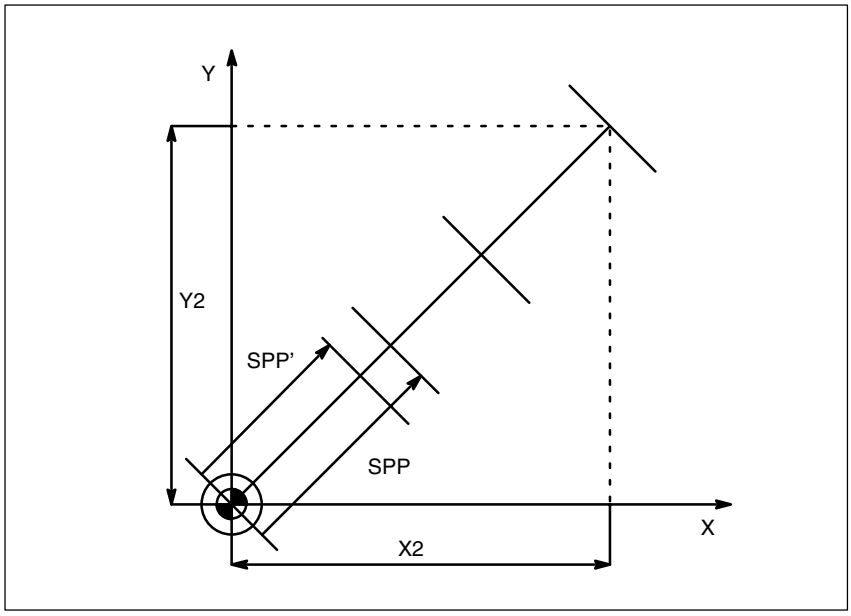


Fig. 2-5 Path segmentation

X2/Y2	Programmed traversing path
SPP	Programmed SPP value
SPP'	Automatically rounded offset path

Example of SPN

The number of path segments per block is programmed via SPN.

A value programmed via SPN takes effect on a non-modal basis for both punching and nibbling applications. The only difference between the two modes is with respect to the first stroke. In the case of nibbling operations, this is executed at the beginning of the first segment. With punching, however, it is executed at the end of the first segment. This means that when n segments are programmed, n strokes are executed with punching operations but n + 1 with nibbling. Furthermore, only one stroke is executed in blocks containing no traversing information, even if several of them are programmed. Should it be necessary to generate several strokes at one specific point, then a corresponding number of blocks without traversing information must be programmed.

```
N1   G01   X0      Y0    SPOF   Position without initiation of punch,
N2   X75   SPN=3   SON          Activate nibbling, the whole path
:                               is divided into 3 segments.
:                               Since nibbling is active, a stroke
:                               is initiated before the first motion
:                               and at the end of each segment.
N3   Y10   SPOF                 Position without initiation of punch
N4   X0    SPN=2   PON          Activate punching, the whole path
:                               is divided into 2 segments.
:                               Since punching is active, the first
:                               stroke is initiated at the
:                               end of the first segment.
```

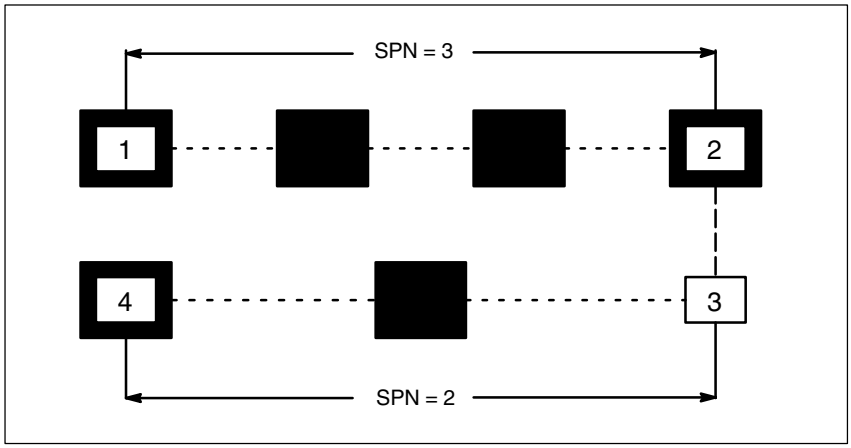


Fig. 2-6

Example

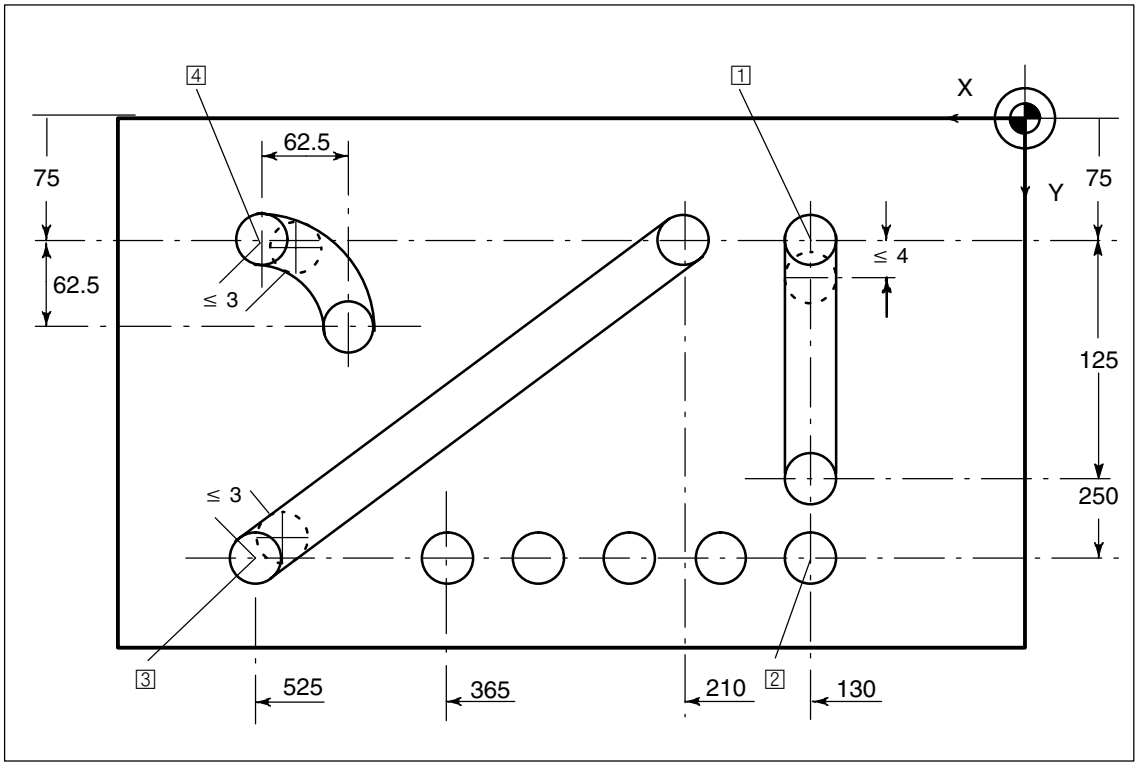


Fig. 2-7 Workpiece

Extract from program

N100	G90	X130	Y75	F60	SPOF	Position at start point 1 of vertical nibble paths
N110	G91	Y125	SPP=4		SON	End point dimensions (incremental); feedrate value: 4 mm, activate nibbling
N120	G90	Y250	SPOF			Absolute dimensioning, Position at start point 2 of horizontal nibble path
N130	X365	SPP=4		SON		End point dimensions, 4 segments, activate nibbling
N140	X525	SPOF				Position at start point 3 of inclined nibble path
N150	X210	Y75	SPP=3		SON	End point dimensions feedrate value: 4 mm, Activate nibbling
N160	X525	SPOF				Position at start point 4 of nibble path on pitch circle path
N170	G02 I0	G91 J62.5	X–62.5 SON		Y62.5	Incremental circular interpolation with interpolation parameters, Activate nibbling
N180	G00	G90	Y300	SPOF		Positioning

2.3.2 Response in connection with single axes

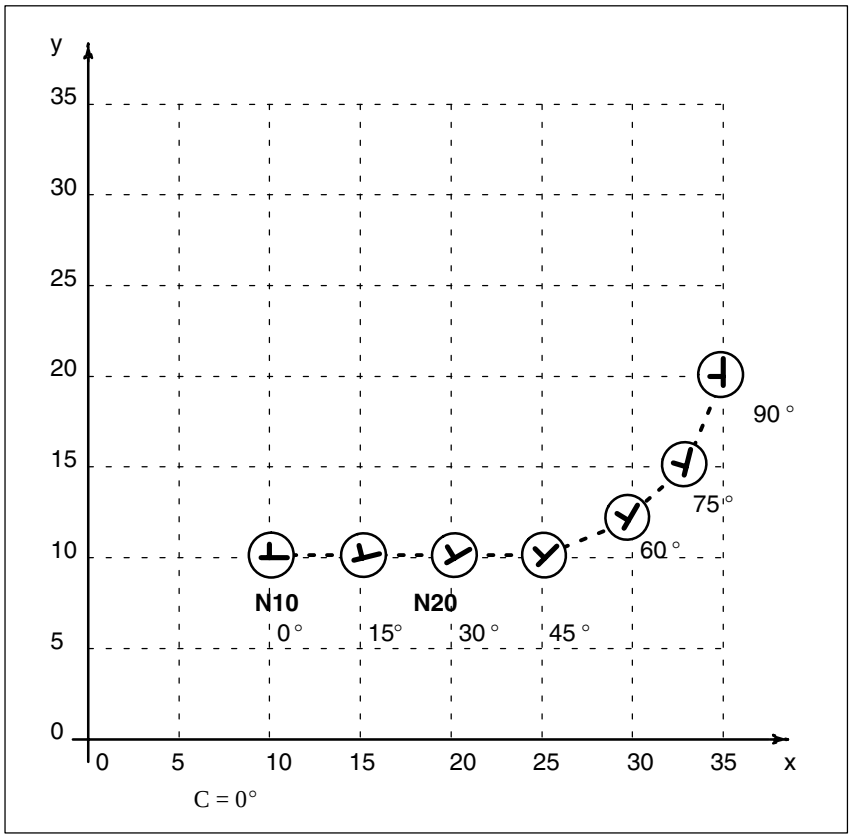
The path of single axes programmed in addition to path axes is distributed evenly among the generated intermediate blocks as standard. In the following example, the additional rotary axis C is defined as a synchronous axis in the system. If this axis is programmed as a “Punch-nibble axis” (via PUNCHNIB_AXIS_MASK = 1 for this axis), then the behavior of the synchronous axis can be varied as a function of machine data PUNCH_PARTITION_TYPE.

Example

```
N10  G1     PON  X10  Y10  C0
N20  SPP=5  X25  C45
N30  SPN=3  X35  Y20  I10  J10  C90
```

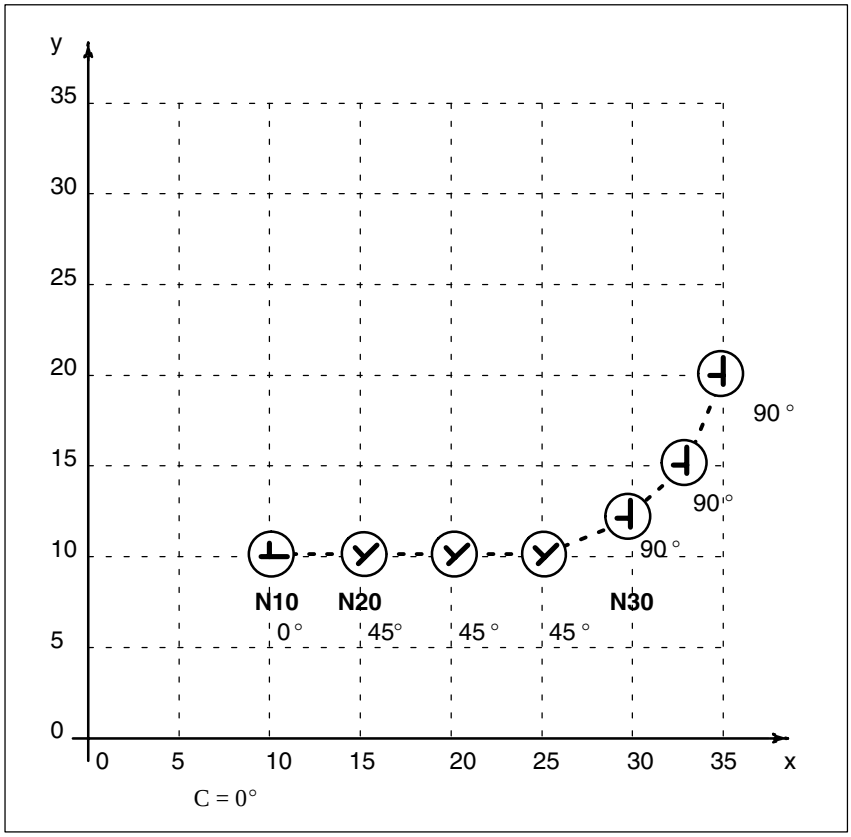
PUNCH_PARTITION_TYPE=0 (default setting)

In the above example, the axes behave as standard, i.e. the programmed special axis motions are distributed among the generated intermediate blocks of the active path segmentation function in all interpolation modes. In block N20, the C axis is rotated through 15° in each of the three intermediate blocks. The axis response is the same in block N30, in the case of circular interpolation (three sub-blocks, each with 15° (axis rotation).



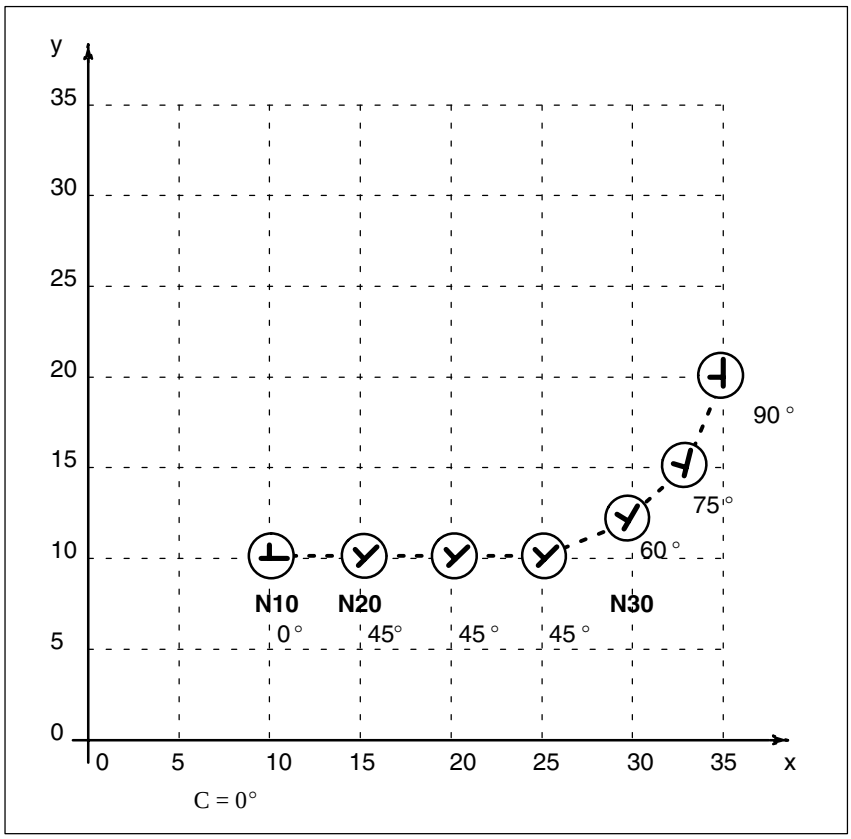
PUNCH_PARTITION_TYPE=1

In contrast to the behavior described above, here the synchronous axis travels the entire programmed rotation path in the first sub-block of the selected path segmentation function. Applied to the above example, the C axis already reaches the programmed end position (C=45) when it reaches X position X=15. It behaves in the same way in the circular interpolation block below.



PUNCH_PARTITION_TYPE=2

MD=2 is set in cases where the axis must behave as described above (PUNCH_PARTITION_TYPE=1) in linear interpolation mode, but according to the default setting in circular interpolation mode (see 1st case). Given the above example, the axis then behaves as follows: In block N20, the C axis is rotated to C=45°. Each of the following circular blocks rotates the C axis through 15°.



The axis response illustrated in the diagram above can be particularly useful when applied to the axis of a rotatable tool in cases where it is used to place the tool in a defined direction (e.g. tangential) in relation to the contour, but where the tangential control function **must not** be applied. However, it is not a substitute for the tangential control function since the start and end positions of the rotary axis must always be programmed.

Note

Additional offset motions of special axes (in this case, rotary axis C) are implemented via a zero offset.

If the C axis is not defined as a “Punch-nibble axis”, then the C axis motion path is not segmented in block N30 in the above example nor is a stroke initiated at the block end.

If the functionality described above is to be implemented in a variant not specific to nibbling applications, but with alignment of the special axis, then stroke initiation can be suppressed by PLC interface signal (stroke suppression). (Application example: Alignment of electron beam during welding).

Similar axis operating characteristics can be obtained by setting
MD 26014 PUNCH_PATH_SPLITTING to “1”.
In this case, the path is segmented irrespective of punching or nibbling functions.

2.4 Rotatable tool

The functions

- “Coupled motion” for synchronous rotation of punch and die and
- “Tangential control” for normal alignment of rotary axes for punching tools in relation to workpiece

can be used on nibbling/punching machines with rotatable punching tool and die to achieve a wide variety of applications for the punch.

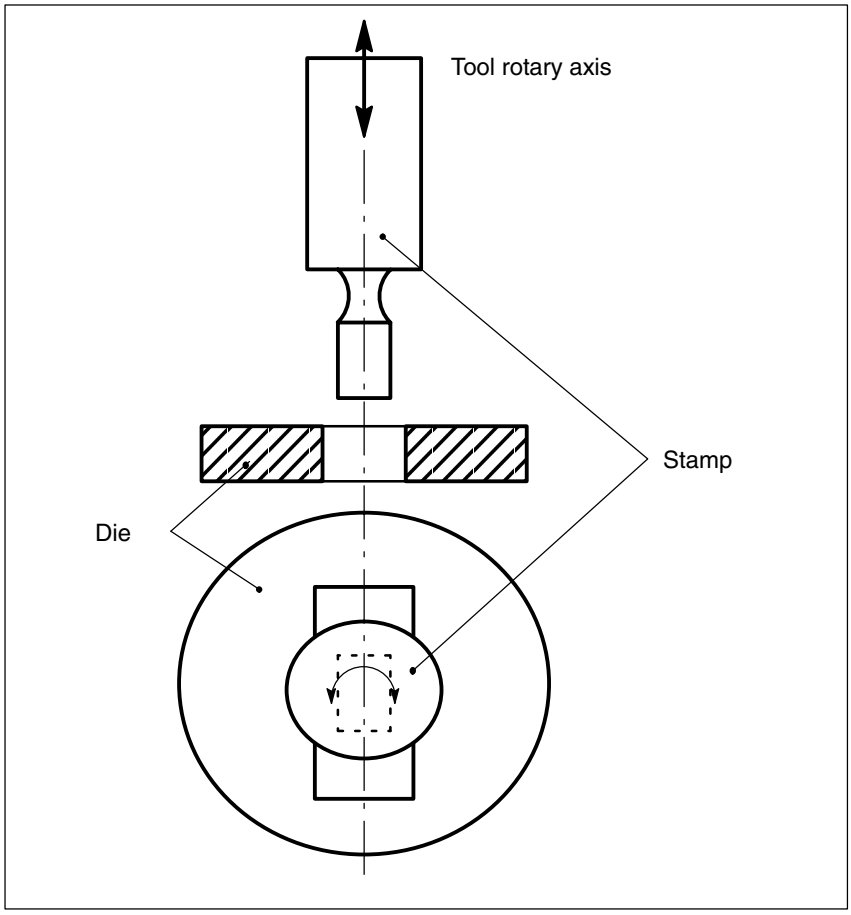


Fig. 2-8 Illustration of a rotatable tool axis

2.4.1 Coupled motion of punch and die

Using the standard function “Coupled motion”, it is possible to assign the axis of the die as a coupled motion axis to the rotary axis of the punch.

Activation

The “Coupled motion” function is activated or deactivated with language commands TRAILON or TRAILOF.

References: /FB/, M3 “Coupled Motion”

Example

Example of a typical nibbling machine with rotatable punching tools where C is the punch axis and C1 the die axis

```
:
:
TRAILON (C, C1, 1);                    Switch on coupled motion grouping
G01   X100  Y100     C0        PON     Initiate stroke with C axis and
                                       C1 axis position C=0=C1
X150  C45                              Initiate stroke with C axis/C1 axis
:                                      position C=45=C1
:
M30
```

Initial setting

No coupled motion groupings are active after power-up. Once the two tool axes have approached the reference point, the coupled-axis grouping is not generally separated again. This can be achieved by activating the coupled-axis grouping once (see above example) and setting machine data 20110 RESET_MODE_MASK, bit 8=1. In this way, the coupled-axis grouping remains active after Reset/part program start or end.

2.4.2 Tangential control

The rotary tool axes on punching/nibbling machines are aligned tangentially to the programmed path of the master axes by means of the tangential control function.

Activation

The “Tangential control” function is activated and deactivated with language commands TANGON or TANGOF respectively.

References: /PA/, Programming Guide, Fundamentals

Mode of operation

The tangential axis is coupled to the interpolation of the master axes. It is therefore not possible to position the axis at the appropriate punching position tangentially to the path independently of velocity. This may lead to a reduction in machining velocity if the dynamics of the rotary axis are unfavorable in relation to those of the master axes. Additional offset angles can be programmed directly via language command TANGON.

Note

If the tool (punch and die) is positioned by 2 separate drives, then the functions “Tangential control” and “Coupled axes” can be used.The tangential control function must be activated first followed by coupled axes.

The tangential control function automatically aligns the punch vertically to the direction vector of the programmed path. The tangential tool is positioned before the first punching operation is executed along the programmed path. The tangential angle is always referred to the positive X axis. A programmed additional angle is added to the calculated angle.

The tangential control function can be used in the linear and circular interpolation modes.

Example

Linear interpolation

The punching/nibbling machine has a rotatable punch and die with separate drives.

```
:
:
N2    TANG (C, X, Y, 1, “B”)
N5    G0    X10     Y5
N8    TRAILON (C, C1, 1)
N10   Y10   C225    PON     F60
N15   X20   Y20     C45
N20   X50   Y20     C90     SPOF
N25   X80   Y20     SPP=10  SON
N30   X60   Y40     SPOF
N32   TANGON (C, 180)
N35   X30   Y70     SPP=3   PON
N40   G91   C45     X–10    Y–10
N42   TANGON (C, 0)
N45   G90   Y30     SPP=3   SON
N50   SPOF  TANGOF
N55   M2
```

Explanations

Table 2-1

Block	Remark
N2	Definition of master and slave axes, C is slave axis for X and Y in the base coordinate system.
N5	Start position
N8	Activation of coupled motion of rotatable tool axes C/C1.
N10	C/C1 axis rotates to 225° stroke
N15	C/C1 axis rotates to 45° stroke

N20	C/C1 axis rotates to 90°, no stroke initiation
N25	Path segmentation: 4 strokes are executed with tool rotated to 90°.
N30	Positioning
N32	Activate tangential control, offset angle of rotatable tool axes 180°
N35	Path segmentation: 3 strokes with active tangential control and an offset angle of 180°
N40	C–/C1 rotates to 225° (180° + 45° INC) tangential control deactivated because path is not segmented ↓ stroke
N42	Tangential control without offset
N45	Path segmentation: 3 strokes with active tangential control but without offset angle
N50	Deactivation of stroke initiation + tangential control

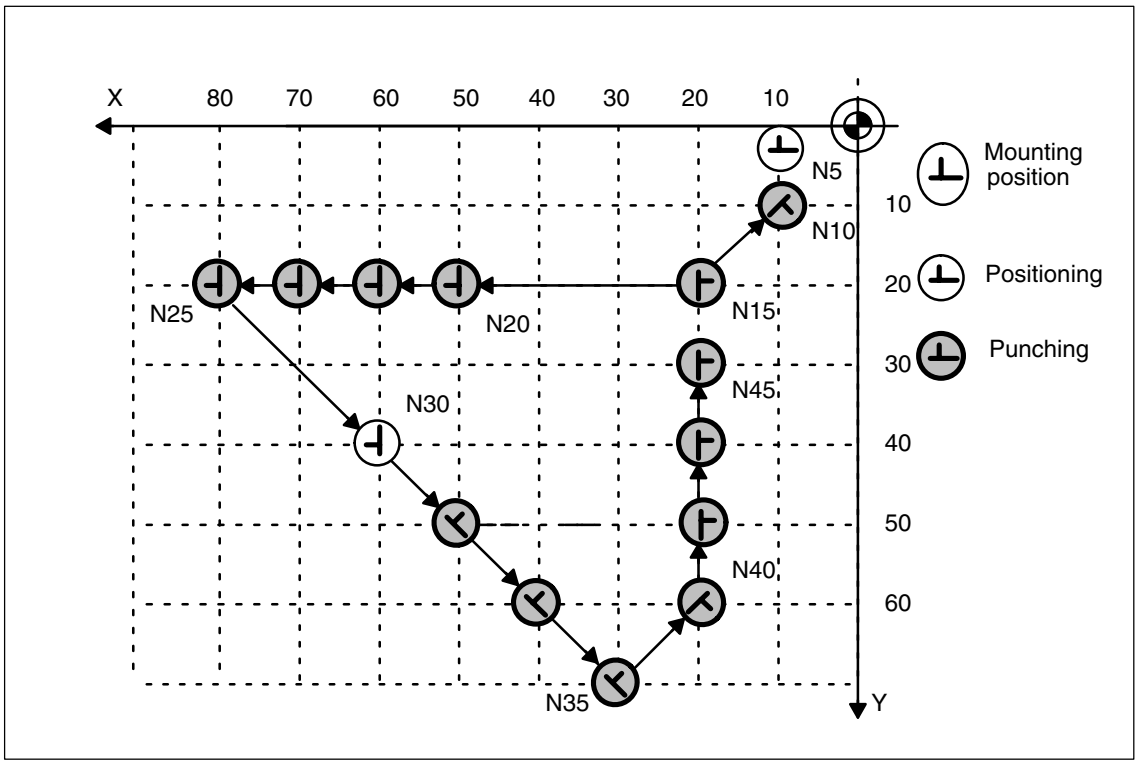


Fig. 2-9 Illustration of programming example in XY plane

Example Circular interpolation

In circular interpolation mode, particularly when path segmentation is active, the tool axes rotate along a path tangentially aligned to the programmed path axes in each sub-block.

```
:
:
N2    TANG (C, X, Y, 1, “B”)
N5    G0    F60     X10     Y10
N8    TRAILON (C, C1, 1)
N9    TRAILON (C, –90)
N10   G02   X30     Y30     I20     J0      SPN=2   PON
N15   G0    X70     Y10     SPOF
N17   TANGON (C, 90)
N20   G03   X35.86  Y24.14  CR=20   SPP=16          SON
N25   G0    X74.14  Y35.86  C0      PON
N27   TANGON (C, 0)
N30   G03   X40     Y50     I–14.14 J14.14 SPN =5   SON
N35   G0    X30     Y65     C90     SPOF
N40   G91   X–10    Y–25    C180
N43   TANGOF
N45   G90   G02     Y60     I0      J10     SPP=2   PON
N50   SPOF
N55   M2
```

Explanations

Table 2-2

Block	Remark
N2	Definition of master and slave axes, C is slave axis for X and Y in the base coordinate system.
N5	Start position
N8	Activate coupled motion of rotatable tool axes C/C1 for punch and die.
N9	Activate tangential control with offset 270°.
N10	Circular interpolation with path segmentation, 2 strokes are executed with 270° offset angle and tangential alignment along circular path.
N15	Positioning.
N17	Activate tangential control with offset 90°.
N20	Circular interpolation with path segmentation, 4 strokes are executed with 90° offset angle and tangential alignment along circular path.
N25	Rotation of tool axes to 0°, stroke.
N27	Activate tangential control with offset 0°.
N30	Circular interpolation with path segmentation, 5 strokes with offset angle 0° and tangential alignment along circular path.
N32	Activate tangential control, offset angle of rotatable tool axes 180°
N35	Position without active tangential control.
N40	Positioning, C axis rotates to 270°
N43	Deactivate tangential control.
N45	Circular interpolation with path segmentation, 2 strokes without tangential control where C=270°.
N50	Punching OFF.

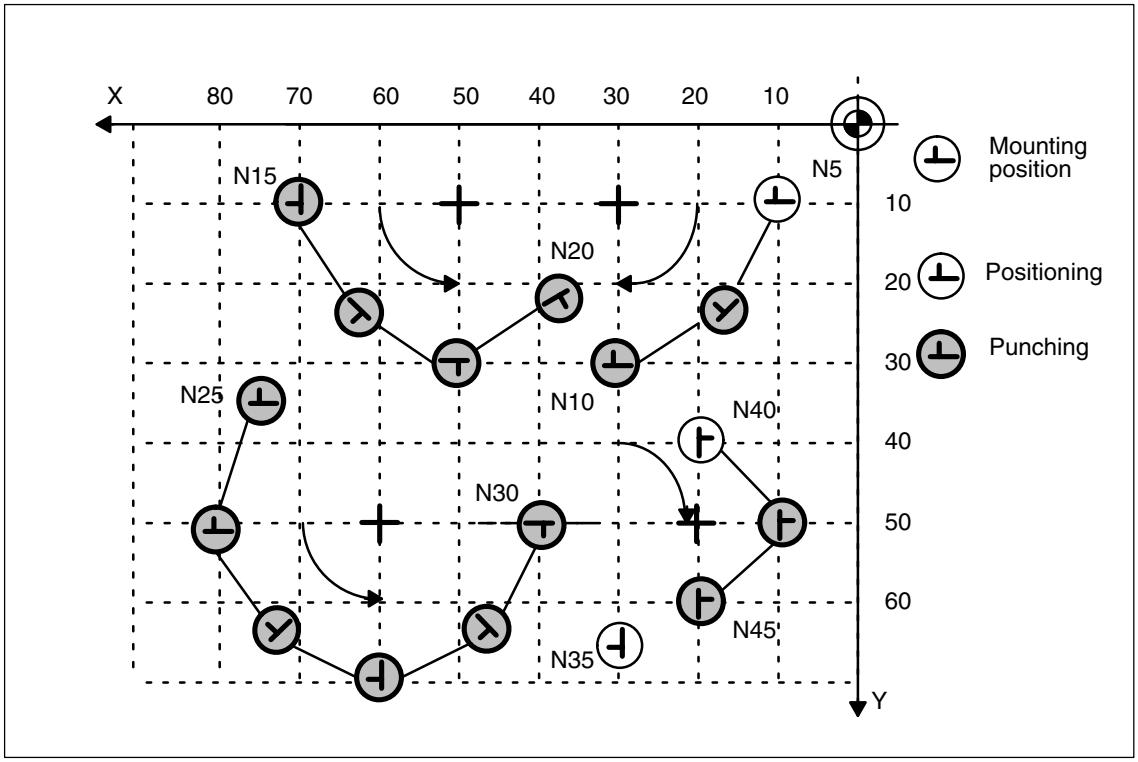


Fig. 2-10 Illustration of programming example in XY plane

2.5 Protection zones

The “clamping protection zone” function is contained as a subset in the “Protection zones” function. Its purpose is to simply monitor whether clamps and tool could represent a mutual risk.

Note

No by-pass strategies are implemented for cases where the clamp protection is violated.

References: /FB/, A3, “Axis Monitoring, Protection Zones”

■

Notes

Supplementary Conditions 3

Availability of "Punching and nibbling" function

This function is an option and available for

- SINUMERIK 840D with NCU 572 and 573, SW 3 and higher

Data Descriptions (MD, SD) 4

4.1 Channel-specific machine data

26000 MD number	**PUNCHNIB_ASSIGN_FASTIN** Hardware assignment for input byte with stroke control		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies from SW version: 3.1		
Significance:	This data defines which input byte is to be used for the signal "Stroke active". = 1: On-board inputs (4 high-speed NCK inputs) are used. = 2, 3, 4, 5 The external digital NCK inputs are used **Example**: Value "0000 0001" Stroke active is HIGH active Value "0001 0001" Stroke active is LOW active **Note:** This MD is not compatible with earlier SW versions. The HIGH word now acts as an inversion mask. In earlier SW versions (< 3.2), the "Stroke active" signal was always LOW active.		
Related to	NIBBLE_PUNCH_INMASK[n]		
References	/FB/, A4, Digital and Analog NCK I/Os The signal is high-active as standard with SW 3.2 and higher, i.e. open-circuit monitoring is implemented. If the signal needs to be low-active, the MD must be set, for example, to a value of "H 0001 0001" for the outboard inputs.		

26002 MD number	**PUNCHNIB_ASSIGN_FASTOUT** Hardware assignment for output byte with stroke control		
Default value: 0	Min. input limit: 0	Max. input limit: plus	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies from SW version: 3		
Significance:	This data defines which output byte is to be used for the stroke control. = 1: On-board outputs (4 high-speed NCK outputs) are used. = 2, 3, 4, 5 The external digital NCK outputs are used		
Related to	NIBBLE_PUNCH_OUTMASK[n]		
References	/FB/, A4, Digital and Analog NCK I/Os		

26004 MD number	**NIBBLE_PUNCH_OUTMASK[n]** Screen form for high-speed output bits		
Default value: see below	Min. input limit: 0	Max. input limit: 128	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: BYTE	Applies from SW version: 3		
Significance:	A total of 8 byte screen forms for the output of high-speed bits can be defined with this data. Two of these are used at the current time.The standard assignment of this data is as follows: NIBBLE_PUNCH_OUTMASK[0] = 1: 2^0 = first bit for the first punch interface (SPIF1) NIBBLE_PUNCH_OUTMASK[1] = 0: Second punch interface (SPIF2), not available by default NIBBLE_PUNCH_OUTMASK[2] = 0 ... NIBBLE_PUNCH_OUTMASK[7] = 0 Note: The significance of the bit to be defined must be input. Bit: 7 6 5 4 3 2 1 0 Significance: 2^7=128 2^6=64 2^5=32 2^4=16 2^3=8 2^2=4 2^1=2 2^0=1 Example: “8” must be entered in the machine data in order to define bit 3.		
Application	NIBBLE_PUNCH_OUTMASK[0] = 1 —> The first bit (bit 0) is defined NIBBLE_PUNCH_OUTMASK[0] = 4 —> The third bit (bit 2) is defined		
Special cases, errors, ...	Only NIBBLE_PUNCH_OUTMASK[0] is relevant. This is used to define the output bit for the signal “Initiate stroke”.		
Related to	PUNCHNIB_ASSIGN_FASTOUT		

26006 MD number	**NIBBLE_PUNCH_INMASK[n]** Screen form for high-speed input bits		
Default value: see below	Min. input limit: 0	Max. input limit: 128	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: BYTE	Applies from SW version: 3		

26006 MD number	**NIBBLE_PUNCH_INMASK[n]** Screen form for high-speed input bits
Significance:	A total of 8 byte screen forms for the output of high-speed bits can be defined with this data.The standard assignment of this data is as follows: NIBBLE_PUNCH_INMASK[0]=1: 2^0 = first bit for the first punch interface (SPIF1) NIBBLE_PUNCH_INMASK[1]=4: Second punch interface (SPIF2), not available by default NIBBLE_PUNCH_INMASK[2]=0 ... NIBBLE_PUNCH_INMASK[7]=0 Note: The significance of the bit to be defined must be input (refer to MD 26004: NIBBLE_PUNCH_OUTMASK[n]).
Application	NIBBLE_PUNCH_INMASK[0] = 1 —> The first bit (bit 0) is defined NIBBLE_PUNCH_INMASK[0] = 4 —> The third bit (bit 2) is defined
Special cases, errors, ...	Only NIBBLE_PUNCH_INMASK[0] is relevant. This is used to define the input bit for the signal “Stroke active”.
Related to	PUNCHNIB_ASSIGN_FASTIN

26010 MD number	**PUNCHNIB_AXIS_MASK** Definition of punching and nibbling axes	
Default value: 7	Min. input limit: 0	Max. input limit: plus
Changes effective after Power On	Protection level: 2 / 7	Unit: –
Data type: DWORD	Applies from SW version: 3	
Significance:	This data is used to define which channel axes shall be treated as punching/nibbling axes. This data setting determines above all the response of synchronous axes in stroke control mode and with automatic path segmentation. When the axes are defined, the bit significance must be entered according to the following diagram: Bit: 7 6 5 4 3 2 1 0 Significance: 2^7=128 2^6=64 2^5=32 2^4=16 2^3=8 2^2=4 2^1=2 2^0=1 Example: If the first 2 axes are to be defined as the punching/nibbling axes, then “3” must be entered (corresponds to setting of bits 1 and 0). PUNCHNIB_AXIS_MASK = 3 The first 2 axes – typically x and y – are punching/nibbling axes. PUNCH_NIB_AXIS_MASK = 11 The first 2 axes (typically x and y) and the third axis (e.g. an axis for the rotatable tool) are punching/nibbling axes. In this case, the response of the 3rd axis with automatic path segmentation can be defined via PUNCH_PARTITION_TYPE. Independently of the above, this MD also defines whether the path of the A axis must be segmented by means of SPN=<value> if it is programmed without path axes. In addition, this data also allows single path axes to be treated differently in stroke control mode and with path segmentation. To illustrate this option, let us assume that we have a machine with path axes x, y and rotary axis A. However, we do not want the Z axis traversing paths to be segmented into individual sections nor do we want a stroke to be initiated at the end of a block which contains only a Z axis movement. It is possible to achieve the desired response by setting PUNCHNIB_AXIS_MASK = 11. In this case, pure Z movements are not taken into account by the path segmentation function.	
Related to	PUNCH_PARTITION_TYPE	

26012 MD number	**PUNCHNIB_ACTIVATION** Activation of punching and nibbling functions		
Default value: 1	Min. input limit: 0	Max. input limit:	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies from SW version: 3		
Significance:	This MD defines in what way punching and nibbling functions can be activated: PUNCHNIB_ACTIVATION = 0 None of the punching or nibbling functions can be activated. The automatic path segmentation is the only exception if it is enabled via MD: PUNCH_PATH_SPLITTING. PUNCHNIB_ACTIVATION = 1 The functions are activated via language commands. If M functions are to be used, then they must be programmed using macros. PUNCHNIB_ACTIVATION = 2 The M functions are interpreted directly by the software. Language commands can still be used. Note: This option is intended only as a temporary solution.		
Related to	PUNCH_PATH_SPLITTING NIBBLE_PUNCH_CODE[n]		

26014 MD number	**PUNCH_PATH_SPLITTING** Activation of automatic path segmentation		
Default value: 0	Min. input limit: 0	Max. input limit:	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies from SW version: 3		
Significance:	This machine data defines whether it should be possible to activate the automatic path segmentation function, even if punching-specific functionality is not available. PUNCH_PATH_SPLITTING = 0 Automatic path segmentation can only be activated if punching mode is active and is deactivated as soon as punching mode is deactivated. PUNCH_PATH_SPLITTING = 1 In this case, automatic path segmentation can be activated for geometry axes even when punching mode is not active.		

26016 MD number	**PUNCH_PARTITION_TYPE** Behavior of single axes with active automatic path segmentation		
Default value: 0	Min. input limit: 0	Max. input limit:	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies from SW version: 3		
Significance:	This machine data determines how single axes, which are also nibbling axes according to MD: PUNCHNIB_AXIS_MASK, should respond. It is assumed that rotary axis A is defined as the 4th and nibbling axis in addition to the 3 path axes x, y, z. In this case, there are the following options for defining its response with automatic path segmentation and in stroke control mode: PUNCH_PARTITION_TYPE = 0 No special response in the case of automatic path segmentation. If the single axis is programmed together with path axes in one block, then its total traversing path is segmented in accordance with the path axes, i.e. the purely geometric relationship between single axis and path axes is identical to that for non-segmented movements. If the single axis is programmed without path axes, but with SPN=<value>, then the path is segmented according to the programmed SPN value. PUNCH_PARTITION_TYPE = 1 In this case, the path of the single axis is generally (i.e. regardless of the currently active interpolation mode) traversed in the first segment if the axis is programmed together with path axes. PUNCH_PARTITION_TYPE = 2 In this case, the single axis responds to linear interpolation in the same way as described under PUNCH_PARTITION_TYPE = 1, and to all other interpolation modes as described under PUNCH_PARTITION_TYPE = 0.		
Related to	PUNCHNIB_AXIS_MASK		

26018 MD number	**NIBBLE_PRE_START_TIME** Automatic pre-start time		
Default value: 0	Min. input limit: 0	Max. input limit:	
Changes effective after Power On	Protection level:	Unit: –	
Data type:	Applies from SW version: 3.1		
Significance:	To minimize any dead times due to the reaction time of the punching unit, it is possible to release the stroke before reaching the inposition window of the axes. The reference time for this is the interpolation end. Since there is normally a delay of some interpolation cycles after reaching the interpolation end (depending on the machine dynamics) until the axes reach their true position, the prestart time is a delay time with respect to reaching the interpolation end. The function is therefore coupled to G603 (block change at the end of interpolation). The time can be set via the machine data NIBBLE_PRE_START_TIME. Example: With an interpolation cycle of 5 ms, a stroke shall be released 2 cycles after reaching the interpolation end. In this case, the value 0.010 s must be selected for NIBBLE_PRE_START_TIME. If a value that is not integrally divisible by the set interpolaton time is selected, then the stroke is initiated in the interpolation cycle following the set time.		
Related to			

26020 MD number	**NIBBLE_SIGNAL_CHECK** Monitoring of input signal		
Default value: 0	Min. input limit: 0	Max. input limit:	
Changes effective after Power On	Protection level:	Unit: s	
Data type: Flowld	Applies from SW version: 3.1		
Significance:	If for example the stroke active signal is set by punch overshoot between the cycles, the interpolation is stopped. Furthermore, it is possible to generate the “Distorted punch signal” message as a function of machine data NIBBLE_SIGNAL_CHECK.		
Related to			

26008 MD number	**NIBBLE_PUNCH_CODE**[**n**] Definition of M functions (applies only to SW 3.1)		
Default value: see below	Min. input limit: 0	Max. input limit: plus	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies for Software Version 3.1		
Significance:	This machine data defines the special M functions for punching and nibbling. Default setting / Example NIBBLE_PUNCH_CODE[0] = 0 — 20 — End punching, nibble with M20 NIBBLE_PUNCH_CODE[1] = 23 — 23 — End punching, nibble with M23 NIBBLE_PUNCH_CODE[2] = 22 — 22 — Start nibbling NIBBLE_PUNCH_CODE[3] = 25 — 25 — Start punching NIBBLE_PUNCH_CODE[4] = 26 — 26 — Activation of dwell NIBBLE_PUNCH_CODE[5] =122 — 122 — Start nibbling with leader, stroke control on servo level NIBBLE_PUNCH_CODE[6] =125 — 125 — Start punching with leader, stroke control on servo level NIBBLE_PUNCH_CODE[7] = 0 — 0 — Not used (available soon)		
Special cases, errors, ...	If MD: PUNCHNIB_ACTIVATION = 2 (M functions are interpreted directly by software), then MD: NIBBLE_PUNCH_CODE[0] = 20 must be set.		
Related to	PUNCHNIB_ACTIVATION		

4.2 Channel-specific setting data

42400 SD number	**PUNCH_DWELL_TIME** Dwell		
Default value: 1.0	Min. input limit: 0	Max. input limit: plus	
Changes effective immediately			Unit: s
Data type: DOUBLE		Applies from SW version: 3	
Significance:	This machine data is used to set the dwell between reaching the position and initiating the stroke movement. The set value is rounded to whole multiples of the interpolation clock cycle (i.e. the value set here may deviate slightly to the dwell actually applied).		
Related to	MD 10710:$MN_PROG_SD_RESET_SAVE_TAB		

42402 SD number	**NIBPUNCH_PRE_START_TIME** Pre-start time		
Default value: 1.0	Min. input limit: 0	Max. input limit: plus	
Changes effective immediately			Unit: s
Data type: DOUBLE		Applies from SW version: 3	
Significance:	This setting data has exactly the same effect as machine data NIBBLE_PRE_START_TIME. Its primary purpose is to allow the pre-start time to be altered from the NC program so that it can be adapted to different metal sheet sizes and thicknesses. However, setting data 42402 is effective only when the machine data has been set to zero.		
Related to	NIBBLE_PRESTART_TIME		

■

Notes

Signal Descriptions

5

5.1 Signal overview

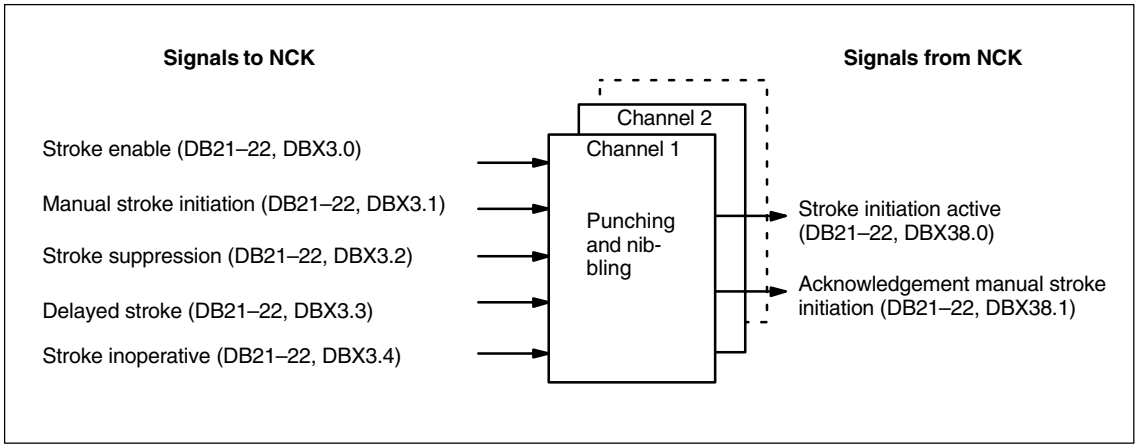


Fig. 5-1 PLC interface signals for “Punching and nibbling”

5.2 Signals to channel

DB 21, 22 **DBX3.0** Data block	**No stroke enable** Signal(s) to channel (PLC ——> NCK)	
Edge evaluation:	Signal(s) updated:	Signal(s) valid from SW vers.: 3
Signal state 1 or signal transition 0 ——> 1	This signal enables the punching strokes via the PLC. 1 signal: Stroke is disabled, the NC must not enable punching strokes	
Signal state 0 or signal transition 1 ——> 0	0 signal: Stroke enable is present, the NC may execute a punching stroke provided the enabling signal is not set.	

DB 21, 22 **DBX3.1** Data block	**Manual stroke initiation** Signal(s) to channel (PLC ——> NCK)	
Edge evaluation:	Signal(s) updated:	Signal(s) valid from SW vers.: 3
Signal state 1 or signal transition 0 ——> 1	This signal permits a single stroke to be initiated in manual mode. 1 signal: Manual stroke is executed.	
Signal state 0 or signal transition 1 ——> 0	0 signal: No effect	

DB 21, 22 **DBX3.2** Data block	**Stroke suppression** Signal(s) to channel (PLC ——> NCK)	
Edge evaluation:	Signal(s) updated:	Signal(s) valid from SW vers.: 3
Signal state 1 or signal transition 0 ——> 1	This signal simply prevents execution of the stroke. The machine continues to operate. The automatic path segmentation remains active if it is already activated. Only the signal “Stroke initiation” is suppressed. The machine traverses in “stop and go” mode. The step length is defined by the path segmentation. 1 signal: Stroke suppression is active	
Signal state 0 or signal transition 1 ——> 0	0 signal: Stroke suppression is not active	

DB 21, 22 **DBX3.3** Data block	**Delayed stroke** Signal(s) to channel (PLC ——> NCK)	
Edge evaluation:	Signal(s) updated:	Signal(s) valid from SW vers.: 3
Signal state 1 or signal transition 0 ——> 1	A “Delayed stroke” can be activated via this signal. This corresponds in function to the programming of PDELAYON. Other PLC signals not corresponding to the standard are not evaluated in the NCK. With the exception of the manual stroke initiation, the evaluation of signals is limited to PON active. 1 signal: Delayed stroke is active	
Signal state 0 or signal transition 1 ——> 0	0 signal: Delayed stroke is not active	

DB 21, 22 **DBX3.4** Data block	**Stroke inoperative** Signal(s) to channel (PLC ——> NCK)	
Edge evaluation:	Signal(s) updated:	Signal(s) valid from SW vers.: 3
Signal state 1 or signal transition 0 ——> 1	The NC reacts to this signal by initiating an immediate movement stop. An alarm is output if any other movement or action needs to be interrupted as a result of this signal. In physical terms, the signal is identical to the signal “Stroke active” for the CNC, i.e. the system is wired such that the two signals are taken to the same NC input via an AND gate. 1 signal: Stroke inoperative (corresponds to signal “Stroke enable”)	
Signal state 0 or signal transition 1 ——> 0	0 signal: Stroke operative (corresponds to signal “Stroke enable”)	

5.3 Signals from channel

DB21, 22 **DBX38.0** Data block	**Stroke initiation active** Signal(s) from channel (PLC ——> NCK)	
Edge evaluation:	Signal(s) updated:	Signal(s) valid from SW vers.: 3
Signal state 1 or signal transition 0 ——> 1	This signal indicates whether the stroke initiation is active. 1 signal: Stroke initiation is active	
Signal state 0 or signal transition 1 ——> 0	0 signal: Stroke initiation is not active	

DB21, 22 **DBX38.1** Data block	**Acknowledgement of manual stroke initiation** Signal(s) from channel (PLC ——> NCK)	
Edge evaluation:	Signal(s) updated:	Signal(s) valid from SW vers.: 3
Signal state 1 or signal transition 0 ——> 1	This signal indicates whether a manual stroke has been initiated. 1 signal: Manual stroke has been initiated	
Signal state 0 or signal transition 1 ——> 0	0 signal: Manual stroke has not been initiated	

■

Examples

6

Examples of defined start of nibbling operation

1st example:

```
:
:
N10   G0    X20   Y120  SPP= 20     Position 1 is approached
N20   X120  SON                     Defined start of nibbling,
                                    first stroke on “1”, last
                                    stroke on “2”
N30   Y20                           Defined start of nibbling,
                                    first stroke on “3”, last
                                    stroke on “4”
N40   X20                           Defined start of nibbling,
                                    first stroke on “5”, last
                                    stroke on “6”
N50         SPOF
N60   M2
```

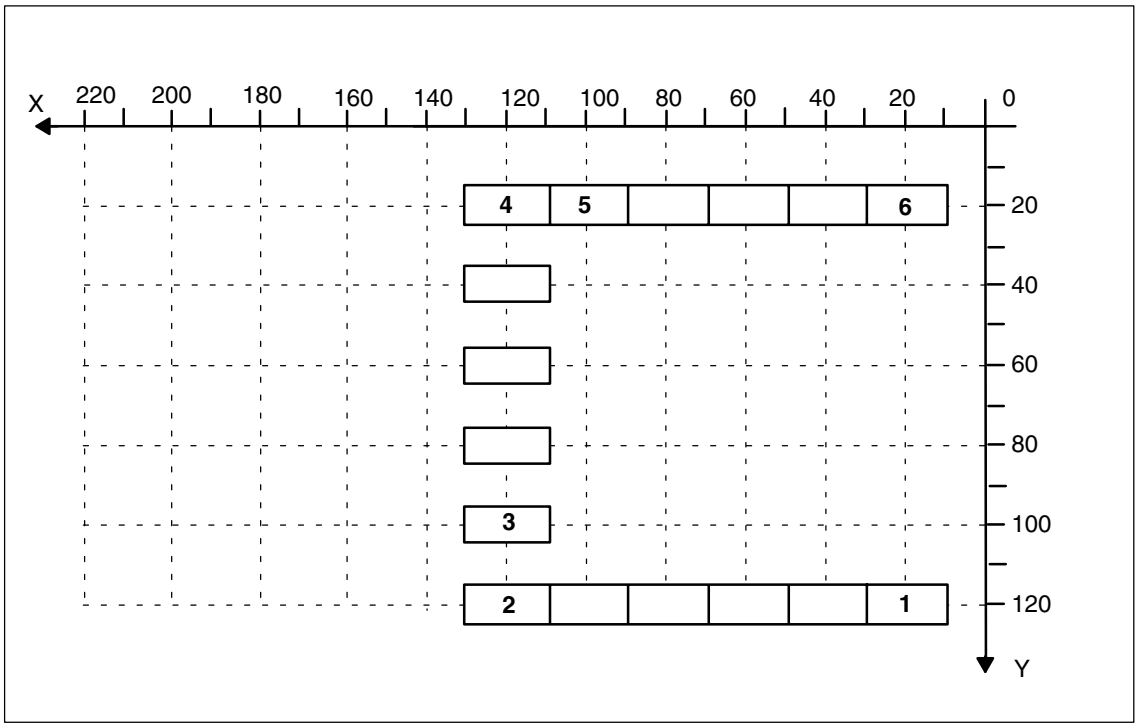


Fig. 6-1

2nd example:

This example utilizes the “Tangential control” function. Z has been selected as the name of the tangential axis.

```
:
:
N5    TANG  (Z, X, Y, 1, “B”)                Definition of tangential axis
N8    TANGON      (Z, 0)                     Selection of tangential control
N10   G0    X20   Y120                       Position 1 is approached
N20   X120  Z     SPN=20      SON            Defined start of nibbling,
                                             tangential control
                                             selected, first stroke on “1”,
                                             last stroke on “2”
N30   SPOF  TANGOF                           Deselection of nibbling mode
                                             and selection of tangential
                                             control
N38   TANGON      (Z, 0)                     Selection of tangential control
N40   Y20   Z     SON                        Defined start of nibbling,
                                             tangential control
                                             selected, first stroke on “2”,
                                             rotated through 90 degrees to
                                             block N20, last stroke on “3”
N50   SPOF
N60   M2
```

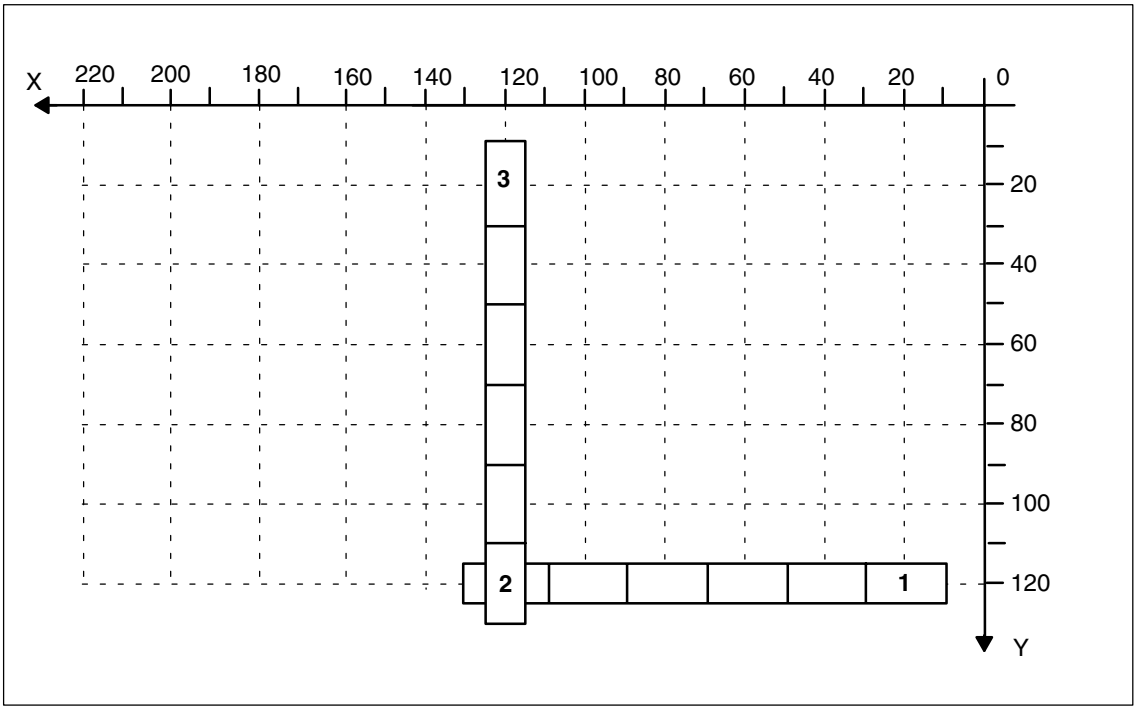


Fig. 6-2

(3) Example of **defined start of nibbling** illustrated in diagram below:

```
:
:
N5    G0    X10   Y10                    Position
N10   X90   SPN=20      SON              Defined start of nibbling,
                                         5 punching operations
N20   X10   Y30   SPP=1                  Initiation of punching
                                         operation at end of path
N30   X90                                4 punching operations with
                                         SPP distance = 20
N40   SPOF
N50   M2
```

(4) Example of **defined start of nibbling** illustrated in diagram below:

```
:
:
N5    G0    X10   Y10                    Position
N10   X90   SPN=4       SON              Defined start of nibbling,
                                         5 punching operations
N20   X10   Y30   PON                    Initiation of punching
                                         operation at end of path
N30   X90   SPN=4                        4 punching operations
N40   SPOF
N50   M2
```

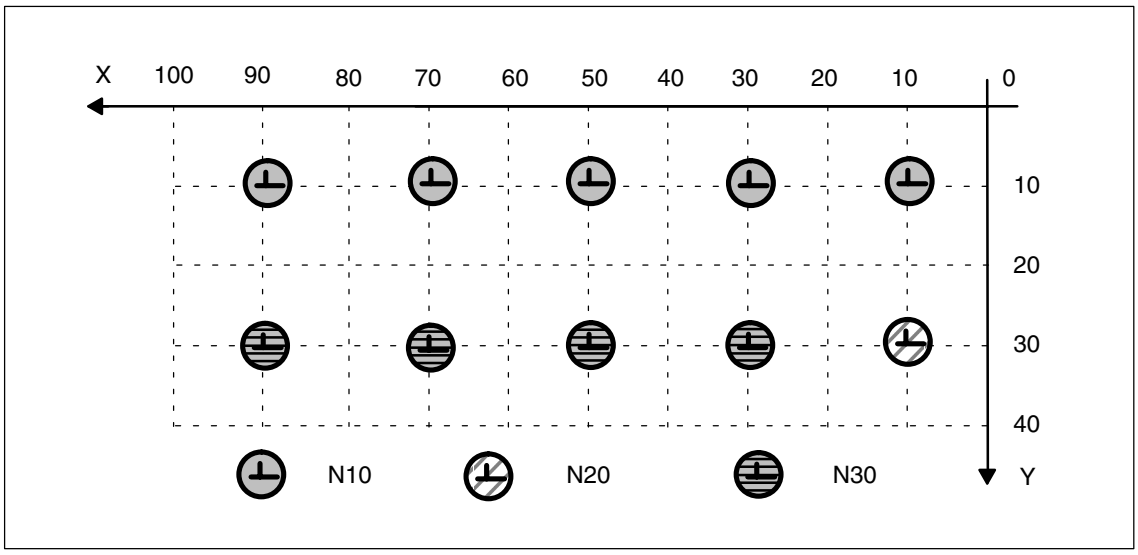


Fig. 6-3 Example of defined start of nibbling operation

(5) Examples of E programming **without defined start of nibbling** illustrated in diagram below:

```
:
:
N5    G0    X10   Y30                    Position
N10   X90   SPN=20      PON              No defined start of nibbling,
                                         4 punching operations
N15   Y10                                Initiation of punching
                                         operation at end of path
N20   X10   SPP=20                       4 punching operations with
                                         distance E20
N25   SPOF
N30   M2
```

(6) Examples of H programming **without defined start of nibbling** illustrated in diagram below:

```
:
:
N5    G0    X10   Y30                    Position
N10   X90   SPN=4       PON              No defined start of nibbling,
                                         4 punching operations
N15   Y10                                Initiation of punching
                                         operation at end of path
N20   X10   SPN=4                        4 punching operations
N25   SPOF
N30   M2
```

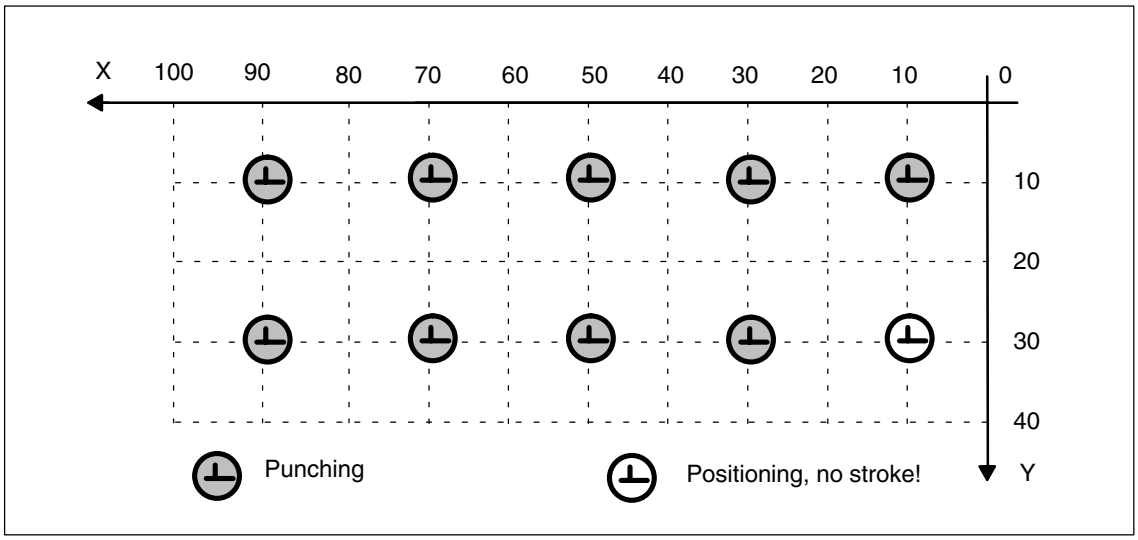


Fig. 6-4 Examples of E/H programming without defined start of nibbling

Example of application: E value for punching

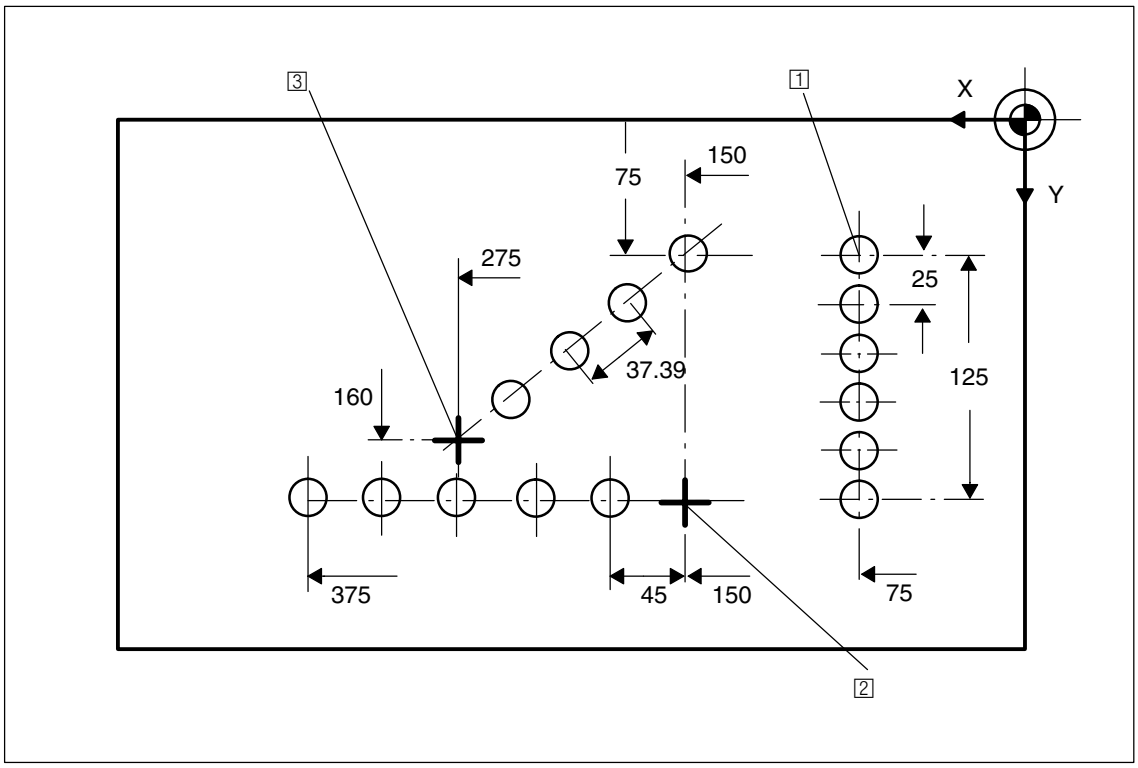


Fig. 6-5 Workpiece

Program extract:

N100	G90	X75	Y75	F60	PON	Position at starting point 1 of vertical row of holes, punch single hole
N110	G91	Y125	SPP=25	PON		End point dimensions (incremental), feedrate value: 25 mm, activate punching
N120	G90	X150	SPOF			Absolute dimensioning, position at starting point 2 of horizontal row of holes
N130	X375	SPP=45	PON			End point dimensions, feedrate value: 45 mm
N140	X275	Y160	SPOF			Position at starting point 3 of inclined row of holes
N150	X150	Y75	SPP=40	PON		End point dimensions, programmed feedrate value: 40 mm, calculated feedrate value: 37.79 mm
N160	G00	Y300	SPOF			Position

■

Notes

Data Fields, Lists 7

7.1 Interface signals

DB number	Bit, byte	Name	Reference
Channel-specific (signals to channel)			
21–22	3.0	No stroke enable	
21–22	3.1	Manual stroke initiation	
21–22	3.2	Stroke suppression	
21–22	3.3	Stroke delay	
21–22	3.4	Stroke inoperative	
Channel-specific (signals from channel)			
21–22	38.0	Stroke initiation active	
21–22	38.1	Acknowledgment of manual stroke initiation	

7.2 Machine data

Number	Identifier	Name	Reference
Channel-specific ($MC_ ...)			
20150	GCODE_RESET_VALUES[n]	Initial setting of G groups	/K1/
26000	PUNCHNIB_ASSIGN_FASTIN	Hardware assignment for input byte with stroke control	
26002	PUNCHNIB_ASSIGN_FASTOUT	Hardware assignment for output byte with stroke control	
26004	NIBBLE_PUNCH_OUTMASK[n]	Screen form for high-speed output bits	
26006	NIBBLE_PUNCH_INMASK[n]	Screen form for high-speed input bits	
26008	NIBBLE_PUNCH_CODE[n]	Definition of M functions (applies only to SW 3.1)	
26010	PUNCHNIB_AXIS_MASK	Definition of punching and nibbling axes	
26012	PUNCHNIB_ACTIVATION	Activation of punching and nibbling functions	
26014	PUNCH_PATH_SPLITTING	Activation of automatic path segmentation	

Number	Identifier	Name	Reference
26016	PUNCH_PARTITION_TYPE	Response of single axes with automatic path segmentation	
26018	NIBBLE_PRE_START_TIME	Automatic activation of the pre-start time	
26020	NIBBLE_SIGNAL_CHECK	Monitoring of the input signal	

7.3 Setting data

Number	Identifier	Name	Reference
Channel-specific ($SC_ ...)			
42400	PUNCH_DWELL_TIME	Dwell	
42402	NIBPUNCH_PRE_START_TIME	Pre-start time	

7.4 Language commands

G group	Language command	Meaning	
35	SPOF	Stroke / Punch Off	Punching and nibbling OFF
35	SON	Stroke On	Nibbling ON
35	SONS	Stroke On	Nibbling ON (position controller)
35	PON	Punch On	Punching ON
35	PONS	Punch On	Punching ON (position controller)
36	PDELAYON	Punch with Delay On	Punching with delay ON
36	PDELAYOF	Punch with Delay Off	Punching with delay OFF
Path segmentation			
	SPP		Path per stroke, modal action
	SPN		Number of strokes per block, non-modal action

7.5 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

■

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

Positioning Axes (P2)

■

Notes

Brief Description 1

In addition to axes for machining, modern machine tools can also be equipped with axes for auxiliary movements, e.g.:

- Axis for tool magazine
- Axis for tool turret
- Axis for workpiece transport
- Axis for pallet transport
- Axis for loader (also multi-axis)
- Axis for tool changer
- Axis for quill/sleeve or steady

Positioning axes

The “Positioning axes” function allows axes for auxiliary motions to be integrated more easily into the control system.
The integration of the positioning axes is simpler

- During programming:
  Programming is performed together with the axes for workpiece machining in the same part program, without having to sacrifice valuable machining time.
- During program testing/startup:
  Program testing and startup is performed simultaneously for all axes.
- During operation:
  Operation and monitoring of the machining process commence simultaneously for all axes.
- During PLC configuring/startup:
  No allowance has to be made on PLC or external computers (PCs) for synchronization between axes for machining and axes for auxiliary movements.
- During system configuration:
  A second channel is not required.

■

Notes

Detailed Description 2

In addition to the axes required for machining, a complex modern machine tool can be equipped with further axes for auxiliary movements. The axes for machining a workpiece are known as path axes: within the channel they are guided by the interpolator so that they start simultaneously and accelerate and reach the end point together.
The auxiliary axes include:

- Axis for tool magazine
- Axis for workpiece transport
- Axis for pallet transport
- Axis for loader
- Axis for tool changer
- Axis for quill/sleeve or steady

Many of these axes were previously manipulated hydraulically and triggered by the part program by means of an auxiliary function.
With control of the axis in the NC, the axis can be addressed by name in the part program and the actual position displayed on the screen.

Positioning axes are traversed independently of the path axes with their own dedicated axis-specific feedrate.
Synchronous axes and geometry axes can be traversed non-modally as positioning axes.
Special traversing instructions are provided for positioning axes, i.e.
POS[...], POSA[...]

2.1 Selection of positioning axes

When axes are provided for auxiliary movements on a machine tool, the required properties will decide whether the axis is to be

- programmed in a separate part program
  —> see section 2.1.1 “Separate channels”
- programmed in the same part program as the machining process
  —> see section 2.1.2 “Positioning axes”
- triggered exclusively from the PLC during machining
  —> see section 2.1.3 “Concurrent positioning axes”

2.1.1 Separate channel

A channel represents a self-contained NC which, with the aid of a part program, can be used to control the movement of axes, spindles and machine functions independently of other channels.

Independence between channels is assured by means of the following provisions:

- One active part program per channel
- Channel-specific signals such as
  - IS “NC Start” (DB21, ... DBX7.1)
  - IS “NC Stop” (DB21, ... DBX7.3)
  - IS “Reset” (DB21, ... DBX7.7)
- One feedrate override per channel
- One rapid traverse override per channel
- Channel-specific evaluation and display of alarms
- Channel-specific display, e.g. for:
  - Actual axis positions
  - Active G functions
  - Active help functions
  - Current program block
- Channel-specific testing and channel-specific modification of programs:
  - Single block
  - Dry run (DRY RUN)
  - Block search
  - Program test

Please see the following for more information on channel functionality:
References: /FB/, K1, “Mode Groups, Channels, Program Operation Mode”

2.1.2 Positioning axis

Positioning axes are programmed together with path axes, i.e. with the axes that are responsible for workpiece machining.
Instructions for both positioning axes and path axes can be included in the same NC block. Although they are programmed in the same NC block, the path and positioning axes are not interpolated together and do not reach their end point simultaneously (no direct time relationship, see also section 2.2).
The block change depends on the type of positioning axis programmed (see section 2.3).

- Type 1 The block change occurs when all path and positioning axes have reached their programmed end point.
- Type 2 The block change occurs when all path axes have reached their programmed end point.

With positioning axes of type 2, it is possible to approach the programmed end position across several block boundaries.

Positioning axes permit movements to be activated from the same machining program and such movements to be synchronized at block limits (type 1) or at explicit points by means of a WAITP command (type 2).

Independence of path and positioning axes

The mutual independence of path and positioning axes is ensured by the following measures:

- No shared interpolation
- Each positioning axis has a dedicated axis interpolator
- Dedicated feed override for each positioning axis
- Dedicated programmable feedrate
- Dedicated “axis-specific delete distance-to-go” interface signal.

Dependence of positioning axes

Positioning axes are dependent in the following respects:

- A shared part program
- Starting of positioning axes only at block boundaries in the part program
- No rapid traverse override
- The signals
  - IS “NC Start” (DB21, ... DBX7.1)
  - IS “NC Stop” (DB21, ... DBX7.3)
  - IS “Reset” (DB21, ... DBX7.7)
  - IS “Read-in disable” (DB21, ... DBX6.1)

  act on the entire channel and therefore on positioning axes.
- Program-specific and channel-specific alarms also deactivate positioning axes.
- Program control (dry run feed, program test, DRF, ... etc.) also act on positioning axes
- Block search and single block also act on positioning axes
- Group 1 (movement commands with modal effect) of the G functions (G0, G1, G2, etc.) does not apply to positioning axes.
  References: /PA/, Programming Guide

Applications

The following are typical applications for positioning axes:

- Single-axis loaders
- Multi-axis loaders without interpolation (PTP —> point-to-point traversing)
- Workpiece feed and transport

Positioning axes are not suitable for multi-axis loaders that require interpolation between the axes (path interpolator).

2.1.3 Concurrent positioning axis

Concurrent positioning axes are positioning axes with the following attributes:

- Activation from the PLC need not take place at block limits, but can be implemented at any time in any operating mode (even when a part program is already being processed in the channel).
- Program command “WAITP” is required to move a concurrent positioning axis from the part program immediately after power ON.
- The part program continues to run uninhibited, even if the concurrent positioning axis has not reached the position defined by the PLC.
- Software Version 4.3 and higher
  Depending on the machine data AUTO_GET_TYPE, using the programming command
  - “GET(<axis>)” or “WAITP(<axis>)” from the part program it is possible to traverse a concurrent positioning axis again as a channel axis, or using
  - “RELEASE (<axis>)” or “WAITP(<axis>)” to traverse a channel axis as concurrent positioning axis.

Activation from PLC

The concurrent positioning axis is activated via FC 15 or FC 16 from the PLC.

- Feedrate (with feedrate setting of 0, the feed set in MD 32060: POS_AX_VELO (reset for positioning axis velocity) is applied.
- Absolute coordinates (G90), incremental coordinates (G91), absolute coordinates along the shortest path for rotary axes (rotary axis name = DC(value))

The following functions are defined:

- Linear interpolation (G01)
- Feedrate in mm/min or degrees/min (G94)
- Exact stop (G09)
- Settable zero offsets currently selected are valid

Applications

Typical applications for concurrent positioning axes include:

- Tool magazines with manual loading and unloading during machining
- Tool magazines with tool preparation during machining

2.2 Motional behavior

Path interpolator

Every channel has a path interpolator for a wide range of interpolation modes such as linear interpolation (G01), circular interpolation (G02/G03), spline interpolation, etc.

Axis interpolator

In addition to the path interpolator, each channel also has 5 axis interpolators. When a positioning axis is programmed, an axis interpolator is started in the control (with linear interpolation G01).

This axis interpolator runs independently of the path interpolator until the programmed end position of the positioning axis has been reached.

There is no time relationship between the path interpolator and the axis interpolator, nor between the axis interpolators.

The programmed end position of a positioning axis has been reached when the axis reaches the exact stop fine window (G09). Continuous-path mode (G64) is not possible with positioning axes.

2.3 Block change

Positioning axes can be programmed in the NC block individually or in combination with contour axes.

Contour axes and positioning axes are always interpolated separately (path interpolator and axis interpolators) and this causes them to reach their programmed end positions at different times.

There are two types of positioning axis, whose response differs with respect to block change.

- Type 1 The block change occurs when all path and positioning axes have reached their programmed end point.
- Type 2 The block change occurs when all path axes have reached their programmed end point.

Positioning axis type 1

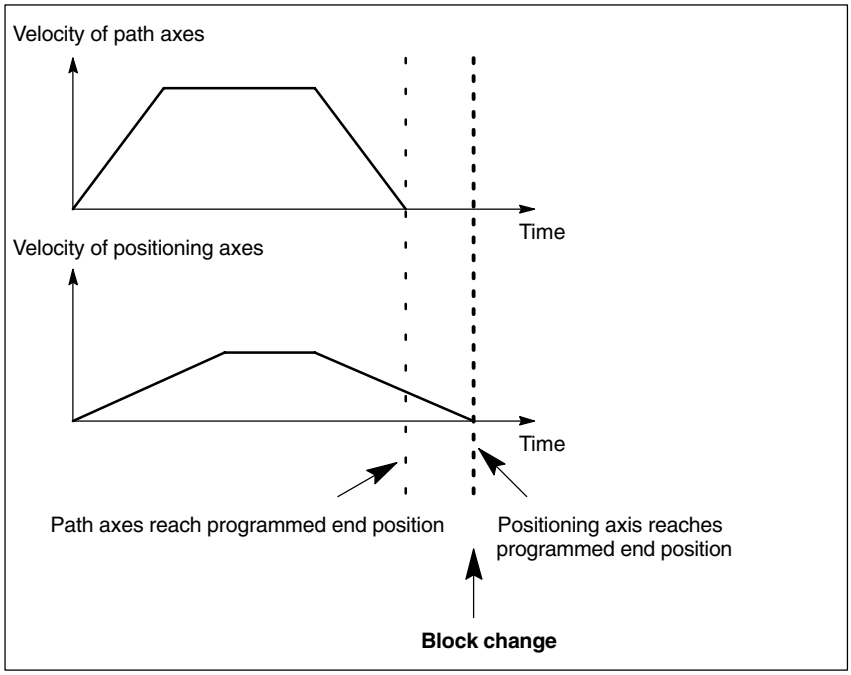


Fig. 2-1 Example of block change for positioning axis type 1

Characteristics of positioning axis type 1

- The block change occurs (NC block finished) when all the path and positioning axes have reached their end positions.
- Continuous-path mode (G64) is only possible for path axes if the positioning axes reach their end positions ahead of the path axes (this is not the case in the example in Fig. 2-1).
- Programming with
  POS[name] = end point FA[name] = feed
  or abbreviated with
  POSA[name] = end point
  in which case the feed set in MD 32060: POS_AX_VELO is applied.
- The programmed instruction is effective on a non-modal basis. The geometry and synchronous axes are separated from the combined path axis and traversed at an axis-specific velocity.

Positioning axis type 2

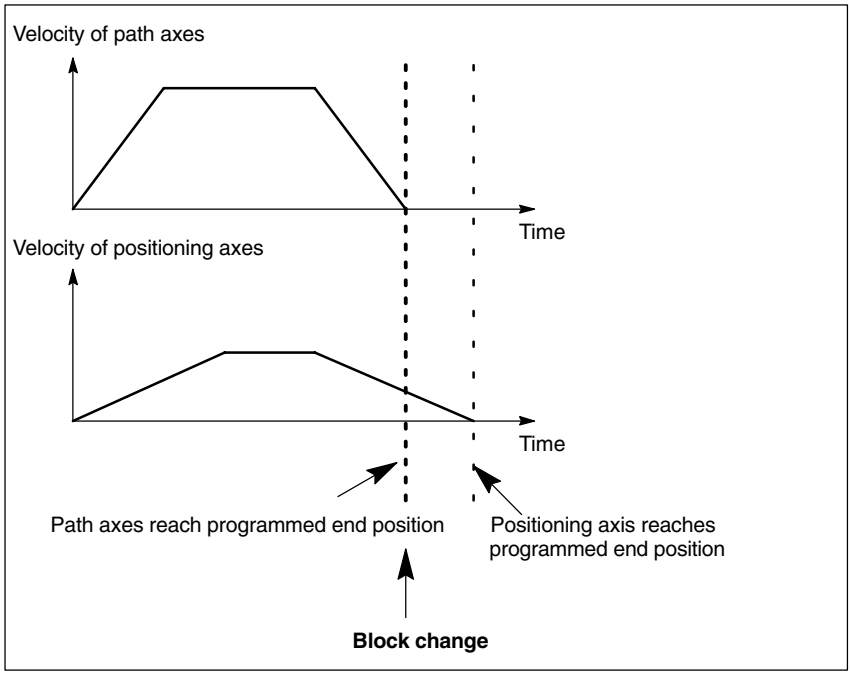


Fig. 2-2 Block change with positioning axis type 2, example of sequence

Characteristics of positioning axis type 2

- The block change occurs (NC block finished) when the path axes have reached their programmed end positions.
- The positioning axes can traverse across several block boundaries to their programmed end positions.
- Since their is no time relationship between “NC block finished” and the point at which type 2 positioning axes reach their programmed end positions, the WAITP coordination command is provided for the synchronization of the positioning axes (see Section 2.6).
- If a positioning axis is reprogrammed before the previous position has been reached, the “axis cannot be repositioned” alarm is issued.
- Programming with
  POSA[name] = end point FA[name] = feed
  or abbreviated with
  POSA[name] = end point
  in which case the feed set in MD 32060: POS_AX_VELO is applied.
- The programmed instruction is effective on a non-modal basis. The geometry and synchronous axes are separated from the combined path axis and traversed at an axis-specific velocity.

2.4 Velocity

The axis-specific velocity limits and acceleration limits are valid for positioning axes.

Positioning axes can be linear axes and rotary axes.

Positioning axes can also be indexing axes, see:
References: /FB/, T1, “Indexing Axes”

Feedrate override

The path and positioning axes have separate feedrate overrides. Each positioning axis can be adjusted by its own axis-specific feed override.

Rapid traverse override

Rapid traverse override applies only to path axes. Positioning axes have no rapid traverse interpolation (only axial linear interpolation G01) and therefore no rapid traverse override.

Feedrate

The positioning axes traverse at the axis-specific feedrate programmed for them. As illustrated in Section 2.2, the feedrate is not influenced by the path axes.

The feedrate is programmed as an axis-specific velocity in units of min/mm, inch/min or degrees/min.

The axis-specific feedrate is always permanently assigned to a positioning axis by the axis name.

If a positioning axis has no programmed feedrate, the control system automatically applies the rate set in axis-specificMD 32060: POS_AX_VELO (initial setting for positioning axis velocity).

The programmed axis-specific feedrate is active until the end of the program.

Revolutional feedrate

In JOG mode, the response of the axis/spindle is also dependent on the setting in SD 41100: JOG_REV_IS_ACTIVE (revolutional feedrate for active JOG).

- If this setting data is active, an axis/spindle is always traversed at revolutional feedrate MD 32050: JOG_REV_VELO (revolutional feedrate for JOG) or MD 32040: JOG_REV_VELO_RAPID (revolutional feedrate for JOG with rapid traverse override) as a function of the master spindle.
- If the setting data is not active, then the axis/spindle responds as a function of the setting in SD 43300: ASSIG_FEED_PER_REV_SOURCE (revolutional feedrate for positioning axes/spindles).
- If the setting data is not active, then a geometry axis influenced by a frame with rotation responds as a function of on channel-specific setting data SD 42600: JOG_FEED_PER_REV_SOURCE. (In JOG mode, revolutional feedrate for geometry axes on which a frame with rotation is effective).

2.5 Control by PLC

Channels-specific signals

All channel-specific signals act to the same extent on path and positioning axes.

The following signals are the only exceptions (see Chapter 5):

- IS “Feedrate override” (DB21, ... DBB4)
- IS “Delete distance-to-go” (DB21, ... DBX6.2)

Axis-specific signals

The following additional signals are available for positioning axes (see Section 5):

- IS “Positioning axis” (DB31, ... DBX76.5)
- F function (feed) for FA positioning axes
- Axis-specific feed override
- IS “Delete distance-to-go” (DB31, ... DBX2.2), axis-specific

Parameters for FC15

When concurrent positioning axes are activated from the PLC, FC15 is called and supplied with the following parameter data:

- Axis name/axis number
- Approach
- Feedrate (with feedrate setting = 0, the feed set in MD 32060: POS_AX_VELO) is applied.
- Absolute coordinates (G90), incremental coordinates (G91), absolute coordinates along the shortest path for rotary axes (rotary axis name = DC(value))

The following functions are defined:

- Linear interpolation (G01)
- Feedrate in mm/min or degrees/min (G94)
- Exact stop (G09)
- Settable zero offsets currently selected are valid

2.6 Programming

Note

The following documentation should be consulted before programming of positioning axes is attempted:
References: /PA/, Programming Guide

The maximum number of positioning axes that can be programmed in a block is limited to five.

Definition

Positioning axes are defined using the following parameters:

- Axis type: positioning axis type 1 or 2
- End point dimensions
- Absolute or incremental dimension for the end position coordinates
- Feedrate for linear axes in [mm/min], for rotary axes in [degrees/min]

Syntax

Positioning axis type 1:

POS[Q1]=200 FA[Q1]=1000; axis Q1 with feedrate 1000 mm/min to position 200

Positioning axis type 2:

POSA[Q2]=300 FA[Q2]=1500; axis Q2 with feedrate 1500 mm/min to position 300

Within a part program, an axis can be a path axis or a positioning axis. Within a movement block, however, each axis must be assigned a unique axis type.

Absolute/ incremental dimensions

The end position coordinates are programmed as absolute dimensions (G90) or incremental dimensions (G91):

Absolute dimensions

G90 POS[Q1]=200
G91 POS[Q1]=AC(200)

Incremental dimensions

G91 POS[Q1]=200
G90 POS[Q1]=IC(200)

Coordination

In the case of positioning axes of type 2 (movement beyond block limits), it must be possible to determine in the part program whether the axis has reached its end position. Only then is it possible to reprogram this positioning axis (otherwise an alarm is issued).

The WAITP coordination command is used to query in the part program whether the end position has been reached.

The coordination command WAITP is programmed in a separate block.

An explicit reference must be made to any axis for which the program is to wait.

Example program:

```
N10 G01 G90 X200 F1000  POSA[Q1]=200 FA[Q1]=500
N15 X400
N20 WAITP(Q1);        Execution of the program stops automatically until Q1 in position
N25 X600 POS[Q1]=300; Q1 is positioning axis type 1 (feed FA[Q1] from block N10)
N30 X800 Q1=500; Q1 is path axis (path feed F1000 from block N10)
```

Tool offset

A tool length compensation for positioning axes can be implemented by means of an axial zero offset, allowing, for example, the positioning path of a loader to be altered. An example where the axial zero offset might be used in place of the tool length compensation is where a loader containing tools of various dimensions has to bypass an obstacle.

Program end

The program end (program status selected) is delayed until all axes (path axes + positioning axes) have reached their programmed end points.

2.6.1 Programming from external

Traversal at an externally supplied revolutional feedrate can be selected via axial data SD 43300: ASSIGN_FEED_PER_REV_SOURCE, (revolutional feedrate for axes) and channel-specific setting data SD 42600: JOG_FEED_PER_REV_SOURCE in JOG mode. The following settings are possible via the setting data:

- >0: The machine axis number of the rotary axis/spindle from which the revolutional feedrate shall be derived
- –1: The revolutional feedrate is derived from the master spindle of the channel in which the axis/spindle is active in each case
- 0: The function is deselected

2.7 Response with special functions

2.7.1 Dry run feedrate (DRY RUN)

The dry run feedrate is also effective for positioning axes unless the programmed feedrate is larger than the dry run feedrate.

2.7.2 Single block

Positioning axis type 1

Single-block mode is effective with positioning axes of type 1.

Positioning axis type 2

Positioning axes of type 2 continue to operate beyond block limits even in single-block mode.

■

Notes

Supplementary Conditions 3

There are no supplementary conditions stipulated for this Description of Functions.

Data Descriptions (MD, SD) 4

4.1 Axis/spindle-specific machine data

30450 MD number	**IS_CONCURRENT_POS_AX** Default on RESET: Neutral axis or channel axis		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after RESET	Protection level: 2	Unit: None	
Data type: BOOLEAN	Applies from SW version: 1.1		
Significance:	The axis is a concurrent positioning axis. SW 4.3 and higher (not FM-NC): If FALSE: A neutral axis becomes a channel axis again on RESET. If TRUE: A neutral axis remains in the neutral axis state on RESET and a channel axis becomes a neutral axis		

32060 MD number	**POS_AX_VELO** Initial setting for positioning axis velocity		
Default value: 0	Min. input limit: 0	Max. input limit:	
Changes effective after Power On	Protection level: 2	Unit: mm/min, rpm	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	Where a positioning axis is programmed in the part program without specifying the axis-specific feedrate, the feedrate entered in MD: POS_AX_VELO is automatically used. The feedrate from MD: POS_AX_VELO applies until an axis-specific feedrate is programmed in the part program for this positioning axis.		
MD irrelevant for	POS_AX_VELO is irrelevant as a positioning axis for all other axis types.		
Special cases, errors, ...	If a zero velocity setting is entered in POS_AX_VELO, the positioning axis does not traverse if it is programmed without feed. If a velocity setting is entered in POS_AX_VELO that is higher than the maximum velocity of the axis (MD 32000: MAX_AX_VELO), the velocity is automatically restricted to the maximum rate.		

Notes

Signal Descriptions 5

The following signals or commands on the NCK/MMC/PLC interface are only of significance for the positioning axis:

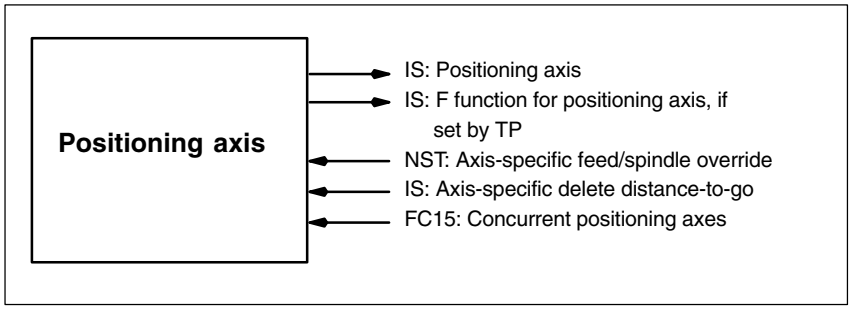


Fig. 5-1 Signal modification by the PLC

5.1 Axis/spindle-specific signals

DB31, ... **DBB0** Data block	**Feedrate override/spindle speed override, axis-specific** Signal(s) from axis/spindle (NCK ——> PLC)		
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1	
Signal state 1 or signal transition 0 ——> 1	Positioning axes have a dedicated axis-specific feed override. The evaluation of the axis-specific feed override is identical to the channel-specific feed override.		
Signal irrelevant for.....	IS “Positioning axis” (DB31, ... DBX74.5) = ZERO		
References	For evaluation, see IS “Feedrate override” (DB21, ... DBB4), channel-specific		

DB31, ... **DBX2.2** Data block	**Delete distance-to-go, axis-specific** Signal(s) from axis/spindle (NCK ——> PLC)		
Edge evaluation: yes	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1	
Signal state 1 or signal transition 0 ——> 1	The axis-specific distance-to-go of the positioning axis is cancelled. The positioning axis is decelerated and the following error is eliminated. The programmed end position is deemed to have been reached. The path axes are not influenced by the axis-specific “delete distance-to-go” interface signal. The channel-specific “delete distance-to-go” interface signal is used for this purpose.		
Special cases, errors, ...	If the axis-specific “delete distance-to-go” interface signal is enabled, even if no positioning axes have been programmed in this block, the NCK does not respond.		
Related to	IS “Delete distance-to-go” (DB21, ... DBX6.2), channel-specific, for path axes		

DB31, ... **DBX76.5** Data block	**Positioning axis** Signal(s) from axis/spindle (NCK ——> PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The NCK treats the axis as a positioning axis. It therefore has: a dedicated axis interpolator (linear interpolator) a dedicated feedrate (F value) a dedicated feed override exact stop (G09) at the programmed end position	

DB31, ... **DBB78–81** Data block	**F function (feedrate) for positioning axis** Signal(s) from axis/spindle (NCK ——> PLC)	
Edge evaluation: no	Signal(s) updated: on change	Signal(s) valid from SW vers.: 1.1
Signal state 1 or signal transition 0 ——> 1	The programmed axial feed is assigned to a positioning axis by means of the programmed axis name and output to the PLC for this axis. There is no output of the value preset via FC15. See below	
Signal irrelevant for.....	IS: positioning axis = ZERO	
Special cases, errors, ...	If the positioning axis is traversed at the feedrate set in MD 32060: POS_AX_VELO (initial setting for positioning axis velocity), the NC does not output an F function (feed) to the PLC.	
Related to	IS “Positioning axis” (DB31, ... DBX74.5) MD 22240: AUXFU_F_SYNC_TYPE Output time of F functions	

5.2 Function call

FC15

PLC function call FC15 can be used to start concurrent positioning axes from the PLC. The following parameters are passed to the function call:

- Axis name/axis number
- Approach
- Feedrate (with feedrate setting = 0, the feed set in MD 32060: POS_AX_VELO) is applied.
  The F value of FC15 is **not** transferred to the axis-specific IS “F function (feedrate) for positioning axis” DB31, ...DBB78-81.
- Absolute coordinates (G90), incremental coordinates (G91), absolute coordinates along the shortest path for rotary axes (rotary axis name = DC(value))

Since each axis is assigned to exactly one channel, the control can select the correct channel from the axis name/axis number and start the concurrent positioning axis on this channel.

References: /FB/, P3, “Basic PLC Program”

■

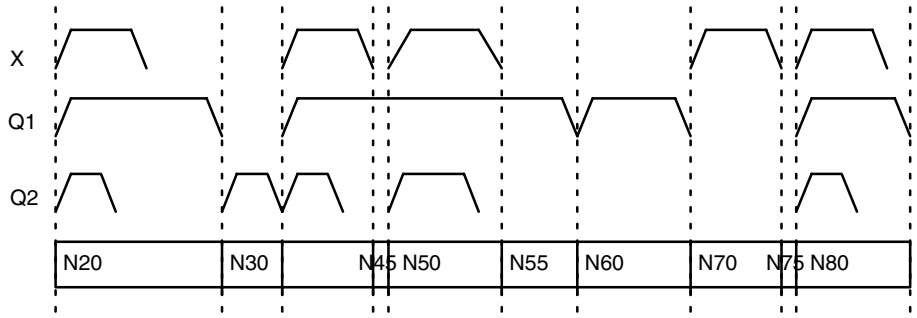
Example

6

In the following example, the two positioning axes Q1 and Q2 represent two separate units of movement. There is no interpolation relationship between the two axes. In the example, the positioning axes are programmed as type 1 (e.g. in N20) and type 2 (e.g. in N40).

Program example

```
N10 G90 G01 G40 T0 D0 M3 S1000
N20    X100 F1000        POS[Q1]=200 POS[Q2]=50 FA[Q1]=500
                         FA[Q2]=2000
N30                      POS[Q2]=80
N40    X200              POSA[Q1] = 300 POSA[Q2]=200] FA[Q1]=1500
N45 WAITP[Q2]
N50    X300              POSA[Q2]=300
N55 WAITP[Q1]
N60                      POS[Q1]=350
N70    X400
N75 WAITP[Q1, Q2]
N80 G91 X100             POS[Q1]=150 POS[Q2]=80
N90 M30
```



X .... path axes
Q1, Q2 .... positioning axes

Fig. 6-1 Timing of path axes and positioning axes

■

Notes

Data Fields, Lists

7

7.1 Interface signals

DB number	Bit, byte	Name	Reference
Axis/spindle-specific			
31–48	0	Axis-specific feed override	
31–48	2.2	Axis-specific delete distance-to-go	
31–48	74.5	Positioning axis	
31–48	78–81	F function (feed) for positioning axis	

7.2 Machine data

Number	Identifier	Name	Reference
Channel-specific ($MC_ ...)			
22240	AUXFU_F_SYNC_TYPE	Output time of F functions	H2
Axis-specific ($MC_ ...)			
30450	IS_CONCURRENT_POS_AX	Concurrent positioning axis	
32060	POS_AX_VELO	Feedrate for positioning axis	

7.3 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

■

Notes

SINUMERIK 840D/810D/FM-NC Description of Functions, Extension Functions (FB2)

Oscillation (P5)

■

Notes

Brief Description 1

Definition

When the “Oscillation” function is selected, an oscillation axis oscillates backwards and forwards at the programmed feedrate or a derived feedrate (revolutional feedrate) between two reversal points. Several oscillation axes can be active at the same time.

Oscillation variants

Oscillation functions can be classified according to the axis response at reversal points and with respect to infeed:

- Asynchronous oscillation beyond block limits.
  While the oscillation movement is in progress, other axes can interpolate freely. The oscillation axis can act as the input axis for dynamic transformation or as the master axis for gantry or coupled-motion axes. Oscillation is not automatically linked to the AUTOMATIC mode.
- Oscillation with continuous infeed.
  Infeed can be executed simultaneously in several axes. However, there is no interpolative connection between the infeed and oscillation movements.
- Oscillation with infeed in both reversal points or only in the left-hand or right-hand reversal point. The infeed can be initiated at a programmable distance from the reversal point.
- Sparking-out strokes after oscillation.

Response at reversal points

The change in direction is initiated:

– without the exact stop limit being reached (exact stop fine or coarse)

– after the programmed position is reached or

– after the programmed position is reached and expiry of a dwell.

Control methods

Oscillation movements can be controlled by various methods:

- The oscillation movement and/or infeed can be interrupted by delete distance-to-go.
- The reversal points can be altered via NC program, PLC, MMC, handwheel or directional keys.
- The feedrate velocity of the oscillation axis can be altered through a value input in the NC program, PLC, MMC or via an override. The feedrate can be programmed to be dependent on a master spindle, rotary axis or spindle (revolutional feedrate).

References: /FB/; V1, “Feedrates”

- The oscillation movement can be controlled entirely by the PLC.

■

Detailed Description 2

Methods of oscillation control

There are two modes of oscillation:

1. Asynchronous oscillation
   which is active beyond block limits and can also be started from PLC/MMC,

   and

2. Oscillation as controlled by motion-synchronous actions.
   In this case, asynchronous oscillation and an infeed movement are coupled with one another via synchronized actions. In this way, it is possible to program oscillation with infeed at the reversal points which is active on a non-modal basis.

2.1 Asynchronous oscillation

Characteristics

The characteristics of asynchronous oscillation are as follows:

- The oscillation axis oscillates backwards and forwards between reversal points at the specified feedrate until the oscillation movement is deactivated or until there is an appropriate response to a supplementary condition. If the oscillation axis is not positioned at reversal point 1 when the movement is started, then it traverses to this point first.
- Linear interpolation G01 is active for the oscillation axis regardless of the G code currently valid in the program. Alternately, revolutional feedrate G95 can be activated.
- Asynchronous oscillation is active on an axis-specific basis beyond block limits.
- Several oscillation axes (i.e. maximum number of positioning axes) can be active at the same time.
- During the oscillation movement, axes other than the oscillation axis can be freely interpolated. A continuous infeed can be achieved via a path movement or with a positioning axis. In this case, however, there is no interpolative connection between the oscillation and infeed movements.
- If the PLC does not have control over the axis, then the axis is treated like a normal positioning axis during asynchronous oscillation. In the case of PLC control, the PLC program must ensure via the appropriate stop bits of the VDI interface that the axis reacts in the desired way to VDI signals. These signals include program end, operating mode changeover and single block.
- The oscillation axis can act as the input axis for the transformations (e.g. inclined axis).

References: /FB/, M1, “Transmit/Peripheral Surface Transformation”

- The oscillation axis can act as the master axis for gantry and coupled motion axes.

References: /FB/, G1, “Gantry Axis”

- It is possible to traverse the axis with jerk limitation (SOFT) and/or with knee-shaped acceleration characteristic (as for positioning axes).
- Block-synchronous activation of the oscillation movement is assured via the part program.
- The oscillation movement can likewise be started, influenced and stopped from the PLC/MMC.
- Interpolatory oscillation is not possible (e.g. oblique oscillation).

2.1.1 Influences on asynchronous oscillation

Setting data

The setting data required for oscillation can be set with special language commands in the NCK part program, via the MMC and/or the PLC.

Feed velocity

The feed velocity for the oscillation axis is selected or programmed as follows:

- The velocity defined for the axis as a positioning axis is used as the feed velocity. This value can be programmed via FA[axis] and has a modal action. If no velocity is programmed, then the value stored in machine data POS_AX_VELO is used (see positioning axes).
- When an oscillation movement is in progress, the feed velocity of the oscillation axis can be altered via setting data. It can be specified via the part program and setting data whether the changed velocity must take effect immediately or whether it should be activated at the next reversal point.
- The feed velocity can be influenced via the override (axial VDI signal and programmable).
- If “Dry run” is active, the dry run velocity setting is applied if it is higher than the currently programmed velocity.
- Velocity overlay/path overlay can be influenced by the handwheel.
  See also Table 2-1.

  References: /FB/, H1, “JOG with and without Handwheel”
- The oscillation axis can be moved with revolutional feedrate.

Revolutional feedrate

The reversal feed can also be used for oscillation axes.

Reversal points

The positions of the reversal points can be entered via setting data before an oscillation movement is started or while one is in progress.

- The reversal point positions can be entered by means of manual traverse (handwheel, JOG keys) before or in the course of an oscillation movement, regardless of whether the oscillation movement has been interrupted or not.

The following applies to alteration of a reversal point position: When an oscillation movement is already in progress, the altered position of a reversal point does not become effective until this point is approached again. If the axis is already approaching the position, the correction will take effect in the next oscillation stroke.

Note

If a reversal point must be altered at the same time as VDI interface signal *"Activate DRF"* is set, the handwheel signals are applied both to the DRF offset and to the offset of the reversal point, i.e. the reversal point is shifted absolutely by an amount corresponding to twice the distance.

Stop times

A stop time can be programmed via setting data for each reversal point.
The setting can be altered in the following blocks of the NC program. It is then effective in block synchronism from the next applicable reversal point.
The stop time can be altered asynchronously via setting data. It is then effective from the instant that the appropriate reversal point is next traversed.

The following table explains the motional behavior in the exact stop range or at the reversal point depending on the stop time input.

Table 2-1 Effect of stop time

Stop time setting	Response
–2	Interpolation continues without wait for exact stop
–1	Wait for coarse exact stop at reversal point
0	Wait for fine exact stop at reversal point
>0	Wait for fine exact stop at reversal point followed by wait for stop time

Deactivate oscillation

One of the following options can be set for termination of the oscillation movement when oscillation mode is deactivated:

- Termination of oscillation movement at the next reversal point
- Termination of oscillation movement at reversal point 1
- Termination of oscillation movement at reversal point 2

Following this termination process, sparking-out strokes are processed and an end position approached if programmed.

On switchover from asynchronous oscillation to spark-out and during spark-out, the response at the reversal point regarding exact stop corresponds to the response determined by the stop time programmed for the appropriate reversal point. A sparking-out stroke is the movement towards the other reversal point and back. See table:

Note

Oscillation with motion-synchronous actions and stop times “OST1/OST2”

Once the set stop times have expired, the internal block change is executed during oscillation (indicated by the new distances to go of the axes). The deactivation function is checked when the block changes. The deactivation function is defined according to the control setting for the motional sequence “OSCTRL”.

This dynamic response can be influenced by the feed override.

An oscillation stroke may then be executed before the sparking-out strokes are started or the end position approached.

Although it appears as if the deactivation response has changed, this is not in fact the case.

Table 2-2 Operational sequence for deactivation of oscillation

Function	Inputs	Explanation
Deactivation at defined reversal point	Number of sparking-out strokes equals 0, no end position active	The oscillation movement is stopped at the appropriate reversal point
Deactivation with specific number of sparking-out strokes	Number of sparking-out strokes is not equal to 0, no end position active	After appropriate reversal point is reached, the number of sparking-out strokes specified in command is processed.
Deactivation with sparking-out strokes and defined end position (optional)	Number of sparking-out strokes is not equal to 0, end position active	After appropriate reversal point is reached, the number of sparking-out strokes specified in command is processed followed by approach to specified end position.
Deactivation without sparking-out strokes, but with defined end position (optional)	Number of sparking-out strokes equals 0, end position active	After appropriate reversal point is reached, axis is traversed to specified end position.

NC language

The NC programming language allows asynchronous oscillation to be controlled from the part program. The following functions allow asynchronous oscillation to be activated and controlled as a function of the NC program processing.

Note

If the setting data are directly written in the part program, then the data change takes effect prematurely with respect to processing of the part program (at the preprocessing time). It is possible to re-synchronize the part program and the oscillation function commands by means of a preprocessing stop (STOPRE).

References: /PA/, Programming Guide

1. **Activate, deactivate oscillation:**

- OS[oscillation axis] = 1; Activate oscillation for oscillation axis
- OS[oscillation axis] = 0; Deactivate oscillation for oscillation axis

---

Note

Every axis may be used as an oscillation axis.

---

2. **End of oscillation:**

- WAITP(oscillation axis)
  Positioning axis command – stops block until oscillation axis is at fine stop and synchronizes preprocessing and main run. The oscillation axis is entered as a positioning axis again and can then be used normally.
  If an axis is to be used for oscillation, then it must be enabled beforehand with a WAITP(axis) call.
  This also applies if oscillation must be initiated from the PLC/MMC. In this case, the WAITP(axis) call is also needed if the axis was programmed beforehand via the NC program. With SW version 3.2 and higher it is possible to select via machine data $MA_AUTO_GET_TYPE, whether WAITP() shall be performed with programming or automatically.

Note

WAITP effectively implements a time delay until the oscillation movement has been executed. Termination of the movement can be initiated, for example, through a programmed deactivation command in the NC program or via the PLC or MMC by means of deletion of distance-to-go.

---

3. **Setting reversal points:**

- OSP1[oscillation axis] = position of reversal point 1
- OSP2[oscillation axis] = position of reversal point 2

A position is entered into the appropriate setting data in synchronism with the block in the main run and thus remains effective until the setting data is next changed.

If incremental traversal is active, then the position is calculated incrementally to the last appropriate reversal point programmed in the NC program.

4. **Stop times at reversal points**

- OST1[oscillation axis] = stop time at reversal point 1 in [s]
- OST2[oscillation axis] = stop time at reversal point 2 in [s]

A stop time is entered into the appropriate setting data in synchronism with the block in the main run and thus remains effective until the setting data is next changed.

The unit for the stop time is identical to the unit selected for the stop time programmed with G04.

5. **Setting feedrate:**

- FA[axis] = Fvalue
  Positioning axis feedrate.

The feedrate is transferred to the appropriate setting data in synchronism with the block in the main run. If the oscillation axis is moved with revolutional feedrate, the corresponding dependencies must be indicated as described in Description of Functions V1.

6. **Setting control settings for sequence of movements:**

- OSCTRL[oscillation axis] = (setting options, resetting options)

  The set options are defined as follows (the reset options deselect the settings):

Table 2-3 Set/reset options

Option value	Meaning
0	Stop at next reversal point on deactivation of the oscillation movement (default). Can only be achieved by resetting option values 1 and 2.
1	Stop at reversal point 1 on deactivation of the oscillation movement
2	Stop at reversal point 2 on deactivation of the oscillation movement.
3	On deactivation of oscillation movement, do not approach reversal point unless sparking-out strokes are programmed.
4	Approach an end position after spark-out process.
8	If the oscillation movement is terminated by deletion of distance-to-go, the specified sparking-out strokes must then be executed and the end position (if programmed) approached.
16	If the oscillation movement is terminated by deletion of distance-to-go, the programmed reversal point must be approached on deactivation of the oscillation movement.
32	Altered feedrate will only take effect from the next reversal point.
64	If feedrate setting is 0, path overlay is active, or otherwise velocity overlay
128	For rotary axis DC (shortest path)

Note

The option values 0–3 encode the behavior at reversal points at Power OFF. You can choose one of the alternatives 0–3. The other settings can be combined with the selected alternative according to individual requirements. A “+” character can be inserted to create a string of options.

Example: The oscillation movement for axis Z must stop at reversal point 1 on deactivation; an end position must then be approached and a newly programmed feedrate take immediate effect; the axis must stop immediately after deletion of distance-to-go.
OSCTRL[Z] = (1+4, 16+32+64)

The set/reset options are entered into the appropriate setting data in synchronism with the block in the main run and thus remain effective until the setting data is next changed.

Note

First the control evaluates the reset options, then the setting options.

---

7. **Sparking-out strokes:**

- OSNSC[oscillation axis] = number of sparking-out strokes

The number of sparking-out strokes is entered into the appropriate setting data in synchronism with the block in the main run and thus remains effective until the setting data is next changed.

8. **End position to be approached after deactivation of oscillation:**

- OSE[oscillation axis] = end position of oscillation axis

The end position is entered into the appropriate setting data in synchronism with the block in the main run and thus remains effective until the setting data is next changed. Option value 4 is set implicitly according to Table 2-3, such that the set end position is approached.

Programming example

Chapter 6 gives an example containing all the important elements for asynchronous oscillation.

2.1.2 Asynchronous oscillation under PLC control

Activation

The function can be selected from the PLC via setting data OSCILL_IS_ACTIVE in all operating modes except for MDA Ref and JOG Ref.

Settings

The following criteria can be controlled from the PLC via setting data: Activation and deactivation of oscillation movement, positions of reversal points 1 and 2, stop times at reversal points, feedrate velocity, the options in the reversal points, the number of sparking-out strokes and the end position after deactivation. However, these values can also be set beforehand as a setting data via the MMC directly or via an NC program. These settings remain valid after power ON and the PLC can also start an oscillation movement set in this way directly via setting data OSCILL_IS_ACTIVE (via variable service).

Supplementary conditions

A spindle which must act as an axis to execute an oscillation movement started via the PLC must fulfill the conditions required to allow traversal as a positioning axis, i.e. the spindle must, for example, have been switched to the position control (SPOS) beforehand.

The axes always react to the two stop bits of the VDI interface (DBX28.5, DBX28.6) regardless of whether or not they are being controlled by the PLC.

2.1.3 Special reactions during asynchronous oscillation

With PLC control

The PLC program can take over the control of an oscillation axis via VDI signals. These VDI signals also include program end, operating mode changeover and single block. The following VDI interface signals are ignored: feed/spindle stop and NC STOP; the resulting deceleration request is suppressed in the case of delete distance-to-go.

Without PLC control

If the PLC does not have control over the axis, then the axis is treated like a normal positioning axis (POSA) during asynchronous oscillation.

References: /FB/, P2, “Positioning axes”

EMERGENCY STOP

In the event of an EMERGENCY STOP, the axis is decelerated by the servo (by cancellation of servo enable and follow-up). The oscillation movement is thus terminated and must be restarted if necessary.

Delete distance-to-go

Channel-specific delete distance-to-go is ignored. Axial delete distance-to-go:

Without PLC control

If the oscillation axis is not under PLC control, it is stopped by means of a braking ramp.

With PLC control

In this case, deceleration is suppressed and must be initiated by the PLC. The following applies to **both** cases: After the axis has been stopped, the appropriate reversal point is approached (see OSCILL_CONTROL_MASK, Section 4) and the distance-to-go deleted. The sparking-out strokes are then executed and the end position approached if this has been set such in OSCILL_CONTROL_MASK.
The oscillation movement is thus terminated.

Note

During grinding, the calipers can be put into action via axial delete distance-to-go.

Reset

The oscillation movement is interrupted and deselected with a braking ramp. The options selected subsequently are not processed (sparking-out strokes, end point approach).

Special reactions during asynchronous oscillation (continued)

Working area limitation, limit switches

If it is detected during preprocessing that the oscillation movement would violate an active limitation, then an alarm is output and the oscillation movement not started. If the oscillation axis violates a limitation which has been activated in the meantime (e.g. 2nd software limit switch), then the axis is decelerated down along a ramp and an alarm output.

Caution

Protection zones are not effective!

Follow-up mode

There is no difference to positioning axes.

Program end

If the axis is not controlled by the PLC, then the program end is not reached until the oscillation movement is terminated (reaction as for POSA:Positioning beyond block limits).

If the axis is controlled by the PLC, then it continues to oscillate after program end.

Operating mode changeover

The following table shows the operating modes in which oscillation can be implemented. Changeover to an operating mode which allows oscillation does not affect the oscillation movement. Changeover to inadmissible operating modes is rejected with an alarm. It is not possible to traverse an axis in oscillation mode while applying control commands from the NC program or via operator inputs (JOG) simultaneously; an alarm is output if this is attempted. The general rule is that the type of movement first started has priority.

Table 2-4 Operating modes which allow oscillation

Operating mode	Allows oscillation
AUTO	yes
MDA	yes
MDA Repos	yes
MDA Teachin	yes
MDA Ref	no
JOG	yes
JOG Ref	no
JOG Repos	yes

Single-block processing

If the axis is not controlled by the PLC, then it responds to a single block in the same way as a positioning axis (POSA), i.e. it continues the movement.

Override

The override is specified by the

VDI interface

Axial override acts on the oscillation axis.

Programming

The override acts on oscillation axes in the same way as on positioning axes.

Block search

In the case of a block search, the last valid oscillation function is registered and is activated – depending on machine data OSCILL_MODE_MASK – either immediately after NC start (on approach to approach position after block search) or after the approach position has been reached after block search (default setting).
OSCILL_MODE_MASK bit 0:

0	Oscillation starts after approach position is reached
1	Oscillation starts immediately after NC start

REORG

Reversal point 1 is always approached first before oscillation continues.

ASUB

The oscillation movement continues while an ASUB (asynchronous subprogram) is in progress.

2.2 Oscillation controlled by synchronized actions

Principle

An asynchronous oscillation movement is coupled via synchronized actions with an infeed motion and controlled accordingly.

References: /FB/, S5, “Synchronized Actions”

The following description concentrates solely on the motion-synchronous actions associated with the oscillation function.

Functions

The following function complexes can be implemented by means of the language tools described in detail below:

1. Infeed at reversal point (see 2.2.1).
2. Infeed in reversal point range (see 2.2.2).
3. Infeed at both reversal points (see 2.2.3).
4. Stopping oscillation movement at reversal point until infeed is terminated (see 2.2.4).
5. Enable oscillation movement (see 2.2.5).
6. Preventing premature start of partial infeed (see 2.2.6).

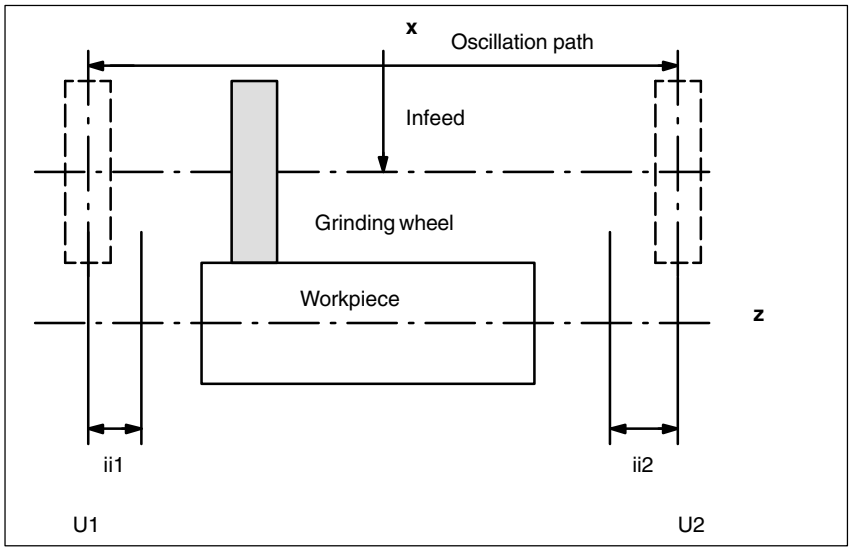


Fig. 2-1 Arrangement of oscillation and infeed axes plus terms

Legend:		
	U1	Reversal point 1
	U2	Reversal point 2
	ii1	Reversal range 1
	ii2	Reversal range 2

Programming

The parameters governing oscillation (see 2.2.7) must be defined before the movement block containing the assignment between the infeed and oscillation axes (see 2.1.1), the infeed definition (POSP) and the motion-synchronous actions:

The oscillation axis is enabled via a WAITP [oscillation axis] (see MD $MA_AUTO_GET_TYPE), allowing the oscillation parameters to be transferred, i.e. into the setting data, simultaneously. The symbolic names, e.g. $SA_REVERSE_POS1 can then be used to program the motion-synchronous actions.

Note

For motion-synchronous actions with $SA_REVERSE_POS values, the comparison values at the **time of interpretation** are valid. If setting data are modified subsequently, this has no influence.
For motion-synchronous actions with $$AA_REVERSE_POS values, the comparison values within **interpolation** are valid. This ensures a reaction to the modified reversal positions.

- **Motion-synchronous conditions WHEN, WHENEVER**
- Activation through motion block
  - Assign oscillation axis and infeed axes to one another OSCILL
  - Specify infeed response POSP.

The elements which have not yet been discussed are explained in more detail in the following sections.
Some examples are described in Chapter 6.

Note

If the condition with which the motion-synchronous action (WHEN and WHENEVER) has been defined is no longer valid, the OVERRIDE for this condition is **automatically** set to 100% if the OVERRIDE had been set to 0% before.

Main run evaluation

From software version 3.2, it is possible to compare the synchronization conditions in the interpolation cycle in the main run with the current actual values ($$ variable on the right of comparison conditions). With normal system variable comparison, the expressions are evaluated in the first run. The complete extended possibilities for synchronized actions are listed in the following documentation:

References: /FB/, S5, “Motion-synchronous actions”.

Example 1

Oscillation, reversal position firmly set via setting data:

```
$SA_OSCILL_REVERSE_POS1[Z]=–10
$SA_OSCILL_REVERSE_POS2[Z]=10

G0 X0 Z0
WAITP(Z)

ID=1 WHENEVER $AA_IM[Z] < $SA_OSCILL_REVERSE_POS1[Z] DO
        $AA_OVR[X]=0
ID=2 WHENEVER $AA_IM[Z] > $SA_OSCILL_REVERSE_POS2[Z] DO
        $AA_OVR[X]=0
                                        ; If the actual value of the oscillation
                                        ; axis has exceeded the reversal point,
                                        ; the infeed axis is stopped.

OS[Z]=1 FA[X]=1000 POS[X]=40            ; Oscillation on
OS[Z]=0                                 ; Oscillation off

M30
```

Example 2

Oscillation with online change of the reversal position, i.e. any modification of reversal position 1 via the user surface are immediately taken into account with active oscillation movement.

```
$SA_OSCILL_REVERSE_POS1[Z]=–10
$SA_OSCILL_REVERSE_POS2[Z]=10

G0 X0 Z0
WAITP(Z)

ID=1 WHENEVER $AA_IM[Z] < $$SA_OSCILL_REVERSE_POS1[Z] DO
        $AA_OVR[X]=0
ID=2 WHENEVER $AA_IM[Z] > $$SA_OSCILL_REVERSE_POS2[Z] DO
        $AA_OVR[X]=0
                                        ; If the actual value of the oscillation
                                        ; axis has exceeded the reversal point,
                                        ; the infeed axis is stopped.

OS[Z]=1 FA[X]=1000 POS[X]=40            ; Oscillation on
OS[Z]=0                                 ; Oscillation off

M30
```

2.2.1 Infeed at reversal point 1 or 2

Function

As long as the oscillation axis has not reached the reversal point, the infeed axis does not move.

Application

Direct infeed in reversal point

Programming

For reversal point 1:
WHENEVER $AA_IM[Z] <> $SA_OSCILL_REVERSE_POS1[Z] DO
$AA_OVR[X] = 0 $AA_OVR[Z] = 100

For reversal point 2:
WHENEVER $AA_IM[Z] <> $SA_OSCILL_REVERSE_POS2[Z] DO
$AA_OVR[X] = 0 $AA_OVR[Z] = 100

Explanation of system variables:

$AA_IM[Z]	Actual position of oscillation axis Z in Machine coordinate system
$SA_OSCILL_REVERSE_POS1[Z]	Position of reversal point 1 of oscillation axis
$AA_OVR[X]	Axial override of infeed axis
$AA_OVR[Z]	Axial override of oscillation axis

Explanation of keywords:

WHENEVER ... DO ...	Whenever condition is fulfilled, then ....

Infeed

The absolute infeed value is defined by instruction POSP.
See 2.2.8

Assignment

The assignment between the oscillation axis and the infeed axis is defined by instruction OSCILL.
See 2.2.7

2.2.2 Infeed in reversal point range

Function

Reversal point range 1:
No infeed takes place provided the oscillation axis has not reached the reversal point range (position at reversal point 1 plus contents of variable ii1). This applies on the condition that reversal point 1 is set to a lower value than reversal point 2. If this is not the case, then the condition must be changed accordingly.

Application

Reversal point range 1:
The purpose of the synchronized action is to prevent the infeed movement from starting until the oscillation movement has reached reversal point range 1.
See Fig. 2-1.

Programming

Reversal point range 1:
WHENEVER $AA_IM[Z] > $SA_OSCILL_REVERSE_POS1[Z] + ii1 DO
$AA_OVR[X] = 0

Explanation of system variables:

$AA_IM[Z]	Actual position of oscillation axis Z
$SA_OSCILL_REVERSE_POS1[Z]	Position of reversal point 1 of oscillation axis
$AA_OVR[X]	Axial override of infeed axis
ii1	Size of reversal range(user variable)

Explanation of keywords:

WHENEVER ... DO ...	Whenever condition is fulfilled, then ....

Function

Reversal point range 2:
The infeed axis stops until the current position (value) of the oscillation axis is lower than the position at reversal point 2 minus the contents of variable ii2. This applies on condition that the setting for reversal point position 2 is higher than that for reversal point position 1. If this is not the case, then the condition must be changed accordingly.

Application

Reversal point range 2:
The purpose of the synchronized action is to prevent the infeed movement from starting until the oscillation movement has reached reversal point range 2.
See Fig. 2-1.

Programming

Reversal point range 2:
WHENEVER $AA_IM[Z] < $SA_OSCILL_REVERSE_POS2[Z] – ii2 DO
$AA_OVR[X] = 0

Explanation of system variables:

$AA_IM[Z]	Actual position of oscillation axis Z
$SA_OSCILL_REVERSE_POS2[Z]	Position of reversal point 2 of oscillation axis
$AA_OVR[X]	Axial override of infeed axis
ii2	Size of reversing point range 2 (user variable)

Infeed

The absolute infeed value is defined by instruction POSP.
See 2.2.8.

Assignment

The assignment between the oscillation axis and the infeed axis is defined by instruction OSCILL.
See 2.2.7.

2.2.3 Infeed at both reversal points

Principle

The functions described above for infeed at the reversal point and in the reversal point range can be freely combined.

Combinations

Infeed:

at U1	at U2
at U1	range U2
range U1	at U2
range U1	range U2

One-sided infeed

at U1
at U2
range U1
range U2

These options are described in Sections 2.2.1 and 2.2.2.

2.2.4 Stopping oscillation movement at reversal point

Function

Reversal point 1:
Every time the oscillation axis reaches reversal position 1, it must be stopped by means of the override and the infeed movement started.

Application

The synchronized action is used to hold the oscillation axis stationary until part infeed has been executed. This synchronized action can be omitted if the oscillation axis need not wait at reversal point 1 until part infeed has been executed. At the same time, this synchronized action can be used to start the infeed movement if this has been stopped by a previous synchronized action which is still active.

Programming

```
WHENEVER $AA_IM[oscillation axis] ==
$SA_OSCILL_REVERSE_POS1[oscillation axis]
DO $AA_OVR[oscillation axis] = 0   $AA_OVR[infeed axis] = 100
```

Explanation of system variables:

$AA_IM[oscillation axis]	Current position of oscillation axis
$SA_OSCILL_REVERSE_POS1[oscillation axis]	Reversal point 1 of oscillation axis
$AA_OVR[oscillation axis]	Axial override of oscillation axis
$AA_OVR[infeed axis]	Axial override of infeed axis

Function

Reversal point 2:
Every time the oscillation axis reaches reversal position 2, it must be stopped by means of override 0 and the infeed movement started.

Application

The synchronized action is used to hold the oscillation axis stationary until part infeed has been executed. This synchronized action can be omitted if the oscillation axis need not wait at reversal point 2 until part infeed has been executed. At the same time, this synchronized action can be used to start the infeed movement if this has been stopped by a previous synchronized action which is still active.

Programming

```
WHENEVER $AA_IM[oscillation axis] ==
$SA_OSCILL_REVERSE_POS2[oscillation axis]
DO $AA_OVR[oscillation axis] = 0   $AA_OVR[infeed axis] = 100
```

Explanation of system variables:

$AA_IM[oscillation axis]	Current position of oscillation axis
$SA_OSCILL_REVERSE_POS2[oscillation axis]	Reversal point 2 of oscillation axis
$AA_OVR[oscillation axis]	Axial override of oscillation axis
$AA_OVR[infeed axis]	Axial override of infeed axis

2.2.5 Oscillation movement restarting

Function

The oscillation axis is started via the override whenever the distance-to-go for the currently traversed path section of the infeed axis = 0, i.e. part infeed has been executed.

Application

The purpose of this synchronized action is to continue the movement of the oscillation axis on completion of the part infeed movement. If the oscillation axis need not wait for completion of partial infeed, then the motion-synchronous action with which the oscillation axis is stopped at the reversal point must be omitted.

Programming

```
WHENEVER $AA_DTEPW[infeed axis] == 0 DO
$AA_OVR[oscillation axis] =100
```

Explanation of system variables:

$AA_DTEPW[infeed axis]	Axial distance-to-go for infeed axis in workpiece coordinate system Path section of infeed axis
$AA_OVR[oscillation axis]	Axial override for oscillation axis

Explanation of keywords:

WHENEVER ... DO ...	Whenever condition is fulfilled, then ....

2.2.6 Preventing premature starting of partial infeed

Function

The functions described above prevent any infeed movement outside the reversal point or the reversal point range. On completion of an infeed movement, however, restart of the next partial infeed must be prevented.

Application

A channel-specific flag is used for this purpose. This flag is set at the end of the partial infeed (partial distance-to-go == 0) and is deleted when the axis leaves the reversal point range. The next infeed movement is then prevented by a synchronized action.

Programming

```
WHENEVER $AA_DTEPW[infeed axis] == 0 DO
$AC_MARKER[index]=1
```

and, for example, for reversal point 1:

```
WHENEVER $AA_IM[Z]<> $SA_OSCILL_REVERSE_POS1[Z] DO
$AC_MARKER[index]=0

WHENEVER $AC_MARKER[index]==1 DO $AA_OVR[infeed axis]=0
```

Explanation of system variables:

$AA_DTEPW[infeed axis]	Axial distance-to-go for infeed axis in workpiece coordinate system Path section of infeed axis
$AC_MARKER[index]	Channel-specific flag with index
$AA_IM[oscillation axis]	Current position of oscillation axis
$SA_OSCILL_REVERSE_POS1[oscillation axis]	Reversal point 1 of oscillation axis
$AA_OVR[infeed axis]	Axial override for infeed axis

Explanation of keywords:

WHENEVER ... DO ...	Whenever condition is fulfilled, then ....

2.2.7 Assignment of oscillation and infeed axes OSCILL

Function

One or several infeed axes are assigned to the oscillation axis with command OSCILL. The oscillation movement is started.
The PLC is informed of which axes have been assigned via the VDI interface. If the PLC is controlling the oscillation axis, it must now also monitor the infeed axes and use the signals for the infeed axes to generated the reactions on the oscillation axis via 2 stop bits of the interface.

Application

The axes whose response has already been defined by synchronous conditions are assigned to one another for activation of oscillation mode. The oscillation movement is started.

Programming

OSCILL[oscillation axis] = (infeed axis1, infeed axis2, infeed axis3)

Infeed axis2 and infeed axis3 in brackets plus their delimiters can be omitted if they are not required.

2.2.8 Definition of infeeds POSP

Function

The control receives the following data for the infeed axis:

- Total infeed
- Part infeed at reversal point/reversal point range
- Part infeed response at end

Application

This instruction must be given after activation of oscillation with OSCILL to inform the control of the required infeed values at the reversal points/reversal point ranges.

Programming

POSP[infeed axis] = (end position, part section, mode)

End position	End position for infeed axis after all partial infeeds have been executed.
Part section	Part infeed at reversal point/reversal point range
Mode	0

For the last two part steps, the remaining path up to the target point is divided into two equally large residual steps (default setting).

Mode 1

The part length is adjusted such that the total of all calculated part lengths corresponds exactly to the path up to the target point.

■

Notes

Supplementary Conditions 3

Availability of "Oscillation" function

Oscillation is an option with order number 6FC5 251-0AB04-0AA0.
Asynchronous and modal oscillation is available as from SW2 for NCU570, 571, 572, 573.
Oscillation with motion-synchronous actions is available with NCU 572 and 573.

Data Descriptions (MD, SD) 4

4.1 Machine data

11460 MD number	**OSCILL_MODE_MASK** Mode screen form for asynchronous oscillation		
Default value: 0	Min. input limit: 0	Max. input limit: 0xFFFF	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies from SW version: 2.0		
Significance:	Bit 0 Value 1 In the case of block search, the oscillation movement is started immediately after NC start, i.e. during approach to approach position, provided it has been activated in the program section being processed. Value 0 The oscillation movement is not started until the approach position is reached.		

4.2 Setting data

Axis/spindle-specific data

43700 SD number	**OSCILL_REVERSE_POS1[axis]** Oscillation reversal point 1	
Default value: 0	Min. input limit: ***	Max. input limit: ***
Changes effective immediately		Unit: mm, degrees
Data type: DOUBLE	Applies from SW version: 2.0	
Significance:	Position of oscillation axis at reversal point 1	
Application	NC language: OSP1[axis]=position	
Related to	OSCILL_REVERSE_POS2 MD 10710 $MN_PROG_SD_RESET_SAVE_TAB	

43710 SD number	**OSCILL_REVERSE_POS2[axis]** Oscillation reversal point 2	
Default value: 0	Min. input limit: ***	Max. input limit: ***
Changes effective immediately		Unit: mm, degrees
Data type: DOUBLE	Applies from SW version: 2.0	
Significance:	Position of oscillation axis at reversal point 2	
Application	NC language: OSP2[axis]=position	
Related to	OSCILL_REVERSE_POS1 MD 10710 $MN_PROG_SD_RESET_SAVE_TAB	

43720 SD number	**OSCILL_DWELL_TIME1[axis]** Stop time at oscillation reversal point 1	
Default value: 0	Min. input limit: –2	Max. input limit: ***
Changes effective immediately		Unit: s
Data type: DOUBLE	Applies from SW version: 2.0	
Significance:	Stop time of oscillation axis at reversal point 1	
Application	NC language: OST1[axis]=time	
Related to	OSCILL_DWELL_TIME2 MD 10710 $MN_PROG_SD_RESET_SAVE_TAB	

43730 SD number	**OSCILL_DWELL_TIME2[axis]** Stop time at oscillation reversal point 2	
Default value: 0	Min. input limit: –2	Max. input limit: ***
Changes effective immediately		Unit: s
Data type: DOUBLE	Applies from SW version: 2.0	
Significance:	Stop time of oscillation axis at reversal point 2	
Application	NC language: OST2[axis]=time	
Related to	OSCILL_DWELL_TIME1 MD 10710 $MN_PROG_SD_RESET_SAVE_TAB	

43740 SD number	**OSCILL_VELO[axis]** Feed velocity of oscillation axis	
Default value: 0	Min. input limit: ***	Max. input limit: ***
Changes effective immediately		Unit: mm/min rev/min
Data type: DOUBLE	Applies from SW version: 2.0	
Significance:	Feed velocity of oscillation axis	
Application	NC language: FA[axis]=Fvalue	
Related to	MD 10710 $MN_PROG_SD_RESET_SAVE_TAB	

43750 SD number	**OSCILL_NUM_SPARK_CYCLES[axis]** Number of sparking-out strokes	
Default value: 0	Min. input limit: 0	Max. input limit: ***
Changes effective immediately		Unit: 1
Data type: DWORD	Applies from SW version: 2.0	
Significance:	Number of sparking-out strokes which are executed on completion of oscillation movement.	
Application	NC language: OSNSC[axis]=number of strokes	
Related to	MD 10710 $MN_PROG_SD_RESET_SAVE_TAB	

43760 SD number	**OSCILL_END_POS[axis]** End position of oscillation axis	
Default value: 0	Min. input limit: ***	Max. input limit: ***
Changes effective immediately		Unit: mm, degrees
Data type: DOUBLE	Applies from SW version: 2.0	
Significance:	Position to be approached by oscillation axis after execution of sparking-out strokes.	
Application	NC language: OSE[axis]=position	
Related to	MD 10710 $MN_PROG_SD_RESET_SAVE_TAB	

43770 SD number		**OSCILL_CTRL_MASK[axis]** Oscillation sequence control screen form	
Default value: 0		Min. input limit: –	Max. input limit: –
Changes effective immediately			Unit: –
Data type: BYTE		Applies from SW version: 2.0	
Significance:		Bit screen form, see following table 4-1	
Application		NC language: OSCTRL[axis]=(setting options, resetting options)	
Related to		MD 10710 $MN_PROG_SD_RESET_SAVE_TAB	

Table 4-1 Bit significance in screen form OSCILL_CTRL_MASK

Bit No.	Meaning in OSCILL_CTRL_MASK
0,1	0: Stop at next reversal point on deactivation of oscillation movement 1: Stop at reversal point 1 on deactivation of oscillation movement 2: Stop at reversal point 2 on deactivation of oscillation movement 3: On deactivation of oscillation movement. do not approach reversal point unless sparking-out strokes are programmed.
2	1: Approach end position after next sparking-out
3	1: If the oscillation movement is aborted by delete distance-to-go, the sparking-out strokes must then be executed and the end position approached (if programmed)
4	1: If the oscillation movement is aborted by delete distance-to-go, then the appropriate reversal position is approached as for deactivation
5	1: New feedrate setting not effective until the next reversal point

Table 4-1 Bit significance in screen form OSCILL_CTRL_MASK

Bit No.	Meaning in OSCILL_CTRL_MASK
6	1: If feedrate setting is 0, path overlay is active, or otherwise velocity overlay
7	1: For rotary axes DC (shortest path)

43780 SD number	**OSCILL_IS_ACTIVE[axis]** Activate oscillation movement	
Default value: 0	Min. input limit: 0	Max. input limit: 1
Changes effective immediately		Unit: –
Data type: BOOLEAN	Applies from SW version: 2.0	
Significance:	Activate and deactivate oscillation movement	
Application	NC language: OS[axis]=1 OS[axis]=0	
Related to	MD 10710 $MN_PROG_SD_RESET_SAVE_TAB	

■

Signal Descriptions

5

VDI input signals

The PLC user program uses the following signals to control the oscillation process.

DB31, ... **DBX28.3** Data block	**Set reversal point** Signal(s) to axis/spindle		
Edge evaluation: no		Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.0
Signal state 1 or signal transition 0 ——> 1	Reversal point 2		
Signal state 0 or signal transition 1 ——> 0	Reversal point 1		

DB31, ... **DBX28.4** Data block	**Alter reversal point** Signal(s) to axis/spindle		
Edge evaluation: no		Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.0
Signal state 1 or signal transition 0 ——> 1	The selected reversal point can be altered by manual traverse.		
Signal state 0 or signal transition 1 ——> 0	The selected reversal point cannot be altered by manual traverse.		
Related to	DBX28.3		

DB31, ... **DBX28.5** Data block	**Stop at next reversal point** Signal(s) to axis/spindle		
Edge evaluation: no		Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.0
Signal state 1 or signal transition 0 ——> 1	The oscillation movement is interrupted at the next reversal point.		
Signal state 0 or signal transition 1 ——> 0	The oscillation movement continues after the next reversal point.		
Related to	DBX28.6, DBX28.7		

DB31, ... **DBX28.6** Data block	**Stop along braking ramp** Signal(s) to axis/spindle		
Edge evaluation: no		Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.0
Signal state 1 or signal transition 0 ——> 1	The axis is decelerated along a ramp, the oscillation movement is interrupted.		
Signal state 0 or signal transition 1 ——> 0	The oscillation movement continues without interruption.		
Related to	DBX28.5, DBX28.7		

DB31, ... **DBX28.7** Data block	**PLC controls axis** Signal(s) to axis/spindle	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.0
Signal state 1 or signal transition 0 ——> 1	Axis is controlled by the PLC. The reaction to interface signals is controlled by the PLC by means of the 2 stop bits, other signals with deceleration action are ignored.	
Signal state 0 or signal transition 1 ——> 0	Axis is not controlled by the PLC.	
Related to	DBX28.5, DBX28.6	

VDI output signals

The NCK makes the following signals available to the PLC user program.

DB31, ... **DBX100.3** Data block	**Oscillation cannot start** Signal(s) from axis/spindle	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The oscillation axis cannot be started owing to incorrect programming. This status can occur even when axis has already been traversed.	
Signal state 0 or signal transition 1 ——> 0	The oscillation movement can be started.	

DB31, ... **DBX100.4** Data block	**Error during oscillation movement** Signal(s) from axis/spindle	
Edge evaluation:	Signal(s) updated:	Signal(s) valid from SW vers.: 2..1
Signal state 1 or signal transition 0 ——> 1	The oscillation movement has been aborted.	
Signal state 0 or signal transition 1 ——> 0	The oscillation movement is being executed correctly.	

DB31, ... **DBX100.5** Data block	**Sparking-out active** Signal(s) from axis/spindle	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The axis is executing sparking-out strokes.	
Signal state 0 or signal transition 1 ——> 0	The axis is not currently executing sparking-out strokes.	
Related to	DBX100.7	

DB31, ... **DBX100.6** Data block	**Oscillation movement active** Signal(s) from axis/spindle	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The axis is executing an oscillation movement between 2 reversal points.	
Signal state 0 or signal transition 1 ——> 0	The axis is not currently oscillating.	
Signal irrelevant for.....	DBX100.7 = 0	
Related to	DBX100.7	

DB31, ... **DBX100.7** Data block	**Oscillation active** Signal(s) from axis/spindle	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The axis is currently being traversed as an oscillation axis.	
Signal state 0 or signal transition 1 ——> 0	The axis is a positioning axis.	
Related to	DBX100.5, DBX100.6	

DB31, ... **DBX104.0 – 7** Data block	**Active infeed axes** Signal(s) from axis/spindle	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The axis sending the signal is currently the oscillation axis and is indicating its active infeed axes in this field (104.0 axis 1 is infeed axis, 104.1 axis 2 is infeed axis, etc.).	
Signal state 0 or signal transition 1 ——> 0	The associated axis is not an infeed axis.	
Related to	DBX100.7	

■

Notes

Examples 6

Preconditions

The examples given below require components of the NC language specified in the sections entitled

– Asynchronous oscillation

and

– Oscillation controlled by synchronized

actions.

6.1 Example of asynchronous oscillation

Task

The oscillation axis Z must oscillate between –10 and 10. Approach reversal point 1 with exact stop coarse and reversal point 2 without exact stop. The oscillation axis feedrate must be 5000. 3 sparking-out strokes must be executed at the end of the machining operation followed by approach by oscillation axis to end position 30. The feedrate for the infeed axis is 1000, end of the infeed in X direction is at 15.

Program section

```
OSP1[Z]=–10          ; Reversal point 1
OSP2[Z]=10           ; Reversal point 2
OST1[Z]=–1           ; Stop time at reversal point 1: Exact stop coarse
OST2[Z]=–2           ; Stop time at reversal point 2: Without exact stop
FA[Z]=5000           ; Feedrate for oscillation axis
OSNSC[Z]=3           ; Three sparking-out strokes
OSE[Z]=30            ; End position
OS[Z]=1 F500 X15     ; Start oscillation, infeed X axis
                     ; with feedrate 5000, infeed target 15
```

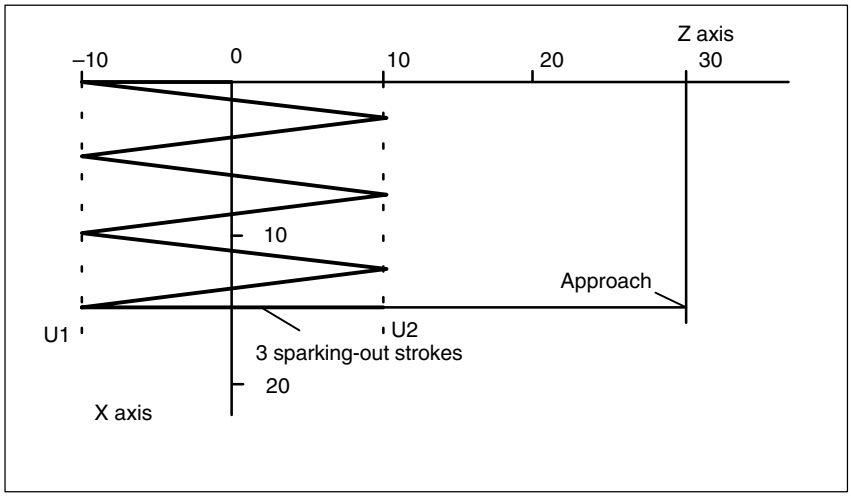


Fig. 6-1 Sequences of oscillation movements and infeed, example 1

6.2 Example 1 of oscillation with synchronized actions

Task

Direct infeed must take place at reversal point 1; the oscillation axis must wait until the part infeed has been executed before it can continue traversal. At reversal point 2, the infeed must take place at a distance of -6 from reversal point 2; the oscillation axis must not wait at this reversal point until part infeed has been executed. Axis Z is the oscillation axis and axis X the infeed axis. (see 2.2).

Note

The setting data OSCILL_REVERSE_POS_1/2 are values in the machine coordinate system; therefore comparison is only suitable with $AA_IM[n].

Program section

```
;   Example 1: Oscillation with synchronized actions
OSP1[Z]=10 OSP2[Z]=60     ; Define reversal points 1 and 2
OST1[Z]=–2 OST2[Z]=0      ; Reversal point 1: Without exact stop
                          ; Reversal point 2: Exact stop fine
FA[Z]=5000 FA[X]=250      ; Feedrate for oscillation axis, feedrate
                          ; for infeed axis
OSCTRL[Z]=(1+8+16,0)      ; Deactivate oscillation movem. at reversal point 1
                          ; Sparking-out after deletion of distance-to-go
                          ; and approach end position
                          ; Approach programmed reversal
                          ; point after deletion of distance-to-go
OSNSC[Z]=3                ; 3 sparking-out strokes
OSE[Z]=0                  ; End position = 0;
WAITP(Z)                  ; Enable oscillation for Z axis

;  Motion-synchronous actions.
;
; Whenever        the current position of the oscillation axis in the machine coordinate system
; is not equal to reversal position 1,
; then            set flag with index 1 to value 0 (reset flag 1)
WHENEVER $AA_IM[Z]<>$SA_OSCILL_REVERSE_POS1[Z] DO $AC_MARKER[1]=0
;
; Whenever        the current position of the oscillation axis in the machine coordinate system
; is lower than   the beginning of reversal point range 2 (here: Reversal point 2 –6),
; then            set the axial override of the infeed axis to 0%
; and             set flag with index 2 to value 0 (reset flag 2).
WHENEVER $AA_IM[Z]<$SA_OSCILL_REVERSE_POS2[Z]–6 DO $AA_OVR[X]=0 $AC_MARKER[2]=0
;
; Whenever        the current position of the oscillation axis in the machine coordinate system
; is the same as  reversal position 1,
; then            set the axial override of oscillation axis to 0%
; and             set the axial override of infeed axis to 100% (i.e. to cancel the preceding synchronized
                  action).
WHENEVER $AA_IM[Z]==$SA_OSCILL_REVERSE_POS1[Z] DO $AA_OVR[Z]=0 $AA_OVR[X]=100
;

; Whenever        the distance-to-go of the part infeed
; is equal to     0,
; then            set flag with index 2 to a value of 1
; and             sets flag with index 1 to a value of 1
```

```
WHENEVER $AA_DTEPW[X]==0 DO $AC_MARKER[2]=1 $AC_MARKER[1]=1
;
;Whenever        the flag with index 2
; is equal to    1,
; then           set the axial override of the infeed axis to 0% to prevent premature infeed
;                (oscillation axis has not yet exited from reversalposition 1).
WHENEVER $AC_MARKER[2]==1 DO $AA_OVR[X]=0
;
; Whenever       the flag with index 1
; is equal to    1,
; then           set the axial override of the infeed axis to 0% to prevent premature infeed
;                (oscillation axis has not yet exited from reversal point range 2)
; and            set axial override of oscillation axis to 100% ('start' oscillation).
WHENEVER $AC_MARKER[1]==1 DO $AA_OVR[X]=0 $AA_OVR[Z]=100
;

; If             the current position of the oscillation axis in the machine coordinate system
; is equal to    reversal position 1,
; then           set the axial override of the oscillation axis to 100%
; and            set the axial override of the infeed axis to 0% (in order
;                to cancel the second synchronized action once).
WHEN $AA_IM[Z]==$SA_OSCILL_REVERSE_POS1[Z] DO $AA_OVR[Z]=100 $AA_OVR[X]=0
;
;_____________________________________________
OSCILL[Z]=(X) POSP[X]=(5,1,1)   ; Assign axis X as infeed axis
                                ; to oscillation axis Z; axis X must
                                ; infeed to end position 5 in part steps of
                                ; 1 and the total of all part lengths
                                ; must correspond exactly to the end position
;
M30                             ; Program end
```

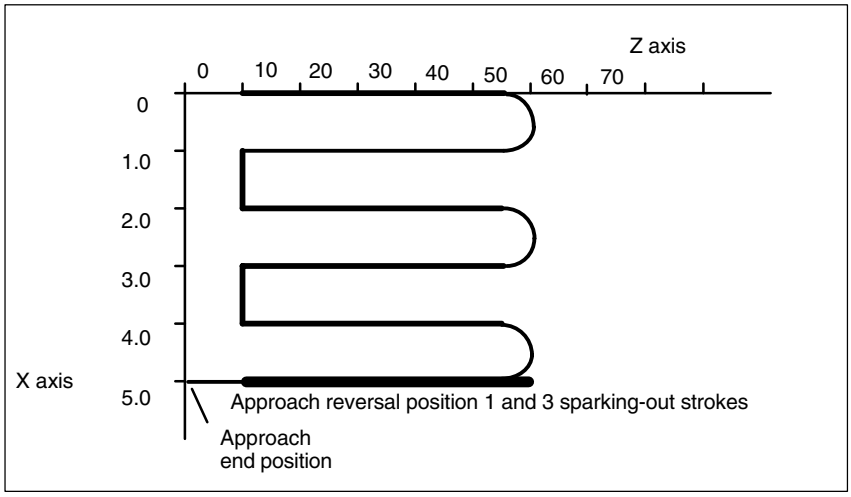


Fig. 6-2 Sequences of oscillation movements and infeed, example 1

6.3 Example 2 of oscillation with synchronized actions

Task

No infeed must take place at reversal point 1. At reversal point 2, the infeed must take place at distance ii2 from reversal point 2; the oscillation axis must wait at this reversal point until part infeed has been executed. Axis Z is the oscillation axis and axis X the infeed axis.

Program section

Example 2: Oscillation with synchronized actions

```
DEF  INT  ii2                ; Define variables for reversal point range 2
;
OSP1[Z]=10 OSP2[Z]=60        ; Define reversal points 1 and 2
OST1[Z]=0 OST2[Z]=0          ; Reversal point 1: Exact stop fine
                             ; Reversal point 2: Exact stop fine
FA[Z]=5000 FA[X]=100         ; Feedrate for oscillation axis, feedrate for infeed axis
OSCTRL[Z]=(2+8+16,1)         ; Deactivate oscillation movement at reversal point 2
                             ; After deletion of distance-to-go sparking-out and approach end position
                             ; After deletion of distance-to-go approach appropriate reversal position
OSNSC[Z]=3                   ; 3 sparking-out strokes
OSE[Z]=70                    ; End position = 70;
ii2=2                        ; Set reversal point range
WAITP(Z)                     ; Enable oscillation for Z axis

; Motion-synchronous actions:
; Whenever                   the current position of the oscillation axis in the machine coordinate system
; is lower than              the start of reversal point range 2,
; then                       set the axial override of the infeed axis to 0%
; and                        set the flag with index 0 to a value of 0
WHENEVER $AA_IM[Z]<$SA_OSCILL_REVERSE_POS2[Z]–ii2 DO $AA_OVR[X]=0 $AC_MARKER[0]=0
;
; Whenever                   the current position of the oscillation axis in the machine coordinate system
; is equal to or greater than reversal position 2,
; then                       set the axial override of the oscillation axis to 0%
WHENEVER $AA_IM[Z]>=$SA_OSCILL_REVERSE_POS2[Z] DO $AA_OVR[Z]=0
;
; Whenever                   the distance-to-go of the part infeed
; is equal to                0,
; then                       set the flag with index 0 to a value of 1
WHENEVER $AA_DTEPW[X]==0 DO $AC_MARKER[0]=1
;
; Whenever                   the flag with index 0
; is equal to                1,
; then                       set the axial override of the infeed axis to 0% in order to prevent premature
;                            infeed (oscillation axis has not yet exited from reversal point range 2,
;                            infeed axis is ready to infeed again)
; and                        set the axial override of the oscillation axis to 100% (thus cancelling
;                            the 2nd synchronized action)
WHENEVER $AC_MARKER[0]==1 DO $AA_OVR[X]=0 $AA_OVR[Z]=100
;
  OSCILL[Z]=(X) POSP[X]=(5,1,1)      ; Start axes
                                     ; Axis X is assigned to oscillation axis Z
                                     ; as the infeed axis
                                     ; Axis X must traverse to end position
                                     ; 5 in steps of 1
;
M30
```

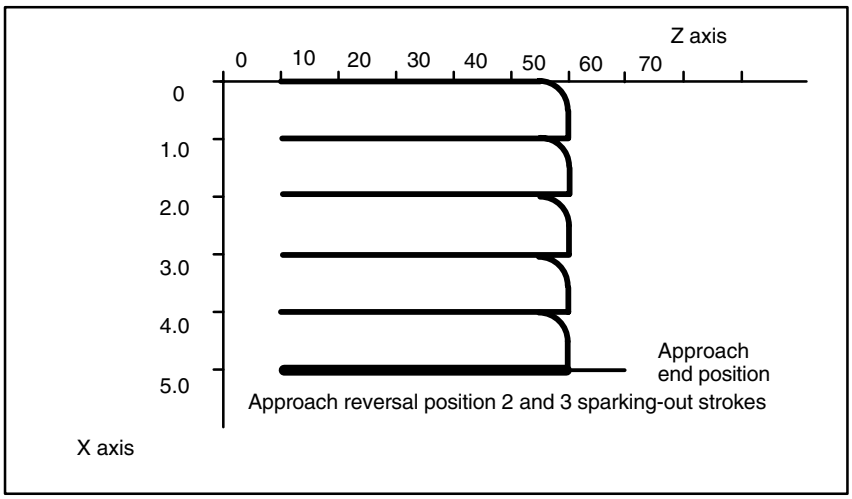


Fig. 6-3 Sequences of oscillation movements and infeed, example 2

■

Data Fields, Lists 7

7.1 Interface signals

DB number	Bit, byte	Name	Reference
Axis-specific/spindle-specific		**Signals to axis/spindle**	
31, ...	28.3	Set reversal point	
31, ...	28.4	Alter reversal point	
31, ...	28.5	Stop at next reversal point	
31, ...	28.6	Stop along braking ramp	
31, ...	28.7	PLC controls axis	
		Signals from axis/spindle	
31, ...	100.3	Oscillation cannot be started	
31, ...	100.4	Error during oscillation movement	
31, ...	100.5	Sparking-out active	
31, ...	100.6	Oscillation movement active	
31, ...	100.7	Oscillation active	

7.2 Machine data

Number	Identifier	Name	Reference
General ($MN_ ...)			
11460	OSCILL_MODE_MASK	Control screen form for asynchronous oscillation	

7.3 Setting data

Number	Identifier	Name	Refer-ence
Axis-specific ($SA_...)			
43700	OSCILL_REVERSE_POS1	Position at reversal point 1	
43710	OSCILL_REVERSE_POS2	Position at reversal point 2	
43720	OSCILL_DWELL_TIME1	Stop time at reversal point 1	
43730	OSCILL_DWELL_TIME2	Stop time at reversal point 2	
43740	OSCILL_VELO	Feed velocity of oscillation axis	
43750	OSCILL_NUM_SPARK_CYCLES	Number of sparking-out strokes	
43760	OSCILL_END_POS	Position after sparking-out strokes/at end of oscillation movement	
43770	OSCILL_CTRL_MASK	Control screen form for oscillation	
43780	OSCILL_IS_ACTIVE	Oscillation movement on/off	

7.4 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

7.5 Main run variables for motion-synchronous actions

The following variables are provided for main run variable_read, :

Main run variable_read:

I$A_IN[<arith. expression>**]**	Digital input (Boolean)
I$A_OUT[<arith. expression>**]**	Digital output (Boolean)
I$A_INA[<arith. expression>**]**	Analog input (Boolean)
I$A_OUTA[<arith. expression>**]**	Analog output (Boolean)
I$A_INCO[<arith. expression>**]**	Comparator inputs (Boolean)
I$AA_IW[<axial expression>**]**	Actual position, axis PCS (Real)
I$AA_IB[<axial expression>**]**	Actual position, axis BCS (Real)
I$AA_IM[<axial expression>**]**	Actual position, axis MCS (IPO setpoints) (Real) With $AA_IM[S1], actual values for spindles can be evaluated. For spindles and rotary axis modulo calculation applies depending on the machine data $MA_ROT_IS_MODULO and $MA_DISPLAY_IS_MODULO modulo calculation.
I$AC_TIME	Time from block start (Real) in s
I$AC_TIMEC	Time from block start (Real) in IPO cycles
I$AC_DTBB	Distance to block start in BCS (distance to begin, baseCoor) (Real)
I$AC_DTBW	Distance to block start in PCS (distance to begin, workpieceCoor) (Real)
I$AA_DTBB[<axial expression>**]**	Axial path to block start in BCS (distance to begin, baseCoor) (Real)
I$AA_DTBW[<axial expression>**]**	Axial path to block start in PCS (distance to begin, workpieceCoor) (Real)
I$AC_DTEB	Distance to block end in BCS (distance to end) (distance to end, baseCoor) (Real)
I$AC_DTEW	Distance to block end in PCS (distance to end, workpieceCoor) (Real)
I$AA_DTEB[<axial expression>**]**	Axial path to end of movement in BCS (distance to begin, baseCoor) (Real)
I$AA_DTEW[<axial expression>**]**	Axial path to end of movement in PCS (distance to end, workpieceCoor) (Real)
I$AC_PLTBB	Path length from block start in BCS (path length from begin, baseCoor) (Real)
I$AC_PLTEB	Path length to block end in BCS (distance to end) (path length to end, baseCoor) (Real)
I$AC_VACTB	Path velocity in BCS (velocity actual, baseCoor) (Real)
I$AC_VACTW	Path velocity in PCS (velocity actual, workPieceCoor) (Real)
I$AA_VACTB[<axial expression>**]**	Axis velocity in BCS (velocity actual, baseCoor) (Real)
I$AA_VACTW[<axial expression>**]**	Axis velocity in PCS (velocity actual, workPieceCoor) (Real)
I$AA_DTEPB[<axial expression>**]**	Axial distance-to-go for infeed, oscillation in BCS (distance to end, pendulum,baseCoor) (Real)
I$AA_DTEPW[<axial expression>**]**	Axial distance-to-go for infeed, oscillation PCS (distance to end, pendulum, workpieceCoor) (Real)
I$AC_DTEPB	Remaining path for infeed, oscillation in BCS

(not P2) (distance to end, pendulum,baseCoor)
(Real)

I$AC_DTEPW Remaining path for infeed, oscillation in PCS
(not P2)
(distance to end, pendulum, workpieceCoor) (Real)

I$AC_PATHN (path parameter normalized)(Real) Normalized path parameter: 0 for block start, up to 1 for block end

I$AA_LOAD[<axial expression>**]** Drive load (for 611D only)

I$AA_POWER[<axial expression>**]** Real drive output in W (for 611D only)

I$AA_TORQUE[<axial expression>**]** Drive torque setpoint in Nm (for 611D only)

I$AA_CURR[<axial expression>**]** Current value of axis (for 611D only)

I$AC_MARKER[<arithmetic expression>**]** (int)
Marker variable: can be used in synchronized actions for creating complex conditions
There are 8 markers (index 0–7).
With reset, the markers are set to 0.
Ex.: WHEN .....DO $AC_MARKER[0]=2
WHEN .....DO $AC_MARKER[0]=3
WHEN $AC_MARKER[0]==3 DO $AC_OVR=50
It is possible to read or overwrite the markers independently of synchronized actions in the part program:
IF $AC_MARKER == 4 GOTOF SPRUNG

I$AC_PARAM[<arithmetic expression>**]** (Real)
Floating point parameter for synchronized actions.
Serves for buffering and evaluation of synchronized actions.
There are 50 parameters (index 0–49) available.

I$AA_OSCILL_REVERSE_POS1[<axial expression>**]** (Real)
I$AA_OSCILL_REVERSE_POS2[<axial expression>**]** (Real)
Current reverse positions 1 and 2 for oscillation:
In each case, the current setting data value is read from $SA_OSCILL_REVERSE_POS1 or $SA_OSCILL_REVERSE_POS2.
Changes to the reversal positions in in the setting data thus become effective when oscillation is active, i.e. during an active synchronized action.

Conditions

Conditions for motion-synchronous actions are formulated:

Main run variable Relation operator Expression
For details, please refer to:

References: /FB/, S5, “Synchronized Actions”

■

SINUMERIK 840D/810D/FM-NC Description of Functions, Extension Functions (FB2)

Rotary Axes (R2)

■

Notes

Brief Description

1

Rotary axes in machine tools

Rotary axes are used on many modern machine tools. They are required for tool and workpiece orientation, auxiliary movements and various other technological or kinematic purposes.

A typical example of a machine tool requiring the use of rotary axes is the 5-axis milling machine. Only with the aid of rotary axes can the tip of the tool be positioned on any point of the workpiece on this type of machine.

Depending on the type of machine, many different demands are placed on a rotary axis. In order that the control can be adapted to the various types of machine, the individual rotary axis functions can be activated by means of machine data or special programming.

Rotary axes are always programmed in degrees. They are generally characterized by the fact that they assume the same position again after exactly one rotation (modulo 360°). Depending on the application in question, the traversing range of the rotary axis can be limited to less than 360° (e.g. on swivel axes for tool holders) or may be endless (e.g. when tool or workpiece is rotated).

The behavior and features of rotary axes are, in many aspects, identical to those of linear axes. The following functional description is limited to a description of the special features of rotary axes and the differences compared with linear axes.

■

Notes

Detailed Description 2

2.1 General

Definition of rotary axis

An axis can be declared as a rotary axis by means of machine data IS_ROT_AX. If the axis is already defined as a geometry axis, alarm 4200 is output. Only when an axis has been declared as a rotary axis can it perform or use the functions described on the following pages (e.g. unlimited traversing range, modulo display of axis position, etc.). Several axes can be simultaneously declared as rotary axes.

Types of rotary axis

Depending on the particular application, the operating range of a rotary axis can be endless (i.e. endlessly turning in both directions MD: ROT_IS_MODULO = 1) or limited by software limit switches (e.g. operating range between 0 ... 60°) or restricted to a specific number of rotations (e.g. 1000°).

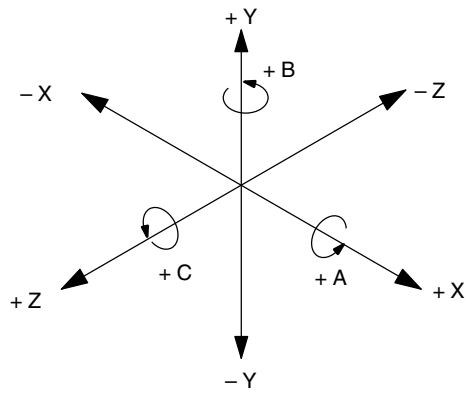
The following list presents some typical applications of rotary axes:

Typical applications

- 5-axis machining (operating range limited or unlimited)
- Rotary axis for eccentric machining (unlimited operating range)
- Rotary axis for cylindrical or contour grinding (unlimited operating range)
- C axis with TRANSMIT (unlimited operating range)
- Rotary axis on winding machines (unlimited operating range)
- Rotary workpiece axis (C) on hobbing machines (unlimited operating range)
- Round tool magazines and tool turrets (unlimited operating range)
- Rotary axis for peripheral surface transformation (limited operating range)
- Swivel axes for gripping (operating range 360°)
- Rotary axes for swivelling (operating range < 360°; e.g. 60°)
- Milling swivel axis (A) on hobbing machines (operating range e.g. 90°)

Axis addresses

Coordinate axes and directions of movement of numerically controlled machine tools are designated according to DIN. DIN 66025 specifies the following axis addresses for rotary or swivel axes: A, B and C with X, Y and Z as middle axis; i.e. A rotates around X, B rotates around Y and C rotates around Z (see diagram below). The positive direction of a rotary axis corresponds to a movement to the right looking in the positive axis direction of the corresponding middle axis.



Cartesian coordinate system

Fig. 2-1 Identification of axes and directions of movements for rotary axes

Extended addressing (e.g. C2=) or freely configured axis addresses can be used for additional rotary axes.

Note

If the X1, Y1 or Z1 axis is declared to be a rotary axis, then the operating modes cannot be selected or changed.
MD 20050: AXCONF_GEOAX_ASSIGN_TAB (assignment of geometry axis to channel axis) must be adapted to suit the corresponding axis.

Units of measurement

The following units of measurement apply as standard to data inputs and outputs for rotary axes:

Table 2-1 Units of measurement for rotary axes

Physical quantity	Unit
Angular position	degrees
Programmed angular speed	Degrees/minute
MD for angular speed	Rev/min [1]
MD for angular acceleration	Rev/sec^2 [1]
MD for angular smoothing	Rev/sec^3 [1]

[1] These units are interpreted by the control with the axis-specific machine data as soon as the axis is declared as a rotary axis.
The user has the option of defining other units for data input/output using machine data.

References: /FB/, G2, “Velocities, Setpoint/Actual-Value Systems, Closed-Loop Control”

Operating range

The axis operating range can be defined by means of axis-specific machine and setting data (software limit switches and working area limitations). As soon as the modulo conversion is activated for the rotary axis (MD: ROT_IS_MODULO = “1”), the operating range is unlimited and the software limit switches and working area limitations are inactive.

Position display

The value range for the position display can be set to the modulo 360° representation that is frequently selected for rotary axes (MD:DISPLAY_IS_MODULO = “1”).

Feedrate

The programmed feedrate F corresponds to an angular speed [degrees/min] in the case of rotary axes.

If rotary axes and linear axes traverse along a common path with G94 or G95, the feed should be interpreted in the unit of measurement of the linear axes [e.g. mm/min, inch/min].

The tangential speed of the rotary axis refers to the diameter D_E (unit diameter D_E=360/π). In the case of unit diameter D=D_E, the programmed angular speed in degrees/min and the tangential velocity in mm/min (or inch/min) are numerically identical.

The following applies generally for the tangential speed:

$F = F_W * D / D_E$ F = Tangential speed [mm/min]
F_W = Angular speed [degrees/min]
D = Diameter at which F effective [mm]

where $D_E = 360 / \pi$ D_E = Unit diameter [mm]
π = Circle constant Pi

Revolutional feedrate

In JOG mode the behavior of the axis/spindle also depends on the setting of setting data JOG_REV_IS_ACTIVE (revolutional feedrate when JOG active).

- If this setting data is active, an axis/spindle is always moved with revolutional feedrate MD JOG_REV_VELO (revolutional feedrate with JOG) or MD JOG_REV_VELO_RAPID (revolutional feedrate with JOG with rapid traverse overlay) depending on the master spindle.
- If the setting data is not active, the behavior of the axis/spindle depends on the setting data ASSIGN_FEED_PER_REV_SOURCE (revolutional feedrate for positioning axes/spindles)
- If the setting data is not active, the behavior of a geometry axis on which a frame with rotation is effective depends on the channel-specific setting data JOG_FEED_PER_REV_SOURCE. (In JOG mode, revolutional feedrate for geometry axes on which a frame with rotation is effective).

2.2 Modulo 360°

Term Modulo 360°

Rotary axes are frequently programmed in the 360° representation mode. The axis must be defined as a rotary axis in order to use the modulo feature.

With respect to a rotary axis, the term “Modulo” refers to imaging of the axis position internally in the control within the range from 0° to 359.999°. With path settings > 360° (e.g. for incremental dimension programming using G91), the axis position is imaged in the value range between 0° to 360° through a conversion process in the control. The imaging process is applied in JOG and AUTOMATIC mode. The service display is an exception.

In the diagram below, the absolute position of the rotary axis in the positive direction of rotation is represented as a spiral. An arrow marks the actual absolute position (Example: Point C‘ = 420°). By sliding the arrow back around the circle (position 0° of the spiral and circle are identical), it is possible to determine a modulo position within the 360° range corresponding to every absolute position. In the example below, absolute position point C’=420° is mapped onto point C = 60° through the modulo conversion process.

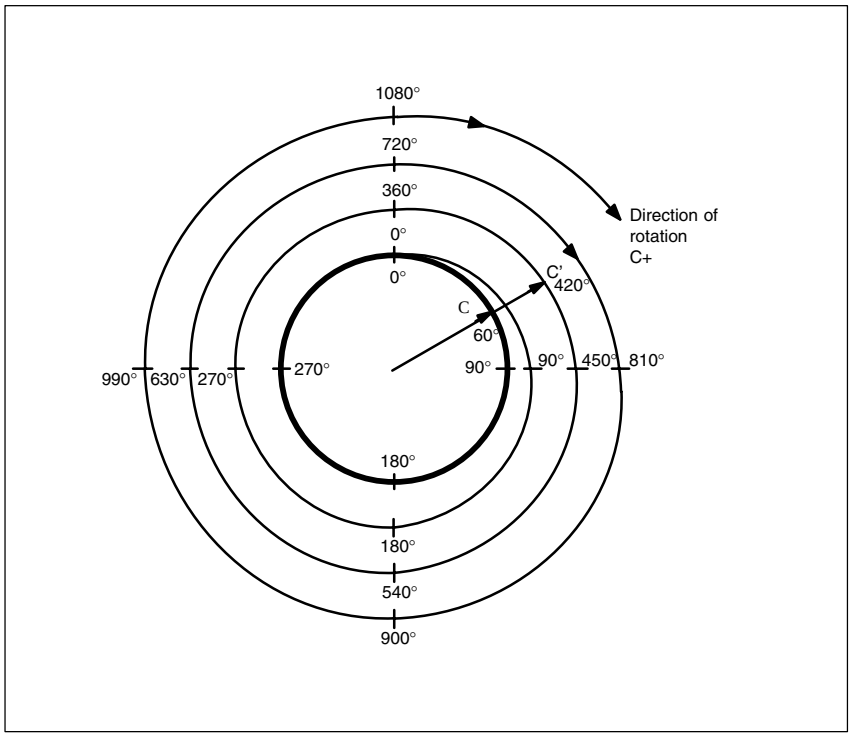


Fig. 2-2 Modulo 360° imaging

Machine data settings

Using machine data it is possible to define the programming and positioning settings (MD: ROT_IS_MODULO) and the position display (MD DISPLAY_IS_MODULO) in modulo 360° for each individual rotary axis to suit the requirements of individual machine tools.

Axis is modulo

MD: ROT_IS_MODULO = “1”:

Activation of this machine data allows the special rotary axis action implemented in the system to be utilized (see Section 2.3.1), defining the positioning action of the rotary axis for programming (G90, AC, ACP, ACN or DC). A modulo 360° imaging process is executed internally in the control after the current zero offsets have been taken into account. The calculated destination position is subsequently approached **within a single revolution**. The software limit switches and the working area limitations **are ineffective** and the operating range is unlimited (endlessly turning rotary axis).
Please see Section 2.3 on the programming of rotary axes or MD: ROT_IS_MODULO for further information.

Modulo position display

MD: DISPLAY_IS_MODULO = 1:
A “modulo 360°” (1 rotation) position display is frequently required for rotary axes, i.e. when the axis is rotating in the positive direction, the display is periodically reset from 359.999° to 0.000° in the control system; with a negative direction of rotation, the axis positions are also displayed in the 0°...359.999° range.
MD: DISPLAY_IS_MODULO = 0:
In contrast to the modulo 360° display method, absolute positions are indicated by the absolute position display, e.g. +360°, after 1 rotation and +720° after 2 rotations, etc. In this case, the display range is limited by the control in accordance with the linear axes.

Note

The modulo 360° display method should always be selected for a modulo axis (ROT_IS_MODULO = “1”).

2.3 Programming of rotary axes

Note

For general information on programming, please refer to:
References: /PAG/Programming Guide Fundamentals

General

The machine data ROT_IS_MODULO (modulo conversion for rotary axis) defines whether the rotary axis behaves in the same way as a linear axis during programming and positioning or whether the special features of the rotary axis are incorporated. These features and the differences (mainly with respect to absolute dimension programming) are explained on the following pages.

2.3.1 Rotary axis with active modulo conversion (endlessly turning rotary axis)

Activate modulo conversion

⇒ MD: ROT_IS_MODULO = “1”
Recommendation: it is also advisable to set the position display to modulo 3605 (MD: DISPLAY_IS_MODULO = “1”).

Absolute dimension programming (AC, ACP, ACN, G90)

Example of positioning axis: **POS[axis name] = ACP(value)**

- The value identifies the target position of the rotary axis within the 0 ... 359.999° range. In the case of values with a negative sign or ≧ 360° alarm 16830 “Incorrect modulo position programmed” is output.
- ACP (positive) and ACN (negative) define the traversing direction of the rotary axis unambiguously (irrespective of actual position).
- When programming exclusively with AC or with G90, the traversing direction depends on the actual position of the rotary axis. If the destination position is larger than the actual position, the axis traverses in the positive direction, otherwise it traverses in the negative direction.
- Use of ACP and ACN: with asymmetrical workpieces, it must be possible to define the traversing direction to prevent collisions during rotation.

Example:
(see diagram below): Start position of C is 0˚

	Programming	Effect
1	POS[C] = ACP(100)	Rotary axis C traverses in the positive rotational direction to position 100˚
2	POS[C] = ACN(300)	C traverses in the negative rotational direction to position 300˚
3	POS[C] = ACP(240)	C traverses in the positive rotational direction to position 240˚
4	POS[C] = AC (0)	C traverses in the negative rotational direction to position at 0˚

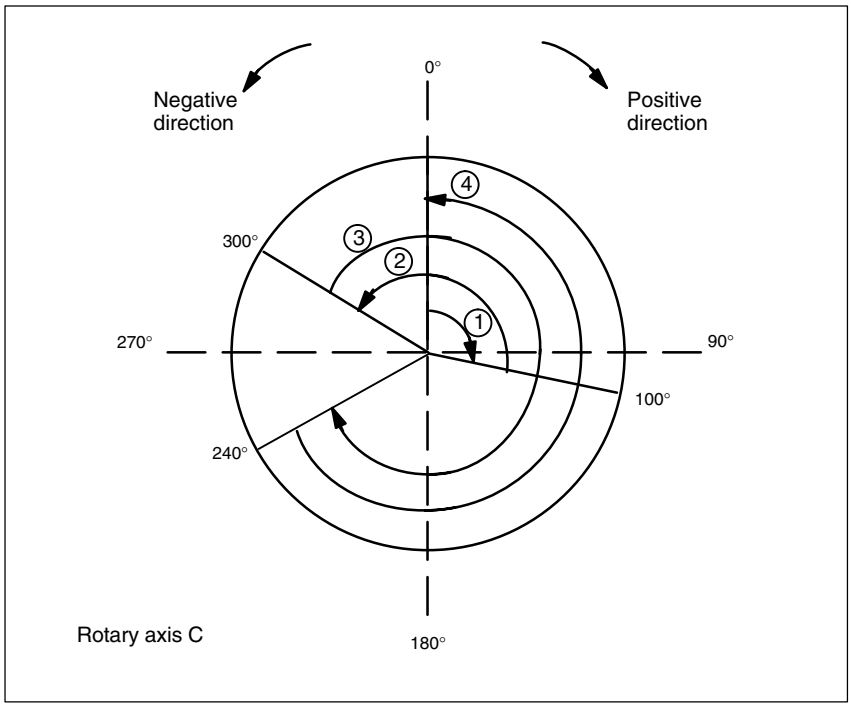


Fig. 2-3 Examples of absolute dimension programming for modulo axes

Absolute dimension programming via shortest route (DC)

POS[axis name] = DC(value)

- The value identifies the destination position of the rotary axis in a range from 0 to 359.999˚. For values with a negative sign or ≥ 360˚ alarm 16830 “incorrect modulo position programmed” is output.
- With DC (Direct Control), the rotary axis approaches the programmed absolute position via the **shortest route** within one revolution (traversing movement max. ±180˚).
- The control calculates the direction of rotation and the traversing distance according to the actual position. If the distance to be traversed is the same in both directions (180˚), the positive direction receives preference.

– Example application of DC: the rotary table is required to approach the changeover position in the shortest time (and therefore across the shortest path).

– If DC is programmed with a linear axis, alarm 16800 “DC traversing instruction cannot be used”.

Example:
(see diagram below): Start position of C is 0º

	Programming	Effect
1	POS[C] = DC(100)	C axis traverses along the shortest path to position 100˚
2	POS[C] = DC(300)	C axis traverses along the shortest path to position 300˚
3	POS[C] = DC(240)	C axis traverses along the shortest path to position 240˚
4	POS[C] = DC (60)	C axis traverses along the shortest path to position 60˚. Since the distance in this case is equal to 180˚ in both directions, the positive direction has preference.

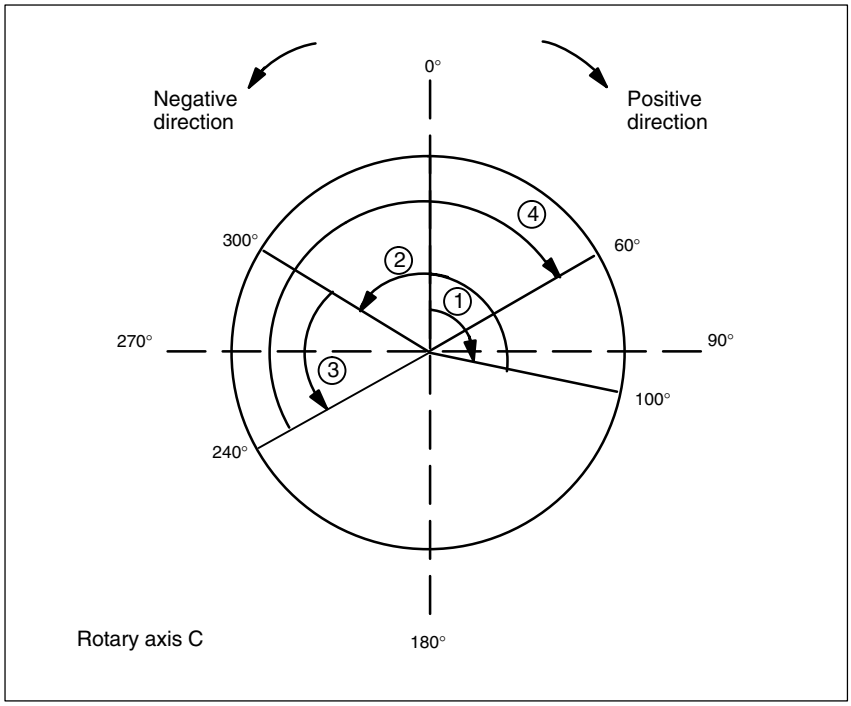


Fig. 2-4 Examples of DC programming

Incremental dimension programming (IC, G91)

Example of positioning axis: **POS[axis name] = IC(+/–value)**

– The value identifies the traversing distance of the rotary axis. The value can be negative or ≥ +/–360˚.

– The **sign** of the value mandatorily specifies the **traversing direction** of the rotary axis.

– Example application: milling a spiral groove across several revolutions

Example:

POS[C] = IC(720)	C axis traverses incrementally through 720° (2 revolutions) in the positive direction
POS[C] = IC(–180)	C axis traverses incrementally through 180° in the negative direction

Endless traversing range

As soon as the modulo function is active, no limit is placed on the traversing range (software limit switches are not active). The rotary axis can now be programmed to traverse continuously.

Example:

```
LOOP:
POS[C] = IC(720)
GOTOB LOOP
```

2.3.2 Rotary axis without modulo conversion

Deactivate modulo conversion

⇒ MD:ROT_IS_MODULO = “0”

Absolute dimension programming (AC, G90)

Example of positioning axis: **POS[axis name] = AC (+/–value)**

- The value and its leading sign provide a unique identification of the destination position of the rotary axis. The value can also be ≥ +/–360°. The position value is limited by the software limit switch positions.
- The traversing direction is ascertained by the control according to the leading sign of the actual position of the rotary axis.
- If ACP or ACN are programmed, alarm 16810 “ACP traversing instruction cannot be used” or alarm 16820 “ACN traversing instruction cannot be used” is output.
- Example application: linear movements are incorporated in the rotary axis (cam gear); certain end positions may therefore not be overtraveled.

Example:

Programming	Effect
POS[C] = AC (–100)	Rotary axis C approaches position –100°; the traversing direction depends on the starting position
POS[C] = AC (1500)	Rotary axis C traverses to the position at 1500°

Absolute dimension programming across the shortest path (DC)

POS[axis name] = DC(value)

Even if the rotary axis is not defined as a modulo axis, the axis can still be positioned with DC (direct control). The response is the same as on a modulo axis.

- The value identifies the destination position of the rotary axis **in a range from 0 to 359.999° (modulo 360°).** In the case of values with a negative sign or of ≥ 360º, alarm 16830 “Incorrect modulo position programmed” is output.
- With DC (Direct Control), the rotary axis approaches the programmed absolute position via the **shortest route** within one revolution (traversing movement max. ±180°).
- The control calculates the direction of rotation and the traversing distance according to the actual position (in relation to modulo 360°). If the distance to be traversed is the same in both directions (180°), the positive direction receives preference.
- Example application of DC: the rotary table is required to approach the changeover position in the shortest time (and therefore across the shortest path).
- If DC is programmed with a linear axis, alarm 16800 “DC traversing instruction cannot be used”.

Example:

Programming	Effect
POS[C] = AC (7200)	Rotary axis C traverses to position 7200°; the traversing direction depends on the starting position
POS[C] = DC (300)	Rotary axis C traverses along the shortest path to the “modulo” position at 300°. C therefore traverses through 60° in the negative direction of rotation and stops at the absolute position at 7140°.
POS[C] = AC (7000)	Rotary axis C traverses to the absolute position at 7000°; here, C traverses through140° in the negative direction of rotation

Note: In this example, it is advisable to activate the modulo 360° display (MD: DISPLAY_IS_MODULO = “1”).

Incremental dimension programming (IC, G91)

Example of positioning axis: **POS[axis name] = IC(+/–value)**

When programming with incremental dimension, the rotary traverses across the same path as with the modulo axis. In this case, however, the traversing range is limited by the software limit switches.

- The value identifies the traversing distance of the rotary axis. The value can be negative or ≥ +/–360°.
- The **leading sign** of the value defines the **traversing direction**
  Please see section 2.3.2 for an example.

Limited traversing range

The traversing range is limited as with linear axes. The range limits are defined by the plus and minus software limit switches.

2.3.3 Miscellaneous programming features relating to rotary axes

Offsets

TRANS (absolute) and ATRANS (additive) can be applied to rotary axes.

Scales

SCALE or ASCALE are not suitable for rotary axes since the control system always bases its modulo calculation on a 360° full circle.

Set actual value

PRESETON is possible.

Indexing axes

References: /FB/, T1, “Indexing Axes”

2.4 Start-up of rotary axes

Procedure

The procedure for starting up rotary axes is identical to that for linear axes with a small number of exceptions. It should be noted that the units of the axis-specific machine and setting data on the control are interpreted as follows as soon as the axis has been defined as a rotary axis (MD: IS_ROT_AX = 1):

Position	in “degrees”
Speed	in “rev/minute”
Acceleration	in “rev/second^2”
Smoothing	in “rev/second^3”

Special MDs

The special machine data of the rotary axis described in Section 4 must also be entered depending on the application.

- MD: ROT_IS_MODULO — Modulo conversion for positioning and programming
- MD: DISPLAY_IS_MODULO — Modulo conversion for position display
- MD: INT_INCR_PER_DEG
  Precision of angular position calculation

The following overview lists the possible combinations of these machine data for a rotary axis.

Table 2-2 Possibilities for combining machine data of rotary axes

MD: IS_ROT_AX “rotary axis”	**MD: ROT_IS_MODULO “modulo conversion for rotary axis”**	**MD: DISPLAY_IS _MODULO “modulo actual value display”**	**Application permitted**	**Remark**
0	0	0	yes	The axis is a linear axis (standard case)
0	0 or 1	0 or 1	not recom-mended	The axis is not a rotary axis; the signal status of the other machine data is there-fore irrelevant
1	0	0	yes	The axis is a rotary axis; modulo conversion is not used for positioning, i.e. the software limit switches are active; the position display is absolute
1	1	0	yes (not recom-mended)	The axis is a rotary axis; positioning is performed with modulo conversion, i.e. the software limit switches are inactive, the operating range is unlimited; the position display is absolute

Table 2-2 Possibilities for combining machine data of rotary axes

MD: IS_ROT_AX "rotary axis"	MD: ROT_IS_MODULO "modulo conversion for rotary axis"	MD: DISPLAY_IS _MODULO "modulo actual value display"	Application permitted	Remark
1	1	1	yes	The axis is a rotary axis; positioning is performed with modulo conversion, i.e. the software limit switches are inactive, the operating range is unlimited; the position display is modulo (**setting most frequently used for rotary axes**)
1	0	1	yes	The axis is a rotary axis; modulo conversion is not used for positioning, i.e. the software limit switches are active; the position display is modulo; application: e.g. for axes with an operating range of +/–10005

JOG velocity for rotary axes

A jog velocity that is valid for all rotary axes can be set with SD: JOG_ROT_AX_SET_VELO (JOG velocity for rotary axes).

If a value of "0" is entered in the setting data, then the axis MD: JOG_VELO (conventional axis velocity) acts as the JOG velocity for the rotary axis.

References: /FB/, H1, "Manual and Handwheel Travel"

2.5 Special features of rotary axes

Software limit switches

The software limit switches and working area limitations are operative and are required for swivel axes with a restricted operating range. For endlessly turning rotary axes with (MD: ROT_IS_MODULO=1), however, the software limit switches and working area limitations are set inactive.

References: /FB/, A3, “Axis Monitoring”

Mirroring of rotary axes

Mirroring can be implemented for rotary axes with programming commands MIRROR(C) and AMIRROR(C).

Reference point approach

References: /FB/, R1, “Reference Point Approach”

Spindles as rotary axes

For notes concerning the use of spindles and rotary axes (C axis operation), please refer to:

References: /FB/, S1, “Spindles”

■

Notes

Supplementary Conditions 3

There are no supplementary conditions stipulated for this Description of Functions.

Data Descriptions (MD, SD) 4

4.1 Axis/spindle-specific machine data

30320 MD number	**DISPLAY_IS_MODULO** Position display is modulo 360°		
Default value: 1	Min. input limit: 0	Max. input limit: 1	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: BOOLEAN	Applies from SW version: 1.1		
Significance:	1: "Modulo 360º" position display is active: The position display of the rotary axis or spindle (for basic or machine coordinate system) is defined as "Modulo 360°". The control resets the position display internally to 0.000 degrees following each cycle of 359.999 degrees. The display range is always positive and always between 0° and 359.999°. 0: Absolute position display is active: In contrast to the modulo 360° display method, absolute positions are indicated by the absolute position display, e.g. +360° after 1 rotation and +720° after 2 rotations, etc. In this case, the display range is limited by the control in accordance with the linear axes.		
MD irrelevant for	Linear axes MD: IS_ROT_AX = "0"		
Application	● With continuously rotating axes (MD: ROT_IS_MODULO = "1") it is advisable to activate the position display with modulo 360° . ● The position display for spindles must always be activated with modulo 360° .		
Related to	MD: IS_ROT_AX = 1 "axis is rotary axis"		

34220 MD number	**ENC_ABS_TURNS_MODULO[n]** Absolute encoder range for rotary encoders: 0 ... max. no. encoders –1		
Default value: 4096, 4096	Min. input limit: 1	Max. input limit: 4096	
Changes effective after Power On	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies from SW version: 2.2		
Significance:	The absolute position of a rotary axis is reduced to the following range after switching on an absolute encoder: That is, if a MODULO conversion is performed, the actual position read is greater than the position allowed by the MD ENC_ABS_TURNS_MOTOR. 0 degrees <= position <= n*360 degrees, (with n = ENC_ABS_TURNS_MODULO) **Note:** In SW 2.2., the position is reduced to this range when the control system/encoder is switched on. With SW 3.6 and higher, half this value represents the maximum permissible traversing path when the control or encoder is switched off.		
Special cases, errors, ...	Only powers of two are allowed as values (1, 2, 4, 8, 16, ..., 4096) If other values are entered, then in the software versions up to 4.1 they are “rounded off” without a message being issued. In SW 4.1 and higher, rounding off is performed and shown in the machine data; the change is displayed in alarm 26025. **The MD is only relevant for rotary encoders (with linear and rotary axes).** **Important recommendation:** The default value “1 encoder revolution” was changed to “4096” from software version 3.6. The new value provides a more robust setting for the most commonly used encoder types. When using an encoder with smaller multi-turn information, or when using single-turn encoders, the value must be decreased accordingly. In any case, for multi-turn absolute encoders the value should be changed to the maximum quantity supported by the encoder so that the unambiguous traversing range that is increased as a result can be utilized (Note: This value also influences the permissible position offset when the encoder or power supply is switched off).		
Related to	Drive MD 1021, ENC_ABS_TURNS_MOTOR, Drive MD 1031, ENC_ABS_TURNS_DIRECT		

30300 MD number	**IS_ROT_AX** Rotary axis		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: BOOLEAN	Applies from SW version: 1.1		
Significance:	1: Axis: The axis is defined as a “rotary axis” • The special functions of the rotary axis are active or can be activated by means of additional machine data according to the type of machine required(see below). • The unit of measurement is degrees. • The units of the axis-specific machine and setting data are interpreted as follows with the standard control setting: • Position in “degrees” • Speed in “rev/minute” • Acceleration in “rev/second^2” • Smoothing in “rev/second3” Spindle: The machine data should always be set to “1” for a spindle, otherwise alarm 4210 “rotary axis declaration missing” is output. 0: The axis is defined as a “linear axis”.		
Special cases, errors, ...	for axis: alarm 4200 if the axis is already defined as a geometry axis. for spindle: alarm 4210		
Related to	The following machine data are effective only after activation of MD:IS_ROT_AX_ = “1”: • MD: ROT_IS_MODULO “Modulo conversion for rotary axis” • MD: DISPLAY_IS_MODULO “Position display is modulo” • MD: INT_INCR_PER_DEG “Calculation precision for angle positions”		
References	Table 2.2 Possible combinations of machine data		

30310 MD number	**ROT_IS_MODULO** Modulo conversion for rotary axis		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: BOOLEAN	Applies from SW version: 1.1		
Significance:	1: A modulo conversion is performed on the setpoints for the rotary axis. The software limit switches and the working area limitations are **ineffective**; the traversing range is therefore unlimited in both directions. MD: ROT_IS_AX must be set to “1” See Section 2.2 for further information 0: No modulo conversion		
MD irrelevant for	MD: IS_ROT_AX = “0” (linear axes)		
Table 2.2	Possible combinations of machine data		
Application	Continuously rotating axes (e.g. for eccentric rotation, grinding, winding)		
Related to	MD: DISPLAY_IS_MODULO MD: IS_ROT_AX = 1 MD: POS_LIMIT_MINUS MD: POS_LIMIT_PLUS SD: WORKAREA_LIMIT_MINUS SD: WORKAREA_LIMIT_PLUS	“Position display is modulo 360°” “Rotary axis” “Software limit switch minus” “Software limit switch plus” “Working area limitation minus” “Working area limitation plus”	

■

Notes

Signal Descriptions 5

No separate signals exist for this Description of Functions.

Example 6

Fork head, inclined axis head

Rotary axes are frequently used on 5-axis milling machines to swivel the tool axis or rotate the workpiece. These machines can position the tip of a tool on any point of the workpiece and take up any position on the tool axis. Various milling heads are required according to the application. Fig. 6-1 illustrates a fork head and an inclined axis head as example arrangements for rotary axes.

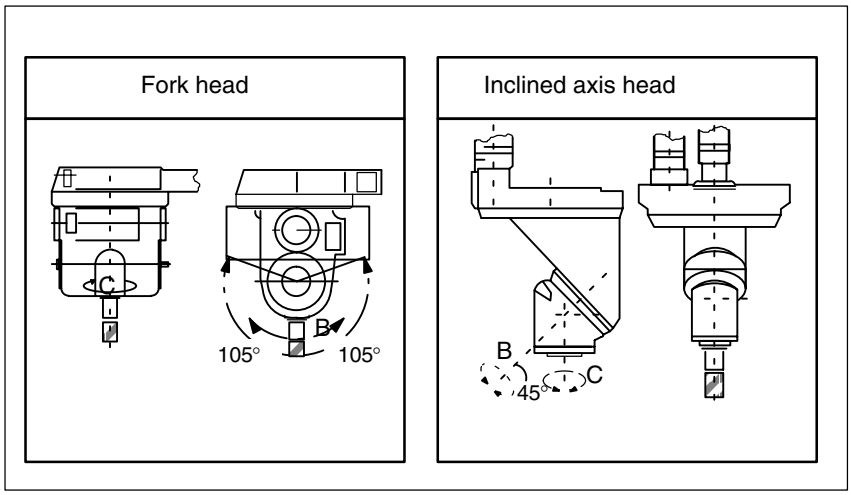


Fig. 6-1 Fork head, inclined axis head

Notes

Data Fields, Lists 7

7.1 Interface signals

– None –

7.2 Machine data

Number	Identifier	Name	Reference
General ($MN_ ...)			
10210	INT_INCR_PER_DEG	Calculation precision for angular positions	G2
Axis/spindle-specific ($MA_ ...)			
30320	DISPLAY_IS_MODULO	Modulo actual value display	
30300	IS_ROT_AX	Axis is rotary axis	
36100	POS_LIMIT_MINUS	Software limit switch minus	A3
36110	POS_LIMIT_PLUS	Software limit switch plus	A3
30310	ROT_IS_MODULO	Modulo conversion for rotary axis	

7.3 Setting data

Number	Identifier	Name	Reference
General ($MN_ ...)			
41130	JOG_ROT_AX_SET_VELO	JOG velocity for rotary axes	H1
Axis-specific ($SA_...)			
43430	WORKAREA_LIMIT_MINUS	Working area limitation minus	A3
43420	WORKAREA_LIMIT_PLUS	Working area limitation plus	A3

7.4 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

■

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

Synchronous Spindle (S3)

■

Brief Description 1

Synchronous spindle

This function (see Chapter 3) enables synchronization between a leading and following spindle with high angle accuracy.

It also offers the option of on-the-fly transfer of the workpiece from spindle 1 to spindle 2 during operation on turning machines for the purpose, for example, of final machining. Advantage: Avoidance of downtimes.

Apart from synchronizing the spindle speed, it is also possible to specify the relative angular position of the spindles in relation to one another, e.g. for on-the-fly, position-oriented transfer of workpieces.

On-the-fly transfer of workpieces between leading spindle (LS) and following spindle (FS):

- $n_{FS} = n_{LS}$ Speed synchronism
- $\varphi_{FS} = \varphi_{LS}$ Position synchronism

or

- $\varphi_{FS} = \varphi_{LS} + \Delta\varphi$ Position synchronism with angular offset

Polygonal machining

As an additional feature, specification of an integer, multiple speed ratio $k_Ü$ between the main spindle and a “tool spindle” provides the basis for polygonal machining (polygonal turning).

Polygonal turning:

- $n_{FS} = k_Ü \cdot n_{LS}$

Synchronous operation is selected and deselected via the CNC part program.

The synchronous spindle pairs for each machine can be assigned a fixed configuration by means of channel-specific machine data or defined for specific applications via the CNC part program.

Up to 2 synchronous spindle pairs can be operated in each NC channel.

SW 5 and higher

Any number of following spindles in any channels on an NCU can be coupled to one leading spindle. The only possible restriction could be imposed by the real CPU time requirement.

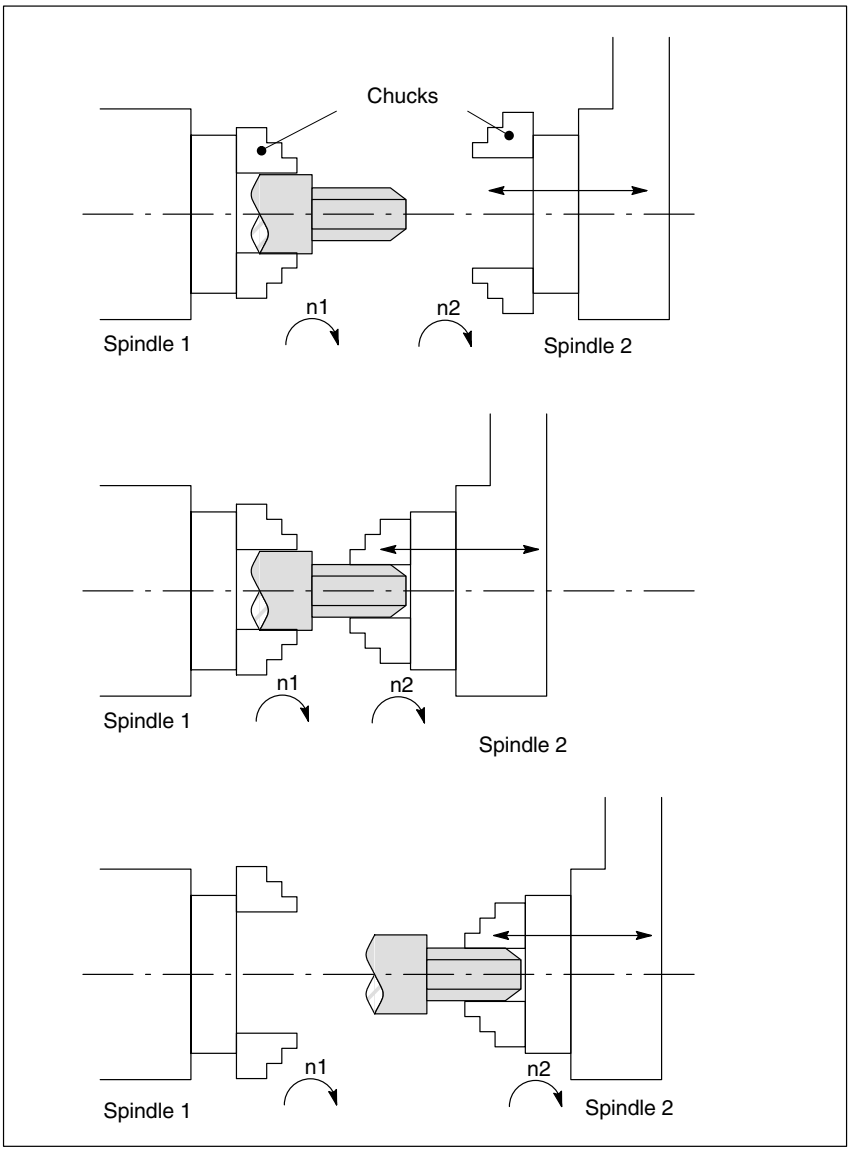


Fig. 1-1 Synchronous mode, on-the-fly workpiece transfer from spindle 1 to spindle 2

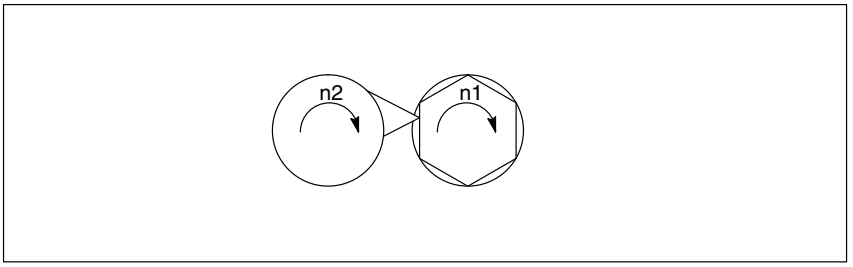


Fig. 1-2 Polygonal turning

■

Detailed Description 2

2.1 General functionality

2.1.1 Synchronous mode

Explanations

<Axial expression>:	can be: – Axis identifier – Spindle identifier
<Axis identifier>:	C (if spindle 1 has theidentifier C in axis mode
<Spindle identifier>:	Sn, SPI(n) with n = Spindle number 1, 2, ...
<Spindle number>:	according to the spindle number defined in MD 35000:SPIND_ASSIGN_TO_MACHAX

Synchronous spindle pair

Synchronous operation involves a following spindle (FS) and a leading spindle (LS), referred to as the **synchronous spindle pair**. The following spindle imitates the movements of the leading spindle when a coupling is active (synchronous operation) in accordance with the defined functional interrelationship.

Synchronous mode

Synchronous mode (also referred to as “Synchronous spindle operation”) is another spindle operating mode. Before synchronous mode is activated, the following (slave) spindle must have been switched to position control. Synchronous operation is activated for the following spindle when the coupling is activated. As soon as the coupling is deactivated, the following spindle switches to back to open-loop control mode.

As soon as synchronous operation has been activated for the following spindle, IS “Synchronous operation” (DB31, ... DBX84.4) = 1 is transmitted to the PLC.

Number of synchronous spindles (SW 4 and earlier)

Two synchronous spindle couplings can be operated simultaneously in each NC channel.

Number of synchronous spindles (SW 5 and higher)

Any number of following spindles in any channels on an NCU can be coupled to one leading spindle.

Configured or user-defined coupling

Synchronous spindle couplings can be defined both

- as a fixed configuration via channel-specific machine data (referred to below as “**fixed coupling configuration**”) and
- as a freely defined coupling via language instructions (COUP...) in the part program (referred to below as “**user-defined coupling”**).

The following variants are possible:

1. A fixed configuration for a coupling can be programmed via machine data. In addition, a second coupling can be freely defined via the part program.

2. No coupling is configured via machine data. In this case, two couplings can be user-defined and parameterized via the part program.

Options in synchronous mode

The following functional options are available in synchronous mode:

- Following and leading spindle rotate at the same speed
  ($n_{FS} = n_{LS}$; speed ratio $k_Ü = 1$)
- LS and FS rotating in the same direction or in opposite directions
  (can be defined by specifying positive or negative speed ratio $k_Ü$)
- Following and leading spindles rotate at different speeds
  ($n_{FS} = k_Ü \cdot n_{LS}$; speed ratio $k_Ü \neq 1$)
  Application: Polygonal turning
- Adjustable angular position between FS and LS ($\varphi_{FS} = \varphi_{LS} + \Delta\varphi$)
  The spindles rotate at synchronous speed with a defined angular offset between the LS and FS (position-synchronous coupling).
  Application: Shaped workpieces
- Activation of synchronous operation between LS and FS can take place when the spindles are in motion or at standstill.
- The full functionality of the open-loop and position control modes is available for the leading spindle.
- When synchronous mode is not active, the FS and LS can be operated in all other spindle modes.
- The speed ratio can also be altered when the spindles are in motion in active synchronous mode.
- With synchronous spindle coupling switched on, the offset of the FS to the LS (overlaid movement) can be altered.

Definition of synchronous spindle pair

The spindles to be coupled (FS, LS) must be defined before synchronous mode is activated.

This can be done in two ways depending on the application in question:

1. Fixed coupling configuration

   The machine axes which are to act as the leading and following spindles must be defined in channel-specific MD: COUPLE_AXIS_1[n].

   The machine axes programmed as the LS and FS for this coupling configuration cannot be altered by the NC part program.

   If necessary, the coupling parameters can be modified with the NC part program.

2. User-defined coupling:

   Couplings can be created and altered in the NC part program with language instruction “COUPDEF(FS,LS**...**”). If a new coupling relationship is to be defined, it may be necessary to delete an existing user-defined coupling beforehand (with language instruction COUPDEL(FS,LS)).

   The axis identifiers (Sn, SPI(n)) for the following and leading spindles must be programmed with FS and LS for every language instruction COUP**...**, thus ensuring that the synchronous spindle coupling is unambiguously defined.

The valid spindle number must be assigned to a machine axis in the axis-specific MD: SPIND_ASSIGN_TO_MACHAX.

References: /PA/, Programming Guide

The PLC is informed of whether a machine axis is programmed as a following or leading spindle by means of interface signals “FS active” (DB31, ... DBX99.1) and “LS active“ (DB31, ... DBX99.0).

Speed ratio

The speed ratio is programmed with separate numerical values for numerator and denominator (speed ratio parameters). It is therefore possible to specify the speed ratio very exactly, even with rational numbers.

General rule: $k_{Ü} = \frac{\text{Speed ratio parameter numerator}}{\text{Speed ratio parameter denominator}} = \frac{Ü_{numerator}}{Ü_{denominator}}$

The value range of the speed ratio parameter ($Ü_{numerator}$, $Ü_{denominator}$) is virtually unlimited internally in the control.

The speed parameters for the coupling configured via machine data can be set in channel-specific setting data (SD: COUPLE_RATIO_1[n]. In addition, the ratio can be altered with language instruction COUPDEF(FS, LS, $Ü_{numerator}$, $Ü_{denominator}$,**...**). The values entered in the setting data are not overwritten in this case (default settings).

The ratio for the coupling defined via the NC part program can only be input with language instruction COUPDEF (**...**).

The new ratio parameters take effect as soon as the COUPDEF instruction has been processed.

Coupling characteristics

The following characteristics can be defined for every synchronous spindle coupling:

- **Block change response**

  The condition to be fulfilled for a block change can be defined on activation of synchronous operation or on alteration of the ratio or the speed defined angular offset when the coupling is active:

  – Block changes immediately
  – Block change in response to “Fine synchronism”
  – Block change in response to “Coarse synchronism”
  – Block change in response to IPOSTOP (i.e. after setpoint-based synchronism)
  – Check of the synchronism conditions at an arbitrary moment with WAITC.

- **Type of coupling** between FS and LS

  The position setpoint or the actual position value of the leading spindle can be used as the reference value for the following spindle. The following coupling types can therefore be selected:

  1. Setpoint coupling (DV)
     Application in position-controlled mode. The control dynamic response of both spindles should coincide as far as possible. The setpoint coupling should be used preferably.
  2. Actual-value coupling (AV)
     Application if position control of the LS is not possible or with great deviation of the control characteristics between FS and LS. The setpoints for the FS are derived from the actual values of the LS. The synchronism is with changing LS speed of worse quality than with setpoint coupling.
  3. Speed coupling (VV)
     Internally, the speed coupling is a setpoint coupling. The requirements for FS and LS are lower. Position control and measuring systems are not required for FS and LS.

     The position offset between FS and LS is undefined.

The coupling characteristics are selected via machine data for fixed coupling configurations (see Section 2.3) and via language instruction COUPDEF for user-defined couplings (see Section 2.2.1).

In addition, coupling characteristics “Type of coupling” and “Block change response” can be altered for fixed coupling configurations by means of language instruction COUPDEF.

Change protection for coupling characteristics

Channel-specific MD: COUPLE_IS_WRITE_PROT_1 is set to define whether or not the configured coupling parameters “Speed ratio”, “Type of coupling” and “Block change response” can be altered by the NC part program:

0: Coupling parameters can be altered by the NC part program with COUPDEF.

1: Coupling parameters cannot be altered by the NC part program. Attempts to make changes will be rejected with an alarm message.

Overlaid movement

In synchronous operation, the synchronous spindle copies the movement of the leading spindle in accordance with the programmed speed ratio.

At the same time, the synchronous spindle can also be traversed with overlay so that the LS and FS can operate at a specific angular position in relation to one another.

The overlaid traversing movement of the FS can be initiated in various ways:

1. Programmable position offset of FS in AUTOMATIC and MDA:
   The position reference between LS and FS can be altered in active synchronous operation with language instructions COUPON and SPOS (see Section 2.1.2)

2. Manual position offset of FS:
   - In JOG operating mode (continuous JOG or incremental JOG):
     Overlay of FS by handwheel or plus or minus traversing key in active synchronous operation.
   - In AUTOMATIC and MDA operating modes:
     Overlay of FS by handwheel via DRF offset

As soon as the FS executes the overlaid traversing movement, the IS “Overlaid movement“ (DB31, ... DBX98.4) is set to “1”. The overlaid movement is executed optimally in terms of time at the maximum possible FS speed. With offset change by SPOS, the positioning speed can be set with FA[Sn] and influenced with override (selection possibility with DB31,... DBX17.0).

References: /FB/, S1, “Spindles”

2.1.2 Selecting synchronous mode

Activation of coupling

Language instruction COUPON activates the coupling between the programmed spindles with the last valid parameters and thus also activates synchronous mode. This coupling may be a fixed configuration or user-defined. The leading spindle and/or following spindle may be at standstill or in motion at the instant of activation.

Certain conditions must be fulfilled before synchronous operation can be activated (see Section 2.1.4).

Activation methods

Two different methods can be selected to activate synchronous mode:

1. Fastest possible activation of coupling with **any angular reference** between leading and following spindles.
   COUPON(FS, LS)
2. Activation of coupling with a **defined angular offset** POS_{FS} between leading and following spindles. With this method, the angular offset must be programmed on selection.
   COUPON(FS, LS, POS_{FS})

Note

If the LS and/or FS is in axis mode before switching on the synchronous coupling, the axis mode is left and spindle mode is activated with use of the spindle identifier with SW 3.2 and higher.

If the spindle is switched on with use of the axis identifier, no changeover takes place.

Block change response

Before synchronous operation is selected, it must be determined under what conditions the block change must occur when synchronous mode is activated (see Section 2.2.1).

Determining current coupling status

It is possible to determine the current coupling status for the specified axis/spindle in the NC part program by means of axial system variable $AA_COUP_ACT[<axial expression >] (see Section 2.2.3 Axis system variables for synchronous spindle). As soon as the synchronous spindle coupling is active for the following spindle, bit 2 must be “1” when read.

Change defined angular offset

Language instructions COUPON and SPOS allow the defined angular offset to be changed while synchronous mode is active. The following spindle is positioned as an overlaid movement at the angular offset programmed with POS_{FS}. IS “Overlaid movement” (DB31, ... DBX98.4) is set during this period.

Angular offset POS_{FS}

The defined angular offset POS_{FS} must be specified as an absolute position referred to the zero degrees position of the leading spindle in a positive direction of rotation.

The “Zero degrees position” of a position-controlled spindle is calculated from the zero mark signal or Bero signal of the measuring system and the offsets stored in axis-specific machine data (MD: REFP_SET_POS, REFP_MOVE_DIST, REFP_MOVE_DIST_CORR)

Range of POS_{FS}: 0 **...** 359.999 degrees.

References: /FB/, R1, “Reference Point Approach”

Read current angular offset

Using axial system variables, it is possible to read the current position offset between the FS and LS in the NC part program. The following two position offsets exist:

a) Current position offset of setpoint between FS and LS
$AA_COUP_OFFS [<axis identifier for FS>]

b) Current position offset of actual value between FS and LS
$VA_COUP_OFFS [<axis identifier for FS>]

(For more information about <axis identifier>, see Section 2.1.1)

Activation after POWER ON

Synchronous operation can also be activated for LS or FS which are not referenced/synchronized (IS: “Referenced/synchronized 1 or 2“ DB31, ... DBX60.4 or DBX60.5 = 0). In this case, a warning message is displayed.

Example: LS and FS are already coupled in a friction lock via a workpiece after power ON.

2.1.3 Deselecting synchronous mode

Deactivation of coupling

Language instruction COUPOF cancels synchronous mode between the programmed spindles. The coupling concerned can be a fixed configuration or user-defined. The leading and following spindles can be at standstill or in motion when synchronous operation is deactivated.

On switching off the synchronous mode, the following spindle is put into **control mode**. The originally programmed S-word is no longer valid for the FS, the following spindle can be operated like any other normal spindle.

When the coupling is deactivated, a block preprocessing stop (STOPRE) is generally initiated internally in the control.

Deactivation while spindles are moving

If synchronous mode is deselected while the spindles are in motion, the following spindle continues to rotate at the current speed (n_{FS}). The current speed can be read with system variable $AA_S in the NC part program.

The spindle can then be stopped from the part program with M05, SPOS or SPOSA or from the PLC with the appropriate interface signal.

Deselection methods

Three different methods can be used to deselect synchronous mode:

1. Fastest possible deactivation of coupling.
   The block change is enabled immediately.
   COUPOF(FS, LS)
2. The coupling is not deselected until the following spindle has overtraveled the programmed deactivation position POS_{FS}.
   The block change is then enabled.
   COUPOF(FS, LS, POS_{FS})
3. The coupling is not deselected until the following spindle and the leading spindle have overtraveled the programmed deactivation positions POS_{FS} and POS_{LS}.
   The block change is then enabled.
   COUPOF(FS, LS, POS_{FS} , POS_{LS})

Note

If the LS and/or FS is in axis mode before switching off the synchronous coupling, the axis mode is left with use of the spindle identifier and the speed control mode is activated with SW 3.2 and higher.

If the spindle is switched off with use of the axis identifier, no changeover takes place. Before shutdown, the LS must be in the setpoint-side standstill.

POS_{FS}, POS_{LS}

The deactivation positions POS_{FS} and POS_{LS} correspond to the actual positions of the FS and LS referred to the defined reference point value (see Section 2.1.2).

Range of POS_{FS}, POS_{LS}: 0 **...** 359.999 degrees.

References: /FB/, R1, “Reference Point Approach”

2.1.4 Prerequisites for synchronous mode

Conditions on selection of synchronous mode

The following conditions must be fulfilled before the synchronous spindle coupling is activated or else alarm messages will be generated.

- The synchronous spindle coupling must have been defined beforehand (either a fixed configuration via machine data or according to user definition via part program).
- The spindles to be coupled must be defined in the NC channel in which the coupling is activated.

  Channel-specific MD 20070: AXCONF_MACHAX_USED
  Axis-specific MD 35000: SPIND_ASSIGN_TO_MACHAX
- The following spindle must be assigned to the NC channel in which the coupling is activated.

  Default setting with axis-specific MD30550: AXCONF_ASSIGN_MASTER_CHAN
- LS and FS must be equipped with at least one position measuring system for position sensing.
- If the FS is in speed control mode (IS “Position controller active“ DB31, ... DBX61.5 = 0) before synchronous operation is activated, it must be switched to position control with language instruction SPCON.

Note

When position control is activated, the maximum setpoint speed of the LS is automatically limited to 90 % (control reserve) of the maximum speed as determined by the limit frequency of the position encoder. The limitation is reported via IS “Setpointspeed limited” (DB31, ... DBX83.1).

After deactivation of synchronous operation, position control mode can be deselected again with language instruction SPCOF.

References: /FB/, S1, “Spindles”

- To ensure more accurate synchronization, the LS should be in position control mode (language instruction SPCON) before the coupling is activated, thus allowing a setpoint coupling to be established between the LS and FS.

  Actual-value coupling is always possible if there is a measuring system for the LS.
- Before selecting the synchronous mode, the gear stage necessary for FS and LS must be selected. In synchronous mode, gear stage changeover and therefore oscillation mode are not possible for FS and LS. Upon request, an alarm message is generated.
- If FS and/or LS are in the axis mode and if they are actuated with a spindle identifier, spindle mode is activated. The VDI interface signals for the spindle concerned are modified, the active parameter block is changed over and feedforward control is activated.

  If the spindle is activated with use of the axis identifier, no changeover takes place.

Cross-channel coupling (SW4)

- The LS can be programmed either via a part program, PLC (FC18) or, in SW 4 and higher, by means of synchronized actions.

Note

If the LS is swapped between channels with activated speed coupling, and the sequence of the channels is changed, the coupling must be deactivated.

Example:
Channel 1:
Channel 2:
Channel 3: FS in channel 3. COUPON active
Channel 4:
Channel 5:

Easy exchange possible for the LS between:
Channel 1 <—> Channel 2,
Channel 1 <—> Channel 3,
Channel 2 <—> Channel 3,
Channel 4 <—> Channel 5

Exchange possibilities for LS, where the coupling must be deactivated:
from channel 1 <—> channel 4,
from channel 2 <—> channel 4,
from channel 3 <—> channel 4,
from channel 1 <—> channel 5,
from channel 2 <—> channel 5,
from channel 3 <—> channel 5

SW 5 and higher

The LS can belong to any channel.

- The LS can be exchanged between channels by means of “Axis exchange“.
- When several following spindles are coupled to one leading spindle, the dynamic response of the coupling is determined by the weakest response as a function of the coupling factor. The acceleration rate and maximum speed are reduced for the leading spindle to such a degree that none of the coupled leading spindles can be overloaded.

2.1.5 Monitoring of synchronous operation

Fine/coarse synchronism

In addition to conventional spindle monitoring operations, synchronous operation between the FS and LS is also monitored in synchronous mode.

In this case, IS: “Fine synchronism“ (DB31, ... DBX98.0) or “Coarse synchronism“ (DB31, ... DBX98.1) is transmitted to the PLC to indicate whether the current position (AV, DV) or actual speed (VV) of the following spindle is within the specified tolerance window.

When the coupling is switched on, the signals “Coarse synchronism” and “Fine synchronism” are updated with reaching of the setpoint-side synchronism.

The size of the tolerance windows is set with MD of the FS.

Reaching of the synchronism is influenced by the following factors:

- AV, DV: Position deviation between FS and LS
- VV: Speed difference between FS and LS

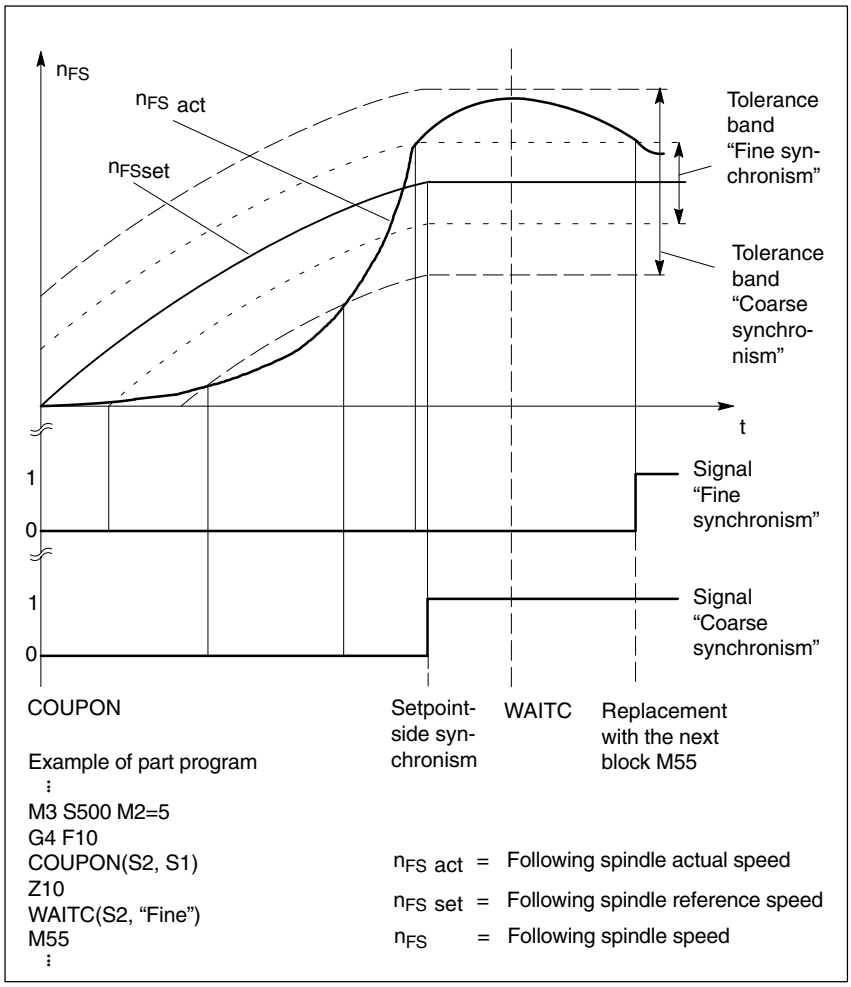


Fig. 2-1 Synchronism monitoring with COUPON and synchronism test marker WAITC

Threshold values

The relevant position or velocity tolerance range for the following spindle in relation to the leading spindle must be specified in degrees of rev/min.

- Threshold value for “Coarse synchronism”
  axis spec. MD 37200: AV, DV: COUPLE_POS_TOL_COARSE
  MD 37220: VV: COUPLE_VELO_TOL_COARSE
- Threshold value for “Fine synchronism”
  axis spec. MD 37210: AV, DV: COUPLE_POS_TOL_FINE
  MD 37230: VV: COUPLE_VELO_TOL_FINE

Speed/acceleration limits

In synchronous mode, the speed and acceleration limit values of the leading spindle are adjusted internally in the control in such a way that the following spindle can imitate its movement, allowing for the currently selected gear stage and effective speed ratio, without violating its own limit values.

For example, the LS is automatically decelerated to prevent the FS from exceeding the maximum speed in order to maintain synchronism between the spindles.

2.2 Programming of synchronous spindle couplings

Table 2-1 Overview

Programmed coupling	Configured coupling(s)	Remark
Coupling definition: **COUPDEF()**	Modification of configured data: **COUPDEF()**	Setting the coupling parameters
Switching on a coupling: **COUPON()**		Switching on and off
Switching off a coupling: **COUPOF()**		
Deleting the coupling data: **COUPDEL()**	Reactivating the configured data: **COUPRES()**	Arrangement, restore

2.2.1 Preparatory programming instructions

User-defined coupling (SW 4 and earlier)

Up to two synchronous spindle couplings can be active simultaneously in each channel (SW 4 and earlier). Provided no fixed coupling configuration has been programmed, both couplings can be freely defined by the NC part program.

These couplings must also be parameterized by the NC part program. Default values are used for parameters which are not programmed.

A new synchronous spindle coupling is defined if an FS/LS coupling relationship which has no fixed configuration is programmed in language instruction COUPDEF. This coupling can be invalidated again with language instruction COUPDEL if, for example, a further synchronous spindle coupling between other spindles is needed. These programming options, i.e. re-definition and deletion of couplings, allow more than two coupling relationships to be successively created in the NC channel (SW 4 and earlier).

SW 5 and higher

Any number of couplings can be programmed. Furthermore, one coupling can also be configured via machine data as in earlier SW versions.

Permanent coupling configuration

The coupling characteristics and speed ratio for a permanently configured synchronous spindle coupling can be altered by the NC part program provided that they are not write-protected. The machine axes for LS and FS cannot be changed.

Define new couplings

Language instruction “COUPDEF“ can be used to create new synchronous spindle couplings (user-defined) and to modify the parameters for existing couplings.

When the coupling parameters are fully specified, the following applies:

COUPDEF
(FS, LS, $Ü_{numerator}$, $Ü_{denominator}$, block change response, coupling type)

The spindle coupling is unambiguously defined with FS and LS which must be programmed for every COUP**...** instruction. Alarm messages will otherwise be generated.

The other coupling parameters must only be programmed when they need to be changed. The last valid status remains applicable for non-specified parameters.

The individual coupling parameters are explained below:

- **FS, LS**: Axis identifiers for following and leading spindles

  e.g.: S1, SPI(1), S2, SPI(2)

  The applicable spindle number must be assigned to a machine axis in axis-specific MD: SPIND_ASSIGN_TO_MACHAX.

- $\mathbf{Ü_{numerator}}$, $\mathbf{Ü_{denominator}}$: Speed ratio parameters for numerator and denominator

  The speed ratio is input in the form of numeric values for numerator and denominator (see Section 2.1.1).

  The numerator must always be programmed. If no denominator is specified, then its value is always assumed to be “1,0”.

- **Block change response**
  This parameter allows the condition for block change on selection of synchronous operation to be defined:

  NOC ⇒ Block change is immediately enabled
  FINE ⇒ Block change in response to “Fine synchronism”
  COARSE ⇒ Block change in response to “Coarse synchronism”
  IPOSTOP ⇒ Block change in response to IPOSTOP (e.g. after setpoint-based synchronism)

  The block change response is entered as a character string (i.e. with quotation marks).

  The block change response can be specified simply by writing the letters in bold print. The remaining letters can be entered to improve legibility of the part program but they are not otherwise significant.

  If no block change response is specified, then the currently selected response continues to apply.

  With the programmable synchronism test marks **WAITC**, the replacement with new blocks is delayed until reaching the synchronism indicated.

- **Type of coupling**

  DV (Desired Values) ⇒ Setpoint coupling between FS and LS
  AV (Actual Values) ⇒ Actual-value coupling between FS and LS
  VV (Velocity Values) ⇒ Speed coupling between FS and LS

  If no coupling type is specified, then the currently selected type continues to apply.

Note

The coupling type may only be changed when synchronous operation is deactivated!

Examples

COUPDEF (SPI(2), SPI(1), 1.0 , 1.0, “FINE”, “DV”)
COUPDEF (S2, S1, 1.0 , 4.0)
COUPDEF (S2, SPI(1), 1.0)

Default settings

The following default settings apply to user-defined couplings:

- $Ü_{numerator}$ = 1.0
- $Ü_{denominator}$ = 1.0
- Block change response = **IPOSTOP** (block change enabled with setpoint synchronism)
- Type of coupling = **DV** (setpoint coupling)

Delete couplings

Language instruction “COUPDEL“ is used to delete user-defined couplings.

COUPDEL (FS, LS)

SW 4 and earlier

If a new synchronous spindle coupling relationship needs to be defined and all available, freely configurable couplings (1 or 2) are already configured, then one of the couplings will have to be deleted first.

SW 5 and higher

There is no limit to the number of programmable couplings. The COUPDEL command can be used, but is not absolutely necessary.

An alarm message is generated if COUPDEL is programmed for an active coupling. Synchronous operation remains active. It must be deselected beforehand with COUPOF.

Note

A fixed coupling configuration cannot be deleted with COUPDEL!

Activate original coupling parameters

Language instruction “COUPRES“ can be used to re-activate the configured coupling parameters.

COUPRES (FS, LS)

The parameters programmed with COUPDEF (including speed ratio) are then overwritten.

Language instruction “COUPRES”

- activates the parameters stored in the machine and setting data (fixed coupling configuration) and
- activates the default settings (user-defined coupling).

Programmable block change

With Software Version 3.2 and higher, it is possible to mark a point in the NC program using “WAITC”. The system waits at this point for fulfillment of the synchronism conditions for the specified FS and delays changes to new blocks until the specified state of synchronism is reached (see Fig. 2-1).

WAITC (FS)

Advantage: The time between switching on the synchronous coupling and reaching the synchronism can be technologically useful.

Note

Basically, it is always possible to write WAITC. If the spindle indicated is not active as FS, the instruction for this spindle is without effect.

If no synchronism condition is indicated, the check is always performed for the synchronism condition programmed/configured on the respective coupling, at least for the setpoint synchronism.

Examples

WAITC(S2),
WAITC(S2, “Fine”),
WAITC(S2, ,S4, “Fine”)

2.2.2 Programming instructions for activating and deactivating the coupling

Activate synchronous mode

Language instruction COUPON is used to activate couplings and synchronous mode.

Two methods by which synchronous operation can be activated are available:

1. **COUPON(FS, LS)**
   Fastest possible activation of synchronous operation with any angular reference between the leading and following spindles.
2. **COUPON(FS, LS, POS_{FS})**
   Activation of synchronous operation with a defined angular offset POS_{FS} between the leading and following spindles. This offsets is referred to the zero degrees position of the leading spindle in a positive direction of rotation. The block change is enabled according to the defined setting. Range of POS_{FS}: 0 **...** 359.999 degrees.

By programming COUPON(FS, LS, POS_{FS}) or SPOS when synchronous operation is already active, the angular offset between LS and FS can be changed.

Deactivate synchronous mode

Three different methods can be selected to deactivate synchronous mode:

1. **COUPOF(FS, LS)**
   Fastest possible deactivation of synchronous operation. Block change is immediately enabled.

2. **COUPOF(FS, LS, POS_{FS})**
   Deselection of synchronous operation after deactivation position POS_{FS} has been overtraveled. Block change is not enabled until this position has been overtraveled.

3. **COUPOF(FS, LS, POS_{FS} , POS_{LS})**
   Deselection of synchronous operation after the two deactivation positions POS_{FS} and POS_{LS} have been overtraveled. Block change is not enabled until **both** positions have been overtraveled.
   Range of POS_{FS}, POS_{LS}: 0 **...** 359.999 degrees.

If continuous path control (G64) is programmed, a non-modal stop is generated internally in the control.

Examples

```
COUPDEF (S2, S1, 1.0 , 1.0, “FINE”, “DV”)
        :
COUPON (S2, S1, 150)
        :
COUPOF (S2, S1, 0)
        :
COUPDEL (S2, S1)
```

2.2.3 Axis system variables for synchronous spindle

Reading the current coupling status

The current coupling status for the following spindle can be read in the NC part program with the following axis system variable:

$AA_COUP_ACT[<axial expression>]

(Explanation about <axial expression>, see Section 2.1.1)

Example

$AA_COUP_ACT[SPI(2)]

The value read has the following significance for the following spindle:
Byte = 0: No coupling active
Bit 2 = 1: Synchronous spindle coupling active

Read current angular offset

The current position offset between the FS and LS can be read in the NC part program by means of the following axial system variables:

a) Setpoint-based position offset between FS and LS:

$AA_COUP_OFFS[<axial expression>]

b) Actual-value-based position offset between FS and LS:

$VA_COUP_OFFS[<axial expression>]

Example

$AA_COUP_OFFS[S2]

If an angular offset is programmed with COUPON, this coincides with the value read after reading the setpoint synchronism.

Note

After cancellation of the servo enable signal when synchronous operation and follow-up mode are active, the position offset applied when the controller is enabled again is different to the originally programmed value. In this case, the altered position offset can be read and corrected in the NC part program if necessary.

2.3 Configuration of a synchronous spindle pair via machine data

Coupling parameters

One synchronous spindle coupling per NC channel can be configured permanently via machine data.

It is then necessary to define the machine axes (spindles) which are to be coupled and what characteristics this coupling should have.

The following parameters can be configured as fixed settings for the synchronous spindle coupling:

- **Synchronous spindle pair** (channel-specific MD: COUPLE_AXIS_1[n])

  This machine data defines the two machine axes which are to form the synchronous spindle pair (following spindle (n=0), leading spindle (n=1)).

  A “0” as the setting for the axis number means that no coupling is configured via the machine data. The machine data for the coupling characteristics are then irrelevant.

  The machine axis numbers for the LS and FS can not be changed by the NC part program for a configured coupling configuration.

- **Speed ratio**

  This ratio is input in the form of numerator and denominator via setting data (currently POWER ON active!) in two speed parameters (channel-specific SD: COUPLE_RATIO_1[n]). The quotient is generated internally in the control.

  $$k_{Ü} = \frac{\text{Speed ratio parameter nominator}}{\text{Speed ratio parameter denominator}} = \frac{\$SC\_COUPLE\_RATIO[0]}{\$SC\_COUPLE\_RATIO[1]}$$

  Provided it is not write-protected, the speed ratio can be changed by the NC part program with language instruction COUPDEF.

- **Block change response**
  (channel-specific MD: COUPLE_BLOCK_CHANGE_CTRL_1)

  One of the following options can be selected as the condition for a block change:

  0: Block changes immediately

  1: Block change in response to “Fine synchronism”

  2: Block change in response to “Coarse synchronism”

  3: Block change in response to IPOSTOP (i.e. after setpoint-based synchronism)

- **Type of coupling** between LS and FS:
  (channel-specific MD: COUPLING_MODE_1)

  0: Actual-value coupling

  1: Setpoint coupling

  2: Speed coupling

- **Abortion of coupling with NC start**

  Channel-specific MD; COUPLE_RESET_MODE_1 (see Table 2-3)

- **Write-protection for coupling parameters**:

  (channel-specific MD: COUPLE_IS_WRITE_PROT_1)

  It can be defined in this machine data whether or not the configured coupling parameters “Speed ratio”, “Type of coupling” and “Block change response” may be influenced by the NC part program.

  0: Coupling parameters can be changed by the NC part program

  1: Coupling parameters cannot be changed by the NC part program. Attempts to make changes are rejected with an alarm message.

2.3.1 Configuration of the behavior with NC start

The response to NC machining program start is defined by the channel-specific machine data.

Table 2-2 Synchronous coupling behavior with NC start

	Configured coupling	**Programmed coupling (see Section 2.3)**
	MD: COUPLE_RESET_MODE	MD: START_MODE_MASK
Coupling is maintained	Bit 0 = 0	Bit 10 = 0
Deselect coupling	Bit 0 = 1	Bit 10 = 1
Activate configured data	Bit 5 = 1	–
Switch on coupling	Bit 9 = 1	–

2.3.2 Configuration of the behavior with Reset

With SW 3.2 and higher, the following behavior can be set with the channel-specific machine data with reset and end of NC machining program:

Table 2-3 Synchronous coupling behavior with end of NC machining program and after reset

	Configured coupling	**Programmed coupling (see Section 2.3)**
Coupling is maintained	MD: COUPLE_RESET_MODE Bit 1 = 0	MD: RESET_MODE_MASK- Bit 10 = 1
Deselect coupling	MD: COUPLE_RESET_MODE Bit 1 = 1 MD: RESET_MODE_MASK Bit 0 = 1 (Generating a block on RESET)	MD: RESET_MODE_MASK- Bit 10 = 0 Bit 0 = 1
Activate configured data	MD: COUPLE_RESET_MODE Bit 6 = 1 MD: RESET_MODE_MASK Bit 0 = 1	–

2.4 Special features of synchronous operation

2.4.1 Special features of synchronous operation in general

Control dynamics

When a setpoint coupling is used, the position controller parameters of FS and LS (e.g. K_V factor) must be matched. It may be necessary to activate different parameter sets for speed control mode and synchronous operation (M41...M45).

Feedforward control

Due to the improved control system dynamic response it provides, feedforward control is **always active** for the following and leading spindles in synchronous mode. It can, however, be deselected for FS and LS with axis-specific MD: FFW_MODE (=0). The NC part program cannot deactivate the feedforward control for LS and FS with FFWOF.

The feedforward control mode (speed or torque feedforward control) is defined in axis-specific MD: FFW_MODE.

References: /FB/, K3, “Compensations”

Speed/acceleration limits

The speed and acceleration limits of the spindles operating in synchronous mode are determined by the “weakest” spindle in the coupling. The current gear stages, the programmed acceleration and, for the leading spindle, the effective position control status (On/Off) are taken into account for this purpose.

As an example, the maximum speed of the leading spindle is calculated internally in the control on the basis of the speed ratio and the spindle limitations of the following spindle.

Multiple couplings

If the system detects that a coupling is already active for an FS and LS when synchronous mode is activated, then the activation process is ignored and an alarm message generated.

Example of multiple couplings:

- A spindle is acting as the FS for several LS

SW 5.1 and later

Number of configurable spindles per channel:

- Every axis in the channel can be configured as a spindle. The number of axes per channel is dependent on the control system model.

SW 5.2 and later

Cross-channel setpoint linkage and optional number of following spindles in optional channels of an NCU:

- Cross-channel synchronous spindle setpoint links (DV) can be implemented with no additional restrictions.
- Any number of following spindles in any channels on an NCU can be coupled to one leading spindle.

Number of following spindles to one leading spindle

In SW 5.2 and later, any number of following spindles in any channels on an NCU can be coupled to one leading spindle. The only possible restriction to the number of spindles could be imposed by the real CPU time requirement.

The dynamic response of a coupling group is determined by the weakest response as a function of the coupling factor. The acceleration rate and maximum speed are reduced for the leading spindle down to the load limit of the coupled leading spindles.
Further notes: See above **Speed/acceleration limits**

Knee-shaped acceleration characteristic

The effect of a knee-shaped acceleration characteristic (identified by axis-specific MD: ACCEL_REDUCTION_SPEED_POINT and ACCEL_REDUCTION_FACTOR) on the following spindle is taken into account for the leading spindle.

The acceleration should, however, be **constant** over the entire speed range for the following spindle. If a knee-shaped acceleration characteristic is nevertheless stored in the above-mentioned machine data for the following spindle, the knee-shaped acceleration characteristic for the LS must be set such that the FS is not overloaded in acceleration terms or that the programmed acceleration for the LS is used accordingly.

Direct control of synchronous mode by PLC

ASUBs (activation of asynchronous subprograms) processed by the PLC can be used to activate or terminate synchronous mode at any chosen time in the AUTOMATIC or MDA modes.

Response to alarms

When alarms (e.g. servo alarms) occur during synchronous operation which cause cancellation of the servo enable signal in the control and active follow-up mode, the subsequent response is the same as if IS “Servo enable” (DB31, ... DBX2.1) had been cancelled by the PLC (and IS “Follow-up mode” (DB31, ... DBX1.4) is set) → see Section 2.4.2 .

2.4.2 Influence on synchronous operation via PLC interface

PLC interface signals

In synchronous operation, the influence of the PLC on the coupling resulting from the setting of LS and FS interface signals must be noted.

The effect of the main PLC interface signals on the synchronous spindle coupling is described below.

Spindle speed override (DB31, ... DBB19)

The spindle speed override value input by the PLC in synchronous operation is applied only to the leading spindle.

Axis/spindle disable (DB31, ... DBX1.3)

The participating axes behave as shown in the following table (SW 4 and higher):

set: 1 not set: 0

No.	LS/LA	FS/FA	Coupling	Response
1	0	0	Off	Axis setpoints are output
2	0	1	Off	no setpoint output for FS/FA
3	1	0	Off	no setpoint output for LS/LA
4	1	1	Off	no setpoint output for LS/LA and FS/FA
5	0	0	On	Axis setpoints are output
6	0	1	On	Disable not effective for FS/FA
7	1	0	On	Disable also effective for FS/FA
8	1	1	On	no setpoint output for LS/LA and FS/FA

- The signal is not effective for the FS/FA when the coupling is active. → No. 6
- If the signal for the LS/LA is enabled, this also acts on FS/FA(s), → No. 7
- If a workpiece is clamped between two spindles (transfer of workpiece from front face to rear face machining), it cannot be damaged.

Servo enable (DB31, ... DBX2.1)

Cancellation of “Servo enable” for LS (either via PLC interface or internally in control in the event of faults):

If the servo enable signal of the LS is set to “0” during synchronous operation and a setpoint coupling is active, a switchover to actual-value coupling is executed in the control. If the LS is in motion at this instant, it is decelerated to a standstill and an alarm message generated. Synchronous operation remains active.

Cancellation of “Servo enable” for FS in synchronous operation (either via PLC interface or internally in control in the event of faults):

The coupling is internally cancelled until the signals are reset.

If the “Servo enable” signal is not set for either of the spindles when synchronous operation is selected, synchronous operation is still activated when the coupling is switched on. The LS and FS however remain at standstill until the servo enable signal is set for both of them.

Setting the “Servo enable” signal for LS and FS:
When the signal edge of IS “Servo enable” switches to 1, the spindle either moves back to the old position (position on cancellation of servo enable) (signal status = 0: Stop active) or the current positions (position offset) are used again (signal status = 1: Follow-up active), depending on IS “Follow-up mode”.

Note

If the “servo enable” signal is cancelled for the FS after Spindle Stop without the coupling being deactivated beforehand, then any synchronism error resulting from external intervention (e.g. manual rotation) will not be compensated when the “servo enable” signal is applied again.

This may result in loss of the defined angular reference between the FS and LS for special applications.

Follow-up mode (DB31, ... DBX1.4)

Interface signal “Follow-up mode” is relevant only if the “servo enable” for the drive is cancelled. When “servo enable” is set for the FS and LS, either the spindle will return to the position recorded on cancellation of the servo enable signal (signal state = 0: Stop active) or the current positions will be used again (signal status = 1: Follow-up active), depending on IS “Follow-up mode”.

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

Switchover between the position measuring systems for the FS and LS is not locked out in synchronous operation. A switchover would not affect the coupling. It is however recommended that the measuring systems only be switched when synchronous mode is not active.

If “Park” status is selected for the FS or LS in synchronous operation, then the system responds as if “servo enable” had been cancelled.

Spindle Reset (delete distance to go) (DB31, ... DBX2.2)

When Spindle Reset is set for the LS in synchronous operation, the LS is braked down to standstill at the selected acceleration rate. The FS and LS continue to operate in synchronous mode. The overlaid motion (except with COUP...) is terminated as quickly as possible.

Spindle Stop (Feed Stop) (DB31, ... DBX4.3)

When “Spindle Stop” is set for the FS or LS, both coupled spindles are braked down to standstill via a ramp, but continue to operate in synchronous mode.

As soon as IS “Spindle Stop” is no longer active for any of the spindles in the coupling, it is accelerated back up to the previous speed setpoint.

Application

“Spindle Stop” can halt the synchronous spindle pair without offset since the servo loop remains operative.

Example

When the protective door is opened with an active synchronous spindle coupling, the FS and LS must be stopped without the coupling relationship being altered. This can be achieved by applying IS “Spindle Stop” to halt the FS and LS (IS “Axis/spindle stationary” (DB31, ... DBX61.4) = 1). “Servo enable” can then be cancelled for both spindles.

Delete S value (DB31, ... DBX16.7)

The S value programmed for the LS is deleted and the LS decelerated down to zero speed via a ramp. The FS and LS continue to operate in synchronous mode.

IS “Delete S value” has no affect on the FS in synchronous operation.

Re-synchronize spindle 1/2 (DB31, ... DBX16.4 and 16.5)

It is possible to synchronize the spindle (LS) with its positioning measuring system when it is operating in synchronous mode. It is however recommended that the leading spindle only be re-synchronized when synchronous mode is not active.

Traversing keys in JOG (DB31, ... DBX4.6 and 4.7)

The “plus and minus traversing keys” for JOG are **not disabled** internally for the FS in synchronous operation, i.e. the FS executes an overlaid motion if one of these keys is pressed.

Note

If overlaid traversing movements are to be precluded, they must be locked out by measures in the PLC user program.

NC Stop axes plus spindles (DB21, ... DBX7.4)

“NC Stop axes plus spindles” in synchronous operation decelerates the coupled spindles in accordance with the selected dynamic response. They continue to operate in synchronous mode.

NC Start (DB21, ... DBX7.1)

See Section 2.3.1.

Note

NC Start after NC Stop does not deselected synchronous operation.

2.4.3 Special points regarding start-up of a synchronous spindle coupling

Spindle start-up

The leading and following spindles must be started up initially like a normal spindle. This start-up procedure is described in:

References: /IAD/, SINUMERIK 840D Installation and Start-Up Guide
and
References: /FB/, S1, “Spindles”

Parameters

The following parameters must then be set for the synchronous spindle pair:

- The machine axis numbers for the leading and following spindles (for a permanently configured coupling with channel-spec. MD: COUPLE_AXIS_1[n])
- The required coupling type (setpoint, actual-value or velocity coupling) (for a permanently configured coupling with channel-spec. MD: COUPLING_MODE_1[n])
- The gear stage(s) of FS and LS for synchronous operation
- Plus the following coupling properties (see Section 4.1) for a permanently configured synchronous spindle coupling:
  - Block change behavior in synchronous spindle operation
    Channel-spec. MD: COUPLE_BLOCK_CHANGE_CTRL_1
  - Coupling abort behavior
    Channel-spec. MD: COUPLE_RESET_MODE_1
  - Modification protection for coupling parameters
    Channel-spec. MD: COUPLE_IS_WRITE_PROT_1
  - Speed ratio parameters for synchronous spindle coupling
    Channel-spec. SD: COUPLE_RATIO_1[n]

Response to setpoint changes

In order to obtain the best possible synchronism in **setpoint couplings**, the FS and LS must **have the same dynamic response to setpoint changes**. Each of the axial servo loops (position, speed and current controller) should be set **optimally** to eliminate any interference as quickly and efficiently as possible. The **dynamic response adaptation** function in the setpoint branch is used to match axial dynamic responses without loss of control quality.

The following control parameters must each be set optimally for the FS and LS:

- K_V factor (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 identically for the FS and LS:

- 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 FS and the coupled LS must have the same dynamic response to setpoint changes. The “same dynamic response” means that their following errors must be equal at any given speed.

The dynamic response adaptation function in the setpoint branch is capable of accurately matching the response to setpoint changes on dynamically unequal axes (servo loops). The difference in the equivalent time constants between the dynamically “weakest” spindle and the other spindle in the coupling must be entered as the dynamic response adaptation time constant.

Example

With active speed feedforward control, the dynamic response is mainly determined by the equivalent time constants of the “slowest” speed control loop.

Leading spindle: MD 32810: EQUIV_SPEEDCTRL_TIME [n] = 5ms

Following spindle: MD 32810: EQUIV_SPEEDCTRL_TIME [n] = 3ms

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

The dynamic response adaptation must be activated axially via MD 32900 DYN_MATCH_ENABLE.

The dynamic adaptation setting can be checked by comparing the following errors of the FS and LS (in Diagnosis operating area; Service Axes display). Their following errors must be identical when they are operating at the same speed!

By way of fine adjustment, it may be necessary to slightly adjust K_V factors or feedforward control parameters in order to achieve an optimum result.

Control parameter sets

A separate parameter set with servo loop setting is assigned to each gear stage on coupled spindles.

These parameter sets can be used, for example, to adapt the dynamic response of the leading spindle to the following spindle in synchronous operation. When the coupling is deactivated (speed or positioning mode), it is therefore possible to select other position controller parameters for the FS and LS. To utilize this option, a separate gear stage must be reserved for synchronous operation and selected before synchronous mode is activated.

References: /FB/, G2, “Velocities, Setpoint/Actual Value Systems, Closed-Loop Control”

Actual-value coupling

In an actual-value coupling, the drive for the FS must be considerably more dynamic than the leading spindle drive. The individual drives in an actual-value coupling are also set optimally according to their dynamic response.

An actual-value coupling should only be used in exceptional cases.

Velocity coupling

The velocity coupling corresponds internally to a setpoint coupling, but with lower dynamic requirements of the FS and LS. A servo loop is not needed for the FS and/or LS and no measuring systems are needed.

Threshold values for coarse/fine synchronism

After controller optimization and feedforward control setting, the threshold values for coarse and fine synchronism must be entered for the FS.

- Threshold value for “Coarse synchronism”
  axis spec. MD 37200: AV, DV: COUPLE_POS_TOL_COARSE
  MD 37220: VV: COUPLE_VELO_TOL_COARSE
- Threshold value for “Fine synchronism”
  axis spec. MD 37210: AV, DV: COUPLE_POS_TOL_FINE
  MD 37230: VV: COUPLE_VELO_TOL_FINE

The values must be calculated according to the accuracy requirements of the machine manufacturer (check via the PLC interface or in the FS Service display).

Angular offset LS/FS

If there must be a defined angular offset between the FS and LS, e.g. when synchronous operation is selected, the “zero degree positions” of the FS and LS must be mutually adapted. This can be done with the following machine data:

MD 34100 REFP_SET_POS
MD 34080 REFP_MOVE_DIST
MD 34090 REFP_ MOVE_DIST_CORR

References: /FB/, R1, “Reference Point Approach”

Service display for FS

The following values are displayed for the following spindle for start-up in synchronous operation in the “Service Values Axes” display in the “Diagnosis” operating area:

- Actual deviation between setpoints of FS and LS
  Display value: Position offset in relation to leading spindle (setpoint)
  (value corresponds to angular offset between FS and LS that can be read with axis variable $AA_COUP_OFFS in the part program)
- Actual deviation between actual values of FS and LS
  Display value: Position offset in relation to leading spindle (actual value)

■

Notes

Supplementary Conditions 3

Availability of "Synchronous spindle" function

This function is an option and available for

- SINUMERIK 840D, SW 2 and higher

Availability of "WAITC" function

This function is available together with synchronous spindle for

- SINUMERIK 840D, SW 3 and higher

Data Descriptions (MD, SD) 4

4.1 Description of machine data

4.1.1 Channel-specific machine data

21300 MD number	**COUPLE_AXIS_1[n]** Definition of synchronous spindle pair [n]		
Default value: 0	Min. input limit: 0	Max. input limit: 8	
Changes effective after Power On	Protection level: 2/7	Unit: –	
Data type: BYTE	Applies from SW version: 2.1		
Significance:	**One** synchronous spindle pair per NC channel can be defined in a **fixed configuration** with this machine data. The machine axis numbers (channel-specific MD: AXCONF_MACHAX_USED) applicable in the NC channel must be entered for the following spindle [n=0] and the leading spindle [n=1]. If a value of "0" is entered, then the coupling is not configured, thus leaving 2 couplings to be configured freely via the NC part program.		
MD irrelevant for	User-defined coupling		
Related to	Channel-specific MD; COUPLING_MODE_1 (type of coupling in synchronous spindle mode) Channel-specific MD: COUPLE_IS_WRITE_PROT_1 (write-protection for coupling parameters) Channel-specific MD: COUPLE_RESET_MODE_1 (coupling abortion response) Channel-specific MD: COUPLE_BLOCK_CHANGE_CTRL_1 (block change response in synchronous spindle mode) SD: $SC_COUPLE_RATIO_1 (speed ratio parameters for synchronous spindle mode)		

21310 MD number	**COUPLE_IS_DES_POS_1** Type of coupling in synchronous spindle mode		
Default value: 1	Min. input limit: 0	Max. input limit: 2	
Changes effective after Power On	Protection level: 2/7	Unit: –	
Data type: BOOLEAN	Applies from SW version: 2.1		
Significance:	This machine data determines the type of coupling for the fixed coupling configuration defined with machine data COUPLE_AXIS_1[n]. 1: Setpoint coupling activated. With a setpoint coupling, the reference value for the following spindle is calculated from the position setpoint of the leading spindle, allowing the setpoints for the FS and LS to be input simultaneously. This has a particularly positive effect on the spindle synchronism during acceleration and deceleration processes. A better response to setpoint changes is thus obtained with the setpoint coupling than with the actual-value coupling. When a setpoint coupling is selected, the following conditions must be fulfilled before synchronous operation is activated: • The LS must be assigned to the same NC channel as the FS • The FS and LS must be in position control mode (SPCON) • The FS and LS must have the same dynamic control response (see Section 2.4.3) 0: Actual-value coupling activated. With an actual-value coupling, the reference value for the following spindle is calculated from the position actual value of the leading spindle. With this type of coupling, the following drive must be significantly more dynamic than the leading drive, but never vice versa. The actual-value coupling can be used, for example, in the following applications: • The LS must be assigned to a different NC channel than the FS • For leading spindles which are not suitable for position control • In cases where the dynamic control response of the leading spindle is considerably slower than that of the following spindle. As soon as the actual-value coupling is active, the IS “Actual-value coupling” for the FS is set to “1”. 2: Speed coupling activated. The speed coupling is internally a setpoint coupling. The requirements placed on FS and LS are lower. A defined position relation between FS and LS cannot be established. In the following cases, the speed coupling is applied: • LS and/or FS are not in position control. • There are no measuring systems. The coupling type can be altered in the NC part program when the coupling is deactivated by means of language instruction COUPDEF provided that this option is not inhibited in channel-specific MD: COUPLE_IS_WRITE_PROT_1. The parameterized value of channel-specific MD: COUPLING_MODE_1 does not, however, get altered.		
MD irrelevant for	User-defined coupling		
Related to	Channel-specific MD; COUPLE_AXIS_1 (definition of synchronous spindle pair) Channel-specific MD: COUPLE_IS_WRITE_PROT_1 (write-protection for configured parameters) IS “Actual-value coupling” (DB31–48, DBX98.2)		

21320 MD number	**COUPLE_BLOCK_CHANGE_CTRL_1** Block change response in synchronous spindle mode		
Default value: 3	Min. input limit: 0	Max. input limit: 3	
Changes effective after Power On	Protection level: 2/7	Unit: –	
Data type: BYTE	Applies from SW version: 2.1		
Significance:	This machine data determines the condition on which a block change must be executed when synchronous mode is activated for the fixed coupling configuration defined in channel-specific machine data COUPLE_AXIS_1[n]. The following options are available: 0: Block change is enabled immediately 1: Block change in response to “Fine synchronism” 2: Block change in response to “Coarse synchronism” 3: Block change in response to IPOSTOP (i.e. after setpoint-based synchronism) The block change response can be altered in the NC part program with language instructions COUPDEF provided this option has not been inhibited with channel-specific MD: COUPLE_IS_WRITE_PROT_1. The parameterized value of channel-specific MD: COUPLE_BLOCK_CHANGE_CTRL_1 does not, however, get altered. The selected block change response remains valid event when the speed ratio is changed or a defined angular offset is programmed while the coupling is active.		
MD irrelevant for	User-defined coupling		
Related to	Channel-specific MD; COUPLE_AXIS_1 (definition of synchronous spindle pair) Channel-specific MD: COUPLE_IS_WRITE_PROT_1 (change of coupling parameters not possible) Channel-specific MD: COUPLE_POS_TOL_COARSE or COUPLE_VELO_TOL_COARSE (threshold value for coarse synchronism) Channel-specific MD: COUPLE_POS_TOL_FINE or COUPLE_VELO_TOL_FINE (threshold value for fine synchronism)		

21330 MD number	**COUPLE_RESET_MODE_1** Coupling abort response		
Default value: 1	Min. input limit: 0	Max. input limit: 0x3FF	
Changes effective after Power On	Protection level: 2/7	Unit: –	
Data type: BYTE	Applies from SW version: 2.1		
Significance:	The behavior of synchronism for the synchronous spindle pair configured with the machine data COUPLE_AXIS_n is defined with this machine date. Bit 0=0: Synchronism remains active with a new program start and canbe cancelled only with COUPOF as long as the control remains switched on. Bit 0=1: Synchronism is cancelled with program start (from the reset condition). Bit 1=0: Synchronism remains active even with program end and reset and can be cancelled only with COUPOF as long as the control remains switched on. Bit 1=1: Synchronism is cancelled with program end or RESET. Bit 5=1: The configured data are activated with program start. Bit 6=1: The configured data are activated with program end or RESET. Bit 9=1: Synchronism is switched on with program start. Note: Synchronism is not deselected with NC start after NC stop!		
MD irrelevant for	User-defined coupling		
Related to	Channel-specific MD; COUPLE_AXIS_1 (definition of synchronous spindle pair) IS “Synchronous operation” (DB31–48, DBX84.4)		

21340 MD number	**COUPLE_IS_WRITE_PROT_1** Coupling parameters are write-protected		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after Power On	Protection level: 2/7	Unit: –	
Data type: BOOLEAN	Applies from SW version: 2.1		
Significance:	This machine data is used to specify whether or not the coupling parameters (speed ratio, block change response, coupling type) for the synchronous spindle pair configured with channel-specific machine data COUPLE_AXIS_1[n] may be altered by the NC part program. 1: Coupling parameters may **not** be altered by the NC program (write-protection active) An alarm message is generated if an attempt is made to change the parameters. 0: NC part program may alter coupling parameters using language instructions COUPDEF.		
MD irrelevant for	User-defined coupling		
Related to	Channel-specific MD; COUPLE_AXIS_1 (definition of synchronous spindle pair) Channel-specific MD: COUPLING_MODE_1 (type of coupling in synchronous spindle mode) Channel-specific MD: COUPLE_RESET_MODE_1 (coupling abort response) Channel-specific MD: COUPLE_BLOCK_CHANGE_CTRL_1 (block change response in synchronous spindle mode) SD: $SC_COUPLE_RATIO_1 (speed ratio parameters for synchronous spindle mode)		

4.1.2 Axis-specific machine data

37200 MD number	**COUPLE_POS_TOL_COARSE** Threshold value for coarse synchronism		
Default value: 1.0	Min. input limit: 0.0	Max. input limit: PLUS	
Changes effective after NEW_CONF	Protection level: 2/7	Unit: Linear axis: mm Rotary axis: degrees	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	In synchronous operation, the positional deviation between the leading and following spindles is monitored (only DV and AV mode). IS “Coarse synchronism” is set if the current positional deviation is within the tolerance band specified by the threshold value. Furthermore, this threshold value represents the criterion for a block change on activation of synchronous operation or on alteration of the transmission parameters when the coupling is active in cases where “Coarse synchronism” is selected as the block change response condition (see channel-specific MD: COUPLE_BLOCK_CHANGE_CTRL_1 or language instruction COUPDEF). If the value “0” is input, IS “Coarse synchronism” is always set to “1” in DV and AV mode.		
Related to	Channel-specific MD; COUPLE_BLOCK_CHANGE_CTRL_1 (block change response in synchronous spindle operation) IS “Coarse synchronism” (DB31–48, DBX98.1)		

37210 MD number	**COUPLE_POS_TOL_FINE** Threshold value for fine synchronism		
Default value: 0.5	Min. input limit: 0.0	Max. input limit: PLUS	
Changes effective after NEW_CONF	Protection level: 2/7	Unit: Linear axis: mm Rotary axis: degrees	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	In synchronous operation, the positional deviation between the leading and following spindles is monitored (only DV and AV mode). IS “Fine synchronism” is set if the current positional deviation is within the tolerance band specified by the threshold value. Furthermore, this threshold value represents the criterion for a block change on activation of synchronous operation or on alteration of the transmission parameters when the coupling is active in cases where “Fine synchronism” is selected as the block change response condition (see channel-specific MD: COUPLE_BLOCK_CHANGE_CTRL_1 or language instruction COUPDEF). If the value “0” is input, IS “Fine synchronism” is always set to “1” in DV and AV mode.		
Related to	Channel-specific MD; COUPLE_BLOCK_CHANGE_CTRL_1 (block change response in synchronous spindle operation) IS “Fine synchronism” (DB31–48, DBX98.0)		

37220 MD number	**COUPLE_VELO_TOL_COARSE** “Coarse” speed tolerance between leading and following spindles		
Default value: 1.0	Min. input limit: 0.0	Max. input limit: PLUS	
Changes effective after NEW_CONF	Protection level: 2/7	Unit: Linear axis: mm/min Rotary axis: rpm	
Data type: DOUBLE	Applies from SW version: 3.1		
Significance:	In synchronous operation, the speed difference between the leading and following spindles is monitored (VV mode only). IS “Coarse synchronism” is set if the current speed difference is within the tolerance band specified by the threshold value. Furthermore, this threshold value represents the criterion for a block change on activation of synchronous operation or on alteration of the transmission parameters when the coupling is active in cases where “Coarse synchronism” is selected as the block change response condition (see channel specific MD: COUPLE_BLOCK_CHANGE_CTRL_1 or language instruction COUPDEF). If the value “0” is input, IS “Coarse synchronism” is always set to “1” in VV mode.		
Related to	Channel-specific MD; COUPLE_BLOCK_CHANGE_CTRL_1 (block change response in synchronous spindle operation) IS “Coarse synchronism” (DB31–48, DBX98.1)		

37230 MD number	**COUPLE_POS_TOL_FINE** “Fine” speed tolerance between leading and following spindles		
Default value: 0.5	Min. input limit: 0.0	Max. input limit: PLUS	
Changes effective after NEW_CONF	Protection level: 2/7	Unit: Linear axis: mm/min Rotary axis: rpm	
Data type: DOUBLE	Applies from SW version: 3.1		
Significance:	In synchronous operation, the speed difference between the leading and following spindles is monitored (VV mode only). IS “Fine synchronism” is set if the current speed difference is within the tolerance band specified by the threshold value. Furthermore, this threshold value represents the criterion for a block change on activation of synchronous operation or on alteration of the transmission parameters when the coupling is active in cases where “Fine synchronism” is selected as the block change response condition (see channel-specific MD: COUPLE_BLOCK_CHANGE_CTRL_1 or language instruction COUPDEF). If the value “0” is input, IS “Fine synchronism” is always set to “1” in VV mode.		
Related to	Channel-specific MD; COUPLE_BLOCK_CHANGE_CTRL_1 (block change response in synchronous spindle operation) IS “Fine synchronism” (DB31–48, DBX98.0)		

4.2 Description of setting data

42300 SD number	**COUPLE_RATIO_1[n]** Speed ratio parameters for synchronous spindle mode [n]		
Default value: 1.0	Min. input limit: –1000	Max. input limit: 1000	
Changes effective after NEW_CONF	Protection level: MMC-MD 9220	Unit: –	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	This setting data determines the speed ratio parameters for the fixed coupling configuration defined with channel-specific MD: COUPLE_AXIS_1[n]. The linear correlation between the leading and following spindles is determined by speed ratio $k_Ü$. This ratio is input by two speed ratio parameters in the form of numerator [n=0] and denominator [n=1], allowing the speed ratio to be specified very exactly. $k_Ü = \frac{\text{Speed ratio parameter nominator}}{\text{Speed ratio parameter denominator}} = \frac{\$SC\_COUPLE\_RATIO[0]}{\$SC\_COUPLE\_RATIO[1]}$ The speed ratio parameters can be altered in the NC part program with language instruction COUPDEF provided that this option is not inhibited with channel-specific COUPLE_IS_WRITE_PROT_1. The parameterized values of SD: $SC_COUPLE_RATIO_1 do not, however, get altered. The calculation of $k_Ü$ is initiated with POWER ON.		
SD irrelevant for	User-defined coupling		
Related to	SD: $SC_COUPLE_RATIO_1 currently has the same action as a machine data (e.g. active after POWER ON). The SD data are therefore displayed and input in the same way as channel-specific machine data.		
References	Channel-specific MD: COUPLE_AXIS_1 (definition of synchronous spindle pair)		

■

Signal Descriptions

5

5.1 Axis/spindle-specific signals

5.1.1 Signals from axis/spindle

DB31–48 **DBX84.4** Data block	**Synchronous mode** Signal(s) from axis/spindle to PLC (NCK → PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The spindle is operating in “Synchronous operation” mode. The following spindle is therefore following the motions of the leading spindle according to the transmission ratio. The monitoring functions for coarse and fine synchronism are implemented in synchronous operation. Note: The signal is set only for the machine axis which is acting as following spindle (IS “FS active“ = 1)	
Signal state 0 or signal transition 1 ——> 0	The spindle is not operated as the following spindle in “synchronous mode”. When the coupling is deactivated (deselection of synchronous operation), the following spindle is switched to “open-loop control mode”.	
Related to	IS “Coarse synchronism” (DB31–48, DBX98.1) IS “Fine synchronism” (DB31–48, DBX98.0) IS “FS active” (DB31–48, DBX99.1)	

DB31–48 **DBX98.0** Data block	**Fine synchronism** Signal(s) from axis/spindle to PLC (NCK → PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The positional deviation between the following spindle and its leading spindle is within the “Fine synchronism” tolerance band (see Section 2.1.5).	
Signal state 0 or signal transition 1 ——> 0	The positional deviation between the following spindle and its leading spindle is within the “Fine synchronism” tolerance band (see Section 2.1.5). Note: The signal is relevant only for the following spindle in synchronous operation.	
Application example	Clamping of workpiece in following spindle on transfer from the leading spindle: Clamping of the workpiece is not initiated by the PLC user program until the spindles are sufficiently synchronized.	
Related to	IS “Synchronous operation” (DB31–48, DBX84.4) MD: $MA_COUPLE_POS_TOL_FINE threshold value for fine synchronism or MD: $MA_COUPLE_VELO_TOL_FINE “fine” speed tolerance	

DB31–48 **DBX98.1** Data block	**Coarse synchronism** Signal(s) from axis/spindle to PLC (NCK → PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The positional deviation between the following spindle and its leading spindle is within the "Coarse synchronism" tolerance band (see Section 2.1.5). Note: The signal is relevant only for the following spindle in synchronous operation.	
Signal state 0 or signal transition 1 ——> 0	The positional deviation between the following spindle and its leading spindle is not within the "Coarse synchronism" tolerance band (see Section 2.1.5).	
Application example	Clamping of workpiece in following spindle on transfer from the leading spindle: Clamping of the workpiece is not initiated by the PLC user program until the spindles are sufficiently synchronized.	
Related to	IS "Synchronous operation" (DB31–48, DBX84.4) Axis-specific MD: COUPLE_POS_TOL_COARSE threshold value for coarse synchronism or axis-specific MD: COUPLE_VELO_TOL_COARSE "coarse" speed tolerance	

DB31–48 **DBX98.2** Data block	**Actual-value link** Signal(s) from axis/spindle to PLC (NCK → PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The actual-value coupling is active as the coupling type between the leading and following spindles (see channel-specific MD: COUPLING_MODE_1). Note: The signal is relevant only for the active following spindle in synchronous operation.	
Signal state 0 or signal transition 1 ——> 0	The setpoint coupling is active as the coupling type between the leading and following spindles (see channel-specific MD: COUPLING_MODE_1).	
Special cases, errors, ...	In the case of faults/disturbances on the following spindle which result in cancellation of the FS "servo enable", the coupling relationship between the FS and LS is reversed and switched over to an actual-value coupling internally in the control under certain circumstances.	
Related to	IS "Synchronous operation" (DB31–48, DBX84.4) Channel-spec. MD: COUPLING_MODE_1 (coupling type in synchr. spindle oper.)	

DB31–48 **DBX98.4** Data block	**Overlaid motion** Signal(s) from axis/spindle to PLC (NCK → PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The following spindle traverses an additional motional component which is overlaid on the motion from the coupling with the leading spindle. Examples of overlaid movement of FS: • Activation of synchronous operation with defined angular offset between FS and LS • Activation of synchronous operation with LS in rotation • Alteration of transmission ratio when synchronous operation is selected • Input of a new defined angular offset when synchronous operation is selected • Traversal of FS with plus or minus traversing keys or handwheel in JOG when synchronous operation is selected As soon as the FS executes an overlaid movement, IS "Fine synchronism" or IS "Coarse synchronism" (depending on threshold value) may be cancelled immediately. Note: The signal is relevant only for the following spindle in synchronous operation.	
Signal state 0 or signal transition 1 ——> 0	The following spindle does not traverse any additional motional component or this motion has been terminated.	
Related to	IS "Synchronous operation" (DB31–48, DBX84.4)	

DB31–48 **DBX99.0** Data block	**LS (leading spindle) active** Signal(s) from axis/spindle to PLC (NCK → PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The machine axis is currently active as the leading spindle. Note: The signal is relevant only in synchronous operation.	
Signal state 0 or signal transition 1 ——> 0	The machine axis is not currently active as the leading spindle.	
Related to	In the case of faults/disturbances on the following spindle which result in cancellation of the FS “servo enable”, the coupling relationship between the FS and LS is reversed and switched over to an actual-value coupling internally in the control under certain circumstances. In this case, the leading spindle becomes the new, active following spindle (IS “FS active”).	
Related to	IS “Synchronous operation” (DB31–48, DBX84.4) IS “FS active” (DB31–48, DBX99.1)	

DB31–48 **DBX99.1** Data block	**FS (following spindle) active** Signal(s) from axis/spindle to PLC (NCK → PLC)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	The machine axis is currently operating as the following spindle. The following spindle thus follows the movements of the leading spindle in synchronous operation in accordance with the transmission ratio. Note: The signal is relevant only in synchronous operation.	
Signal state 0 or signal transition 1 ——> 0	The machine axis is not currently operating as the following spindle.	
Related to	In the case of faults/disturbances on the following spindle which result in cancellation of the FS “servo enable”, the coupling relationship between the FS and LS is reversed and switched over to an actual-value coupling internally in the control under certain circumstances.	
Related to	IS “Synchronous operation” (DB31–48, DBX84.4) IS “LS active” (DB31–48, DBX99.0)	

■

Notes

Examples

6

```
                                        ; Leading spindle = master spindle =
                                        ; spindle 1
                                        ; Following spindle = spindle 2
N05   M3 S3000  M2=4 S2=500             ; Leading spindle rotates at 3000/min
                                        ; FS: –500/min.
N10   COUPDEF (S2, S1, 1, 1, “No”, “Dv”)        ; Definition of coupling;
                                                ; can also be configured
...
N70   SPCON                             ; Take leading spindle into position control
                                        ; (Setpoint coupling).
N75   SPCON(2)                          ; Take following spindle into position control.
N80   COUPON (S2, S1, 45)               ; On-the-fly coupling
                                        ; to offset position = 45 degrees
...
N200  FA [S2] = 100                     ; Positioning speed = 100 degrees/min
N205  SPOS[2] = IC(–90)                 ; Travel 90 degrees overlaid in the negative
                                        ; direction
N210  WAITC(S2, “Fine”)                 ; Wait for “fine” synchronism
N212   G1 X.., Y..  F...                ; Processing
...
N215  SPOS[2] = IC(180)                 ; Travel 180 degrees overlaid in the positive
                                        ; direction
N220  G4 S50                            ; Dwell = 50 rotations ofmaster spindle
N225   FA [S2] = 0                      ; Activate configured speed (MD).
N230  SPOS[2] = IC (–7200)              ; 20 rotations with configured speed
; in the negative direction
...
N350  COUPOF (S2, S1)                   ; On-the-fly decoupling, S = S2 = 3000
N355  SPOSA[2] = 0                      ; Stop FS at zero degrees
N360   G0 X0 Y0
N365 WAITS(2)                           ; Wait for spindle 2
N370  M5                                ; Stop FS
N375   M30
```

■

Notes

Data Fields, Lists

7

7.1 Interface signals

DB number	Bit, byte	Name	Reference
Channel-specific			
21, ...	7.1	NC start	K1
21, ...	7.4	NC stop axes plus spindle	K1
21, ...	24.6	Dry run feedrate selected	V1
21, ...	25.3	Feedrate override for rapid traverse selected	V1
Axis/spindle-specific			
31, ...	1.3	Axis/spindle disable	A2
31, ...	1.4	Follow-up mode	A2
31, ...	1.5/1.6	Position measuring system 1, position measuring system 2	A2
31, ...	2.1	Servo enable	A2
31, ...	2.2	Spindle RESET	A2
31, ...	4.3	Spindle stop/feed stop	V1
31, ...	4.6–4.7	Traversing keys for JOG	V1
31, ...	16.4/16.5	Re-synchronize spindle 1, re-synchronize spindle 2	S1
31, ...	16.7	Delete S value	S1
31, ...	17.0	Feedrate override valid	S1
31, ...	19	Spindle speed override	V1
31, ...	60.4/60.5	Referenced/synchronized 1, Referenced/synchronized 2	R1
31, ...	84.4	Synchronous operation	
31, ...	98.0	Fine synchronism	
31, ...	98.1	Coarse synchronism	
31, ...	98.2	Actual-value coupling	
31, ...	98.4	Overlaid movement	
31, ...	99.0	LS/LA active	
31, ...	99.1	FS/FA active	

7.2 Machine data

Number	Identifier	Name	Reference
General ($MN_...)			
10000	AXCONF_MACHAX_NAME_TAB	Machine axis name	K2
Channel-specific ($MC_...)			
21300	COUPLE_AXIS_1	Definition of synchronous spindle pair	
21320	COUPLE_BLOCK_CHANGE_CTRL_1	Block change response in synchronous spindle operation	
21310	COUPLE_IS_DES_POS_1	Coupling type in synchronous spindle operation	
21330	COUPLE_RESET_MODE_1	Coupling abort response	
21340	COUPLE_IS_WRITE_PROT_1	Write-protection for coupling parameters	
20070	AXCONF_MACHAX_USED	Machine axis number valid in channel	K2
Axis/spindle-specific ($MA_...)			
30550	AXCONF_ASSIGN_MASTER_CHAN	Initial setting of channel for change of axis	K5
32200	POSCTRL_GAIN	Servo gain factor	G2
32400	AX_JERK_ENABLE	Axial jerk limitation	B2
32410	AX_JERK_TIME	Time constant for axial jerk filter	B2
32420	JOG_AND_POS_JERK_ENABLE	Basic setting of axial jerk limitation	B2
32430	JOG_AND_POS_MAX_JERK	Axial jerk	B2
32610	VELO_FFW_WEIGHT	Feedforward control factor for speed feedforward control	K3
32620	FFW_MODE	Feedforward control type	K3
32650	AX_INERTIA	Moment of inertia for torque feedforward control	K3
32800	EQUIV_CURRCTRL_TIME	Equivalent time constant, current control loop for feedforward control	K3
32810	EQUIV_SPEEDCTRL_TIME	Equivalent time constant, speed control loop for feedforward control	K3
34080	REFP_MOVE_DIST	Reference point distance	R1
34090	REFP_MOVE_DIST_CORR	Reference point offset	R1
34100	REFP_SET_POS	Reference point value	R1
35000	SPIND_ASSIGN_TO_MACHAX	Assignment of spindle to machine axis	S1
37200	COUPLE_POS_TOL_COARSE	Threshold value for coarse synchronism	
37210	COUPLE_POS_TOL_FINE	Threshold value for fine synchronism	
37220	COUPLE_VELO_TOL_COARSE	Speed tolerance “coarse” between leading and following spindles	
37230	COUPLE_VELO_TOL_FINE	Speed tolerance “fine” between leading and following spindles	

7.3 Setting data

Number	Identifier	Name	Refer-ence
Axis-specific ($SA_...)			
42300	COUPLE_RATIO_1	Transmission parameters for synchronous spindle operation	

7.4 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

■

Notes

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

Synchronized Actions (S5)

up to and including SW 3

■

Notes

Brief Description

1

Auxiliary function output to PLC

For software version 4 and higher, a description of the auxiliary functions is contained in /FB/ H2, Auxiliary Function Output to PLC

Motion-synchronous actions

With the support of motion-synchronous actions, it is possible to execute programmed actions in synchronism with the block and/or along the programmed path as a function of programmed conditions.

The following are some of the possible actions:

- Output of M and H auxiliary functions to the PLC
- Output of digital and analog signals
- Execution of NC commands parallel to machining (e.g. delete distance-to-go)
- Switchover between several feedrates in a programmed block as a function of binary and/or analog signals
- Start of evaluation function (e.g. for AC control)
- Feedrate overrides
- Axial motion overlays
  this enables e.g. to implement a distance control for laser processing.

These actions can be started as a function of parameterized conditions such as:

- Status of a digital or analog input
- Status of a digital or analog output
- Actual position of an axis
- Time from block start
- Distance from block start
- Distance to block end

With SW version 3.2 and higher, it is possible to form the condition by comparison of the status sizes in the IPO cycle.

Example:
Comparison of the actual value of an axis with the setting data for the reversal oscillation position. Any changes to the setting data are effective at once.

■

Notes

Detailed Description 2

2.1 Auxiliary function output to PLC

For SW 4 and higher, a description of the signals of the auxiliary functions is provided in /FB/ H2, Auxiliary Function Output to PLC

2.2 Motion-synchronous actions (up to and including SW 3)

2.2.1 General motion-synchronous actions

Application

Reaction to events/states during execution of a motion block.

Motion-synchronous actions allow the user to initiate actions **in synchronism with machining operations**. The instant of activation of these actions can be defined by a condition which means that actions of this type do not need to be started in connection with block limits.

These synchronous actions are performed in the IPO cycle.

Examples of typical motion-synchronous actions are:

- **Transfer of M and H auxiliary functions to the PLC**
  The auxiliary functions programmed in an NC block are transferred to the PLC as a function of a relational operation or an external signal.
- **Axis-specific deletion of distance-to-go**
  For example, high-speed inputs effect a conditional stop and deletion of distance-to-go with respect to path or positioning axes.
- **Programmed read-in disable**
  The read-in disable can be set or cancelled in the part program as a function of, for example, an external input.

Schematic

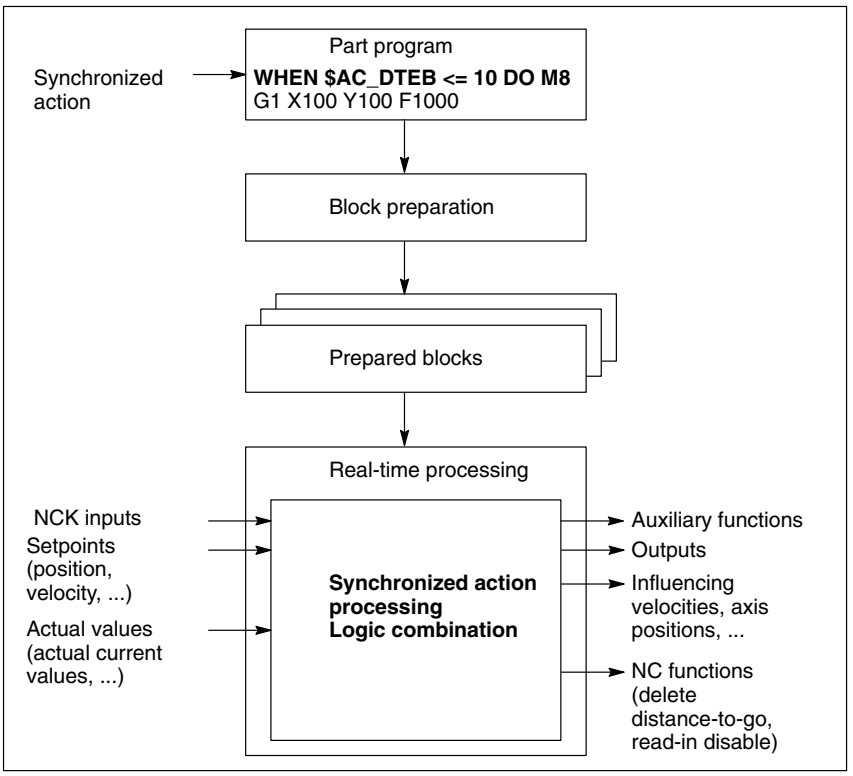


Fig. 2-1 Motion-synchronous actions

Functionality and syntax

A motion-synchronous action consists of a condition, up to 16 associated synchronous commands and, in some cases, an identification number.

The schematic of motion-synchronous actions is as follows:

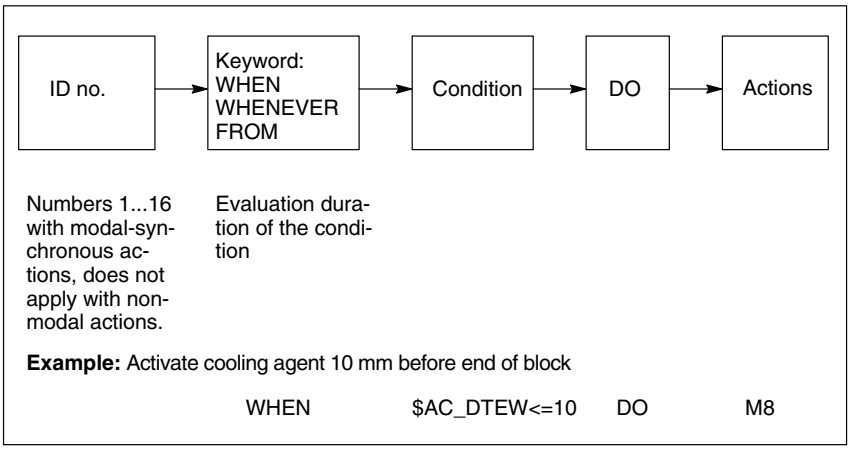


Fig. 2-2 Structure of motion-synchronous actions

Note

A motion-synchronous action is positioned on its own in a block and acts in the next block with a machine function (e.g. block with G01, G02, G04, auxiliary function output).

The instruction can act modally or non-modally.

A total maximum of 16 modal and non-modal motion-synchronous actions can be operative in a machining block.

Several actions can be specified for a condition, e.g.
... DO M8 M800.

A motion-synchronous action is terminated with the end of the output block unless it is identified by means of an identification number ('ID='<number>;) as having a modal action. Identification numbers from 1 to 16 may be used in each channel.

Motion-synchronous actions are processed in order of their ID numbers (i.e. block with ID number 1 before block with ID number 2...). After processing of the modal motion-synchronous actions, the non-modal actions are processed in the order in which they were programmed.

Note

If a variable is written several times by motion-synchronous actions, then the last value to be written is valid.

Modal motion-synchronous actions can be terminated with the 'CANCEL('<expression>{1,16} ')' command.

The system variable $PC_ACTID[<1, 16>] can be used to read whether an ID number is currently active.

Depending on the keyword programmed in the synchronized action instruction, the function responds as follows:

Table 2-1 Keywords for synchronized action instructions

Keyword	Response
‘WHEN’...’DO’...	A condition is evaluated continuously until it is fulfilled once. The synchronous commands programmed after ’DO’ are then executed **once**.
‘WHENEVER’...’DO’...	The synchronous commands are executed cyclically for **as long as** the condition is fulfilled.
‘FROM’...’DO’...	The synchronous commands are executed cyclically **if** the condition is fulfilled **once**.
– – – ...’DO’...	In the absence of ’WHEN’, ’WHENEVER’ or ’FROM’, the condition is regarded as being permanently fulfilled and the synchronous commands are executed cyclically.

Conditions

The condition is the comparison of a real-time variable with a value calculated at the time of block preparation:

WHEN $AA_IM[Y] > 10 * SIN(R10) DO ...

or the comparison of two real-time variables:

WHEN $AC_VACTW > $$A_INA[1] DO ...

In this case, the real time variable must be marked on the right with **$$** as real-time variable.

The form of the conditions which may be used in motion-synchronous actions is as follows:

<read_real-time variable><relational operator><**$$**read real-time variable>
<read_real-time variable><relational operator><expression>

Note

The data types BOOL, CHAR, INTEGER and REAL are admissible in the condition. The comparison is made on the left of the data type of the real-time variable. The type conversion rules for value allocation apply. If two real-time variables are compared, these must have the same data type.

A block change to the following block without fulfillment of the specified conditions results in the following response:

- Non-modal motion-synchronous actions are made inoperative
- Modal motion-synchronous actions remain operative

All motion-synchronous actions are deleted by a reset.

Example

Priorities when several motion-synchronous actions are programmed

```
ID= 2 WHEN ($A_IN[1] = 1) DO $A_OUTA[3]= 20
      :
      :
ID= 1 WHEN ($A_IN[5] = 1) DO $A_OUTA[3]= 5
      :
WHEN ($A_IN[3] = 1) DO $A_OUTA[4]= 10
WHEN ($A_IN[5] = 1) DO $A_OUTA[4]= 30
WHEN ($A_IN[7] = 1) DO $A_OUTA[3]= 7
```

Since the modal motion-synchronous actions are processed in ascending order of their ID numbers and the non-modal actions have priority, analog output 3 is set to a value of 7 and output 4 to a value of 30 in the next block with a machine function if inputs 1, 3, 5 and 7 are simultaneously set.

Real time variable evaluation

With SW 3.2 and higher, it is possible to compare the synchronization conditions in the IPO cycle with real-time variable with the current actual values ($$ variable on the right or left of comparison conditions). With normal system variable comparison, the expressions are evaluated in the first run.

With comparison of the $$ real-time variable, the change of the reversal position during oscillation may become effective during oscillation in synchronization.

References: /FB/, P5 “Oscillation”

Example 1

On the left-hand side of the comparison, there is a variable evaluated in real time while, on the right-hand side any random expression is shown and not any of the permitted real-time processing variables that start with $$.

WHEN $AA_IM[X] > **$A_INA**[1] DO M120

M120 is output during the movement programmed in the following block if the actual value of the X axis is greater than the value of analog input 1. The actual value is reevaluated in each IPO cycle while the value of the analog input is formed at the time of interpretation.

The comparison is initiated in real time.

Example 2

On the left-hand side there is a comparison variable evaluated in real time and on the right-hand side of the comparison a real-time variable permitted for the synchronized action that begins with $$.

WHEN $AA_IM[X] > **$$A_INA**[1] DO M120

Comparison of the current actual value of the X axis in IPO cycles with the analog input 1 because a $$ variable is on the right-hand side of the comparison.

The two variables are compared in IPO cycles.

Example 3

$$ variables are permissible on the left-hand side of the comparison.

WHEN **$$AA_IM[X]** > **$$A_INA[1]** DO M120

Identical to example 2. The left-hand and right-hand sides are always compared in real time.

Real-time variables As a condition, the following real-time variables can be read, their status scanned and a comparison made between them and another value:

Table 2-2 Read real-time variable

Real-time variable	Meaning	Type
$A_IN[<arith. expression>]	Digital input	BOOL
$A_OUT[<arith. expression>]	Digital output	BOOL
$A_INA[<arith. expression>]	Analog input	REAL
$A_OUTA[<arith. expression>]	Analog output	REAL
$A_INCO[<arith. expression>]	Comparator inputs	BOOL
$AA_IW[<axial expression>]	Actual axis position in WCS	REAL
$AA_IB[<axial expression>]	Actual axis position in BCS	REAL
$AA_IM[<axial expression>]	Actual axis position in MCS (IPO setpoints) $AA_IM[S1] can be used to evaluate actual values for spindles. Depending on the machining data $MA_ROT_IS_MODULO and $MA_DISPLAY_IS_MODULO modulo calculation is made for spindles and rotary axes.	REAL
$AC_TIME	Time from block start in s	REAL
$AC_TIMEC	Time from block start in IPO cycles	REAL
$AC_DTBB	Distance to block start in BCS	REAL
$AC_DTBW	Distance to block start in WCS	REAL
$AA_DTBB[<axial expression>]	Axial path from block start in BCS (applies to positioning and synchronous axes)	REAL
$AA_DTBW[<axial expression>]	Axial path from block start in WCS (applies to positioning and synchronous axes)	REAL
$AC_DTEB	Distance to block end in BCS	REAL
$AC_DTEW	Distance to block end in WCS	REAL
$AA_DTEB[<axial expression>]	Axial path to end of movement in BCS (applies to positioning and synchronous axes)	REAL
$AA_DTEW[<axial expression>]	Axial path to end of movement in WCS (applies to positioning and synchronous axes)	REAL
$AC_PLTBB	Path distance from block start in BCS	REAL
$AC_PLTEB	Path distance to block end in BCS	REAL
$AC_PATHN	Normalized path parameter (0: block beginning, 1: block end)	REAL
$AC_VACTB	Path velocity in BCS	REAL
$AC_VACTW	Path velocity in WCS	REAL
$AA_VACTB[<axial expression>]	Axis velocity in BCS (valid for positioning axes)	REAL
$AA_VACTW[<axial expression>]	Axis velocity in WCS	REAL
$AA_DTEPB[<axial expression>]	Axial distance-to-go for infeed oscillation in BCS References: /FB/, P5, “Oscillation”	REAL
$AA_DTEPW[<axial expression>]	Axial distance-to-go for infeed oscillation in WCS References: /FB/, P5, “Oscillation”	REAL
$AA_OSCILL_REVERSE_POS1 [<axial expression>]	Current reversal position 1 for oscillation References: /FB/, P5, “Oscillation”	REAL
$AA_OSCILL_REVERSE_POS1 [<axial expression>]	Current reversal position 1 for oscillation References: /FB/, P5, “Oscillation”	REAL

Table 2-2 Read real-time variable

Real-time variable	Meaning	Type
$AA_LOAD[<axial expression>]	Drive capacity utilization (for 611D only) Unit: % The drive capacity utilization is specified by the ratio torque setpoint/torque limit value. The response to a reduction in the value applied to the highest-priority input can be defined in MD: AC_FILTER_TIME (filter smoothing time constant for adaptive control) can be set to smooth the actual value via a settable filter in the NCK. The response to a reduction in the value applied to the highest-priority input can be defined in MD: DRIVE_SIGNAL_TRACKING (acquisition of additional drive actual values) can be set to activate sensing of drive capacity utilization.	REAL
$AA_POWER[<axial expression>]	Drive active power (for 611D only) Unit: W The actual value can be smoothed by means of a settable filter in the NCK with MD: AC_FILTER_TIME. The response to a reduction in the value applied to the highest-priority input can be defined in MD: DRIVE_SIGNAL_ TRACKING can be set to activate sensing of drive loading.	REAL
$AA_TORQUE[<axial expression>]	Drive torque setpoint (for 611D only) Unit: Nm There are two smoothing filters: 1. In the 611D 2. In the NCK area (parameterized in MD: AC_FILTER_TIME) The two filters can be connected in series if both a strong and a weak smoothing action is required in the system. Otherwise, the second filter must be deactivated by entering “0” as the smoothing time. The response to a reduction in the value applied to the highest-priority input can be defined in MD: DRIVE_SIGNAL_TRACKING can be set to activate sensing of the drive torque setpoint.	REAL
$AA_CURR[<axial expression>]	Actual current value of axis or spindle (for 611D only) Unit: A value of “1” corresponds to 1 A. There are two smoothing filters: 1. In the 611D 2. In the NCK area (parameterized in MD: AC_FILTER_TIME) The two filters can be connected in series if both a strong and a weak smoothing action is required in the system. Otherwise, the second filter must be deactivated by entering “0” as the smoothing time. The response to a reduction in the value applied to the highest-priority input can be defined in MD: DRIVE_SIGNAL_TRACKING can be set to activate sensing of the actual current value.	REAL
$AC_MARKER[<arith. expression>]	Marker variable: Can be used in synchronized actions for the generation of complex conditions. There are 8 markers (index 0...7). With reset, the markers are set to 0. Example: WHEN ... DO $AC_MARKER[0]=2 WHEN ... DO $AC_MARKER[0]=3 WHEN $AC_MARKER[0]==3 DO $AC_OVR=50 The parameters can be read and overwritten in the part program even independently of synchronized actions: IF$AC_MARKER[0] == 4 GOTOF SPRUNG	IN-TEGER

Table 2-2 Read real-time variable

Real-time variable	Meaning	Type
$AC_PARAM[<arith. expression>]	Floating point parameter for synchronized actions. Serves for buffering and evaluating synchronized actions. There are 50 parameters (index 0...49). Example: WHEN $A_IN[3]==1 DO $AC_PARAM[49] = $$AA_IM[X]) WHENEVER $AC_PARAM[1]>=$$AC_PARAM[49] DO ... FCTDEF(1,–1000,1000,0,1,–2) DO SYNFCT(1,$AC_OVR, $AC_PARAM[49]) If input 3 is applied, the actual value of the X axis is stored in parameter 0. The parameters can also be read and written in the part program independently of synchronized actions.	REAL

Note

Details about the resolution and rated range of the analog input and output modules used can be found in:

References: /PHD/, SINUMERIK 840D, NCU 571–573 Manual
/PHF/, SINUMERIK FM-NC, NCU 570 Manual
/S7H/, SIMATIC S7, Manual

The distance from the block start or to the block end on paths is always the distance between the point on the center path of the tool and the programmed start or end point.

Contrary to this, the path distance from block start/block end is indicated via $AC_PLTBB and $AC_PLTEB.

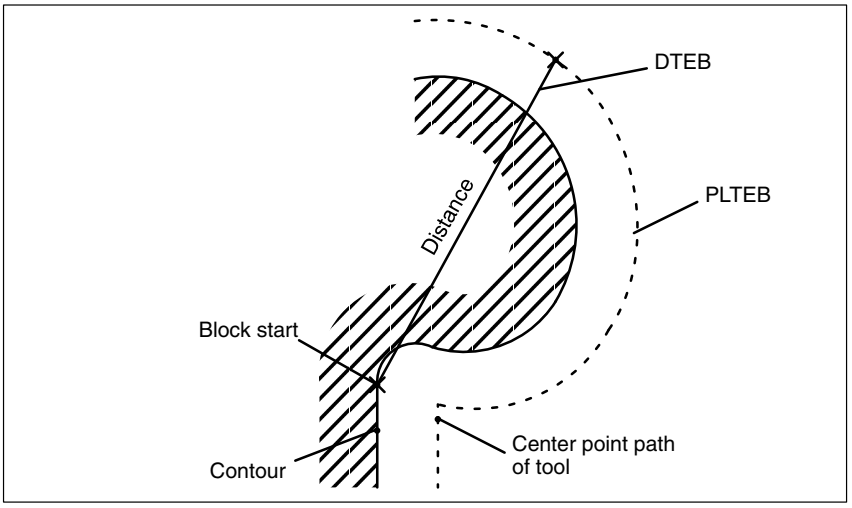


Fig. 2-3 Calculation of distance for path conditions

Example

Activation of auxiliary function before end of block

WHEN $AC_DTEW<=10 DO M8

Activation of cooling agent 10 mm before end of block.

Action/ synchronous commands

The following synchronous commands can be programmed as the reaction to a fulfilled condition:

- Write data to real-time variable
- Output of M and H auxiliary functions to the PLC
- Activation of synchronous procedures
- Activation of evaluation functions

< write_real_time_variable > '=' < read_real_time_variable >
< write_real_time_variable > '=' < expression >
< auxiliary function >
< synchronous_procedure >
< evaluation_function >

If there is a $$ real-time variable admissible for the synchronized action on the right of the allocation, the value is formed currently in the IPO cycle.

Example:
DO **$$**A_OUT[1] = **$$**AA_IB[Z] ; The actual value is set to the output in the ; IPO cycle.

If there is an arbitrary expression on the right, the value is formed for block preparation. Variables with $ or $$ are admissible on the left side.

Write real-time variable

A value or status can be assigned to one of the following real-time variables as a synchronous command:

Table 2-3 Write real-time variable

Real-time variable	Meaning	Type
$A_OUT[<arith. expression>]	Digital output	BOOL
$A_OUTA[<arith. expression>]	Analog output (see under "Constant analog value output" in Section 3) Unit: A value of 1000 corresponds to 1 V.	REAL
$AC_MARKER[<arith. expression>]	Marker variable: Can be used in synchronized actions for the generation of complex conditions. There are 8 markers (index 0 ... 7). With reset, the markers are set to 0. Example: WHEN ... DO $AC_MARKER[0]=2 WHEN ... DO $AC_MARKER[0]=3 WHEN $AC_MARKER[0]==3 DO $AC_OVR=50	IN-TEGER
$AC_VC	Additive path feedrate override	REAL
$AA_VC[<axial expression>]	Additive axis feedrate override (applies only to positioning axes). The effective feedrate override is calculated as follows: $F_{eff} = F_{prog} + F_{over}$ or $FA[X]_{eff} = FA[X]_{prog} + FA[X]_{over}$ The additional feed offset is interpreted in the current feed unit (e.g. mm/min, degree/min). The additive path feedrate override is not effective with G0, G33, G331, G332 and G63. When override = 0, the value entered in the real-time variable is not effective; otherwise the override has no effect on the override value. The total feedrate cannot become negative as a result of the override value. An upper limit is imposed to ensure that the maximum axis speeds and acceleration rates and the path velocities stipulated by the Look Ahead are not exceeded.	REAL

Table 2-3 Write real-time variable

Real-time variable	Meaning	Type
$AC_OVR	Path override factor for synchronized actions.	REAL
$AA_OVR[<axial expression>]	Axial override factor for synchronized actions (applies only to positioning axes). Multiplicative override component which acts in addition to the programmed override factors or to those specified by handwheel or via PLC interface. The effective total factor is limited to 200%. The multiplicative override component must be rewritten in every IPO cycle, otherwise a value of 100%) is applied $AA_OVR[S1] is used to modify the spindle speed override. The override values defined in machine data MD: $MN_OVR_FACTOR_LIMIT_BIN, MD: $MN_OVR_FACTOR_FEEDRATE[30], MD: $MN_OVR_FACTOR_AX_SPEED[30], MD: $MN_OVR_FACTOR_SPIND_SPEED must not be exceeded with additive feedrate override. The additive feedrate override is limited in such a way that the resulting feedrate does not exceed the maximum override value of the original feedrate.	REAL
$AA_OFF	Overlaid movement for the programmed axis. The offset is traversed immediately independently of whether the axis is programmed or not. In this way, a distance control can be implemented. The type of calculation is defined via the axial MD: $MA_AA_OFF_MODE (proportional or integrating). The speed of the overlaid movement is limited by axial MD: $MA_CORR_VELO. The offset is reset on Reset. No axial movement is performed. The current position is resynchronized. The current offset is not displayed separately. It is indicated by the difference between the MCS and the WCS in the actual value display.	REAL
$AC_PARAM[<arith. expression>]	Floating point parameter for synchronized actions. Used to buffer and evaluate synchronized actions. 50 parameters (index 0...49) are available. Example: WHEN $A_IN[3]==1 DO $AC_PARAM[0] = $$AA_IM[X] If input 3 is set the actual value of the X axis in parameter 0 is stored. The parameters can also be read and written in the part program independently of synchronized actions.	REAL

Example 1 Set a digital output

WHEN $AA_IW[Q1]>10 DO $A_OUT[3]=1

If the actual value of axis Q1 exceeds 10mm (assuming that the metric measuring system is selected), set output 3.

Example 2 Analog output

WHEN $A_IN[3]=TRUE DO $A_OUTA[5]=700

If digital output 3 is set, output a 0.7 V voltage at output 5 (assuming value of 1000 corresponds to 1 V).

Output of auxiliary functions to PLC

An M or H auxiliary function can be output to the PLC as a synchronous command.

A total of 10 M or H auxiliary functions in a machining block can be output as synchronous commands.

The auxiliary functions are output to the PLC if the associated conditions are fulfilled. The output time specified in MD: AUXFU_GROUP_SPEC (auxiliary function group specification) or AUXFU_[M, H]_SYNC_TYPE (instant of output of [M, H] functions) are not effective.

The PLC acknowledges an auxiliary function either after a complete PLC user cycle or immediately depending on whether it is defined as a normal or as a high-speed auxiliary function.
The block change is not affected by the acknowledgment.

An auxiliary function may only be output **once** to the PLC, i.e. it may only be programmed with keyword “WHEN” and with a non-modal action.

M functions in groups 1 to 6 and M6 or the M functions for tool change set via MD may **not** be output as motion-synchronous actions.

Example

Several auxiliary functions in block

WHEN $AA_IW[Q1]>5 DO M172 H510

If the actual value of axis Q1 exceeds 5 mm, output auxiliary functions M172 and H510 to the PLC interface.

Synchronous procedure

One of the following synchronous procedures can be activated as a synchronous command.

Table 2-4 Synchronous procedures

Synchronous procedure	Meaning
‘RDISABLE’	**Programmed read-in disable:** If a block is processed after a synchronized action was activated with action RDISABLE, then further processing of the block is stopped if the associated condition is fulfilled. Only the programmed motion-synchronous actions are processed. RDISABLE means interruption of continuous-path operation, even if the read-in disable is not active. If the condition for the RDISABLE instruction is not longer fulfilled, the read-in disable is cancelled. Examples are shown below. **Application:** This procedure allows, for example, the program to be started in the IPO cycle as a function of external inputs.
‘STOPREOF’	**Acknowledgment of preprocessing stop:** A motion-synchronous action with a STOPREOF command stops block preparation after the next machining block. The stop is cancelled again at the end of the machining block or if the condition for the STOPREOF command is fulfilled. STOPREOF may only be programmed with keyword “WHEN” and as a non-modal command. **Application:** Rapid program branching at block end.

Table 2-4 Synchronous procedures

Synchronous procedure	Meaning
‘DELDTG’	Delete distance-to-go for path axes
‘DELDTG’ (<axial expression>)	**Delete distance-to-go for positioning axes** A motion-synchronous action with a DELDTG command effects a preprocessing stop after the next block with machine function. If the condition for the DELDTG command is fulfilled, delete distance-to-go is implemented and the preprocessing stop is cancelled. The preprocessing stop is also cancelled at the end of the block with machine function. No continuous-path mode or overgrinding is possible in a block for which DELDTG applies. Continuation of the following block is faster if G603: “Block change with IPO end” is active. The distance to the programmed target point (in WCS) is entered in system variable $AC_DELT if the path distance-to-go has been deleted. The axial distance to the target point for axial deletion of distance-to-go is stored in system variable $AA_DELT[<axis>]. ’DELDTG’ and ’DELDTG’[<axial expression>] may only be programmed with keyword ’WHEN’ and as a non-modal command. **Application:** Deletion of distance-to-go for path or positioning axes.

Example 1 Programmed read-in disable

a)
N10 WHEN $A_IN[1] == 0 DO RDISABLE
N15 G01 X10 F1000
N20 Y20
...
If input 1 at the end of N15 has value 0, then changing N20 is delayed until the input has value 1. During the blocking, exact stop is triggered between N15 and N20.
Application:
RDISABLE can be used if the continuation of processing at a part program location depends on reception of an enabling signal. The example can be used so that processing is only continued in N20 when an enabling signal from the PLC is present. (For example, another axis has left the working space or the door to the automatic tool changer is open).

b)

WHENEVER $A_INA[2]<7000 DO RDISABLE

If the voltage at input 2 drops below 7 V, program processing is stopped (assuming value of 1000 corresponds to 1 V).

Example 2

Program branching

```
WHEN $AC_DTEB<5 DO STOPREOF
G01 X100
IF $A_INA[7]>5000 GOTOF Label 1
```

If the distance to the block end is less than 5 mm, terminate preprocessing stop. If the voltage exceeds 5 V at input 7, jump forward to label 1 (assuming value of 1000 corresponds to 1 V).

Example 3

Axis-specific deletion of distance-to-go

```
WHEN $A_INA[5]>8000 DO DELDTG(X1)
```

If the voltage exceeds 8 V at input 5, delete the distance-to-go of axis X1 (assuming value of 1000 corresponds to 1 V).

Evaluation function

An evaluation function (SYNFCT) can be activated as a synchronous command.

With an evaluation function, it is possible – in synchronism with a machining operation – to read a variable, evaluate it with a polynomial function and write the result to another variable.
This function can be used, for example, to influence the feedrate of an axis as a function of a measured loop current.

An evaluation function has the following syntax:

‘SYNFCT(‘arithmetic expression,
<write_real-time variable>,
<read_real-time variable> ‘)’

The first parameter selects one of three evaluation functions defined by the user in the channel.

An evaluation function consists of a 3rd order polynomial and is determined in each case by 4 polynomial coefficients $AC_FCTxxC[0,...,3].
(xx = number of function: 1, 2 or 3).

Polynomial: $y = a_3x^3 + a_2x^2 + a_1x + a_0$
Polynomial coefficients:
a_3: $AC_FCTxxC[3]
a_2: $AC_FCTxxC[2]
a_1: $AC_FCTxxC[1]
a_0: $AC_FCTxxC[0]

The function value can be limited downwards by means of system variable $AC_FCTxxLL and upwards with $AC_FCTxxUL (xx = 1, 2, 3).

Programmed polynomial coefficients and limits for the function value take immediate effect.

All REAL real-time variables listed under “Synchronous commands” may be specified for <write_real-time variable> and all REAL real-time variables listed under “Conditions” may be specified for <read_real-time variable>.

The operating principle of the ’SYNFCT’ evaluation function is as follows:
The polynomial defined by the arithmetic expression is evaluated with the value of “read_real-time variable”. Upper and lower limits are then applied to the result which is then assigned to “write_real-time variable”.

The form of the synchronous commands which may be used in motion-synchronous actions is as follows:

With the command
‘FCTDEF(‘ <polynomial no.>, <lower_Limit>, <upper_Limit>, <Coeff0>, <Coeff1>, <Coeff2>, <Coeff3> ‘)’
it is possible program the limits and polynomial coefficients of the evaluation functions to be synchronized with machining operations.
The first parameter selects the evaluation function, the next two parameters define the upper and lower limits of the function values and the next four ones determine the polynomial coefficients.

The command for the above example is thus as follows:

```
FCTDEF(1,0,100,0,0.8,0,0)
ID=1 DO SYNFCT(1,$AA_VC[U1], $A_INA[2])
```

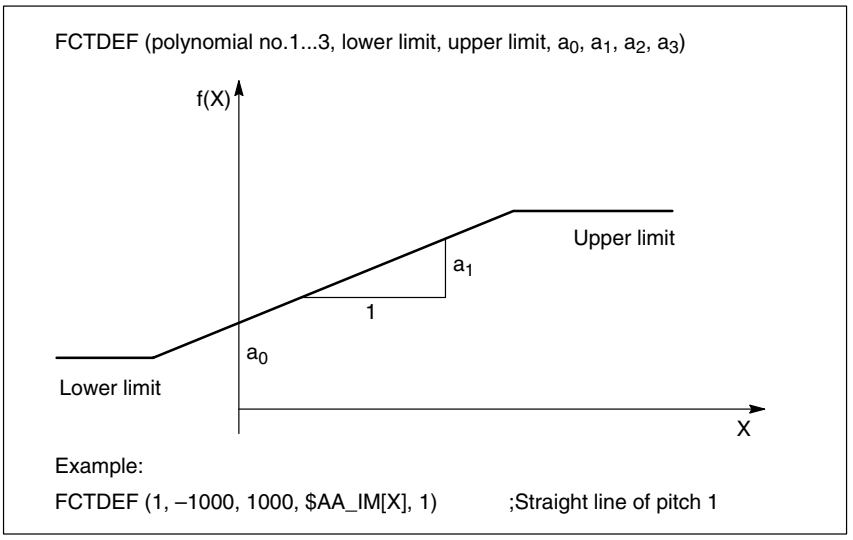


Fig. 2-4 Polynomial definition

Example 1

The **first** evaluation function must be defined with limits 0 and 100 [mm/min] and pitch 0.8.
The 0.8 * value of analog input 2 must then be added to the velocity of axis U1 using the evaluation function.

⇒ Evaluation function: $AA_VC[U1] = 0.8 * $A_INA[2]

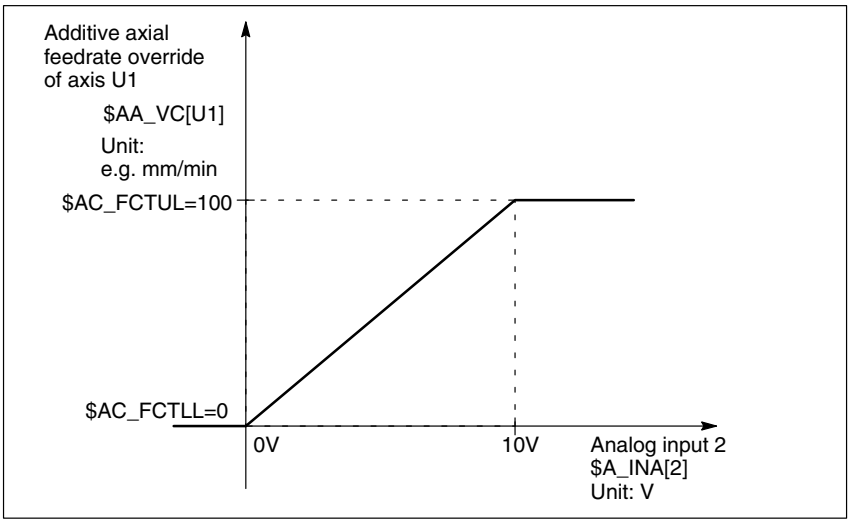


Fig. 2-5 Example of an evaluation function

⇒ Part program:

```
$AC_FCT1C[3]=0
$AC_FCT1C[2]=0
$AC_FCT1C[1]=0.8
$AC_FCT1C[0]=0
$AC_FCT1LL=0
$AC_FCT1UL=100
ID=1 DO SYNFCT(1,$AA_VC[U1], $A_INA[2])
G01 X100 Y200
```

Example 2

The change parameter must be controlled as a function of a value requested according to a polynomial of the 3rd order. The override must be reduced from 100 to 1% during the movement.

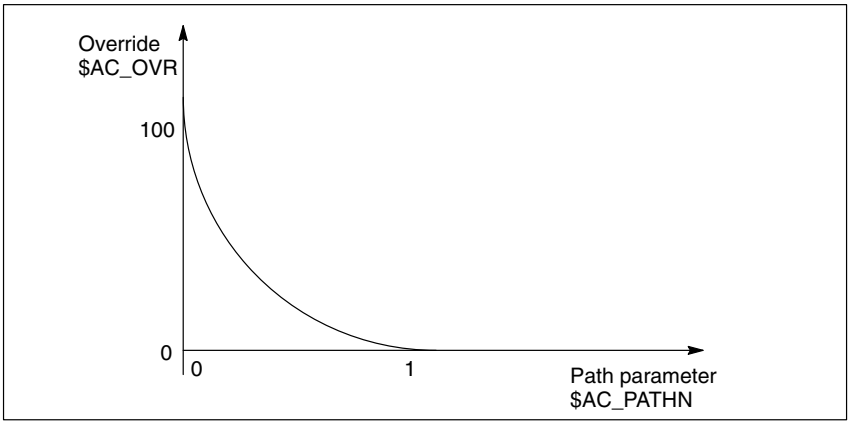


Fig. 2-6 Control velocity continuously

⇒ Part program:

```
FCTDEF(2, 1, 100, 100, –100, –100)
DO SYNFCT(2, $AC_OVR, $AC_PATHN)
G01 X100 Y100 F1000
```

Online override

In the case of grinding applications, machining of the workpiece and dressing of the grinding wheel can be executed either in the same channel or in different channels (machining and dressing channels).

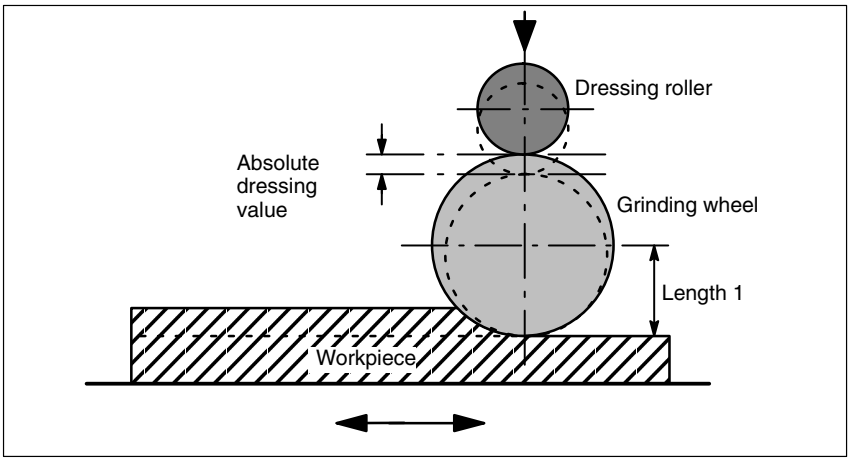


Fig. 2-7 Dressing during machining with a dressing roller

References: /FB/, W4 “Grinding”

With SW 3.2 and higher, online offset FTOC can be activated as synchronous command.

The online offset enables an overlaid movement for a geometry axis according to a polynomial programmed with FCTDEF as a function of a reference value which can be e.g. the actual value of an axis.

The online offset has the following syntax:

```
'FTOC(' polynomial no.,
<read real main variable>,                    ;Reference value
<length 1_2_3>{,
<channel number>}?{,
<spindle number>}?)'
```

Polynomial no.:	Number of the function parameterized before.
Read real main variable:	All main variables of type REAL listed with "Conditions" are admissible.
Length 1_2_3:	Wear parameter to which the offset value is added.
Channel number:	Target channel in which the offset shall be effective. CD dressing is thus possible from a parallel channel. If the channel number is discarded, the offset is effective in the active channel. Online offset with FTOCON must be switched on in the target channel of the offset.
Spindle number:	This is programmed if a grinding wheel which is not active shall be dressed. Prerequisite for this is that "constant wheel circumference speed" or "tool monitoring" is active. If no spindle number is programmed, the active tool is offset.

Example

Correct the length of an active grinding wheel

```
%_N_ABRICHT_MPF
FCTDEF(1,–1000,1000,–$AA_IW[V],1)
ID=1 DO FTOC(1,$AA_IW[V],3,1)  ; Select online tool offset
                               ; derived from V axis movement
                               ; length 3 of the active
                               ; grinding wheel is correctedin channel 1
WAITM (1,1,2)                  ; Synchronization with machining channel
G1 V–0.05 F0.01, G91
G1 V –....
...
CANCEL(1)                      ; Deselect online offset
...
```

Overlaid movement

System variable $AA_OFF allows an overlaid movement to be programmed for each axis in the channel irrespective of the current tool and machining plane. The shift is retracted immediately independently of whether the axis is programmed or not. Distance control can thus be implemented.

Axial MD: AA_OFF_MODE defines the mode of distance calculation.
0: Proportional calculation
1: Integrative calculation

$AC_VACTB and $AC_VACTW as input variables for synchronized actions and the output are disabled by means of option bits ("Feed-dependent analog-value control" ⇒ laser power control)!

$AA_OFF, position offset as output variable for synchronized actions for distance control is disabled via option bit!

Velocity limitation with MD: $MA_CORR_VELO.

Example 1
Distance control

The distance value is calculated on an integrative basis via MD: AA_OFF_MODE[V]=1. This is effective in the basic system of coordinates, that means before transformation. It can thus be used as distance control in the orientation direction (after frame selection with TOFRAME).

References: /PG/, “Programming Guide Fundamentals”

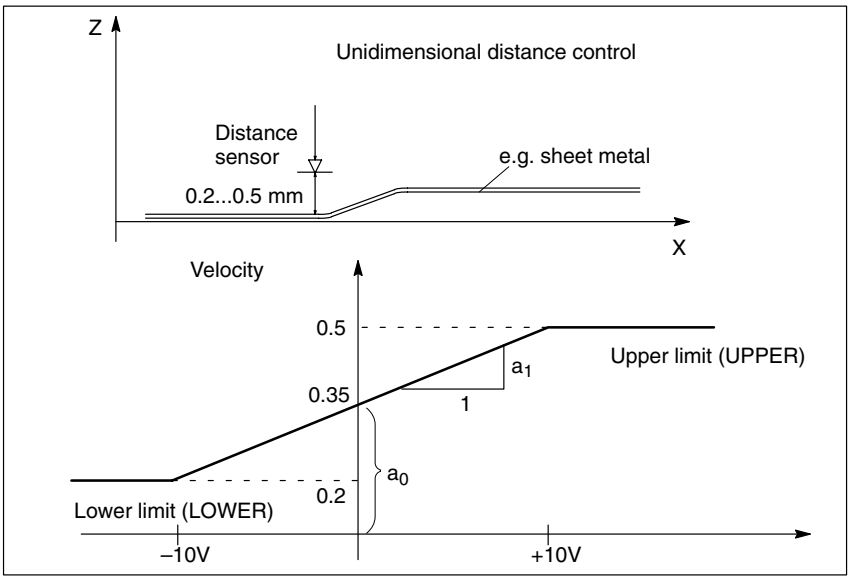


Fig. 2-8 Distance control

```
%_N_AON_SPF
PROC AON                          ; Subprogram for distance control ON
FCTDEF(1, 0.2,  0.5, 0.35, 1.5 EX–5)          ; Polynomial definition:
                                               The offset is
                                               offset is within
                                               the range of 0.2/0.5
ID=1 DO SYNFCT(1,$AA_OFF[Z], $A_INA[3])       ; Distance control
RET
ENDPROC

%_N_AOFF_SPF
PROC AOFF                         ; Subprogram for distance control OFF
CANCEL(1)
RET
ENDPROC

%_N_MAIN_MPF                      ; Main program
AON                               ; Distance control ON
...
G1 X100 F1000
AOFF                              ; Distance control OFF
M30
```

Example 2
Joystick-controlled axis movement

The deflection value is calculated on a proportional basis via the MD: AA_OFF_MODE[V]=0.

```
%_N_AON_SPF
PROC AON                          ; Subprogram for joystick control
                                   ;ON
DEF REAL ABSTAND = 25

FCTDEF(1,–5, +5, ABSTAND,–1)      ; Polynomial definition:
                                  ; The offset is
                                  ; made within the range +/– 5
ID=1 DO SYNFCT(1,$AA_OFF[V], $A_INA[3])        ; Joystick control
RET
ENDPROC

%_N_AOFF_SPF
PROC AOFF                          ; Subprogram for joystick control
                                   ; OFF
CANCEL(1)
RET
ENDPROC

%_N_MAIN_MPF                      ; Main program

AON                               ; Joystick control ON
...
G1 X100 F1000
AOFF                              ; Joystick control OFF
M30
```

Activation/ deactivation

The programmed conditions for the current motion-synchronous actions are acquired in the IPO cycle until they are fulfilled or until the end of the following block with machine function is reached.

With SW 3.2 and higher, a comparison of the synchronization conditions is made in the IPO cycle in the main run with introduction of a $$ main variable approved for synchronized actions.

Note

Whether or not frames are included in the calculation depends on which coordinate system (BCS or WCS) a real-time variable is defined in.

Distances are always calculated in the selected basic system (metric or inches). A switchover with G70 or G71 has no effect.

DRF offsets, external zero point offsets, etc. are only taken into account for real-time variables which are defined in the MCS.

Supplementary conditions

- **Interrupt routines/asynchronous subprograms**
  With activation of an interrupt routine, modal motion-synchronous actions are maintained and are also effective in the asynchronous subprogram. If the subprogram reset is not made with REPOS, the modal-synchronous actions changed in the asynchronous subprogram are effective in the main program.

- **REPOS**
  In the residual block, the synchronized actions are effective as compared to the interruption block. Any modifications on the modal-synchronous actions in the asynchronous subprogram are not effective in the interrupted program. The polynomial coefficients programmed with FCTDEF are not influenced by ASUP and REPOS.

  In the asynchronous subprogram, the coefficients from the program calling up remain effective. In the program calling up, the coefficients from the asynchronous subprogram remain effective.

- **End of program**
  The polynomial coefficients programmed with FCTDET remain effective after the end of program.

- **Block search run**
  In the case of block search run with calculation, they are collected, that means written into the setting data.

2.2.2 Multiple feedrates in a block

Functionality

Using the function “Multiple feedrates in a block” (see Chapter 3), it is possible to activate 6 different feedrate values in one NC block, a dwell and a retraction in synchronism with a movement as a function of external digital and/or analog inputs. The retraction process is initiated on the basis of a predefined value within an IPO cycle. The remaining distance-to-go is deleted.

The HW input signals are combined to form an input byte for the “Multiple feedrates in a block” function. There is a fixed, functional assignment within the byte:

Table 2-5 Input byte for “Multiple feedrates in a block”

	Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Input no.	E7	E6	E5	E4	E3	E2	E1	E0
Feed address	F7	F6	F5	F4	F3	F2	ST	SR

E7 to E2 Activation of feedrates F7 to F2
E1 Activation of dwell ST (in seconds)
E0 Activation of retraction movement SR

The signals are interrogated in ascending order, starting with E0.
The retraction movement (SR) therefore has the highest priority and feedrate F7 the lowest.

SR and ST terminate the feed movements with F2–F7.
SR also terminates ST, e.g. the complete function.

The higher the bit number, the lower the priority of the signals for feedrates F2–F7. The highest-priority signal determines the current feedrate. MD: MULTIFEED_STORE_MASK (store input signals of “Multiple feedrates in a block” function) can be set to define the response to drop-out of the highest-priority input (F2–F7) in each case.

The block end criterion is fulfilled:

- When the programmed end position is reached
- The retraction movement (SR) has been executed
- The dwell (ST) has expired.

A retraction movement or a dwell leads to deletion of the distance-to-go.

Hardware assignments

Channel-specific MD: MULTFEED_ASSIGN_FASTIN (assignment of input bytes of NC I/Os for “Multiple feedrates in a block” function) can be set to assign a maximum of two digital input bytes or comparator input bytes from the NCK I/O devices to the above input byte.

The input bits can also be inverted with MD: MULTFEED_ASSIGN_FASTIN.

If a 2nd byte is entered, the contents of the 1st and 2 bytes are ORed prior to use.

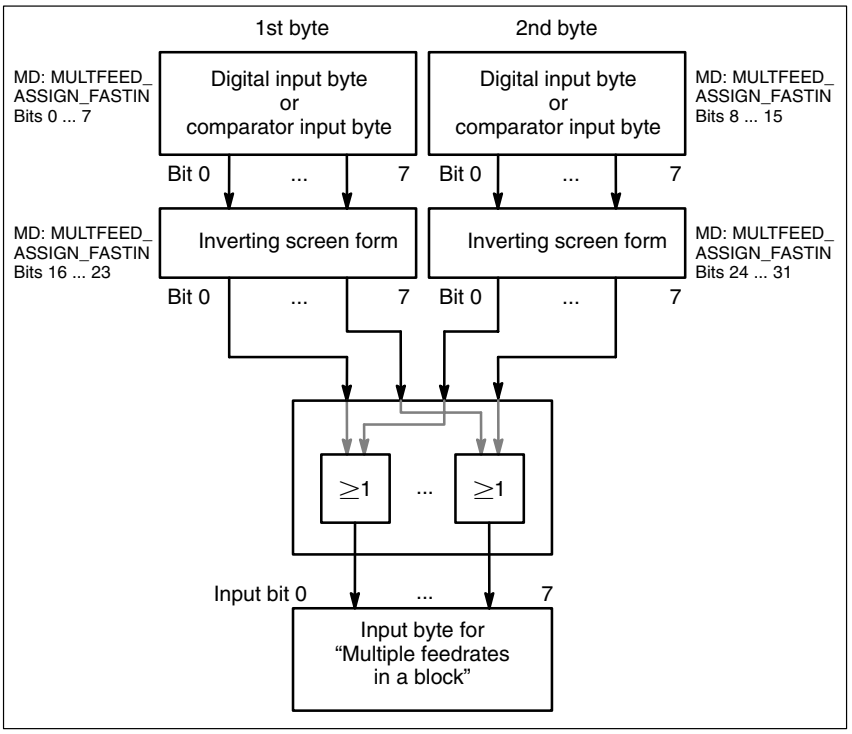


Fig. 2-9 Routing of signals for "Multiple feedrates in a block"

The routing of the digital input bytes and parameter assignments of the comparators are described in

References: /FB/, A2 "Various Interface Signals"

Programming of path movement

The path feed that remains valid while no input signal is applied is programmed in address F. This feedrate setting is modal.

With F2=... to F7=..., it is possible to program 6 further feedrates in the block in addition to the path feedrate.

The numeric extension specifies the bit number of the input which must change state for the feedrate to become effective.

e.g. F7=1000; 7 means input bit 7

Bits 2 to 7 may be specified as the numeric extension of the feedrate. The programmed values have a non-modal action, i.e. in the following block, the path feedrate programmed under F becomes effective again.

Sparking-out time and retraction path are programmed under additional addresses in the block:

ST=... Sparking-out time
SR=... Retraction path

These addresses are effective on a non-modal basis.

Programming of axial movement

Axial feedrates that remain valid as long as no input signal is applied are programmed in address FA. They are modally effective.

With FMA[2,x]=... to FMA[7,x]=..., up to 6 further feedrates per axis can be programmed in the block.

The first expression in square brackets indicates the bit number of the input which must change state for the feedrate to become effective. The second expression specifies the axis to which the feedrate applies.

e.g. FMA[3,Y]=1000;	Axial feedrate for Y axis, 3 means input bit 3

Bits 2 to 7 may be specified as the numerical extension of the axial feedrate. The values programmed under FMA are effective on a non-modal basis, i.e. the feedrate programmed under FA becomes effective again in the following block.

Sparking-out time and retraction path can be specified additionally for individual axes:

STA[x]=...	Sparking-out time, axis-specific
SRA[x]=...	Retraction path, axis-specific

The expression in square brackets indicates the axis to which the sparking-out time and retraction path apply.

STA[X]=2.5;	Sparking-out time is 2.5 seconds
SRA[X]=3.5;	Retraction path is 3.5 (unit e.g.: mm)

These addresses are effective on a non-modal basis.

If feedrates, sparking-out time or retraction path are programmed for an axis on the basis of an external input, this axis may not be programmed as a POSA axis (positioning axis beyond block limit) in this block.

If the input for sparking-out time or retraction path is activated, the distance-to-go for path axes or for the individual axes concerned is deleted and the dwell or retraction operation started.

Note

The unit for the retraction path refers to the currently selected dimension system (mm or inches).

The status of an input can also be interrogated for synchronous commands of different axes.

Look Ahead is effective even when multiple feedrates are programmed in a block. The currently valid feedrate can thus be limited by the Look Ahead function.

Application

Examples of typical applications are:

- Analog or digital calipers
  Various feedrate values, a dwell and a retraction path can be activated as a function of external analog or digital inputs. In this case, threshold values are specified via setting data.
- Switchover from infeed to working feedrate via proximity switch

Example

Internal grinding of a ball-bearing ring. The actual diameter is measured by means of calipers and the appropriate feedrate value for roughing, finishing or fine finishing activated in each case as a function of threshold values. The calipers position also supplies the end position. The block end criterion is therefore not only determined by the programmed axis position of the infeed axis, but also by the caliper.

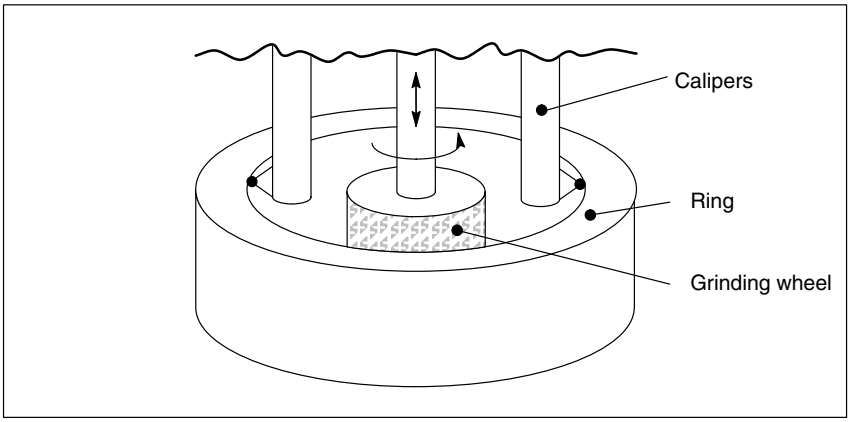


Fig. 2-10 Calipers

Part program

```
G00 X40 F=1000;                 Initial position
G01 X48 F=20 F7=5 F3=2.5 F2=0.5 ST=1.5 SR=–0.5
```

Retraction
Sparking-out time
Feedrate fine finishing
Feedrate finishing
Feedrate roughing
Feedrate air grinding

Note

The axial feedrate or tool feedrate (F value) is the 100% feed. With “Multiple feedrates” in one block (F2 to F7 values), feedrates can be made available that are **less than or equal to** the axial feedrate or tool feedrate.

2.2.3 AC control (adaptive control)

General

The “AC control” function (for Adaptive Control, see Section 3) is used to control one particular process variable (e.g. path or axis-specific feedrate) as a function of other measured process variables (e.g. spindle current, torque, etc.).

This function can be used, for example, to

- keep the cutting volume constant during grinding
- protect the machine and tool against overloads
- achieve shorter machining times
- achieve a better surface quality.

Parameterization

The AC control can be parameterized within the part programming as follows:

- **Additive control**
  With the additive control method, the programmed value (F word in the case of AC control) is corrected as an additive function.

  $F_{effective} = F_{programmed} + F_{AC}$
- **Multiplicative control**
  In the case of multiplicative control, the F word is multiplied by a factor (override in the case of AC control).

  Feffective = $F_{programmed} \cdot \text{Factor}_{AC}$

Functional schematic of additive control

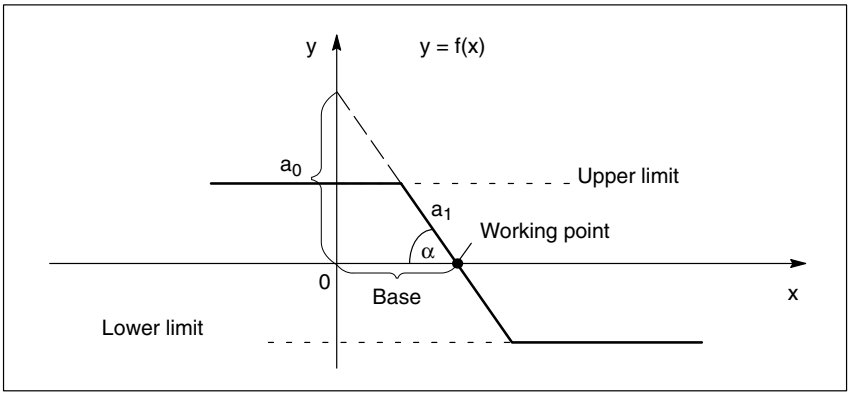


Fig. 2-11 AC additive control

The function is effective within the upper and lower limit. The pitch of straight line

y = mx + n

is negative ⇒ a_1 is negative.

Note:

tan $\alpha = -p$

The gain of the control equals the pitch of straight line

$a_0 = p \cdot \text{Basis}$

Example

The programmed feed (axial or path-related) shall be controlled by the current (positive) of the x-axis (e.g. infeed moment). The working point is defined at 5 A. The feedrate may be altered by ±100 mm/min. In this case, the axial current deviation must not exceed ±1 A.

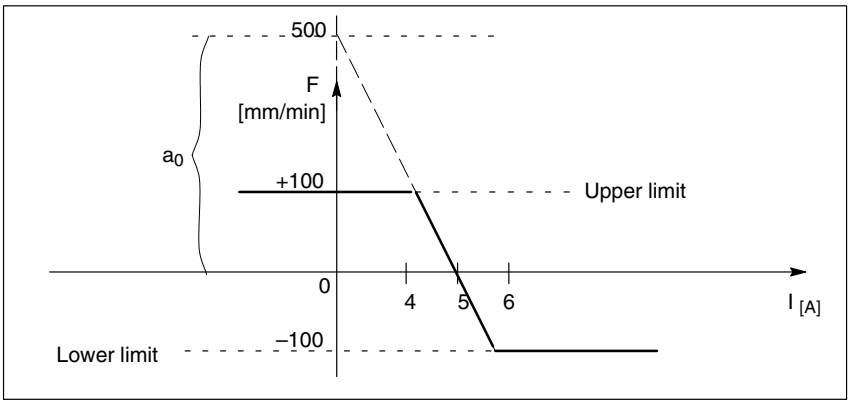


Fig. 2-12 Example of additive control

Definition of coefficients:

$y = f(x) = a_0 + a_1x + a_2x^2 + a_3x^3$

$a_1 = -\frac{100\ \text{mm}}{1\ \text{min} \cdot \text{A}}$

$a_1 = -100 \Rightarrow$ control constant

$a_0 = -(-100) \cdot 5 = 500$

$a_2 = 0$ (no square element)

$a_3 = 0$ (no cubic element)

upper limit = 100

lower limit = –100

FCTDEF(<polynomial no.>, <lower_Limit>, <upper_Limit>, <Coeff0>, <Coeff1>, <Coeff2>, <Coeff3>)

FCTDEF(1, –100, 100, 500, –5, 0, 0)

This function describes the example displayed in Fig. 2-12 completely.

AC control is activated with the following synchronized action:
ID = 1 DO SYNFCT(1, $AC_VC, $AA_LOAD[x])

Functional schematic of multiplicative control

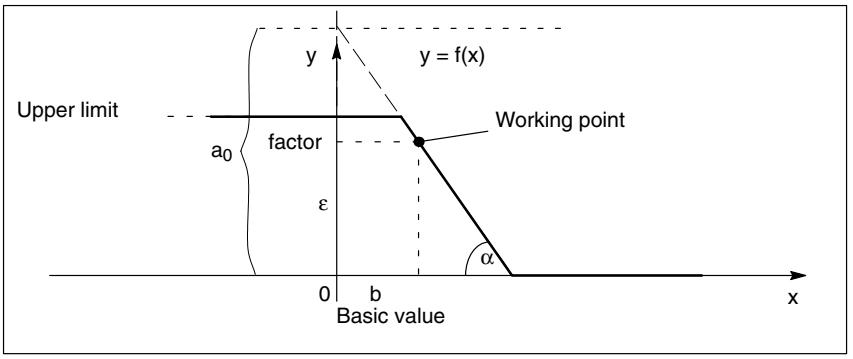


Fig. 2-13 AC multiplicative control

The function is effective between the upper limit and the ordinate zero. The working point is located on the curve.

The working point is characterized by

- the base value
- the corresponding factor (with override, standard = 100 %)

From Fig. 2-13, the following results for a_0:

$a_0 = \varepsilon + (b \cdot p)$

$\varepsilon \Rightarrow$ factor
$b \Rightarrow$ base value
$p \Rightarrow$ gain

Example

The programmed feed (axial or path-related) shall be controlled multiplicatively. The working point is defined at 100 % with 30 % drive load. With 80 % load, the axis (n) shall stand still. A velocity overshoot of 20 % of the programmed velocity is admitted.

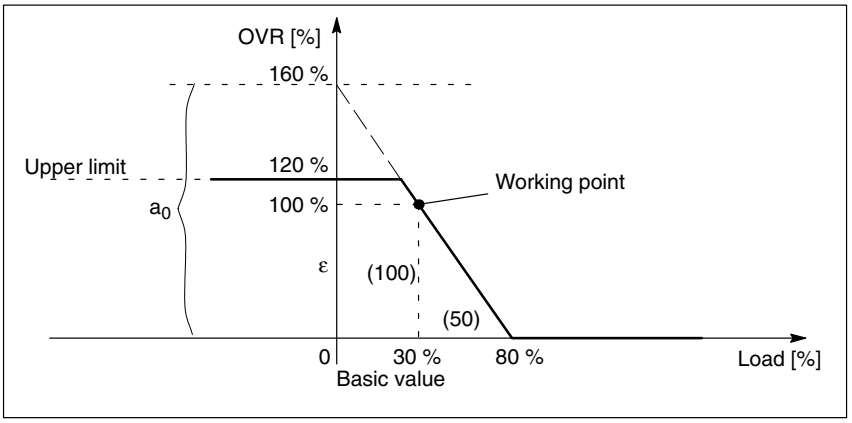


Fig. 2-14 Example of multiplicative control

Definition of coefficients:

$y = f(x) = a_0 + a_1x + a_2x^2 + a_3x^3$

$a_1 = -\frac{100\ \%}{(80 - 30)\ \%} = -2 = -p$

$a_0 = 100 + (2 \cdot 30) = 160$
$a_2 = 0$ (no square element)
$a_3 = 0$ (no cubic element)
upper limit = 120
lower limit = 0

FCTDEF(<polynomial no.>, <lower_Limit>, <upper_Limit>, <Coeff0>, <Coeff1>, <Coeff2>, <Coeff3>)

FCTDEF(1, 0, 120, 160, –2, 0, 0)

This function describes the example displayed in Fig. 2-14 completely.

The corresponding synchronization may be
ID = 1 DO SYNFCT(1, $AC_OVR, $AA_LOAD[x])

Real-time variables

When the function is activated, programming of the following real-time variables is enabled:

- $AA_LOAD[<axial expression>] — Drive capacity utilization
- $AA_POWER[<axial expression>] — Drive active power
- $AA_TORQUE[<axial expression>] — Drive torque setpoint
- $AA_CURR[<axial expression>] — Current actual value of axis or spindle
- $A_INA[INPIT] — Input analog value

Example of AC control with an analog input voltage

A process quantity (measured via $A_INA[INPUT]) must be regulated to 2 V through correction of the path or axial feedrate by the additive control method. The feedrate override must be adjusted within the ±100 [mm/min] range.

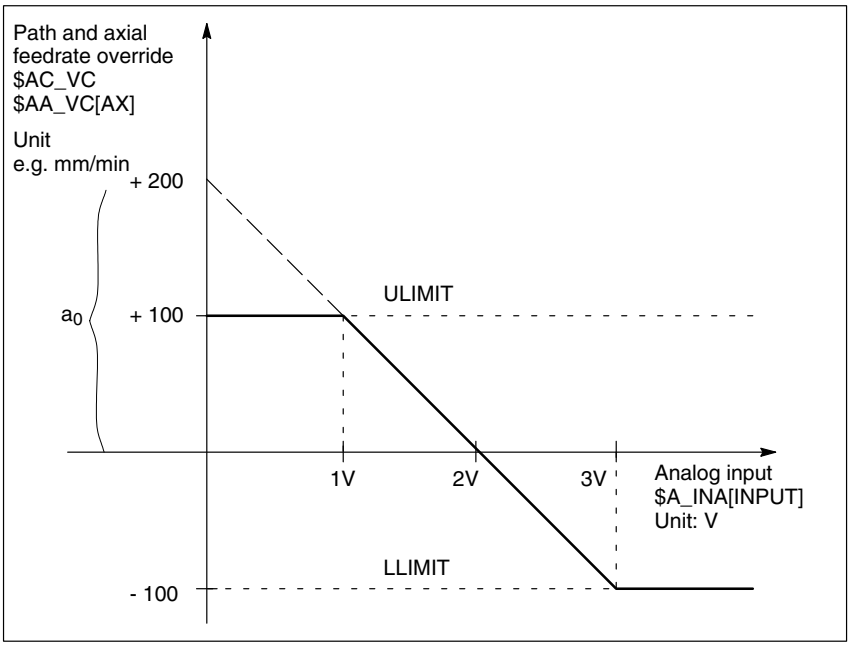


Fig. 2-15 Diagram showing AC control

As an example of how to implement the “AC control” function, two subprograms to activate the function and one subprogram to deactivate it are described below which implement the AC functionality on the basis of **motion-synchronous actions**.The machine manufacturer or user can create and modify these programs to meet his own requirements.

Definition of coefficients:

$y = f(x) = a_0 + a_1x + a_2x^2 + a_3x^3$

$a_1 = -\frac{100\ \text{mm}}{1\ \text{min} \cdot 1\ \text{V}}$

$a_1 = -100 \Rightarrow$ control constant

$a_0 = -(-100) \cdot 2 = 200$

$a_2 = 0$ (no square element)

$a_3 = 0$ (no cubic element)

upper limit = 100

lower limit = –100

FCTDEF(<polynomial no.>, <lower_Limit>, <upper_Limit>, <Coeff0>, <Coeff1>, <Coeff2>, <Coeff3>)

FCTDEF(1, –100, 100, 200, –100, 0, 0)

AC control can be activated with the following synchronized action:
ID = 1 DO SYNFCT(1, $AA_VC[x], $A_INA[1])

■

Notes

Supplementary Conditions 3

Availability of "Constant analog-value output" function

The "Constant analog value output" function (Prerequisite: Analog output) is an option and available for

- SINUMERIK 840D with NCU 572/573, with SW 2 and higher.

Availability of "Adaptive control" function

The function is an option and available for

- SINUMERIK 840D with NCU 572/573, with SW 2 and higher.

The function is contained in the export version 840DE with restricted functionality; it is not contained in the FM-NC, 810DE (SW 3.1 and earlier).

- SINUMERIK 810DE, SW 3.2 and higher

Availability of "Multiple feedrates in a block" function

The function is an option and available for

- SINUMERIK FM-NC with NCU 570, with SW 2 and higher
- SINUMERIK 840D with NCU 572/573, with SW 2 and higher.
- SINUMERIK 810DE, SW 3.2 and higher

Availability of "Laser" function

The function is an option and available for

- SINUMERIK FM-NC with NCU 570, with SW 3.2 and higher
- SINUMERIK 840D with NCU 572/573, SW 3.2 and higher

Availability of "Distance control" function

The function is an option and available for

- SINUMERIK FM-NC with NCU 570, with SW 3.2 and higher
- SINUMERIK 840D with NCU 572/573, SW 3.2 and higher

Availability of "Synchronized actions SW4" function

The new function as of SW 4 is contained in export version 840DE with restricted functionality.

The function is not contained in the export versions FM-NC and 810DE.

SINUMERIK 840Di

This system offers the same functions as the 840D. Special functions that operate on hardware specific to the SINUMERIK 840D, e.g. adaptive control components, are currently available only on request.

Availability of "Motion-synchronous actions" function

Motion-synchronous actions that utilize the listed realtime variables are not configurable on the SINUMERIK 840Di at present:

- $AA_LOAD: Drive capacity utilization
- $AA_POWER: Drive active power
- $AA_TORQUE: Drive torque setpoint
- $AA_CURR: Actual current of axis or spindle

Note

The motion-synchronous action functionality is provided by the NC of the SINUMERIK 840Di. The requisite realtime variables for function evaluation cannot, however, be transferred to or from the drive/slave via the PROFIBUS-DP.

■

Data Descriptions (MD, SD) 4

4.1 Channel-specific machine data

21220 MD number	**MULTFEED_ASSIGN_FASTIN** Assignment of input bytes of NCK I/Os for “Multiple feedrates in a block”		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective after Power On	Protection level: 2/7	Unit: HEX	
Data type: DWORD	Applies from SW version: 2.1		
Significance:	Acquisition of the following drive actual values is activated by means of MD: MULTFEED_ASSIGN_FASTIN (assignment of input bytes of NCK I/Os for “Multiple feedrates in a block”).The assigned input signals can also be inverted with this machine data. The MD is coded as follows: Bits 0–7: Number of 1st digital input byte or comparator input byte used Bits 8–15: Number of 2nd digital input byte or comparator input byte used Bits 16–23: Inverting screen form for writing 1st byte Bits 24–31: Inverting screen form for writing 2nd byte Bit=0: Do not invert Bit=1: Invert If a 2nd byte is entered, the contents of the 1st and 2 bytes are ORed prior to use. The following must be specified as digital input numbers: 1: for on-board byte 2–5: for external bytes The following must be specified as comparator input byte numbers: 128: for comparator 1 (equals 80Hex) 129: for comparator 2 (equals 81Hex)		
Application	The external digital input byte 3 must be used as the 1st byte and the input byte of comparator 2 as the 2nd byte for the “Multiple feedrates in a block” function. The signals must also be inverted: Bit 0, 2, 3 of the digital input byte Bit 0, 1, 5, 7 of the comparator input byte ⇒ MD:MULTFEED_ASSIGN_FASTIN=A30D8103 (in Hex format)		

21230 MD number	**MULTFEED_STORE_MASK** Store input signals of “Multiple feedrates in a block” function		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Changes effective after Power On	Protection level: 2/7	Unit: HEX	
Data type: BYTE	Applies from SW version: 2.1		
Significance:	The higher the bit number of the signals for feedrates F2–F7 of the “Multiple feedrates in a block” function, the lower their priority within the input byte. The highest-priority signal determines the current feedrate. The response to a reduction in the value applied to the highest-priority input can be defined in MD: MULTIFEED_STORE_MASK (store input signals of “Multiple feedrates in a block” function). When bit 2–7 is set, the associated feedrate (F2 to F7) as selected by the highest-priority input signal in each case remains valid even if the input signal drops and a lower-priority input is active. The MD is coded as follows: Bits 0–1: No significance Bits 2–7: Storage of feedrate signals Bits 8–31: Reserved		

4.2 Axis/spindle-specific machine data

32070 MD number	**CORR_VELO** Axis velocity for handwheel, ext. ZO, cont. dressing, distance control		
Default value: 100	Min. input limit: 0	Max. input limit: plus	
Changes effective after Power On	Protection level:	Unit: %	
Data type: DWORD	Applies from SW version: 3.2		
Significance:	Limitation of axis speed for handwheel overlay, external zero offset, continuous dressing, distance control $AA_OFF via synchronized actions related to the JOG speed MD: JOG_VELO, MD: JOG_VELO_RAPID, MD: JOG_REV_VELO, MD: JOG_REV_VELO_RAPID. The maximum admissible speed is the maximum speed in MD: MAX_AX_VELO. Speed limitation is made to this value. An alarm is displayed if this value is exceeded. The conversion into linear or rotary axis speed is made according to MD: IS_ROT_AX.		
Application	The velocity is limited with displacement of overlaid movements.		

32920 MD number	**AC_FILTER_TIME** Filter smoothing constant for adaptive control		
Default value: 0.0	Min. input limit: 0.0	Max. input limit: plus	
Changes effective after Power On	Protection level: 2/7	Unit: s	
Data type: DOUBLE	Applies from SW version: 2.1		
Significance:	The following drive actual values can be acquired with main run variables $AA_LOAD, $AA_POWER, $AA_TORQUE and $AA_CURR: – Drive capacity utilization – Drive active power – Drive torque setpoint – Current actual value of axis or spindle In order to eliminate peaks, the measured values can be smoothed by means of a PT1 filter. The filter time constant is defined with MD: AC_FILTER_TIME (filter smoothing time constant for adaptive control). On acquisition of the drive torque setpoint or current actual value, the filter acts in addition to the filters in the 611D. Both filters are connected in series if both strongly and weakly smoothed values are required in the system. The filter is deactivated by entering a smoothing time of 0 seconds.		
MD irrelevant for	FM-NC with 611A		
Application	Smoothing of current actual value for AC control		

36730 MD number	**DRIVE_SIGNAL_TRACKING** Sensing of additional drive actual values		
Default value: 0	Min. input limit: 0	Max. input limit: 4	
Changes effective after Power On	Protection level: 2/7	Unit: –	
Data type: BYTE	Applies from SW version: 2.1		
Significance:	Acquisition of the following drive actual values is activated by means of MD: DRIVE_SIGNAL_TRACKING (acquisition of additional drive actual values). – Drive capacity utilization ($AA_LOAD[axial expression]) – Drive active power ($AA_POWER[axial expression]) – Drive torque setpoint ($AA_TORQUE[axial expression]) – Current actual value of axis or spindle ($AA_CURR[axial expression]) Coding: 0: No acquisition 1: The above-mentioned drive actual values are acquired.		
MD irrelevant for	FM-NC with 611A		

36750 MD number	**AA_OFF_MODE** Effect of value assignment for axial override on synchronized actions		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after Power On	Protection level: 2/7	Unit: –	
Data type: BYTE	Applies from SW version: 3.2		
Significance:	An overlaid movement can be implemented for the programmed axis within synchronized actions by means of main run variable $AA_OFF. The type of calculation is defined via the axial MD: AA_OFF_MODE. MD: AA_OFF_MODE=0: Absolute value MD: AA_OFF_MODE=1: Incremental calculation		
Application	• Distance control for laser processing (integrative) • Joystick-controlled axis movement (proportional)		

■

Signal Descriptions

5

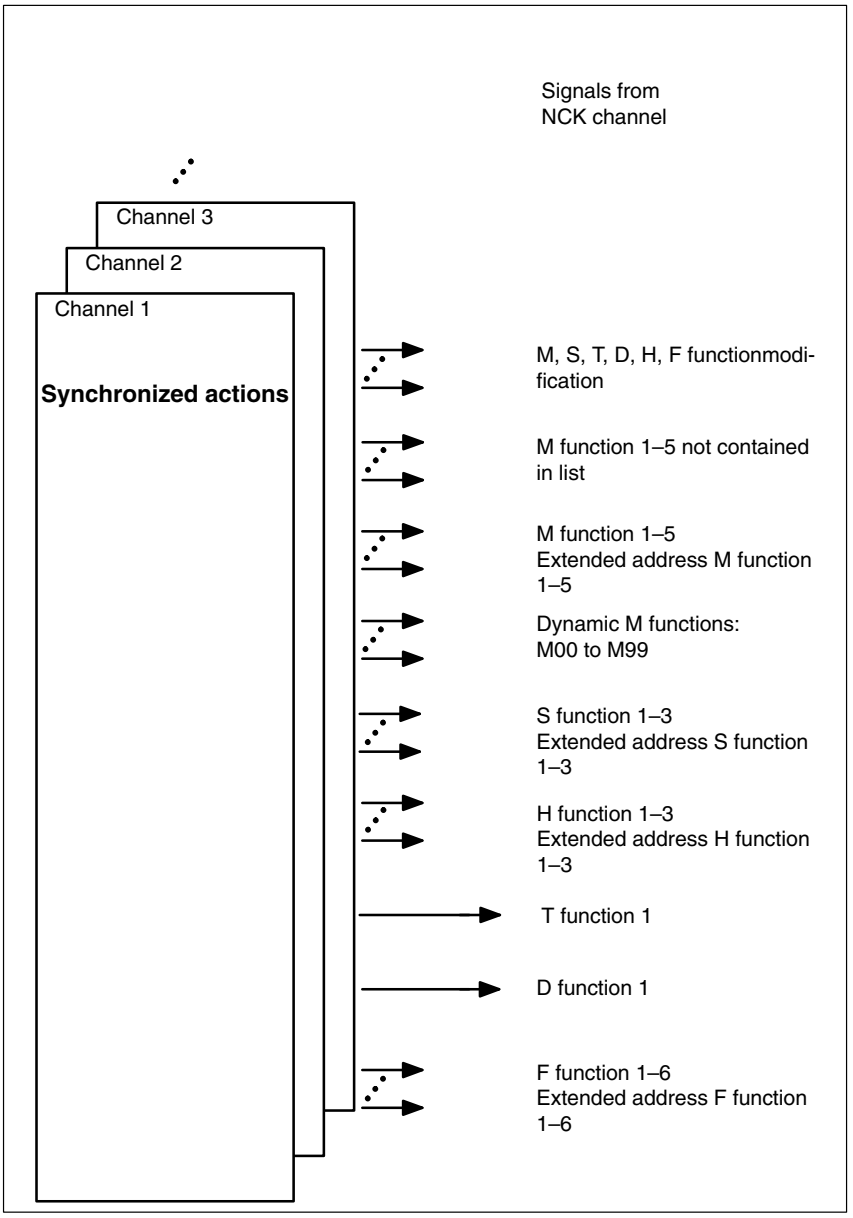


Fig. 5-1 PLC interface signals for synchronized actions

5.1 Channel-specific signals

For SW 4 and higher, a description of the signals of the auxiliary functions is provided in
/FB/ H2, Auxiliary Function Output to PLC.

■

Example

6

See /FBSY/ Description of Functions, Synchronized Actions

■

Notes

Data Fields, Lists

7

7.1 Machine data

Number	Identifier	Name	Reference
General ($MN_ ...)			
10110	PLC_CYCLE_TIME_AVERAGE	Maximum PLC acknowledgment time	B1
11100	AUXFU_MAXNUM_GROUP_ASSIGN	No. of aux. functions assigned to aux. function groups	H2
11110	AUXFU_GROUP_SPEC[n]	Auxiliary function group specification	H2
Channel-specific ($MC_ ...)			
20110	RESET_MODE_MASK	Definition of the basic control setting after start of part program	K2
20112	START_MODE_MASK	Definition of the basic control setting after startup and with reset or end of part program	K2
20120	TOOL_RESET_VALUE	Tool the length offset of which is selected during startup (reset/TP end)	K2
20122	TOOL_RESET_NAME	Definition of the tool used for selecting length offset during startup/reset	FBW
20130	CUTTING_EDGE_RESET_VALUE	Tool cutting edge the tool offset of which is selected during startup (reset/TP end)	K2
20270	CUTTING_EDGE_DEFAULT	Basic setting of tool cutting edge w/o programming	W1
20800	SPF_END_TO_VDI	Subprogram end to PLC	H2
21220	MULTFEED_ASSIGN_FASTIN	Assignment of input bytes of NCK I/Os for “Multiple feedrates in a block” (SW 2 and higher)	
21230	MULTFEED_STORE_MASK	Store input signals of “Multiple feedrates in a block” function (SW 2 and higher)	
22000	AUXFU_ASSIGN_GROUP[n]	Auxiliary function group	H2
22010	AUXFU_ASSIGN_TYPE[n]	Auxiliary function type	H2
22020	AUXFU_ASSIGN_EXTENTION[n]	Auxiliary function extension	H2
22030	AUXFU_ASSIGN_VALUE[n]	Auxiliary function value	H2
22200	AUXFU_M_SYNC_TYPE	Output time of M functions	H2
22210	AUXFU_S_SYNC_TYPE	Output time of S functions	H2
22220	AUXFU_T_SYNC_TYPE	Output time of T functions	H2
22230	AUXFU_H_SYNC_TYPE	Output time of H functions	H2
22240	AUXFU_F_SYNC_TYPE	Output time of F functions	H2
22250	AUXFU_D_SYNC_TYPE	Output time of D functions	H2
22300	AUXFU_AT_BLOCK_SEARCH_END	Auxiliary function output after block search run	H2

Axis-specific ($MC_ ...)			
32070	CORR_VELO	Axis speed for handwheel, ext. ZO, cont. dressing, distance control (SW 3 and higher)	
32920	AC_FILTER_TIME	Filter smoothing time constant for adaptive control (SW 2 and higher)	
36730	DRIVE_SIGNAL_TRACKING	Acquisition of additional drive actual values (SW 2 and higher)	
36750	AA_OFF_MODE	Effect of the value allocation for axial override with synchronized actions (SW 3 and higher)	

7.2 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

■

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

Stepper Motor Control (S6), for SINUMERIK FM-NC only

■

Notes

Brief Description

1

Software basis

The SINUMERIK FM-NC control is based on the software of the SINUMERIK 840D. All FM-NC-specific functions including the stepper motor control are implemented by additions and modifications to this basis software. This documentation only describes the changes and function additions that are required for stepper motor control.

Note

This Description of Functions applies only to SINUMERIK FM-NC with NCU 570.2 up to SW 3.

Reference point approach

A special procedure is required to acquire the actual position of stepper motors (without measuring system). The measuring system zero pulse is replaced here by a BERO.

Rotation monitoring

In order to monitor the rotary motion of a stepper motor, a rotation monitor can be activated via a VDI interface signal. It is thus possible to monitor the stepping coincidence.

Speed control

To achieve optimum speed control with stepper motors, a knee-shaped acceleration characteristic is required owing to the specific frequency characteristic (start/stop frequency with step change). This is available for positioning tasks. A constant speed time can be set in the machine data to minimize acceleration jumps.

■

Notes

Detailed Description 2

2.1 Reference point approach for stepper motors

2.1.1 Traversal for referencing

Reference point approach with incremental encoder

The procedure applied to approach the reference point with an incremental encoder is the same as that used to reference with an analog drive.

MD 30240: ENC_TYPE must be set in accordance with the measuring system type (no feedback of the steps as actual-value pulses).

2.1.2 Defining reference point using a BERO

Reference point approach without incremental encoder

The basic sequence of referencing operations and the possible settings with machine data are the same as those used for referencing with an incremental encoder, see

References: /FB/, R1, “Reference Point Approach”

A BERO replaces the zero pulse of the incremental encoder, which is placed on the motor shaft or spindle. The rising edge of the BERO (single-edge evaluation) or the BERO center (dual-edge evaluation) are used for synchronization.

Procedure

- Phase 1: Approach the reference point cam, see
  References: /FB/, R1, “Reference Point Approach”
- Phase 2: Synchronization with the reference point BERO (zero pulse simulation), basic procedure, see
  References: /FB/, R1, “Reference Point Approach”
- Phase 3: Approach reference point see
  References: /FB/, R1, “Reference Point Approach”

Synchronization with the reference point BERO

A rapid input is required to transfer switching edges to the control system. The FM-NC is equipped with four digital inputs (pins 13 to 16) on connector X1 (connector for handwheel and connection of I/O devices). The rapid digital inputs are permanently assigned to the servo loops.

BERO edge

When the selected BERO edge appears, the corresponding actual-value register is latched.

To obtain a good reduction of the reference point, the search velocity for the BERO edge must no exceed a maximum value depending on the BERO type.

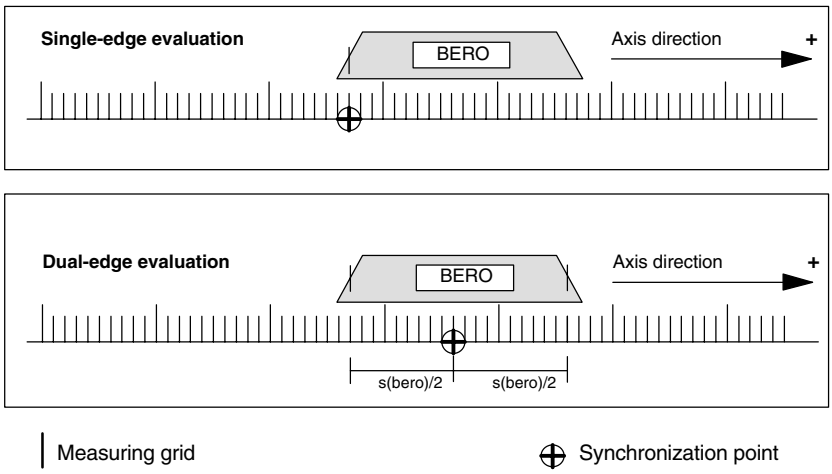


Fig. 2-1 Synchronization point definition

Single-edge evaluation

The positive edge of the BERO is interpreted as a zero pulse.
The selection is made in MD 34200: ENC_REF_MODE = 2.

Dual-edge evaluation

The positive edge and the negative edge of the reference point BERO are traversed one after the other and the actual values for each registered. The mean value is the synchronization point at which phase 2 is completed and phase 3 starts.

The selection is made in MD 34200: ENC_REF_MODE=4.

With dual-edge evaluation any drift of the BERO edges can be compensated for. The maximum speed is also limited by the fact that the time between the two edges including any switching delay by the BERO must be greater than 1 SERVO cycle.

2.2 Rotation monitoring of stepper motor with BERO

Overview

The BERO used for rotation monitoring is connected in the same way as the BERO for referencing.

It is possible to connect this BERO in parallel with the referencing BERO or use of the same for rotation monitoring. However, rotation monitoring must be deactivated during referencing, i.e. no signal edge is allowed to arrive from the reference BERO while rotation monitoring is active.

Modulo counter

A modulo counter (modulo 1 revolution) counts the actual-value increments. The modulo count is stored in the machine data.
MD 31100: BERO_CYCLE Repeat cycle of BERO edges in actual-value increments

Activation

Rotation monitoring can be activated and deactivated via IS “Rotation monitoring ON/OFF” (DB31–..., DBX24.0). The BERO is “zeroed” when it is traversed for the first time, the contents of the modulo counter are recorded as the “BERO zeroing value”.

Comparison

Every time the BERO is traversed again, the contents of the modulo counter are compared with the stored “BERO zeroing value” to see if they are similar.

A BERO tolerance which can be set in MD 31110: BERO_EDGE_TOL can be included in the comparison. If the comparison is negative, IS “Error in rotation monitoring” is signaled to the PLC (DB31–..., DBX96.0). The signal has edge evaluation and is only present for the duration of one PLC cycle. At the same time, the monitoring function is switched off automatically and reference point approach has to be repeated.

Note

The “Error in rotation monitoring” always occurs if the stepper motor is activated incorrectly, even if rotation monitoring is not activated. The user must ensure that the drive is switched off safely.
“Error rotation monitoring” means that the drive has to be switched off!

2.3 Speed control

2.3.1 Knee-shaped acceleration characteristic

One of the characteristic features of stepping drives is the drop in the available torque in the upper speed range (see Fig. 2-2).

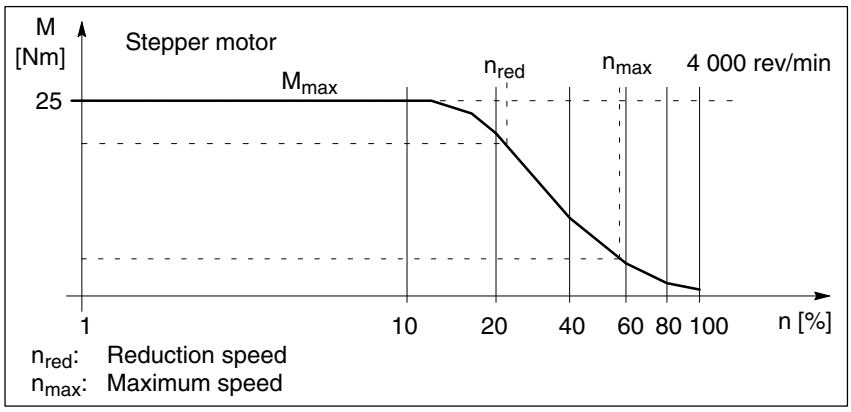


Fig. 2-2 Typical motor characteristics of stepping drive

The best way to achieve optimum loading of motors with such characteristics while protecting them against overloading is to control the acceleration rate as a function of speed.

This method – referred to as a “knee-shaped acceleration characteristic” – can be used for both positioning and path movements.

Parameterization of axis characteristic

The axial response of the acceleration characteristic must be parameterized by the following machine data:

- **MA_MAX_AX_VELO**
  Maximum axial speed (v_{max})
- **MA_ACCEL_REDUCTION_SPEED_POINT**
  Speed at which acceleration rate is reduced with respect to MA_MAX_AX_VELO (v_{red})
- **MA_MAX_AX_ACCEL**
  Maximum axial acceleration (a_{max})
- **MA_ACCEL_REDUCTION_FACTOR**
  Acceleration reduction factor with respect to MA_MAX_AX_ACCEL (a_{red})
- **MA_ACCEL_REDUCTION_TYPE**
  Type of acceleration characteristic
  - 0: Constant
  - 1: Hyperbolic
  - 2: Linear

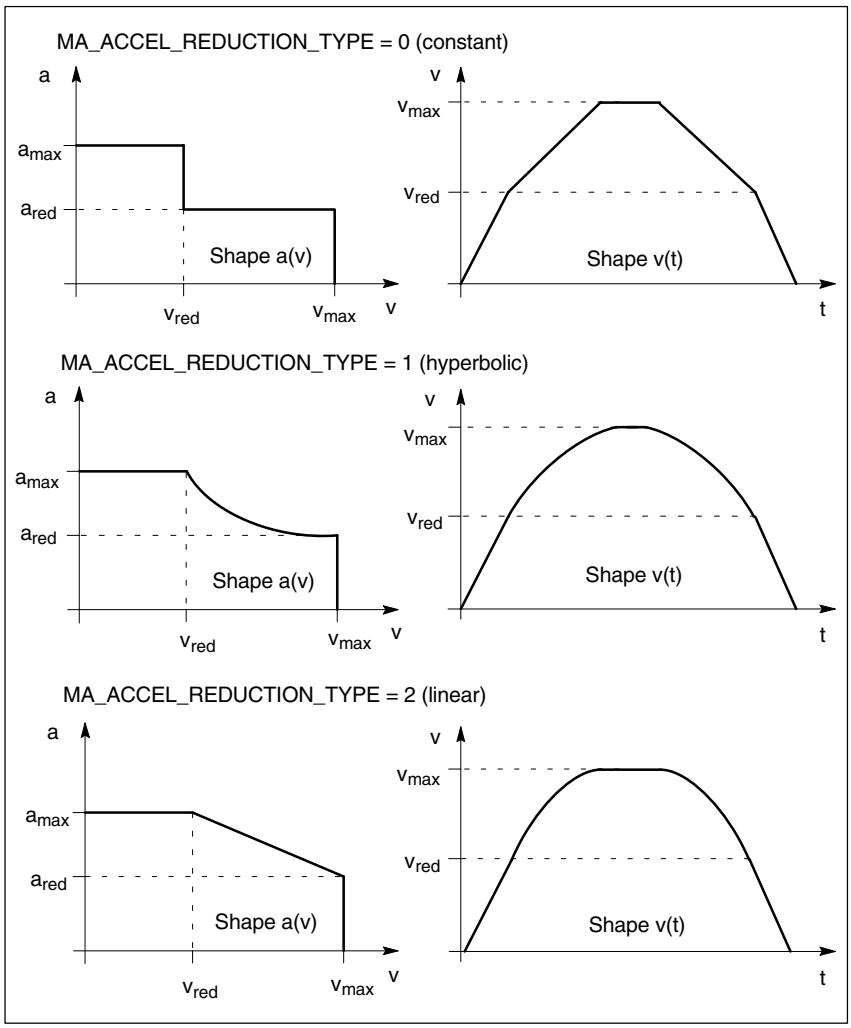


Fig. 2-3 Axial acceleration and speed characteristics

Speeds:

v_{max}: MA_MAX_AX_VELO

v_{red}: MA_ACCEL_REDUCTION_SPEED_POINT · MA_MAX_AX_VELO

Acceleration rates:

a_{max}: MA_MAX_AX_ACCEL

a_{red}: (1-MA_ACCEL_REDUCTION_FACTOR) · MA_MAX_AX_ACCEL

Path characteristic

There are no special machine data for the path movement. The characteristic is based on the parameters of the relevant axes as a function of their share of the path vector (geometry). It is possible to combine axes with different acceleration characteristics.

The most restrictive type in each case – regardless of share of the path vector – determines the acceleration characteristic of the path.

The following order of characteristics applies:

1. Constant
2. Hyperbolic
3. Linear
4. None (MA_ACCEL_REDUCTION_SPEED_POINT = 1 or MA_ACCEL_REDUCTION_FACTOR = 0)

It is therefore possible to operate a mixture of stepping and DC drives.

Line-normal and tangential acceleration characteristics are evaluated together within curved path sections.

The path speed is reduced such that a maximum of 25 % of the speed-dependent accelerating capability of the axes is required for line-normal acceleration. The remaining 75 % is reserved for tangential acceleration processes, i.e. deceleration or acceleration on the path.

Equivalent path characteristic

If the path movement cannot be traversed at a velocity-dependent acceleration rate (e.g. kinematic transformation), then the dynamic limit values are reduced.

A compromise must be found between maximum speed and constant acceleration (e.g. Fig. 2-4).

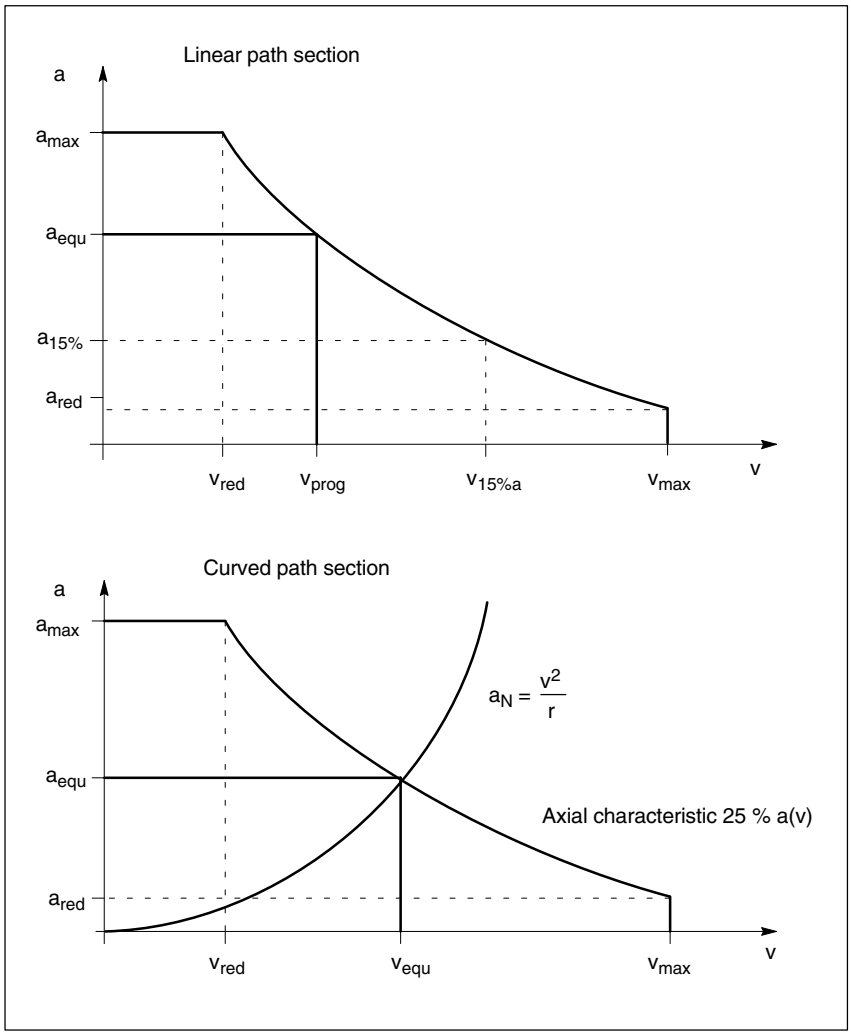


Fig. 2-4 Equivalent path characteristic

a_N: Line normal acceleration
a_{equ}: Constant acceleration of equivalent characteristic
$a_{15\%}$: Minimum constant acceleration
$a_{15\%} = 0.15 \cdot (a_{max} - a_{red}) + a_{red}$
v_{equ}: Speed of equivalent characteristic
v_{prog}: Programmed speed
$v_{15\%a}$: Speed at $a_{15\%}$
r: Radius

If the programmed speed is higher than $v_{15\%a}$, then this value is applied as the limit. A minimum acceleration rate ($a_{15\%}$) is thus ensured.

Activation

The characteristic is activated by means of machine data or NC language commands.

MA_ACCEL_TYPE_DRIVE = 1

The knee-shaped acceleration characteristic is the axial basic setting.All specified parameters of the acceleration characteristic must be observed with every type of movement.

Effect:

1. The knee-shaped acceleration characteristic is always active for single axis movements (positioning, oscillation, JOG...).

   Switchover to BRISKA (constant acceleration over the entire speed range) or SOFTA (constant jerk) is not permitted.

2. If at least one axis for which MA_ACCEL_TYPE_DRIVE = TRUE is programmed is involved in the path movement and DRIVE is **not active**, then the dynamic limit values are reduced (equivalent path characteristic).

NC language commands DRIVE

Activation of knee-shaped acceleration characteristic for path movement is performed with the language command DRIVE or by means of machine data

MC_GCODE_RESET_VALUE[20] = 3.

Value **3** of the MD “MC_GCODE_RESET_VALUE[20]” means that the knee-shaped acceleration characteristic is **active** after activation.

MA_ACCEL_TYPE_DRIVE has no significance in this case.

The path characteristic is determined by the machine data

MA_ACCEL_REDUCTION_SPEED_POINT
MA_ACCEL_REDUCTION_FACTOR
MA_ACCEL_REDUCTION_TYPE

of the relevant axes in combination with the geometry.

Transformation:

The DRIVE acceleration characteristic may not be used in conjunction with an active kinematic transformation. The BRISK characteristic is selected internally instead. If MA_ACCEL_TYPE_DRIVE is set for at least one of the relevant path axes, then the equivalent path characteristic takes effect.

NC language commands DRIVEA(AXIS).

Activation of knee-shaped acceleration characteristic for single-axis movements (positioning axes).

Examples of activation:

Machine data:

DRIVE as power ON setting

MC_GCODE_RESET_VALUE[20] = 3

Parameters of axis characteristic:

X axis
MA_ACCEL_REDUCTION_SPEED_POINT[X] = 0.4
MA_ACCEL_REDUCTION_FACTOR[X] = 0.85
MA_ACCEL_REDUCTION_TYPE[X] = 2
MA_ACCEL_TYPE_DRIVE[X] = TRUE

Y axis
MA_ACCEL_REDUCTION_SPEED_POINT[Y] = 0.0
MA_ACCEL_REDUCTION_FACTOR[Y] = 0.6
MA_ACCEL_REDUCTION_TYPE[Y] = 1
MA_ACCEL_TYPE_DRIVE[Y] = TRUE

Z axis
MA_ACCEL_REDUCTION_SPEED_POINT[Z] = 0.6
MA_ACCEL_REDUCTION_FACTOR[Z] = 0.4
MA_ACCEL_REDUCTION_TYPE[Z] = 0
MA_ACCEL_TYPE_DRIVE[Z] = FALSE

NC program:

N10 G1 X100 Y50 Z50 F700	Path movement (X, Y, Z) with DRIVE
N15 Z20	Path movement (Z) with DRIVE
N20 BRISK	Switchover to BRISK
N25 G1 X120 Y70	Path movement (Y, Z) with equivalent characteristic
N30 Z100	Path movement (Z) with BRISK
N35 POS[X] = 200 FA[X] = 500	Positioning movement (X) with DRIVEA
N40 BRISKA(Z)	Activate BRISKA for Z
N40 POS[Z] = 50 FA[Z] = 200	Positioning movement (Z) with BRISKA
N45 DRIVEA(Z)	Activate DRIVEA for Z
N50 POS[Z] = 100	Positioning movement (Z) with DRIVE
N55 BRISKA(X)	Causes error message
...	

G64 block transition

Axial speed step changes may occur during non-tangential block transitions.

The path speed at the block transition is reduced if the speed of one path axis is higher than the speed at which the acceleration rate is reduced (MA_ACCEL_REDUCTION_SPEED_POINT).

G64 braking ramp

In the case of short blocks, the acceleration or deceleration process may be executed over several path sections.

In this case, the Look Ahead calculation takes the speed-dependent acceleration characteristic into account.

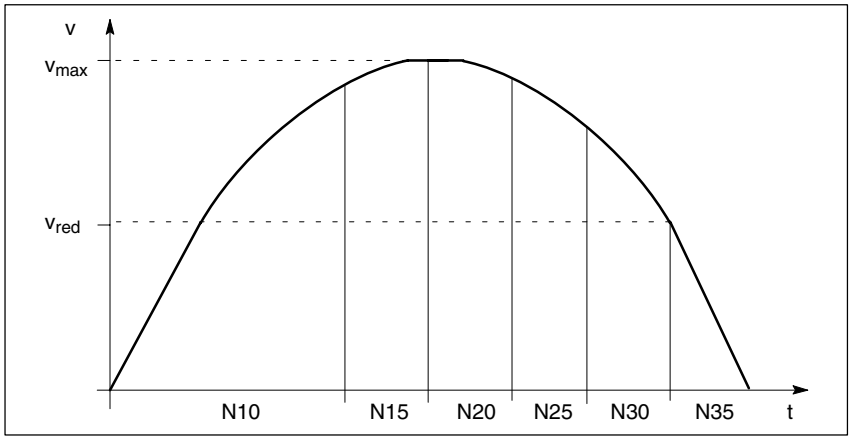


Fig. 2-5 Look Ahead braking ramp

2.3.2 Response to acceleration rate changes

Constant speed time

MD 20500: CONST_VELO_MIN_TIME (minimum time at constant speed) can be programmed to set a time period for which the speed is kept constant in the event of a change from acceleration to deceleration. This function halves any acceleration step change that may occur (e.g. even if the motor stops in an acceleration phase).

Activation

This constant speed time is set in MD 20500: CONST_VELO_MIN_TIME to a value other than zero and applies to positioning axes (additional axes), spindles and geometry axes.

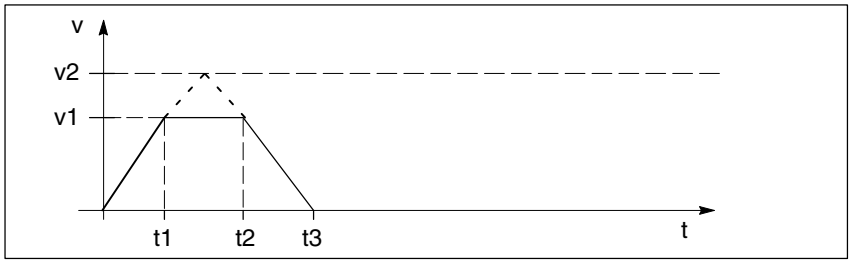


Fig. 2-6 Constant-speed phase

v1: Speed during constant-speed phase
v2: Maximum speed without constant-speed period (not higher than maximum path speed, MD 35220: ACCEL_REDUCTION_SPEED_POINT
t1: Commencement of constant-speed phase
t2: End of constant-speed phase
t3: Stop instants; (t2 – t1) = MD 20500: CONST_VELO_MIN_TIME

2.3.3 Parameterization of stepper motor frequency

Stepper motor frequency

The maximum stepper motor frequency is defined with the machine data

MD 31350: FREQ_STEP_LIMIT [Hz].

$$\text{MD 31350 [Hz]} = \frac{\text{Motor speed [rev/min]} \cdot \text{steps per 360°}}{\text{60 [s]}}$$

This frequency must coincide with MD32000: MAX_AX_VELO.

Example:

MAX_AX_VELO = Variant a or b [mm/min]

a) Speed ratio:

Motor speed [rev/min] · load gear · spindle pitch [mm] = axis velocity

b) Reduction ratio:

$$\frac{\text{Motor speed [rev/min]} \cdot \text{spindle pitch}}{\text{load gear}} = \text{axis velocity}$$

Variant b:	Motor speed:	1 200 rev/min
	Load gear:	1:1
	Spindle pitch:	10 mm
	Steps per 360°:	10 000

$$\frac{\text{1 200 rev/min} \cdot \text{10 mm}}{1} = \text{12 000 mm/min}$$

accordingly, the frequency of MD31350 must be

$$\frac{\text{1 200 rev/min} \cdot \text{10 000}}{\text{60 s}} = \text{200 000 Hz}$$

!

Important

The frequency **must** be entered in **Hz**!

Stepper motor without encoder

When the stepper motor is used **without** an encoder,

MD 31020: ENC_RESOL and MD 31400: STEP_RESOL

the number of steps per 360° must be entered.

Example: Stepper motor pulses per 360° : 10 000
MD 31020: ENC_RESOL [0] = 10 000

Stepper motor with encoder

If a stepper motor is operated **with** encoder, encoder adaptation is made as with analog drives.

Note

The number of steps per 360° must be applied to determine MD 31350.

Example

With encoder

Encoder: 2 500 encoder pulses per revolution

Stepper motor: 10 000 [pulses per motor rotation]

Load gear: 1:1

Spindle pitch: 10 mm

Motor speed: 1 200 rpm

MD 30130: CTRLOUT_TYPE = 2

MD 30240: ENC_TYPE = 2 (with TTL encoder)

MD 31020: ENC_RESOL[0] = 2 500 (pulse quadruplication)

MD 31350: FREQ_STEP_LIMIT[Hz] = 200 000 [Hz]

MD 32 000 = 12 000 mm/min

Without encoder

Stepper motor: 10 000 [pulses per motor rotation]

Load gear: 1:1

Spindle pitch: 10 mm

Motor speed: 1 200 rpm

MD 30130: CTRLOUT_TYPE = 2

MD 30240: ENC_TYPE = 3

MD 31020: ENC_RESOL[0] = 10 000 (no pulse quadruplication)

MD 31350: FREQ_STEP_LIMIT[Hz] = 200 000 [Hz]

MD 32 000 = 12 000 mm/min

Note

To define the adaptation, please observe that the pulses are quadruplicated if an encoder is used.
If no encoder is used, MD 30240: ENC_TYPE = 3

■

Supplementary Conditions 3

Servo gain for SM control without measuring system

The filter of the internal closed-loop system for frequency conversion is rated for a servo sampling time of 5...6 ms at a limit frequency of 60 Hz.
(MD 10050: SYSCLOCK_CYCLE_TIME).

The initial setting for the servo gain $K_v = 2$
(MD 32200: POSCTRL_GAIN, limits 1...5).

Measuring systems

There is no measuring system switchover or disconnected on the SM.

Servo gain for SM control with measuring system

Response and settings as for analog drives.

Servo enabling on SM axes

The interface signal **servo enable SE** (DB31...35DBX2.1) must **not** be used for stepper motor drive axes that have no measuring system. The SE setting must be “1” from the instant of control power-up onwards.

Traverse against fixed stop

The **Traverse against fixed stop** function is **not** available for stepping drive axes that have no measuring system.

References: /FB/, F1, “Traverse against Fixed Stop”

Max. stepper motor frequency

The maximum permissible stepper motor frequency is 1 MHz.

Full/half-step

Interface signal DB 31-35DBX24.1, full-step/half-step for stepper motor is not functional at present.

MDs 31130 to 31160

These machine data are not relevant for the SW 3.2 or for the hardware NCU 570.2.

VDI signals when stepper motor is used in open-loop control mode

When a stepper motor is used as an axis (spindle), VDI signals must be used in the following way:

The “Servo enable” signal is **not used to shut down** the drive via the NC (Drive enable is always active). This affects the following signals:

- Servo enable
- Position measuring system ON/OFF
- Park
- Fault responses

The user is responsible for ensuring that the PLC “safely stops” or shuts down the stepper motor drive concerned.

Drive without measuring system

Machine data $MA_AX_EMERGENCY_STOP_TIME and $MA_SERVO_DISABLE_DELAY_TIME have no effect in stepper motor drives without a measuring system.

■

4 Data Descriptions (MD, SD)

4.1 Axis-specific machine data

31100 MD number	**BERO_CYCLE[n]** Steps between two BERO edges for monitoring stepper motor rotation		
Default value: 2500	Min. input limit: 10	Max. input limit: 10 000 000	
Changes effective after Power On	Protection level: 2	Unit: Steps	
Data type: DWORD	Valid from SW: 1.1		
Significance:	For stepper motor rotation monitoring The number of pulses between two identical BERO edges must be entered		

31110 MD number	**BERO_EDGE[n]** Tolerance of BERO edges for monitoring stepper motor rotation		
Default value: 50	Min. input limit: 10	Max. input limit: 10 000 000	
Changes effective after Power On	Protection level: 2	Unit: Steps	
Data type: DWORD	Valid from SW: 1.1		
Significance:	The number of stepper motor pulses that can be tolerated by the BERO edges during rotation monitoring.		

31350 MD number	**FREQ_STEP_LIMIT** Maximum stepper motor frequency		
Default value: 250.0	Min. input limit: 0.1	Max. input limit: 1 000 000	
Changes effective after NEW_CONF	Protection level: 2	Unit: Hz	
Data type: DOUBLE	Valid from SW: 3.2		
Significance:	MD takes effect when stepper motor is used (FM-NC). Maximum permissible frequency for stepper motor.		

35220 MD number	**ACCEL_REDUCTION_SPEED_POINT** Activation speed/velocity for acceleration reduction		
Default value: 1.0	Min. input limit: 0.0	Max. input limit: 1.0	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DOUBLE	Valid from SW: 3.2		
Significance:	Effective for spindles and positioning axes. The speed/velocity at which the spindle/axis acceleration rate decreases is set here. The reference speed/velocity is the defined maximum speed/velocity of the spindle/axis. The speed/velocity value entered here is a percentage function of the maximum speed/velocity. **Example:** Maximum speed = 10,000 degrees/s, f = 0.7. The reduction in acceleration commences at 70 % of the maximum speed. The motor accelerates at full capacity in the speed range from 0...6,999.99... degrees/s, and at reduced rate in the speed range from 7,000...10,000 degrees/s.		

35230 MD number	**ACCEL_REDUCTION_FACTOR** Factor of acceleration reduction		
Default value: 0.0	Min. input limit: 0.0	Max. input limit: 0.95	
Changes effective after RESET	Protection level: 2	Unit: –	
Data type: DOUBLE	Valid from SW: 3.2		
Significance:	Effective for spindles and positioning axes. The spindle/axis acceleration rate is reduced by the specified factor from the activation speed/velocity onwards. **Example:** a = 50 000 degrees/s^2, v_{ein} = 6 000 degrees/s, f = 0.2: In the speed range from 0 to 5 999.99 degrees/s, the axis/spindle is accelerated/decelerated at a = 50 000 degrees/s^2 . From a speed of 6 000 degrees/s up to maximum speed, the axis/spindle is accelerated/decelerated at a = 40 000 degrees/s^2.		

35240 MD number	**ACCEL_TYPE_DRIVE** Basic setting of acceleration pattern (FM-NC)		
Default value: FALSE	Min. input limit: –	Max. input limit: –	
Changes effective after RESET	Protection level: 2	Unit: –	
Data type: BOOLEAN	Valid from SW: 3.2		
Significance:	Initial setting for acceleration characteristics for all traversing motions: FALSE: No reduction in acceleration TRUE: Acceleration reduction active MD is effective only when JOG_AND_POS_JERK_ENABLE = FALSE		

35242 MD number	**ACCEL_REDUCTION_TYPE** Acceleration reduction characteristics (FM-NC only)		
Default value: 1	Min. input limit: 0	Max. input limit: 2	
Changes effective after RESET	Protection level: 2	Unit: –	
Data type: BYTE	Valid from SW: 3.2		
Significance:	Shape of acceleration reduction characteristic with DRIVE speed control 0: Constant 1: Hyperbolic 2: Linear		

■

Signal Descriptions

5

5.1 Axis-specific signals

DB31, ... **DBX24.0** Data block	**Rotation monitoring ON/OFF** Signal(s) to axis (PLC → NCK)	
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW version: 1.1
Signal state 1 or signal transition 0 → 1	Rotation monitoring active	
Signal state 0 or signal transition 1 → 0	Rotation monitoring off	
Related to	IS “Error speed monitoring” (DB31–38, DBX 96.0)	

DB31, ... **DBX96.0** Data block	**Rotation monitoring error** Signal(s) from axis/spindle (NCK → PLC)	
Edge evaluation: yes	Signal(s) updated: cyclically	Signal(s) valid from SW version: 1.1
Signal state 1 or signal transition 0 → 1	Error during rotation monitoring of this stepper motor axis	
Signal state 0 or signal transition 1 → 0	No error during rotation monitoring of this stepper motor axis	
Related to	IS “Rotation monitoring ON/OFF” (DB31–38, DBX24.0)	

■

Notes

Example 6

An example for the axis configuration with stepper motor is given in the MD file **"md_dr_s.tea"**.

This example MD file can be found under the path **FMNC.TEA** on the "Basic PLC program" diskette which is supplied as standard with the FM-NC.

References: /BU/, SINUMERIK FM-NC/SINUMERIK 840D Ordering Information Catalog NC 60.1

Data Fields, Lists 7

7.1 Machine data

Number	Identifier	Name	Reference
Channel-specific ($MC_ ...)			
20500	CONST_VELO_MIN_TIME	Minimum time with constant velocity	
Axis-specific ($MC_ ...)			
31020	ENC_RESOL[n]	Encoder marks per revolution	G2
31100	BERO_CYCLE[n]	Pulses/stepper motor revolution	
31110	BERO_EDGE[n]	Tolerance of BERO in steps	
34200	ENC_REFP_MODE[n]	Position measuring system type 1 = Incr. measuring system 2 = BERO with single-edge evaluation 3 = Distance-coded reference marks 4 = BERO with dual-edge evaluation	R1
31350	FREQ_STEP_LIMIT	Maximum stepper motor frequency	
35220	ACCEL_REDUCTION_SPEED_POINT	Speed/velocity at which acceleration rate is reduced	
35230	ACCEL_REDUCTION_FAKTOR	Acceleration reduction factor	
35240	ACCEL_TYPE_DRIVE	Basic setting for acceleration characteristics (FM-NC only)	
35242	ACCEL_REDUCTION_TYPE	Shape of acceleration reduction characteristic (FM-NC only)	
32000	MAX_AX_VELO	Max. axial speed	G2
32300	MAX_AX_ACCEL	Max. axial acceleration	B2

7.2 Interface signals

DB number	Bit, byte	Name	Refer-ence
Axis-specific			
31–48	24.0	Rotation monitoring ON/OFF	
13–48	96.0	Error rotation monitoring	

■

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

Memory Configuration (S7)

■

Notes

Brief Description

1

Every CNC requires memory for storing and managing data. This memory can essentially be divided into two areas. One area contains data that doesn't change, such as the software of the CNC. This type of data can be stored on electronic memory chips such as EPROM. The second area contains data stored on the control by the machine manufacturer or user. This data is stored on electronic memory chips such as RAM. The control system enables a RAM area to be set up by the user for various specifications. This description provides information on the areas of RAM that are available to the user and how they can be set up.

Note

The SRAM memory currently available is shown in the Program operating area in the program overview (dialog line).

■

Notes

Detailed Description

2

2.1 General

Active file system

The active file system is the main memory of the CNC. It contains the current system and user data (e.g. machine data) used to operate the control.

Passive file system

The passive file system is the memory area in the control system used to buffer data and programs. The passive file system is organized hierarchically, i.e. in directories and subdirectories.

SRAM

The term SRAM refers to the static RAM of the control system that is available to the user for backing up data. SRAM is also referred to in this documentation as backup, non-volatile or static memory.
The SRAM memory currently available is shown in the Program operating area in the program overview (dialog line).

DRAM

The term DRAM refers to the dynamic RAM of the control system that is available to the user. The data used in this area are generated exclusively by the control, are only required for a certain length of time and do not, therefore, require backup. DRAM is also referred to in this documentation as volatile or dynamic memory.

2.2 Memory organization

Hardware configuration

The following table shows the hardware configuration of the available NC-CPUs:

	D-RAM	S-RAM unbuffered	S-RAM buffered	FLASH	JEIDA
NCU 570		1.5 MB	0.25 MB	2.25 MB	
NCU 571	4 MB		0.5 MB/2.0 MB*		4 MB
NCU 572	8 MB		0.5 MB/2.0 MB*		4 MB
NCU 573	8 MB		0.5 MB/2.0 MB*		4 MB
NCU 573.2	8 MB		2.0 MB		4 MB
NCU 573.2	32 MB*		2.0 MB		4 MB

SRAM

SRAM that is available to the user. It can be configured by means of the machine data described in this Description of Functions.

- NCU 570 128 KB
- NCU 571 256 KB/ 1.5 MB*
- NCU 572 256 KB/ 1.5 MB*
- NCU 573 256 KB/ 1.5 MB*
- NCU 573.2 1.5 MB

*) Available as option, see Catalog NC 60.1

The buffered user memory is entered in KB in MD 18230: MM_USER_MEM_BUFFERED. For NCUs with 2 different memory capacities, the standard entry takes the smallest value into account. If the 2.0 MB version is used, then you have to set MD 18230 explicitly to 1900. (Although the gross value is 2000, it is necessary to make a deduction for internal use).

DRAM

The total amount of DRAM available to the user is displayed in MD 18210: MM_USER_MEM_DYNAMIC (dynamic user memory in DRAM). The value is system-dependent and may vary slightly with different software versions.

The memory areas containing the individual data groups, e.g. global user data, channel-specific user data, axis-specific user data, etc., are arranged contiguously in SRAM and DRAM. The size of a data area in use can be configured in the machine data. The order in which the data areas are arranged is permanently defined by the CNC.

Alteration of memory areas

It is evident from the memory organization described above that any changes to the memory areas must affect the data stored. Every time the system powers up, the CNC compares the current requirement for memory with the existing memory space on the basis of machine data for individual data areas. If the comparison establishes that one or more modified data areas require reallocation of the data areas, the memory is reorganized.

Loss of data during memory reorganization

Reorganization of memory areas always causes loss of all the data in the backup memory.

Backed-up data are also lost if the individual memory areas in the SRAM and DRAM in total exceed the available memory space. It is therefore vital to save the data stored on the CNC before modifying the memory configuration. The SRAM memory available for allocation is indicated in machine data with the identifier INFO_FREE_MEM_.

Memory-configuring machine data

The following list shows some of the machine data which affect the memory configuration:

- System-specific memory settings
- Channel-specific memory settings
- Axis-specific memory settings

The modification of machine data for the SRAM belonging to these groups always leads to the loss of data in the backup memory. The name of these machine data begins with MM_ (e.g. MM_NUM_TOA_MODULES).

The number of active channels also affects the memory configuration. If the number of channels is altered, these channels are set up according to the default settings for the channel-specific memory areas when the system is powered up. Since these areas are also in the SRAM, changing the number of channels also leads to a loss of data in the backup user memory.

After a new value has been entered in a machine data which re-allocates the memory area of the SRAM (see Section 2.4), message 4400 “MD alteration will cause reorganizaton of buffer (data loss!)” is output. This warning indicates that a machine data has been changed, which causes the backup memory to be reorganized when the NCK is powered up, resulting in a loss of all user data stored there. If the memory is to be reorganized and the control contains data which have not been backed up, these should be saved before the next NCK power-up.

Note

The reorganization can be avoided by changing the modified value back to the original value at the time of the last power-up.

Only in exceptional cases does an MD change not cause reorganization of the memory!

In the case of MD 18350: MM_USER_FILE_MM_MINIMUM (minimum part program memory), the memory reorganization is only performed if the remaining RAM is too small.

Loading the memory-configuring standard machine data on the next system power-up through setting system-specific MD 11200: INIT_MD (load standard machine data on next power-up) to 2 causes the backed up user data to be lost if the memory areas are not organized according to the memory default settings before the system power-up process.

2.3 Memory configuration alarms

A modification to the memory allocation that is incorrect or requires memory reorganization causes the output of an alarm message after CNC system power-up. The causes of the faults and the response of the CNC can be summarized as follows:

Alarm 6000

The user memory (static or dynamic) cannot be reallocated because the total memory area available (static or dynamic) is less than the total number of memory areas set by machine data. In this case, all machine data for configuring memory are deleted and assigned their default values. NC machining is no longer possible. This situation is indicated by alarm 6000 “Memory allocation with standard machine data”. It is not possible to pinpoint a particular machine data as the cause of the error in memory allocation. However, it is possible to find the error by altering the machine data for the memory settings one by one. The alarm can be cancelled with RESET. Machining is possible only when the user data have been loaded.

Alarm 6010

After cycle programs, macro definitions or definitions of global user data have been incorporated, alarm 6010 “Channel [name 1] data block [name 2] has not been or is only partially created, error number [identifier]” is output in response to an error. Either the machine data for the corresponding memory areas have been incorrectly configured or the files contain an error. As an example, three files for macro definitions _N_SMAC_DEF, _N_MMAC_DEF and _N_UMAC_DEF contain a total of 30 macro definitions, but the setting in MD 18160: MM_NUM_USER_MACROS (number of macros) restricts the number of macros to 10.

The identifier [name 1] indicates the name of the channel where the error has occurred. The identifier [name 2] indicates the name of the file with the error. The error number is coded as follows with respect to the cause:

Error no.	Explanation
1	No memory available
2	Maximum no. of symbols exceeded
3	Index 1 outside permissible value range
4	Name already exists on channel
5	Name already exists on NCK
>100 000	Unrecoverable system error

If the error number output is between 1 and 5, the user can eliminate the error himself. In cases where the error number is > 100000, the error is an unrecoverable system error.

Machining is possible when the machine data or files have been corrected, or the changes have been cancelled and the system rebooted.

Alarm 6020

The SRAM has been reorganized after a modification to the static memory allocation. All stored data, with the exception of the machine data, have been lost. Alarm 6020 “Machine data altered – memory reallocated” indicates this situation. The SRAM is reallocated when the number of channels on the CNC or the system, channel or axis-specific memory settings for the static memory are altered. The alarm can be cancelled with RESET. Machining can resume when the user data are loaded.

Alarm 6030

The memory area set in MD 18210: MM_USER_MEM_DYNAMIC (user memory in DRAM), MD 18220: MM_USER_MEM_DPR (user memory in dual-port RAM) or MD 18230: MM_USER_MEM_BUFFERED (user memory in SRAM) is larger than the physical memory actually available. In this case, the CNC enters the available memory in the corresponding machine data and displays it with alarm 6030 “User memory limit has been adapted”. In this case, no user data are lost. The alarm can be cancelled with RESET.
Alarm 6000 “Memory allocation with standard machine data” may arise, however, if further machine data were used for the memory allocation assuming that the excessively large data is correct and memory has been allocated over and above the area actually available.

2.4 Memory allocation in SRAM and DRAM

Since in normal practice the SRAM and DRAM memory is only allocated as part of the start-up process, we would recommend the following procedure for allocating memory taking the SRAM as an example:

- Load standard machine data.
- MD 18230: MM_USER_MEM_BUFFERED (user memory in SRAM) is set to a high value (3000). The NCK is then powered up. Alarm 6030 “User memory limit has been adapted” is output and the maximum amount of memory available to the user entered in MD 18230: MM_USER_MEM_BUFFERED.
  All other memory-configuring machine data are set to their default values.
- Activate the number of channels and axes required, for further details see
  References: /IAD/ Installation and Start-Up Guide SINUMERIK 840D
  /IAF/ Installation and Start-Up Guide SINUMERIK FM-NC
- The static memory still available is displayed in MD 18060: INFO_FREE_MEM_STATIC (display of free static memory).
- If the memory default settings do not allocate the memory satisfactorily, then the memory areas can now be re-configured (increase/decrease individual or several areas via machine data) to adapt the memory provided to the requirements on the machine tool.
  Check: Which memory areas require more memory space?
  Which memory areas are less important for the application in question?
- After the appropriate machine data for the selected memory areas have been set to define memory requirements, the NCK is reset in order to reorganize the memory.

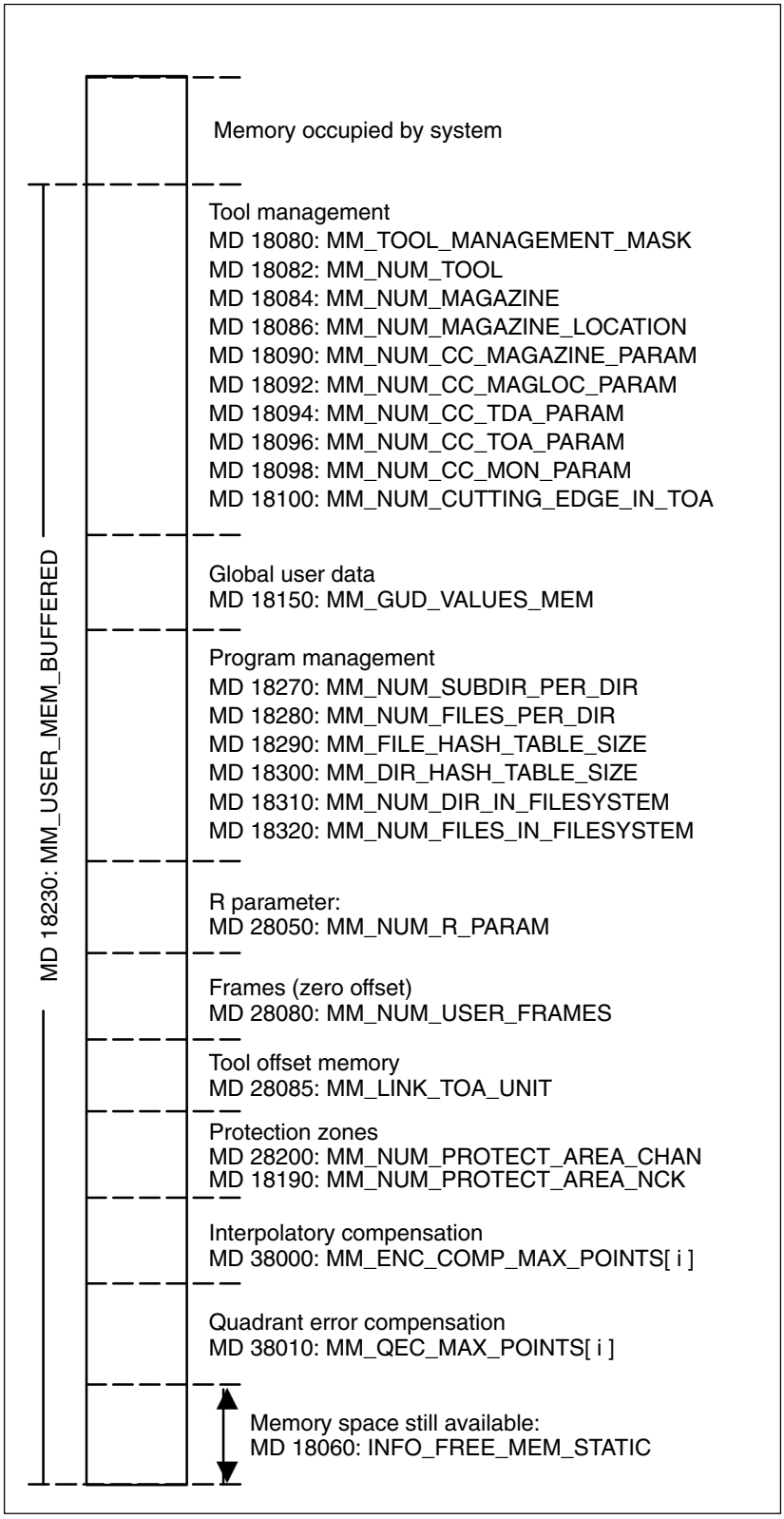


Fig. 2-1 Allocation of static RAM (SRAM)

2.4.1 Memory allocation SRAM

Table 2-1 Allocation of memory space in SRAM

Machine data	Memory requirement	Remark
Tool management		
MD 18080: MM_TOOL_MANAGEMENT_MASK		See detailed description of MD in Chapter 4.
MD 18082: MM_NUM_TOOL	84 bytes per tool	
MD 18084: MM_NUM_MAGAZINE	332 bytes per magazine	
MD 18086: MM_NUM_MAGAZINE_LOCATION	64 bytes per magazine location	
MD 18090: MM_NUM_CC_MAGAZINE_PARAM	Input x no. of magazines x 4 bytes	Corresponds to MD 18084: MM_NUM_MAGAZINE
MD 18092: MM_NUM_CC_MAGLOC_PARAM	Input x no. of magazines x 4 bytes	Corresponds to MD 18084: MM_NUM_MAGAZINE
MD 18094: MM_NUM_CC_TDA_PARAM	Input x no. of tools x 4 bytes	Corresponds to MD 18082: MM_NUM_TOOL
MD 18096: MM_NUM_CC_TOA_PARAM	Input x no. of TOs x 8 bytes	Corresponds to MD 18100: MM_NUM_CUTTING_EDGES_ IN_TOA
MD 18098: MM_NUM_CC_MON_PARAM	Input x no. of TOs x 4 bytes	Corresponds to MD 18100: MM_NUM_CUTTING_EDGES_ IN_TOA
MD 18100: MM_NUM_CUTTING_EDGES_IN_ TOA	Without active monitor: 250 bytes per tool edge With active monitor: Additional 48 bytes per tool edge	
Global user data		
MD 18118: MM_NUM_GUD_MODULES		See following example
MD 18120: MM_NUM_GUD_NAMES_NCK	80 bytes per NCK name	See following example
MD 18130: MM_NUM_GUD_NAMES_CHAN	80 bytes per channel name	See following example
MD 18140: MM_NUM_GUD_NAMES_AXIS	80 bytes per axis name	
MD 18150: MM_GUD_VALUES_MEM		See following example

Example of GUD

An example of how the calculate the memory requirements of global user data is given below.

Supplementary conditions

- Machine with 2 channels.
- The following GUD modules are defined:
  UGUD User-specific
  SGUD Siemens-specific
  MGUD Machine manufacturer-specific
  GUD7 (Contour table stock removal cycle, required for CYCLE95, cycle version 3.4 and higher)
- NCK-specific and channel-specific variables are defined.

NCK variable

```
2   REAL values  –> 2 x 8 bytes     = 16 bytes
1   BOOL values –> 1 x 1 bytes      = 1 byte
                Total 1             = 17  bytes
3 = Total NCK (no. of values)
```

CHAN variable

```
2   BOOL values –> 2 x 1 bytes      = 2 bytes
1   INT values  –> 1 x 4 bytes      = 4 bytes
                Total 2             = 6  bytes
3 = Total CHAN (no. of values)

6 bytes (total 2) x 2 (no. of channels) = 12 bytes (total 3)
```

Calculation of memory required

- MD 18120: MM_NUM_GUD_NAMES_NCK = 3 (total NCK)
  Memory space for management of NCK names
  => 3 x 80 bytes = **240 bytes**
- MD 18130: MM_NUM_GUD_NAMES_CHAN = 3 (total CHAN)
  Memory space for management of CHAN names
  => 3 x 80 bytes = **240 bytes**
- Number of max. defined GUD modules = 7 (GUD7)
  Memory space for management of GUD modules
  => 7 x 120 bytes = **840 bytes**
- Memory requirements for variables
  Total 1 + total 3 =
  17 bytes + 12 bytes = **29 bytes,** rounded up to whole KB gives:
  MD 18150: MM_GUD_VALUES_MEM = 1
- Total memory space required for GUD is calculated as:

Memory space for management of NCK names	=	240 bytes
Memory space for management of CHAN names	=	240 bytes
Memory space for management of GUD modules	=	840 bytes
Memory space required for variables	=	1024 bytes
Total	=	**2344 bytes**

Machine data	Memory requirement	Remark
Program management		
MD 18270: MM_NUM_SUBDIR_PER_DIR	40 bytes per subdirectory	
MD 18280: MM_NUM_FILES_PER_DIR	40 bytes per file	
MD 18290: MM_FILE_HASH_TABLE_SIZE		Assigned internally by the system and must not be altered by user.
MD 18300: MM_DIR_HASH_TABLE_SIZE		Assigned internally by the system and must not be altered by user.
MD 18310: MM_NUM_DIR_IN_FILESYSTEM		See detailed description of MD in Chapter 4.
MD 18320: MM_NUM_FILES_IN_FILESYSTEM	320 bytes per file	
R parameter		
MD 28050: MM_NUM_R_PARAM	8 bytes per R parameter	
Frames (zero offsets)		
MD 28080: MM_NUM_USER_FRAMES	232 bytes per frame An additional 120 bytes are required once for management purposes.	
Tool offset memory		
MD 28085: MM_LINK_TOA_UNIT	500 bytes per unit	
Protection zones		
MD 28200: MM_NUM_PROTECT_AREA_CHAN	400 bytes for each defined block	
MD 18190: MM_NUM_PROTECT_AREA_NCK	400 bytes for each defined area	
Compensations		
MD 18340: MM_NUM_CEC_NAMES	1 KB permanently allocated	
MD 18342: MM_CEC_MAX_POINTS	8 bytes for each compensation point An additional 2 bytes are required once for management purposes.	
MD 38000: MM_ENC_COMP_MAX_POINTS	8 bytes for each compensation point An additional 2 bytes are required once for management purposes.	
MD 38010: MM_QEC_MAX_POINTS	4 bytes for each compensation point An additional 2 bytes are required once for management purposes.	

2.4.2 Memory allocation DRAM

- The dynamic memory still available is displayed in MD 18050: INFO_FREE_MEM_DYNAMIC (display data of available dynamic memory).
- If the memory default settings do not allocate the memory satisfactorily, then the memory areas can now be re-configured (increase/decrease individual or several areas via machine data) to adapt the memory provided to the requirements on the machine tool.
  Check: Which memory areas require more memory space?
  Which memory areas are less important for the application in question?
- After the appropriate machine data for the selected memory areas have been set to define memory requirements, the NCK is reset in order to reorganize the memory.

Note

If more dynamic memory is demanded than the amount actually available, the SRAM(!) is also automatically deleted during the next power-up and all machine data for the memory configuration reset to their default values.

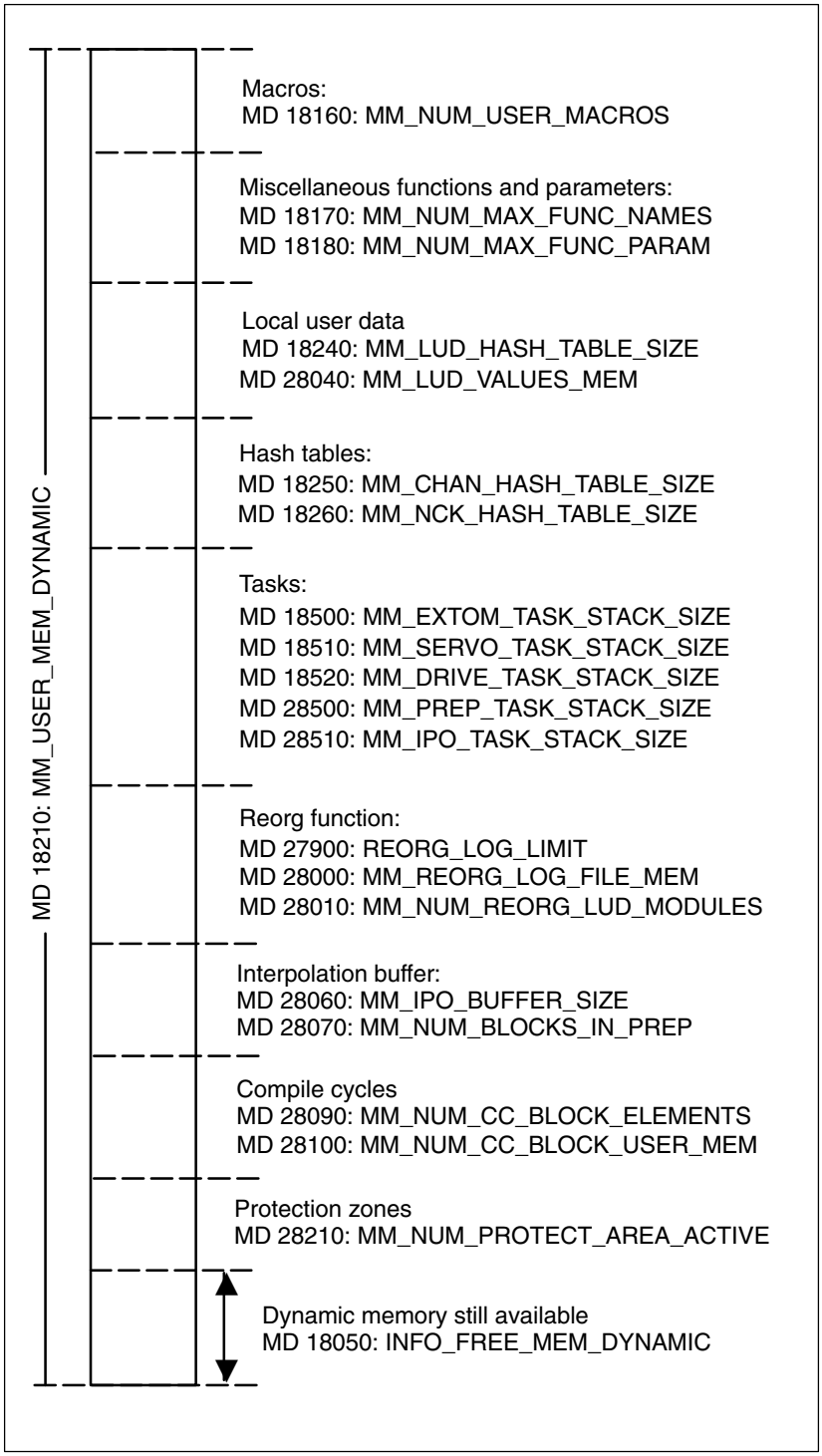


Fig. 2-2 Allocation of memory space in dynamic RAM (DRAM)

Machine data	Memory requirement	Remark
Macros		
MD 18160: MM_NUM_USER_MACROS	375 bytes per macro	
Miscellaneous functions and their additional parameters		
MD 18170: MM_NUM_MAX_FUNC_NAMES	150 bytes per miscellaneous function	
MD 18180: MM_NUM_MAX_FUNC_PARAM	72 bytes per parameter	The entered value is the total of all miscellaneous function parameters required
Local user data		
MD 18240: MM_LUD_HASH_TABLE_SIZE		Assigned internally by the system and must not be altered by user.
MD 18242: MM_MAX_SIZE_OF_LUD_VALUE	Block size depends on variable used Data type Memory requirement REAL 8 bytes INT 4 bytes BOOL 1 byte CHAR 1 byte STRING 1 byte per character, 100 characters permitted per string AXIS 4 bytes FRAME 400 bytes	The machine data must be set for the variable that requires the most memory space. However, it must not be set to a value higher than this variable or else an alarm will be generated See following example
MD 28010: MM_NUM_REORG_LUD_MODULES		Assigned internally by the system and must not be altered by user.
MD 28020: MM_NUM_LUD_NAMES_TOTAL	150 bytes per LUD name	See following example
MD 28040: MM_LUD_VALUES_MEM	Total memory space required for LUD	See following example

Example of local user data

Local user data defined in the part programs are stored in the DRAM while the program in which they are defined is being executed.
An example of how to calculate the memory requirements of local user data is given below.

Parameters of example

The following variables must be used:

1	REAL value	–> 1 x 8 bytes	= 8 bytes
2	BOOL values	–> 2 x 1 bytes	= 2 bytes
		Total 1	= 10 bytes

3 = Total A (no. of values)

Calculation of memory required

- Memory required for variables
  Total 1 = **10 bytes**
- MD 18242: MM_MAX_SIZE_OF_LUD_VALUES = 8 bytes
  The machine data must be set according to the variable that requires the most memory space. In this above example, this is the REAL value with 8 bytes.
- MD 28020: MM_NUM_LUD_NAMES_TOTAL = 3 (total A)
  Memory space for management of LUD NAMES
  => 3 x 150 bytes = **450 bytes**
- MD 28040: MM_LUD_VALUES_MEM

  Total memory space required for LUD is calculated as:

Memory space for management of LUD NAMES	=	450 bytes
Memory space required for variables	=	10 bytes
Total	=	**456 bytes**

  The calculated sum must be rounded in KB and entered in MD 28040: MM_LUD_VALUES_MEM (in this case, 1 KB).
  The memory provided by this setting is allocated in blocks of 8 bytes each in size (according to MD 18242).
  If, for example, 1 REAL value (8 bytes) and 1 BOOL value (1 byte) are used in a program, then 2 blocks of memory are assigned 8 bytes each.

LUD defined in part programs

Local user data defined in part programs are stored in the DRAM while the program in which they are defined is being executed.
During this period, it is possible to view the assigned values under the softkey PARAMETER.

Definition of variables in PP

DEF INT LUD_VARIABLE1	Integer variable with the name VARIABLE1
DEF REAL LUD_VARIABLE2	REAL variable with the name VARIABLE2
DEF REAL LUD_PAUL[19]	Field with 20 REAL variables PAUL[0] – PAUL[19]

Memory management

The system automatically controls the allocation of memory blocks.

- Reservation of a memory block when a part program containing the LUD definition is processed.
- Reservation of further blocks if the memory provided for the number of variables is not sufficient.
- Release of memory space if LUDs are not longer required (at end of program).

Variants of variable definition

When a large number of variables is required, e.g. 20 REAL variables, it is possible to save memory space by defining an ARRAY (field) at the beginning of a part program rather than defining each variable individually.

Example:
Case1
DEF REAL LUD_PAUL[19]
This field with 20 LUD variables PAUL[0] – PAUL[19] requires the following memory space:

MD 28080 = 1	=>	1 x 150 bytes =	150 bytes
Memory for 20 variables	=>	20 x 8 bytes =	160 bytes

Total memory required by 20 variables = 310 bytes

Case2
Individual definition of 20 variables: PAUL0, PAUL1 – PAUL19

MD 28080 = 20	=>	20 x 150 bytes =	3000 bytes
Memory for 20 variables	=>	20 x 8 bytes =	160 bytes

Total memory required by 20 variables = 3160 bytes

Note

This alternative method of variables definition can also be applied to GUD variables.

See MD 18242: MM_MAX_SIZE_OF_LUD_VALUES for LUD variables

Machine data	Memory requirement	Remark
Hash tables		
MD 18250: MM_CHAN_HASH_TABLE_SIZE	Input x no. of channel-spec. names x 68 bytes	Assigned internally by the system and must not be altered by user.
MD 18260: MM_NCK_HASH_TABLE_SIZE	Input x no. of NCK-spec. names x 68 bytes	Assigned internally by the system and must not be altered by user.
Tasks		
MD 18500: MM_EXTCOM_TASK_STACK_SIZE	Input x 1 KB	
MD 18510: MM_SERVO_TASK_STACK_SIZE	Input x 1 KB	
MD 18520: MM_DRIVE_TASK_STACK_SIZE	Input x 1 KB	
MD 28500: MM_PREP_TASK_STACK_SIZE	Input x 1 KB	
MD 28510: MM_IPO_TASK_STACK_SIZE	Input x 1 KB	
Reorg function		
MD 27900: REORG_LOG_LIMIT	Input x 1 KB	Assigned internally by the system and must not be altered by user.
MD 28000: MM_REORG_LOG_FILE_MEM	Input x 1 KB	Assigned internally by the system and must not be altered by user.
MD 28010: MM_NUM_REORG_LUD_MODULES	Input x 1 KB	Assigned internally by the system and must not be altered by user.
Interpolation buffer		
MD 28060: MM_IPO_BUFFER_SIZE	10 KB for each NC block in IPO buffer	
MD 28070: MM_NUM_BLOCKS_IN_PREP	10 KB for each NC block for preparation	
Compile cycles		
MD 28090: MM_NUM_CC_BLOCK_ ELEMENTS	1.2 KB per block element	
MD 28100: MM_NUM_CC_BLOCKS_USER_ MEM	Input / 128 bytes = no. of blocks	The entered value should be a multiple of 128 since the memory is enabled in a grid of 128-byte blocks.
Active protection zones		
MD 28210: MM_NUM_PROTECTED_AREA_ ACTIVE		The value entered should be determined by the total of the settings in MD 18190 and MD 28200. MD 18190 = 2 MD 28200 = 2 => MD 28210 = 4

2.5 Memory requirements calculation

Note

The memory required depends on the software version and type of NC control. The values specified in the table below for the change in memory requirements based on changes in machine data are intended as **standard values** for SW 4 and NCU 572. The standard values and machine data limits for other software versions or other NC controls can be found in:
References: /LIS/, Lists (of software versions used)

Overview

The tables are arranged in the following order:

- DRAM
  - General machine data
  - Channel-specific machine data
  - Axis-specific machine data
- SRAM
  - General machine data
  - Channel-specific machine data
  - Axis-specific machine data

Table entries

1. **MD no.**
   Number of the machine data. The associated identifier can be looked up in /LIS/.
2. **Meaning**
   Meaning of the machine data.
   New line:
   GD: Basic DRAM overhead, **GS:** Basic SRAM overhead
   (This overhead is produced when the function controlled by the MD is used. Values are only specified for MDs which are not directly proportional to the value specified in column 3 or which cannot be calculated.)
3. **Default value** (def)
   Value set when the software is supplied.
4. **Increase def. by 1, extra req. (bytes)**
   Specifies the number of bytes by which the memory requirement changes if the default value is increased by 1.
   The basic overheads for GD and GS specified in column 2 are included in the changes shown.
5. **Increase def. by further (n)**
   Specifies by how many additional units the value of the machine data was increased in the capacity calculation. The increased memory allocation is specified in column 6.
6. **Extra requirement for n, (bytes)**
   Specifies how much additional memory is required if the machine data is increased by the value specified in column 5.
   The basic overheads for GD and GS specified in column 2 are included in the changes shown.
7. **SRAM also affected**
   DRAM also affected
   An x appears in the column if the other type of memory is also occupied proportionally.

Note

The actual dependencies between machine data and required memory are complex. Some MD initiate further functions which also use memory. The relationship between the amount of memory used and the number in the MD is not always linear. The tables below therefore only provide an approximate indication of where memory can be reduced or increased in order to achieve the desired configuration. The information applies both to increasing and reducing the values specified in the machine data.

2.5.1 DRAM memory requirements

Table 2-2 General machine data, DRAM

MD no.	Meaning	Default value	Increase def. by 1 extra req. (bytes)	Increase def. by further (n)	Extra req. for n (bytes)	SRAM also affected
10010	Channels	1	1134608		See SRAM	x
10134	Number of MMC communication partners	6	28236			
18082	Number of tools	30	120	10	1244	x
18088	Number of toolholders **GD:** 588, **GS:** 1293	0	588	n	See SRAM	x
18120	Number of global user variables	10	84	10	828	x
18130	No. of channel-specific user variables	40	84	10	828	x
18140	No. of axis-specific user variables	0	84	10	828	x
18160	No. of macros	10	680	10	6864	
18170	No. of miscellaneous functions (cycles)	40	120	10	1272	
18180	No. of additional parameters for cycles	300	60	10	612	
18190	Number of files for machine-related protection zones **GD:** 504, **GS**: 1062	0	504	n	See SRAM	x
18210	Dynamic user memory	3370	1024	10	10240	
18280	No. of files per directory	100	76			
18342	Max. number of interpolation points for beam sag compensation **GD:** 380, **GS:** 1680	0	380	10	See SRAM	x

Table 2-3 Channel-specific machine data, DRAM

MD no.	Meaning	Default value	Increase def. by 1 extra req. (bytes)	Increase def. by further (n)	Extra req. for n (bytes)	SRAM also affected
28000	Memory capacity for REORG	10	1084	10	10636	
28020	Number of local user variables	200	160	10	1688	
28040	Memory capacity for local user variables	8	1024	10	10260	
28060	No. of NC blocks in IPO buffer	10	15452			
28070	No. of blocks for block preparation	36	15576			
28080	No. of settable frames	5	76	10	784	x
28085	Assignment of TOA unit to a channel	1,2,3...	84		See SRAM	x
28090	No. of block elements for compile cycles	0	924			

Table 2-3 Channel-specific machine data, DRAM

MD no.	Meaning	Default value	Increase def. by 1 extra req. (bytes)	Increase def. by further (n)	Extra req. for n (bytes)	SRAM also affected
28100	Capacity of block memory for compile cycles	256	1056			
28150	Number of elements for writing PLC variables	0	56			
28200	Number of files for channel-spec. protection zones **GD:** 504, **GS**: 1062	0	504		See SRAM	x
28210	Number of simultaneously active protection zones	0	~18000	10	174852	
28250	Number of elements for synchronized action expressions	159	104			
28252	Number of elements for FCTDEF definitions	3	32			

Table 2-4 Axis-specific machine data, DRAM

MD no.	Meaning	Default value	Increase def. by 1 extra req. (bytes)	Increase def. by further (n)	Extra req. for n (bytes)	SRAM also affected
38000	No. of interpolation points for interpolation compensation **GD:** 212, **GS:** 976	0		10	212	x
38010	Number of values for quadrant error compensation **GD:** 548, **GS:** 1932	0		10	548	x

2.5.2 SRAM memory requirements

Table 2-5 General machine data, SRAM

MD no.	Meaning	Default value	Increase def. by 1 extra req. (bytes)	Increase def. by further (n)	Extra req. for n (bytes)	DRAM also affected
10010	Channels	1	10032			x
18082	Number of tools	30	80	10	812	x
18084	Number of magazines	3	244	10	2416	
18086	Number of magazine locations **GD:** 0, **GS:** 6	30	244	31	7612	
18088	Number of toolholders **GD:** 588, **GS:** 1293	0	1408	10	1152	x
18090	Quantity of magazine data for compile cycles **GD**: 0, **GS:** 36	0	40	10	32	
18092	Quantity of magazine data for compile cycles **GD**: 0, **GS:** 36	0	40	10	32	
18094	Quantity of tool-specific data per tool **GD:** 0, **GS:** 31	0	40	9	68	
18096	Quantity of data per tool edge **GD:** 0, **GS:** 31	0	40	9	68	
18098	Quantity of monitoring data per tool edge **GD**: 0, **GS:** 36	0	40	9	32	
18100	Tool offsets per TOA module **GD:** 0, **GS:** 13	30	244	10	2408	
18102	Type of D number programming	0	–2344 (reduced requirement for 1: direct D no. prog.)			
18118	Number of GUD files in active file system	7	628			
18120	Number of global user variables	10	120	10	1200	x
18130	No. of channel-specific user variables	40	120	10	1200	x
18140	No. of axis-specific user variables	0	120	10	1200	x
18150	Memory capacity for user variables	12	1056	10	10548	
18190	Number of files for machine-related protection zones **GD:** 504, **GS**: 1062	0	1464	5	2012	x
18230	User memory in SRAM	1900	1024	10	10232	
18310	No. of directories in passive file system	30	1236			
18320	No. of files in passive file system	100	344			
18342	Number of interpolation points for beam sag compensation **GD:** 380, **GS:** 1680	0		10	1748	x

Table 2-5 General machine data, SRAM

MD no.	Meaning	Default value	Increase def. by 1 extra req. (bytes)	Increase def. by further (n)	Extra req. for n (bytes)	DRAM also affected
18400	Number of curve tables **GD:** 0, **GS:** 4	0	104	1	100	
18402	Number of curve segments **GD:** 0, **GS:** 4	0	128	1	124	
18404	Number of curve table polynomials **GD:** 0, **GS:** 4	0	60	1	56	

Table 2-6 Channel-specific machine data, SRAM

MD no.	Meaning	Default value	Increase def. by 1 extra req. (bytes)	Increase def. by further (n)	Extra req. for n (bytes)	DRAM also affected
28050	No. of channel-specific R parameters	100	8	10	64	
28080	No. of settable frames	5	428	10	4220	x
28085	Assignment of TOA unit to a channel	1,2,3 ...	2124			x
28200	Number of files for channel-specific protection zones **GD:** 504, **GS**: 1062	0	1468	5	2008	x

Table 2-7 Axis-specific machine data, SRAM

MD no.	Meaning	Default value	Increase def. by 1 (bytes)	Increase def. by further (n)	Extra req. (bytes)	DRAM also affected
38000	No. of interpolation points for interpolation compensation **GD:** 212, **GS:** 976	0		10	1040	x
38010	Number of values for quadrant error compensation **GD:** 548, **GS:** 1932	0		10	1996	x

■

Supplementary Conditions

None

Data Descriptions (MD, SD)

4.1 General machine data

18050 MD number	**INFO_FREE_MEM_DYNAMIC** Display data of free dynamic memory		
Default value: –	Min. input limit: –	Max. input limit: –	
Changes effective after Power On	Protection level: 0	Unit: byte	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	The data is used to display the number of bytes available in the dynamic memory. The data cannot be defined. The display is updated with every NCK power-up. Procedure for determining the required memory manually: – Increase the input value by 1 – Power up the NCK – Read off the memory requirements – Calculate possible increase The content of the machine data specifies the amount of dynamic RAM currently available via MD for the expansion of the volatile user data areas. Before expanding a parameter, such as the number of local user data (LUD), it is advisable to check that sufficient memory is available.		
Special cases:	If more dynamic memory is requested than is currently available, the SRAM is deleted on the next power-up and all machine data are initialized with the default settings.		

18060 MD number	**INFO_FREE_MEM_STATIC** Display data of free static memory		
Default value: –	Min. input limit: –	Max. input limit: –	
Changes effective after Power On	Protection level: 0	Unit: byte	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	The contents of the machine data indicate how much non-volatile memory is available for the passive file system **at the time of the power-up**. Then the value is **no longer updated**. To determine the current value at any given time the following operations are available: MMC102: services of programming: free memory NCK MM100: programming; softkey memory info. If MDs that influence the amount of backed-up memory required are altered, then the amount of memory available for the passive file system also changes since the amount of memory allocated to the passive file system consists of the memory setting in MD 18230: MM_USER_MEM_BUFFERED (SRAM user memory) minus all other backup user data (see also description of MD 18350: MM_USER_FILE_MEM_MINIMUM (minimum part program memory)). The data cannot be defined. The display is only updated after every NCK power-up.		
Special cases:	If more static memory is requested than is currently available, the SRAM is deleted on the next power-up and all machine data are initialized with the default settings.		

18070 MD number	**INFO_FREE_MEM_DPR** Display data of free memory in DUAL_PORT RAM		
Default value: –	Min. input limit: –	Max. input limit: –	
Changes effective after Power On	Protection level: 0	Unit: byte	
Data type: DWORD	Applies from SW version:		
Significance:	None		
MD irrelevant for	The functionality is not available with SW 2.		

18080 MD number	**MM_TOOL_MANAGEMENT_MASK** Screen form for reserving memory for TM function		
Default value: 00H	Min. input limit: 00H	Max. input limit: FFFFH	
Changes effective after Power On	Protection level: 1	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	Step-by-step TM-specific memory reservation is implemented on a bit-coded basis by this MD. Reservation of individual memory areas is defined by the following MDs: • MD 18086: MM_NUM_MAGAZINE_LOCATION • MD 18084: MM_NUM_MAGAZINE • MD 18096: MM_NUM_CC_TOA_PARAM • MD 18094: MM_NUM_CC_TDA_PARAM • MD 18098: MM_NUM_CC_MON_PARAM • MD 18092: MM_NUM_CC_MAGLOC_PARAM • MD 18090: MM_NUM_CC_MAGAZINE_PARAM Memory cannot be reserved simply by presetting the individual memory-configuring machine data. The memory configuration is not changed until the appropriate machine data is activated during the next power ON. Bit 1: Make tool management data available: • Memory-reserving MD for basic functionality of tools must be set: MD 18092: MM_NUM_TOOL MD 18100: MM_NUM_CUTTING_EDGES_IN_TOA • Memory-reserving MD for tool management function must be set: MD 18086: MM_NUM_MAGAZINE_LOCATION MD 18084: MM_NUM_MAGAZINE When bit 1 is set, TM-specific memory is added to the memory space defined in MM_NUM_TOOL. Bit 2: Make tool monitoring data available: • Memory-reserving MD for basic functionality of tools must be set: MD 18092: MM_NUM_TOOL MD 18100: MM_NUM_CUTTING_EDGES_IN_TOA When bit 2 is set, memory for monitoring data is made available. TM-specific memory is added to the memory space programmed in MD 18100: MM_NUM_CUTTING_EDGES_IN_TOA. Bit 3: OEM/CC data available: • Memory-reserving MD must be set: MD: MM_NUM_CC_... When bit 3 is set, memory is made available for OEM applications. Bit 4: “Consider adjacent location” tool management: • Make memory available for TM function “Consider adjacent location”		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		
Related to	MD 18084: MM_NUM_MAGAZINE (number of magazines that the NCK can manage) MD 18086: MM_NUM_MAGAZINE_LOCATION (number of magazine locations that the NCK can manage) MD 18090: MM_NUM_CC_MAGAZINE_PARAM (number of magazine data that are set up and evaluated by the CC) MD 18092: MM_NUM_CC_MAGLOC_PARAM (number of magazine location data that are set up and evaluated by the CC) MD 18094: MM_NUM_CC_TDA_PARAM (number of tool-specific data per tool for OEM and compile cycle) MD 18096: MM_NUM_CC_TOA_PARAM (number of data per tool cutting edge for OEM and compile cycle) MD 18098: MM_NUM_CC_MON_PARAM (number of monitoring data per tool cutting edge for OEM and compile cycle)		
References	/FBW/, Description of Functions, Tool Management		

18082 MD number	**MM_NUM_TOOL** Number of tools managed by the NCK		
Default value: 30	Min. input limit: 0	Max. input limit: 600	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	The NC cannot manage more tools than the maximum number entered in the MD. One tool has at least one cutting edge.		
Special cases, errors, ...	The maximum number of possible tools corresponds to the number of cutting edges. This MD must be set even if no tool management function is used. The backup data are lost if this machine data is altered!		
Related to	MD 18100: MM_NUM_CUTTING_EDGES_IN_TOA (number of tool offsets in NCK)		

18084 MD number	**MM_NUM_MAGAZINE** Number of magazines managed by the NC		
Default value: 3	Min. input limit: 0	Max. input limit: 32	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	Number of magazines which NCK can manage. The MDs for TM MD 20310: TOOL_MANAGEMENT_MASK, MD 18080: MM_TOOL_MANAGEMENT_MASK and the optional TM $ON_TECHNO_FUNCTION_MASK must be set.		
MD irrelevant for	MD is irrelevant if the tool management function is not in use.		
Special cases, errors, ...	Only tool management stage 2: Value = 0 –> tool management stage 2 cannot be activated because no memory area has been set up for the data. The backup data are lost if this machine data is altered!		
Related to	MD 18080: MM_TOOL_MANAGEMENT_MASK (screen form for reserving memory for TM function) MD 20310: TOOL_MANAGEMENT_MASK (activation of different variants of tool management function) $ON_TECHNO_FUNCTION_MASK		
References	/FBW/, Description of Functions, Tool Management		

18086 MD number	**MM_NUM_MAGAZINE_LOCATION** Number of magazine locations		
Default value: 30	Min. input limit: 0	Max. input limit: 600	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	Number of magazines which NCK can manage. The MDs for TM MD 20310: TOOL_MANAGEMENT_MASK, MD 18080: MM_TOOL_MANAGEMENT_MASK and the optional TM $ON_TECHNO_FUNCTION_MASK must be set.		
MD irrelevant for	MD is irrelevant if the tool management function is not in use.		
Special cases, errors, ...	Only tool management stage 2: Value = 0 –> tool management stage 2 cannot be activated because no memory area has been set up for the data. The backup data are lost if this machine data is altered!		
Related to	MD 18080: MM_TOOL_MANAGEMENT_MASK (screen form for reserving memory for TM function) MD 20310: TOOL_MANAGEMENT_MASK (activation of different variants of tool management function) $ON_TECHNO_FUNCTION_MASK		
References	/FBW/, Description of Functions, Tool Management		

18090 MD number	**MM_NUM_CC_MAGAZINE_PARAM** TM compile cycles: Number of magazine data		
Default value: 0	Min. input limit: 0	Max. input limit: 10	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	Only if MD for tool management and tool management option are set: Number of magazine data (format IN_int) for which a memory area is set up and which can be evaluated by compile cycles.		
MD irrelevant for	MD is irrelevant if tool management function is not activated.		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		
Related to	MD 18080: MM_TOOL_MANAGEMENT_MASK (screen form for reserving memory for TM function) MD 18084: MM_NUM_MAGAZINE (number of magazines managed by the NC)		
References	/FBW/, Description of Functions, Tool Management		

18092 MD number	**MM_NUM_CC_MAGLOC_PARAM** TM compile cycles: Number of magazine location data		
Default value: 0	Min. input limit: 0	Max. input limit: 10	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	Only if MD for tool management and tool management option are set: Number of magazine data (format IN_int) for which a memory area is set up and which can be evaluated by compile cycles.		
MD irrelevant for	MD is irrelevant if tool management function is not activated.		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		
Related to	MD 18080: MM_TOOL_MANAGEMENT_MASK (screen form for reserving memory for TM function)		
References	/FBW/, Description of Functions, Tool Management		

18094 MD number	**MM_NUM_CC_TDA_PARAM** TM compile cycles: Number of TDA data		
Default value: 0	Min. input limit: 0	Max. input limit: 10	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	Only if MD for tool management and tool management option are set: Number of TDA (tool-specific) data (format IN_int) for which a memory area is set up and which can be evaluated by compile cycles.		
MD irrelevant for	MD is irrelevant if tool management function is not activated.		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		
Related to	MD 18080: MM_TOOL_MANAGEMENT_MASK (screen form for reserving memory for TM function) MD 18082: MM_NUM_TOOL (number of tools managed by the NCK)		
References	/FBW/, Description of Functions, Tool Management		

18096 MD number	**MM_NUM_CC_TOA_PARAM** TM compile cycles: Number of TOA data		
Default value: 0	Min. input limit: 0	Max. input limit: 10	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	Only if MD for tool management and tool management option are set: Number of TOA (tool-specific) data (format IN_int) per cutting edge for which a memory area is set up and which can be evaluated by compile cycles.		
MD irrelevant for	Tool management stages 1 and 2 not activated.		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		
Related to	MD 18080: MM_TOOL_MANAGEMENT_MASK (screen form for reserving memory for TM function) MD 18100: MM_NUM_CUTTING_EDGES_IN_TOA (number of tool offsets in NCK)		
References	/FBW/, Description of Functions, Tool Management		

18098 MD number	**MM_NUM_CC_MON_PARAM** TM compile cycles: Number of monitor data		
Default value: 0	Min. input limit: 0	Max. input limit: 10	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	For tool management compile cycles: Number of monitor data which are created for each tool and which can be evaluated by compile cycles.		
MD irrelevant for	MD is irrelevant if tool management function is not activated.		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		
Related to	MD 18080: MM_TOOL_MANAGEMENT_MASK (screen form for reserving memory for TM function) MD 18100: MM_NUM_CUTTING_EDGES_IN_TOA (number of tool offsets in NCK)		
References	/FBW/, Description of Functions, Tool Management		

18100 MD number	**MM_NUM_CUTTING_EDGES_IN_TOA** Number of tool offsets in NCK		
Default value: 30	Min. input limit: 0	Max. input limit: 600	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the number of tool cutting edges in the NCK. This machine data reserves approximately 250 bytes of backup memory per TOA module for each tool edge, irrespective of the tool type.		
Special cases, errors, ...	The backup data are lost if this machine data is altered.		
References	/FBW/, Description of Functions, Tool Management		

18118 MD number	**MM_NUM_GUD_MODULES** Number of GUD modules		
Default value: 3	Min. input limit: 1	Max. input limit: 9	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	A GUD block corresponds to a file in which user-defined data can be stored. 9 GUD blocks are available of which 3 are already assigned to specific users/applications. UGUD_DEF_USER (block for user) SGUD_DEF_USER (block for SIEMENS) MGUD_DEF_USER (block for machine manufacturer)		
Special cases, errors, ...	The number of GUD modules is determined by the GUD with the highest number. Example: If the following GUD modules are defined, UGUD MGUD GUD5 GUD8 then the machine data must be set to a value of 8, signifying a memory requirement of 8 x 120 bytes = 960 bytes. It is therefore advisable to selected the “lowest” possible GUD module. If GUD modules UGUD and MGUD have not been assigned elsewhere, then they may be used for this purpose.		
Related to:	MD 18150: MM_GUD_VALUES_MEM (memory for user variables)		

18120 MD number	**MM_NUM_GUD_NAMES_NCK** Number of global user variables		
Default value: 10	Min. input limit: 0	Max. input limit: 400	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the number of user variables for NCK global user data (GUD). Approximately 8 bytes of memory per variable are reserved in the SRAM for the name of the variable. The additional memory required for the value of the variable depends on the data type of the variable. The number of available NCK-global user variables is restricted by the limit value set in MM_NUM_GUD_NAMES_NCK or MD 18150: MM_GUD_VALUES_MEM (memory for user variables) is reached.		
Special cases, errors, ...	The backup data are lost if this machine data is altered.		
Related to	MD 18150: MM_GUD_VALUES_MEM (memory for user variables)		

18130 MD number	**MM_NUM_GUD_NAMES_CHAN** Number of channel-specific user variables		
Default value: 10	Min. input limit: 0	Max. input limit: 200	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the number of user variables for channel-specific global user data (GUD). Approximately 8 bytes of memory per variable are reserved in the SRAM for the name of the variable. The additional memory required for the variable value is equal to the size of the data type of the variable multiplied by the number of channels. This means that each channel has its own memory available for the variable values. The number of channel-specific, global user variables available is exhausted when the limit defined in MD 18130: MM_NUM_GUD_NAMES_CHAN or MD 18150: MM_GUD_VALUES_MEM (memory for user variables) is reached.		
Special cases, errors, ...	The backup data are lost if this machine data is altered.		
Related to	MD 18150: MM_GUD_VALUES_MEM (memory for user variables)		

18140 MD number	**MM_NUM_GUD_NAMES_AXIS** Number of axis-specific user variables		
Default value: 0	Min. input limit: 0	Max. input limit: 100	
Changes effective after Power On	Protection level: 0	Unit: –	
Data type: DWORD	Applies from SW version:		
Special cases, errors, ...	The backup data are lost if this machine data is altered.		
MD irrelevant for	The functionality is not available with SW 2.		

18150 MD number	**MM_GUD_VALUES_MEM** Memory space for user variables		
Default value: 2	Min. input limit: 0	Max. input limit: 50	
Changes effective after Power On	Protection level: 2	Unit: KB	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	The specified value reserves memory for the variables of the global user data (GUD). The dimensioning of the memory depends to a large extent on the data types used for the variables. Overview of memory used by data types: Data type Memory used REAL 8 bytes INT 4 bytes BOOL 1 byte CHAR 1 byte STRING 1 byte per character, 100 characters permitted per string AXIS 4 bytes FRAME 400 bytes The total memory required by channel or axis-specific global user data is the memory used by the variables multiplied by the number of channels or axes. The number of global user variables available is exhausted when the limits defined in the MD: MM_NUM_GUD_NAMES_xxxx or MM_GUD_VALUES_MEM are reached. The battery-backed memory is used.		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		
Related to	MD 18118: MM_NUM_GUD_MODULES: (number of GUD modules) MD 18120: MM_NUM_GUD_NAMES_NCK (number of global user variables) MD 18130: MM_NUM_GUD_NAMES_CHAN (number of channel-specific user variables)		

18160 MD number	**MM_NUM_USER_MACROS** Number of macros		
Default value: 10	Min. input limit: 0	Max. input limit: 100	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the total number of macros which can be stored in the files _N_SMAC_DEF, _N_NMAC_DEF and _N_UMAC_DEF. When opened, each of these files requires at least one KB of memory for the file code. If this limit for the file code is exceeded, another Kbytes is reserved for the file. Dynamic user memory is used. Approximately 375 bytes per macro are reserved for the specified number of macros for management tasks.		
Special cases, errors, ...	The backup data are lost if this machine data is altered.		

18170 MD number	**MM_NUM_MAX_FUNC_NAMES** Number of miscellaneous functions		
Default value: 30	Min. input limit: 0	Max. input limit: 50	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	The machine data limits the maximum number of miscellaneous functions over and above the predefined functions (such as sine, cosine, etc.) which can be used in – cycle programs – compile cycle software. The function names are entered in the global NCK dictionary and may not conflict with the names that already exist. The SIEMENS cycle package of SW 1 contains miscellaneous functions that are taken into account by the default setting of the MD. The data are stored in volatile memory. Approximately 150 bytes are required for each miscellaneous function for management purposes.		
Related to	MD 18180: MM_NUM_MAX_FUNC_PARAM (no. of miscellaneous function parameters)		

18180 MD number	**MM_NUM_MAX_FUNC_PARAM** Number of additional parameters		
Default value: 300	Min. input limit: 0	Max. input limit: 500	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the maximum number of parameters required for the miscellaneous functions in – cycle programs – compile cycle software. 50 parameters are required for the miscellaneous functions of the Siemens cycle package of SW 1. The data are stored in volatile memory. Approximately 72 bytes are required for each parameter.		
Related to	MD 18170: MM_NUM_MAX_FUNC_NAMES (number of miscellaneous functions)		

18190 MD number	**MM_NUM_PROTECT_AREA_NCK** Number of protection zones in NCK		
Default value: NCU 570: 0 NCU 571: 0 NCU 572: 0 NCU 573: 0	Min. input limit: NCU 570: 0 NCU 571: 0 NCU 572: 0 NCU 573: 0	Max. input limit: NCU 570: 4 NCU 571: 4 NCU 572: 10 NCU 573: 10	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	This machine data defines how many blocks are created for the protection zones available in the NCK.		
Special cases, errors, ...	The backup data are lost if this machine data is altered.		
References	/FB/, A3, “Axis/Contour Tunnel Monitoring, Protection Zones”		

18210 MD number	**MM_USER_MEM_DYNAMIC** User memory in DRAM		
Default value: NCU 570: 1750 NCU 571: 1750 NCU 572: 3370 NCU 573: 3500 810D: 2000 810D_2: 3500	Min. input limit: NCU 570: – NCU 571: – NCU 572: – NCU 573: – 810D: – 810D_2: –	Max. input limit: NCU 570: – NCU 571: – NCU 572: – NCU 573: – 810D: – 810D_2: –	
Changes effective after Power On	Protection level: 2/7	Unit: KB	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	The DRAM which physically exists in the NC is used jointly by the system and the user. MM_USER_MEM_DYNAMIC defines the amount of memory in the D-RAM that is available to the user. The input limits are dependent on the hardware and software configuration of the CNC. This memory area contains various types of user data such as – local user data – REORG-LOG data. The data in the dynamic memory are not backed up. The input limits ensure that the memory space reserved does not exceed the amount of memory which is actually available in the hardware.		
Application	When the default values are set, the following DRAM memory is available to the user with the NCU 572/573 depending on the number of defined channels: – Approx. 1 MB (1 channel defined) – Approx. 300 KB (2 channels defined).		
Special cases, errors, ...	During power-up, the system software compares the total demands for DRAM with the value set in MD: MM_USER_MEM_DYNAMIC. If the memory required exceeds the capacity defined in the machine data, alarm 6000 “Memory allocation with standard machine data” is output. Alarm 6030 “User memory limit has been adjusted” is output if the control system detects during power-up that the memory capacity requested through MM_USER_MEM_DYNAMIC is greater than the physical memory size.		
Related to:	The available dynamic memory is displayed in MD 18050: INFO_FREE_MEM_DYNAMIC (display data for available DRAM).		

18220 MD number	**MM_USER_MEM_DPR** User memory in dual-port RAM		
Default value: 0	Min. input limit: 0	Max. input limit: 0	
Changes effective after Power On	Protection level: 0	Unit: KB	
Data type: DWORD	Applies from SW version:		
MD irrelevant for	The functionality is not available with SW 2.		

18230 MD number	**MM_USER_MEM_BUFFERED** User memory in SRAM		
Default value: NCU 570: 238 NCU 571: 480 NCU 572: 480 NCU 573: 480	Min. input limit: NCU 570: 100 NCU 571: 100 NCU 572: 100 NCU 573: 100	Max. input limit: NCU 570: 238 NCU 571: 1900 NCU 572: 1900 NCU 573: 1900	
Changes effective after Power On	Protection level: 2	Unit: KB	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the size of the battery-backed user memory. Various types of user data are stored in this area such as, for example: – NC part programs – R parameters – Tool data – User macros – Global user data The settable values are dependent on the hardware and software configurations. 512 KB are available in the hardware for the NCU 571. 512 KB or 2 MB are available for the NCU 572/573 depending on the hardware configuration. The CNC requires approximately 30 KB of this for its operating system, leaving 480 KB. Some of this remaining memory is allocated for further areas permanently reserved for machine data, setting data and data management. The CNC manufacturer guarantees 256 KB user memory in the SRAM. The availability of more than 256 KB user memory cannot be guaranteed in conjunction with the following software versions. **SRAM with 2 MB:** If the NCU 572/573 is used with a larger memory, then the memory must be released. • Enter the value 1900 in MD 18230 • Make a copy of a series start-up file • Execute POWER ON (in order to reorganize the memory) • Load the series start-up file back into the control system During power-up, the system software compares the total amount of battery-backed memory required with the value set in MD 18230: MM_USER_MEM_BUFFERED.If the memory required exceeds the capacity defined in the machine data, alarm 6000 “Memory allocation with standard machine data” is output. Alarm 6030 “User memory limit has been adjusted” is output if the control system detects during power-up that the memory capacity requested in MD 18230: MM_USER_MEM_BUFFERED is larger than the physical memory.		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		

18240 MD number	**MM_LUD_HASH_TABLE_SIZE** Hash table size for user variables		
Default value: 11	Min. input limit: 3	Max. input limit: 107	
Changes effective after Power On	Protection level: 0	Unit: prime number	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the memory size for local user data (LUD). The value entered must be a prime number. The setting allows the optimization of – the interpreter execution time (smaller value = larger execution time) – memory requirements (smaller value = less dynamic memory). A larger table requires a smaller number of decoding operations for decoding the variables and consequently a shorter interpreter execution time. The value set in this machine data affects the amount of dynamic memory required for the management of the modules for the local user variables with REORG, see MD 28010: MM_NUM_REORG_LUD_MODULES (number of modules for local user variables with REORG (DRAM)).		
Note	This machine data is assigned internally by the control and must not be altered by the user.		

18242 MD number	**MM_MAX_SIZE_OF_LUD_VALUE** Maximum field size of LUD variables		
Default value: 8192 Default setting: 496 (from SW4.1)	Min. input limit: 128 240 (from SW4.1)	Max. input limit: 8192 496 (from SW4.1)	
Changes effective after Power On	Protection level: 2	Unit: byte	
Data type: DWORD	Applies from SW version: 2		
Significance:	MD 18242: MM_MAX_SIZE_OF_LUD_VALUE specifies the block size in which the total memory defined in MD 28040: MM_LUD_VALUES_MEM is assigned to the part programs of the channel. The first variable to occur in the part program occupies a block of the size specified in MD 18242: MM_MAX_SIZE_OF_LUD_VALUE. The following variables are also stored in this block. If the block is full of values or cannot accommodate any further variable, then another block is requested. MD 18242: MM_MAX_SIZE_OF_LUD_VALUE must be set to the same value as the memory required by the largest possible variable used. Data type — Memory used REAL — 8 bytes INT — 4 bytes BOOL — 1 byte CHAR — 1 byte STRING — 1 byte per character, 100 characters permitted per string AXIS — 4 bytes FRAME — 400 bytes		
Related to	MD 28040: MM_LUD_VALUES_MEM (memory size for local user variables (DRAM))		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		

18250 MD number	**MM_CHAN_HASH_TABLE_SIZE** Hash table size for channel-specific data		
Default value: 7	Min. input limit: 7	Max. input limit: 193	
Changes effective after Power On	Protection level: 0	Unit: prime number	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the size for channel-specific names. The value entered must be a prime number. The setting allows the optimization of – the interpreter execution time (smaller value = larger execution time) – memory requirements (smaller value = less dynamic memory). A larger table requires a smaller number of decoding operations for decoding the variables and consequently a shorter interpreter execution time. The value of this machine data affects the amount of dynamic memory required. The required memory for each channel in bytes is equal to the value entered multiplied by 68.		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		
Note	This machine data is assigned internally by the control and must not be altered by the user.		

18260 MD number	**MM_NCK_HASH_TABLE_SIZE** Hash table size for global data		
Default value: 1201	Min. input limit: 537	Max. input limit: 1201	
Changes effective after Power On	Protection level: 0	Unit: prime number	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the size for NCK-specific names. The value entered must be a prime number. The setting allows the optimization of – the interpreter execution time (smaller value = larger execution time) – memory requirements (smaller value = less dynamic memory). A larger table requires a smaller number of decoding operations for decoding the variables and consequently a shorter interpreter execution time. The value of this machine data affects the amount of dynamic memory required. The required memory for each channel in bytes is equal to the value entered multiplied by 68.		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		
Note	This machine data is assigned internally by the control and must not be altered by the user.		

18270 MD number	**MM_NUM_SUBDIR_PER_DIR** Number of subdirectories		
Default value: NCU 570: 16 NCU 571: 24 NCU 572: 30 NCU 573: 30	Min. input limit: NCU 570: 16 NCU 571: 24 NCU 572: 24 NCU 573: 24	Max. input limit: NCU 570: 32 NCU 571: 32 NCU 572: 32 NCU 573: 32	
Changes effective after Power On	Protection level: 1	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the maximum number of subdirectories that a directory in the passive file system can have. The number of directories is limited by MD 18310: MM_NUM_DIR_IN_FILESYSTEM (no. of directories in passive file system). The memory requirement is contained in the memory for the number of files per directory (see MD 18260: MM_NUM_FILES_PER_DIR).		
Related to	MD 18310: MM_NUM_DIR_IN_FILESYSTEM (no. of directories in passive file system)		

18280 MD number	**MM_NUM_FILES_PER_DIR** Number of files per directory		
Default value: NCU 570: 32 NCU 571: 24 NCU 572: 100 NCU 573: 100	Min. input limit: NCU 570: 16 NCU 571: 24 NCU 572: 64 NCU 573: 64	Max. input limit: NCU 570: 104 NCU 571: 512 NCU 572: 512 NCU 573: 512	
Changes effective after Power On	Protection level: 1	Unit:	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Specifies the maximum number of files which can be created in a directory or subdirectory. The total number of files is limited by MD 18320: MM_NUM_FILES_IN_FILESYSTEM (no. of files in passive file system). The memory in bytes required for the management of files in the directory is the value entered multiplied by 40. The highest value of MD 18280: MM_NUM_FILES_PER_DIR (number of files per directory) and MD 18270: MM_NUM_SUBDIR_PER_DIR (no. of subdirectories) must be entered as the MD setting. The memory required to manage files in the passive file system is reserved by MD 18320: MM_NUM_FILES_IN_FILESYSTEM.		
Special cases, errors, ...	The backup data are lost if this machine data is altered. Note: An alteration of the MD has an effect on directories created after this. This means that if the number of files in an existing directory is to be altered, the existing directory must first be deleted and then a new directory must be made (but only after having first saved the files)!		
Related to	MD 18320: MM_NUM_FILES_IN_FILESYSTEM (number of files in passive file system)		

18290 MD number	**MM_FILE_HASH_TABLE_SIZE** Hash table size for files of a directory		
Default value: NCU 570: 19 NCU 571: 19 NCU 572: 19 NCU 573: 19	Min. input limit: NCU 570: 3 NCU 571: 3 NCU 572: 3 NCU 573: 3	Max. input limit: NCU 570: 67 NCU 571: 67 NCU 572: 299 NCU 573: 299	
Changes effective after Power On	Protection level: 0	Unit: prime number	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the size of the hash table for the files of a directory. The value entered must be a prime number. The setting allows the optimization of – the interpreter execution time (smaller value = larger execution time) – memory requirements (smaller value = less dynamic memory). The value of this machine data affects the amount of static memory required for the management of directories, see MD 18310: MM_NUM_DIR_IN_FILESYSTEM (no. of directories in passive file system)		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		
Note	This machine data is assigned internally by the control and must not be altered by the user.		

18300 MD number	**MM_DIR_HASH_TABLE_SIZE** Hash table size for subdirectories		
Default value: 7	Min. input limit: 3	Max. input limit: 37	
Changes effective after Power On	Protection level: 0	Unit: prime number	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the size of the hash table for the subdirectories of a directory. The value entered must be a prime number. The setting allows the optimization of – the interpreter execution time (smaller value = larger execution time) – memory requirements (smaller value = less dynamic memory). The value of this machine data affects the amount of static memory required for the management of directories, see MD 18310: MM_NUM_DIR_IN_FILESYSTEM (no. of directories in passive file system)		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		
Note	This machine data is assigned internally by the control and must not be altered by the user.		

18310 MD number	**MM_NUM_DIR_IN_FILESYSTEM** Number of directories in passive file system		
Default value: NCU 570: 32 NCU 571: 30 NCU 572: 30 NCU 573: 30	Min. input limit: NCU 570: 24 NCU 571: 30 NCU 572: 30 NCU 573: 30	Max. input limit: NCU 570: 64 NCU 571: 150 NCU 572: 150 NCU 573: 150	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type:DWORD	Applies from SW version: 1.1		
Significance:	This machine data limits the number of directories in the passive file system and can be used to reserve memory in the SRAM for the management of the directories. The directories and subdirectories of the passive file system set up by the system are included in this machine data. The memory required for the management of the directories can be calculated as follows: Memory required = a (440+28 (b+c)) bytes a = Input value of MD 18310: MM_NUM_DIR_IN_FILESYSTEM (no. of directories in passive file system) b = Input value of MD 19300: MM_DIR_HASH_TABLE_SIZE (HASH table size for subdirectories) c = Input value of MD 18290: MM_FILE_HASH_TABLE_SIZE (hash table size for the files of a directory)		
Special cases, errors, ...	The backup data are lost if this machine data is altered.		
Related to	MD 18270: MM_NUM_SUBDIR_PER_DIR (no. of subdirectories)		

18320 MD number	**MM_NUM_FILES_IN_FILESYSTEM** Number of files in passive file system		
Default value: NCU 570: 64 NCU 571: 128 NCU 572: 100 NCU 573: 100	Min. input limit: NCU 570: 24 NCU 571: 24 NCU 572: 64 NCU 573: 64	Max. input limit: NCU 570: 128 NCU 571: 512 NCU 572: 512 NCU 573: 512	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the number of files available in the part program memory. This machine data is used to reserve memory in SRAM – approximately 320 bytes – for file management. Each file created requires a minimum of one Kbytes of memory for the file code. If the one KB limit for the file code is exceeded another Kbytes is reserved for the file.		
Special cases, errors, ...	The backup data are lost if this machine data is altered.		
Related to:	MD 18280: MM_NUM_FILES_PER_DIR (number of files in directories)		

18330 MD number	**MM_CHAR_LENGTH_OF_BLOCK** Maximum length of an NC block		
Default value: 256	Min. input limit: 32	Max. input limit: 512	
Changes effective after Power On	Protection level: 0	Unit: character	
Data type: DWORD	Applies from SW version: –		
Significance:	None		
MD irrelevant for	The functionality is not available with SW 2.		

18340 MD number	**MM_NUM_CEC_NAMES** Number of LEC tables		
Default value: 4	Min. input limit: 4	Max. input limit: 4	
Changes effective after Power On	Protection level: 0	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	MM_NUM_CEC_NAMES cannot be altered! References to data for CEC (cross error compensation) or leadscrew error compensation are performed in the NC program. For this purpose, a data block is provided that can accommodate the maximum number of names specified in MM_NUM_CEC_NAMES. The actual compensation table is assigned to one of the names. The compensation values are stored in the backed-up user memory, their number can be defined in MD 38000: MM_ENC_COMP_MAX_POINTS (no. of interpolation points with interpolation compensation). The memory required is approximately 1 KB in this software version.		
Special cases, errors, ...	In Software Version 2, the names for the inclusion of CEC data are permanently assigned.The backup data are lost if this machine data is altered.		

18342 MD number	**MM_CEC_MAX_POINTS** Maximum table size for beam sag compensation		
Default value: 0	Min. input limit: 0	Max. input limit: 2000	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	Maximum table size for interpolative compensation between axes. When MM_CEC_MAX_POINTS = 0, no memory is set up for the table. The sag compensation function cannot then be used.		
Special cases:	A change in this machine data causes reconfiguration of the buffered memory area.		

18350 MD number	**MM_USER_FILE_MEM_MINIMUM** Minimum part program memory		
Default value: 20	Min. input limit: 20	Max. input limit: 100	
Changes effective after Power On	Protection level: 1	Unit: KB	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the minimum backup memory area remaining for the files of the passive file system. The settable value depends on the hardware and software configurations (SRAM memory allocation) and on MD 18230: MM_USER_MEM_BUFFERED (SRAM user memory). When the SRAM memory is allocated, the remaining memory is allocated to the files of the passive file system. In order to ensure that the file system can operate, the memory space specified in MM_USER_FILE_MEM_MINIMUM must be available to the memory. If this is not assured, the memory is allocated with the default data on the control and all the data stored in the SRAM by the user is lost. Alarm 6000 “Memory allocation with standard machine data” is also output. The available part program memory capacity is displayed in MD 18060: INFO_FREE_MEM_STATIC (display of free static memory).		
Special cases, errors, ...	The backup data are lost if this machine data is altered and the remaining memory is smaller than the value in MM_USER_FILE_MEM_MINIMUM.		

18500 MD number	**MM_EXTCOM_TASK_STACK_SIZE** Stack size for external communication task		
Default value: 8	Min. input limit: 4	Max. input limit: 20	
Changes effective after Power On	Protection level: 0	Unit: KB	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	The size of the stack for external communication. The dynamic memory area is used.		
Note	This machine data is assigned internally by the control and must not be altered by the user.		

18510 MD number	**MM_SERVO_TASK_STACK_SIZE** Stack size for servo task		
Default value: 8	Min. input limit: 4	Max. input limit: 20	
Changes effective after Power On	Protection level: 0	Unit: KB	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the stack size for the servo task. The dynamic memory is used for this purpose.		
Note	This machine data is assigned internally by the control and must not be altered by the user.		

18520 MD number	**MM_DRIVE_TASK_STACK_SIZE** Stack size for drive task		
Default value: 8	Min. input limit: 4	Max. input limit: 20	
Changes effective after Power On	Protection level: 0	Unit: KB	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	The size of the stack for the SIMODRIVE task is defined with this machine data. The stack is stored in the dynamic memory area.		
Note	This machine data is assigned internally by the control and must not be altered by the user.		

4.2 Channel-specific machine data

27900 MD number	**REORG_LOG_LIMIT** Percentage of IPO buffer for release of log file		
Default value: 1	Min. input limit: –127	Max. input limit: 127	
Changes effective after Power On	Protection level: 0	Unit: –	
Data type: BYTE	Applies from SW version: 1.1		
Significance:	The machine data defines the percentage of the IPO buffer above which data in the REORG LOG memory can be released in stages, if the block preparation has been interrupted due to an overflow of the REORG LOG data memory. The released data are no longer available to the REORG function (References: /FB /, K1, “Mode Groups, Channels, Program Operation Mode”). A consequence of this status is that a further REORG command is cancelled with an error message. If the status of “non-reorganizability” occurs, warning 15110 is output. The output of the warning can be suppressed by enabling the highest significant bit. The bit is set by adding the value 128 to the input value in REORG_LOG_LIMIT. In addition to the instructions of the NC blocks, the size of the IPO buffer and the REORG data memory also affect the frequency of data release.		
Related to	MD 28000: MM_REORG_LOG_FILE_MEM (memory size for REORG) MD 28060: MM_IPO_BUFFER_SIZE (no. of blocks in the IPO buffer)		

28000 MD number	**MM_REORG_LOG_FILE_MEM** Memory size for REORG		
Default value: NCU 570: 10 NCU 571: 10 NCU 572: K1=10, K2=1 NCU 573: K1=10, K2=1	Min. input limit: NCU 570: 0 NCU 571: 0 NCU 572: 0 NCU 573: 0	Max. input limit: NCU 570: 50 NCU 571: 500 NCU 572: 500 NCU 573: 500	
Changes effective after Power On	Protection level: 2	Unit: KB	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the size of dynamic memory for the REORG LOG data. The size of the memory determines the amount of data available for the REORG function.		
References	/FB/, K1, “Mode Groups, Channels, Program Operation Mode”		

28010 MD number	**MM_NUM_REORG_LUD_MODULES** Number of modules for local user variables with REORG		
Default value: 4	Min. input limit: 0	Max. input limit: 100	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the number of additional LUD modules provided for the REORG function (see Description of Functions, Mode Groups, Channels, Program Operation Mode (K1)). If the REORG function is not used, this value can be 0. The CNC always opens 12 LUD modules: 8 for NC programs and 4 for asynchronous subprograms. One LUD module is required for each NC program or asynchronous subprogram containing a definition of a local variable. It may be necessary to increase this value for REORG if a larger IPO buffer is provided and a large number of short NC programs containing LUD variable definitions are active (the NC blocks of the program are stored in prepared format in the IPO buffer. One LUD module is required for each of these programs. The capacity of the reserved memory is affected by the number of LUDs per NC program and their individual memory requirements. The LUD modules are stored in dynamic memory. The memory required for managing the modules for local user variables with REORG can be calculated as follows: Memory = a x (200 + b x 160) bytes a = total number of LUD modules = 8 + 4 + value in MD: MM_NUM_REORG_LUD_MODULES b = Input value of MD 18240: MM_LUD_HASH_TABLE_SIZE (hash table size for user variables) The size of the LUD modules depends on the number of active LUDs and their data types. The memory for LUD modules is limited by MD 28000: MM_REORG_LOG_FILE_MEM (memory size for REORG).		
Application	Example: A main program consisting of 4 NC blocks is started: – A LUD variable is defined in the first block. – A subprogram, nested up to 8 levels, is called in each of the second and third blocks – The fourth block terminates the program. Each subprogram comprises 3 NC blocks: – A LUD variable is defined in the first block. – A subprogram call to the next program level is executed in the second block. – The third block terminates the subprogram. Instead of a subprogram call, the subprogram in the last program level contains a different command, such as a traversing movement. This makes a total of 15 programs with 46 NC blocks which can all be stored in prepared format in the IPO buffer. Since the REORG function requires all the data of the 46 blocks, LUD modules for 3 programs are missing. A value of 3 for the additionally required LUD data blocks must be entered in MM_NUM_REORG_LUD_MODULES for the example given.		

28020 MD number	**MM_NUM_LUD_NAMES_TOTAL** Number of local user variables		
Default value: NCU 570: 200 NCU 571: 200 NCU 572: K1=200, K2=20 NCU 573: K1=200, K2=20	Min. input limit: NCU 570: 0 NCU 571: 0 NCU 572: 0 NCU 573: 0	Max. input limit: NCU 570: 300 NCU 571: 500 NCU 572: 500 NCU 573: 500	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the number of variables for the local user data (LUD) which are permitted to exist in the active sections of the program. Approximately 150 bytes of memory per variable are reserved for the names of the variables and the variable value. The memory required for the variable value is equal to the size of the data type. If the total of the local user variables from the active main program and the related subprograms are larger than the defined limit, the variables which are over the limit are not accepted during execution of the program.Dynamic memory is used for the variable names and variable values. Overview of the memory used by the data types: Data type — Memory used REAL — 8 bytes INT — 4 bytes BOOL — 1 byte CHAR — 1 byte STRING — 1 byte per character, 200 characters per string AXIS — 4 bytes FRAME — 400 bytes		

28040 MD number	**MM_LUD_VALUES_MEM** Memory size for local user variables		
Default value: 20, 20, 20, 20, 20, 20, 20, ...,	Min. input limit: 1	Max. input limit: 100	
Changes effective after Power On	Protection level: 2	Unit: KB	
Data type: DWORD	Applies from SW version: 4.3		
Significance:	This MD defines the amount of memory space available for LUD variables. The number of available LUDs is exhausted when one of the limit values in either MD 28020: MM_NUM_LUD_NAMES_TOTAL or MM_LUD_VALUES_MEM is reached. The memory defined here is subdivided into (MM_LUD_VALUES_MEM * 1024) / MM_MAX_SIZE_OF_LUD_VALUE blocks and allocated to part programs which request memory. Each part program which contains at least one definition of LUD variables or which has call parameters uses at least one such block. It should be remembered that several part programs can be open at once and thus use memory on the NCK. The number depends on the type of programming, the program length and the size of the internal NCK block memory upwards of (MM_IPO_BUFFER_SIZE, MM_NUM_BLOCKS_IN_PREP).		
Related to:	MD 28020: MM_NUM_LUD_NAMES_TOTAL (number of local user variables (DRAM))		

28050 MD number	**MM_NUM_R_PARAM** Number of channel-specific R parameters		
Default value: 100, 100, 100, 100, 100, 100, 100, ...	Min. input limit: 0	Max. input limit: 10000	
Changes effective after Power On	Protection level: 0/0	Unit: –	
Data type: DWORD	Applies from SW version: 4.3		

28050 MD number	**MM_NUM_R_PARAM** Number of channel-specific R parameters
Significance:	Defines the number of R parameters available on the channel. A maximum of 10000 R parameters are available for each channel. This machine data is used to reserve 8 bytes of backup user memory for each R parameter. R parameters require substantially less management overhead compared with LUD and GUD variables.
Special cases, errors, ...	The backup data are lost if this machine data is altered!

28060 MD number	**MM_IPO_BUFFER_SIZE** Number of NC blocks in IPO buffer	
Default value: 10, 10, 10, 10, 10, 10, 10, 10	Min. input limit: 2	Max. input limit: NCU 570: 30 NCU 571: 15 NCU 572, 573: 300 810D: 15 810D_2: 300
Changes effective after Power On	Protection level: 0/0	Unit: –
Data type: DWORD	Applies from SW version: 4.3	
Significance:	Defines the number of blocks in the interpolation buffer. This buffer contains prepared NC blocks which are provided for interpolation. Approximately 10 Kbytes of dynamic user memory is reserved for each NC block. The data also limits the number of Look Ahead blocks for limiting the speed in the Look Ahead function. The MM_IPO_BUFFER_SIZE is set by the system.	
Related to:	MD 28070: MM_NUM_BLOCKS_IN_PREP (number of blocks for block preparation)	

28070 MD number	**MM_NUM_BLOCKS_IN_PREP** Number of blocks for block preparation	
Default value: NCU 570: 36, 36, 36, 36, 36, 36, 36, 36 NCU 571: 26, 26, 26, 26, 26, 26, 26, 26 NCU 572: 36, 36, 36, 36, 36, 36, 36, ... NCU 573: 36, 36, 36, 36, 36, 36, 36, ... 810D: 26, 26, 26, 26, 26, 26, 26, 26 810D_2: 36, 36, 36, 36, 36, 36, 36, 36	Min. input limit: 20	Max. input limit: NCU 570: *** NCU 571: *** NCU 572: *** NCU 573: *** 810D: *** 810D_2: ***
Changes effective after Power On	Protection level: 0/0	Unit: Number of internal blocks
Data type: DWORD	Applies from SW version: 4.3	
Significance:	Defines the number of blocks available for NC block preparation. This figure is determined mainly by the system software and is used for optimization. Approximately 10 Kbytes of dynamic memory is reserved per NC block.	
Related to:	MD 28060: MM_IPO_BUFFER_SIZE (number of NC blocks with IPO buffer)	

28080 MD number	**MM_NUM_USER_FRAMES** Number of settable frames	
Default value: 5	Min. input limit: 5	Max. input limit: 100
Changes effective after Power On	Protection level: 2	Unit: –
Data type: DWORD	Applies from SW version: 1.1	
Significance:	Defines the number of predefined user frames. Approximately 400 bytes of backup memory are reserved per frame. The standard configuration on the system provides four frames for G54 to G57 and one frame for G500.	
Special cases, errors, ...	The backup data are lost if this machine data is altered!	

28085 MD number	**MM_LINK_TOA_UNIT** Assignment of a TO unit to a channel	
Default value: NCU 570: 1 NCU 571: 1 NCU 572: [Channel No.] NCU 573: [Channel No.]	Min. input limit: NCU 570: 1 NCU 571: 1 NCU 572: 1 NCU 573: 1	Max. input limit: NCU 570: 1 NCU 571: 1 NCU 572: 2 NCU 573: 2
Changes effective after Power On	Protection level:	Unit: –
Data type: DWORD	Applies from SW version: 2	
Significance:	A TO unit is assigned to each channel through a default setting. The memory is thus reserved for the data blocks (tools, magazines). A TOA unit can also be assigned to several channels. Def.: The **TOA area** is the sum of all TOA and magazine blocks in the NC. The **TOA unit** consists of a TOA block and, with activated TM function, a magazine block.	
Special cases, errors, ...	The backup data are lost if this machine data is altered!	

28090 MD number	**MM_NUM_CC_BLOCK_ELEMENTS** Number of block elements for compile cycles	
Default value: NCU 570: 0 NCU 571: 0 NCU 572: 0 NCU 573: 0	Min. input limit: NCU 570: 0 NCU 571: 0 NCU 572: 0 NCU 573: 0	Max. input limit: NCU 570: 0 NCU 571: 0 NCU 572: 10 NCU 573: 10
Changes effective after Power On	Protection level: 0	Unit: –
Data type: DWORD	Applies from SW version: 1.1	
Significance:	The value defines the number of block elements used for compile cycles. Approximately 1.2 KB of dynamic memory per block element is required for SW 2.	

28100 MD number	**MM_NUM_CC_BLOCK_USER_MEM** Size of block memory for compile cycles	
Default value: NCU 570: 0 NCU 571: 0 NCU 572: 256 NCU 573: 256	Min. input limit: NCU 570: 0 NCU 571: 0 NCU 572: 0 NCU 573: 0	Max. input limit: NCU 570: 256 NCU 571: 256 NCU 572: 256 NCU 573: 256
Changes effective after Power On	Protection level: 0	Unit: KB
Data type: DWORD	Applies from SW version: 1.1	
Significance:	The value defines the total capacity of block memory available to the user in the dynamic memory area for the compile cycles. The memory is allocated in staggered blocks of 128 bytes.	

28200 MD number	**MM_NUM_PROTECT_AREA_CHAN** Number of blocks for channel-specific protection zones		
Default value: NCU 570: 0 NCU 571: 0 NCU 572: 0 NCU 573: 0	Min. input limit: NCU 570: 0 NCU 571: 0 NCU 572: 0 NCU 573: 0	Max. input limit: NCU 570: 0 NCU 571: 0 NCU 572: 10 NCU 573: 10	
Changes effective after Power On	Protection level: 2	Unit:	
Data type: DWORD	Applies from SW version: 2		
Significance:	This machine data defines how many blocks are created for channel-specific protection zones.		
Related to	MD 28210: MM_NUM_PROTECT_AREA_ACTIVE (number of simultaneously active protection zones) MD 18190: MM_NUM_PROTECT_AREA_NCK (number of files for machine-related protection zones (SRAM))		
References	/FB/, A3, “Axis/Contour Tunnel Monitoring, Protection Zones”		

28210 MD number	**MM_NUM_PROTECT_AREA_ACTIVE** Number of simultaneously active protection zones		
Default value: NCU 570: 0 NCU 571: 0 NCU 572: 0 NCU 573: 0	Min. input limit: NCU 570: 0 NCU 571: 0 NCU 572: 0 NCU 573: 0	Max. input limit: NCU 570: 4 NCU 571: 4 NCU 572: 10 NCU 573: 10	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	The machine data specifies for each channel the number of protection zones that may be activated simultaneously. It is not suitable to enter a value higher than the setting in MD 18190: MM_NUM_PROTECT_AREA_NCK + MD 28200: MM_NUM_PROTECT_AREA_CHAN.		
Related to	MD 28200: MM_NUM_PROTECT_AREA_CHAN (number of blocks for channel-specific protection zones) MD 18190: MM_NUM_PROTECT_AREA_NCK (number of files for machine-related protection zones (SRAM))		
References	/FB/, A3, “Axis/Contour Tunnel Monitoring, Protection Zones”		

28500 MD number	**MM_PREP_TASK_STACK_SIZE** Stack size for preparation task		
Default value: NCU 570: 12 NCU 571: 12 NCU 572: 20 NCU 573: 20	Min. input limit: NCU 570: 4 NCU 571: 4 NCU 572: 4 NCU 573: 4	Max. input limit: NCU 570: 40 NCU 571: 40 NCU 572: 40 NCU 573: 40	
Changes effective after Power On	Protection level: 0	Unit: KB	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the stack size for the preparation task. The stack is stored in dynamic memory.		
Note	This machine data is assigned internally by the control and must not be altered by the user.		

28510 MD number	**MM_IPO_TASK_STACK_SIZE** Stack size for IPO task		
Default value: NCU 570: 8 NCU 571: 8 NCU 572: 12 NCU 573: 12	Min. input limit: NCU 570: 4 NCU 571: 4 NCU 572: 4 NCU 573: 4	Max. input limit: NCU 570: 40 NCU 571: 40 NCU 572: 40 NCU 573: 40	
Changes effective after Power On	Protection level: 0	Unit: KB	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	The stack size for the IPO task is stored in this data. The stack is set up in dynamic memory.		
Note	This machine data is assigned internally by the control and must not be altered by the user.		

28550 MD number	**MM_PRSATZ_MEM_SIZE** Available memory for internal blocks		
Default value: 400	Min. input limit: 100	Max. input limit: 4000	
Changes effective after Power On	Protection level: 0	Unit: KB	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	None. This MD no longer exists in SW 2.		

4.3 Axis-specific machine data

38000 MD number	**MM_ENC_COMP_MAX_POINTS** Number of interpolation points for interpolative compensation		
Default value: 0	Min. input limit: 0	Max. input limit: 5000	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	Defines the number of leadscrew compensation values per encoder for the axis. This value reserves 8 bytes of backup user memory for each compensation value. If more memory for compensation values is required than available in the SRAM, the control outputs alarm 6000 “Memory allocation with standard machine data” on power-up.		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		
References	/FB/, K3, “Compensations”		

38010 MD number	**MM_QEC_MAX_POINTS** Number of values for quadrant error compensation		
Default value: 0	Min. input limit: 0	Max. input limit: 1040	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: DWORD	Applies from SW version: 2		
Significance:	Number of possible values for quadrant error compensation with neural network (option). With a setting of “0”: The quadrant error compensation function cannot be activated, no memory is set up for the function.		
Special cases, errors, ...	The backup data are lost if this machine data is altered!		
References	/IAD/, “SINUMERIK 840D Installation and Start-up Guide” /FB/, K3, “Compensations”		

■

5.6.7 Signal Descriptions

None

Example

None

Data Fields, Lists

7.1 Machine data

Number	Identifier	Name	Reference
General ($MN_ ...)			
18050	INFO_FREE_MEM_DYNAMIC	Display data of free dynamic memory	
18060	INFO_FREE_MEM_STATIC	Display data of free static memory	
18070	INFO_FREE_MEM_DPR	Display data of free memory in dual port RAM	
18080	MM_TOOL_MANAGEMENT_MASK	Screen form for TM memory reservation	/FBW/
18082	MM_NUM_TOOL	Number of tools managed by NCK	
18084	MM_NUM_MAGAZINE	Number of magazines managed by NCK	/FBW/
18086	MM_NUM_MAGAZINE_LOCATION	Number of magazine locations	/FBW/
18090	MM_NUM_CC_MAGAZINE_PARAM	TM compile cycles: Number of magazine data	/FBW/

Number	Identifier	Name	Reference
General ($MN_ ...)			
18092	MM_NUM_CC_MAGLOC_PARAM	TM compile cycles: Number of magazine location data	/FBW/
18094	MM_NUM_CC_TDA_PARAM	TM compile cycles: Number of TDA data	/FBW/
18096	MM_NUM_CC_TOA_PARAM	TM compile cycles: Number of TOA data	/FBW/
18098	MM_NUM_CC_MON_PARAM	TM compile cycles: Number of monitor data	/FBW/
18100	MM_NUM_CUTTING_EDGES_IN_TOA	Number of tool offsets in NCK	
18118	MM_NUM_GUD_MODULES	Number of GUD blocks	
18120	MM_NUM_GUD_NAMES_NCK	Number of global user variables	
18130	MM_NUM_GUD_NAMES_CHAN	No. of channel-specific user variables	
18140	MM_NUM_GUD_NAMES_AXIS	No. of axis-specific user variables	
18150	MM_GUD_VALUES_MEM	Memory capacity for global user variables	
18160	MM_NUM_USER_MACROS	No. of macros	
18170	MM_NUM_MAX_FUNC_NAMES	No. of miscellaneous functions	
18180	MM_NUM_MAX_FUNC_PARAM	No. of miscellaneous function parameters	
18190	MM_NUM_PROTECT_AREA_NCK	Number of protection zones in NCK	/FB/, A3
18210	MM_USER_MEM_DYNAMIC	User memory in DRAM	
18220	MM_USER_MEM_DPR	User memory in dual port RAM	
18230	MM_USER_MEM_BUFFERED	User memory in SRAM	
18240	MM_LUD_HASH_TABLE_SIZE	Hash table size for user variables	
18242	MM_MAX_SIZE_OF_LUD_VALUE	Maximum field size of LUD variables	
18250	MM_CHAN_HASH_TABLE_SIZE	Hash table size for channel-specific data	
18260	MM_NCK_HASH_TABLE_SIZE	Hash table size for global data	
18270	MM_NUM_SUBDIR_PER_DIR	No. of subdirectories	
18280	MM_NUM_FILES_PER_DIR	No. of files per directory	
18290	MM_FILE_HASH_TABLE_SIZE	Hash table size for files of a directory	
18300	MM_DIR_HASH_TABLE_SIZE	Hash table size for subdirectories	
18310	MM_NUM_DIR_IN_FILESYSTEM	No. of directories in passive file system	
18320	MM_NUM_FILES_IN_FILESYSTEM	No. of files in passive file system	
18330	MM_CHAR_LENGTH_OF_BLOCK	Maximum length of an NC block	
18340	MM_NUM_CEC_NAMES	No. of CEC compensation tables	
18342	MM_CEC_MAX_POINTS	Maximum table size for sag compensation	
18350	MM_USER_FILE_MEM_MINIMUM	Minimum part program memory	
18500	MM_EXTCOM_TASK_STACK_SIZE	Stack size for external communication task	
18510	MM_SERVO_TASK_STACK_SIZE	Stack size for servo task	
18520	MM_DRIVE_TASK_STACK_SIZE	Stack size for drive task	
Channel-specific ($MC_ ...)			

Number	Identifier	Name	Refer-ence
General ($MN_ ...)			
20096	T_M_ADDRESS_EXIT_SPINO	Spindle number as address extension (SW 5 and higher)	/FBW/, W1
27900	REORG_LOG_LIMIT	Percentage of IPO buffer for release of log file	
28000	MM_REORG_LOG_FILE_MEM	Memory capacity for REORG	/FB/, K1
28010	MM_NUM_REORG_LUD_MODULES	No. of modules for local user variables with REORG	
28020	MM_NUM_LUD_NAMES_TOTAL	No. of local user variables	
28040	MM_LUD_VALUES_MEM	Memory capacity for local user variables	
28050	MM_NUM_R_PARAM	No. of channel-specific R parameters	
28060	MM_IPO_BUFFER_SIZE	No. of NC blocks in IPO buffer	
28070	MM_NUM_BLOCKS_IN_PREP	No. of blocks for block preparation	
28080	MM_NUM_USER_FRAMES	No. of settable frames	
28085	MM_LINK_TOA_UNIT	Assignment of a TO unit to a channel	/FBW/, W1
28090	MM_NUM_CC_BLOCK_ELEMENTS	No. of block elements for compile cycles	
28100	MM_NUM_CC_BLOCK_USER_MEM	Capacity of block memory for compile cycles	
28200	MM_NUM_PROTECT_AREA_CHAN	Number of blocks for channel-specific protection zones	/FB/, A3
28210	MM_NUM_PROTECT_AREA_ACTIVE	Number of simultaneously active protection zones	/FB/, A3
28500	MM_PREP_TASK_STACK_SIZE	Stack size for preparation task	
28510	MM_IPO_TASK_STACK_SIZE	Stack size for IPO task	
28550	MM_PRSATZ_MEM_SIZE	Available memory for internal blocks	
Axis-specific ($MC_ ...)			
38000	MM_ENC_COMP_MAX_POINTS	No. of interpolation points for interpolation compensation	/FB/, K3
38010	MM_QEC_MAX_POINTS	Number of values for quadrant error compensation	/FB/, K3 /IAD/

7.2 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

■

Notes

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

Indexing Axes (T1)

■

Notes

Brief Description

1

Indexing axes on machine tools

In certain applications, the axis is only required to approach specific grid points (e.g. location numbers).
It is necessary to approach the defined grid points (called indexes) both in automatic and set-up modes. These axes are known as “indexing axes”. The positions defined on the indexing axes are known as “coded positions” or “index positions”.
In SW 4.3 and higher, special functions are available for equidistant indexing on linear and rotary axes and for the Hirth tooth system.

Applications

Indexing axes are used predominantly in connection with specific types of tool magazine such as tool turrets, tool chain magazines or tool cartridge magazines. The coded positions refer to the individual locations of the tools in the magazines. During a tool change, the magazine is positioned at the location containing the tool to be loaded.

■

Notes

Detailed Description

2

2.1 Traversing indexing axes

General

Indexing axes can be traversed manually in the setup mode types JOG and INC, from a part program with special instructions for “Coded positions” and by the PLC. When the indexing position has been reached, the “indexing axis in position” interface signal (DB31–61, DBX76.6) is output to the PLC.

Hirth indexing axes cannot be traversed in JOG mode before reference point approach.

2.1.1 Traversing indexing axes in manual JOG mode

Reference point approach

An indexing axis approaches the reference point in the same way as other axes. It is not necessary for the reference point to match an indexing position. Only when **the reference point has been reached** (IS “Referenced/synchronized 1 or 2” (DB31–61, DBX60.4 or 5) = “1”) does the indexing axis approach only indexing positions in JOG mode with conventional and incremental traversing.
Exception: when traversing with the handwheel, no indexing positions are approached.

If the axis is not referenced (IS “Referenced/synchronized 1 or 2” = “0”), the indexing positions are ignored when the axis is traversed in manual jog mode!

Continuous traversal in JOG

- Jog mode (SD: JOG_CONT_MODE_LEVELTRIGGRD = “1”):

  Pressing a “+” or “–” traversing key causes the indexing axis to move in the same way as with conventional JOG traversing. When the traversing key is released, the indexing axis traverses to the next indexing position in the direction of traversing.

- Continuous mode (SD: JOG_CONT_MODE_LEVELTRIGGRD = “0”):

  Pressing the traversing key briefly (first rising signal edge) starts the traversing movement of the indexing axis in the desired direction. Traversing continues when the traversing key is released. When the traversing key is pressed again (second rising signal edge), the indexing axis traverses to the next indexing position in the direction of traversing.

Indexing axes are generally traversed in JOG mode (standard setting). Continuous mode plays a less important role.

If the operator changes the direction of traversing before the indexing position has been reached, the indexing axis is positioned on the next indexing position in the direction of traversing. The traversing movement must be started in the opposite direction.

For further information on continuous traversing in jog or continuous mode, please see:
References: /FB/, H1 “Manual and Handwheel Travel”

Incremental traversal in JOG (INC)

Irrespective of the increment value currently set (INC1; ... ,INCvar), the indexing axis **always traverses incrementally by 1 indexing position** in the selected direction when a “+” or “–” traversing key is pressed.
In jog mode, the traversing movement is interrupted when the traversing key is released. The indexing position can be approached by pressing the traversing key again.
In continuous mode, the traversing movement is aborted when the traversing key is pressed again. The indexing axis is, in this case, not located on the indexing position.

Between two indexing positions

If an indexing axis is situated between 2 indexing positions, then is approaches the next-higher indexing position when the “+” traversing key is pressed in JOG-INC mode. Similarly, pressing the “–” traversing key causes the next lower indexing position to be approached.

Handwheel traversal

When the indexing axis is traversed by means of the handwheel in JOG mode, the indexing positions **are ignored**. Rotating the handwheel traverses the indexing axis to any position in mm, inches or degrees, according to the standard unit of measurement.
The PLC user program can disable the handwheel for traversing the indexing axis.

Signal from PLC “Indexing axis in position”

When the indexing axis is traversed in JOG mode, the signal “Indexing axis in position” (DB31–61, DBX76.6) is output at the PLC interface to indicate that an indexing position has been reached. The indexing axis must have been referenced (IS “Referenced/synchronized 1 or 2” = “1”.

Alarms in JOG mode

If the indexing axis leaves the traversing range defined in the indexing position table (see 2.2) when traversing in JOG mode, alarm 20054 “wrong index for indexing axis in JOG” is output.

Revolutional feedrate

In JOG mode the behavior of the axis/spindle also depends on the setting of setting data JOG_REV_IS_ACTIVE (revolutional feedrate when JOG active).

- If this setting data is active, an axis/spindle is always moved with revolutional feedrate MD JOG_REV_VELO (revolutional feedrate with JOG) or MD JOG_REV_VELO_RAPID (revolutional feedrate with JOG with rapid traverse overlay) depending on the master spindle.
- If the setting data is not active, the behavior of the axis/spindle depends on the setting data ASSIGN_FEED_PER_REV_SOURCE (revolutional feedrate for positioning axes/spindles).
- If the setting data is not active, the behavior of a geometry axis on which a frame with rotation is effective depends on the channel-specific setting data JOG_FEED_PER_REV_SOURCE. (In the operating mode JOG, revolutional feedrate for geometry axes on which a frame with rotation is effective).

2.1.2 Traversing indexing axes in AUTOMATIC modes

Traversal to selected positions

An axis defined as an indexing axis can be made to approach **any selected position** from the NC part program in AUTOMATIC mode. This includes positions between the defined indexing positions. These positions are programmed, in the usual way, in the unit of measurement (mm/inches or degrees) for the axis. The general programming instructions used for this purpose (G90, G91, AC, IC, etc.) are described in the Programming Guide.

Traversal to “Coded Positions”

Special instructions can also be programmed in the part program:

- CAC Approach absolute coded position
- CACP Approach absolute coded position in positive direction
- CACN Approach absolute coded position in negative direction
- CIC Approach incremental coded position
- CDC Approach coded position along direct (shortest) path

to traverse in the specified manner.

With absolute positioning, the indexing position to be approached is programmed, and with incremental positioning, the number of indexes to be traversed in the “+” or “–” direction is programmed.

On rotary axes, the indexing position can be approached directly across the shortest path (CDC) or with a defined direction of rotation (CACP, CACN). Please refer to Section 2.3 for further information on the special programming instructions for indexing axes.

"Indexing axis in position" interface signal up to SW 4.3/2.3

When an indexing axis is traversing in AUTOMATIC mode, interface signal "Indexing axis in position" is set only if special instructions for approaching "Coded positions" (CAC, CIC, etc.) have been used. In all other cases, the interface signal remains set to "0", even if a position approached with G90 or G91 exactly matches an indexing position.

SW 4.3 and higher:
If the "Exact stop fine" window is reached and the indexing axis is positioned on an indexing position, the signal is enabled regardless of how the indexing position was reached.

2.1.3 Traversing of indexing axes by PLC

Traversal by PLC

Indexing axes can also be traversed from the PLC user program. There are various methods:

- With concurrent positioning axes
  In this case, the indexing position to be approached can be specified by the PLC.
  References: /FB/, P2 "Positioning Axes"
- With asynchronous subprograms (ASUP)
  References: /FB/, K1 "Mode Groups, Channels, Program Operation

2.2 Parameterization of indexing axes

Definition of the indexing axis

An axis (linear or rotary axis) can be defined as an indexing axis with the aid of the axial machine data INDEX_AX_ASSIGN_POS_TAB. The number of the indexing position table (1 or 2) must be entered in the machine data.

Several axes can be assigned to an indexing position table on the condition that they are all of the same type (linear axis, rotary axis, modulo 360° function). If they are not, alarm 4080 “Incorrect configuration of indexing axis in MD [name]” is output during power-up.

Indexing position tables

The axis positions (in mm or degrees) assigned to the indexes must be stored for each indexing axis in the form of a table in machine data. Any value can be entered for the distance between the individual indexing positions.
The following should be noted when entering the indexing positions:

Number of tables

Up to 2 indexing position tables are permitted:

MD 10910: INDEX_AX_POS_TAB_1 [n]
MD 10930: INDEX_AX_POS_TAB_2 [n]

Number of entries for each table

Up to 60 positions can be entered in each indexing position table [n = 0 ... 59].

The actual number of entries used must be defined with machine data

MD 10900: INDEX_AX_LENGTH_POS_TAB_1 bzw.
MD 10920: INDEX_AX_LENGTH_POS_TAB_2

for table 1 and/or 2.

All positions entered in the table which are higher than the number defined in the machine data are inactive.

Inch/metric switchover

Up to SW 5, when inputting the rotary axis indexing position on an inch machine (MD 10240: SCALING_SYSTEM_IS_METRIC=0), the value still has to be divided by 25.4 despite the input in degrees.

From SW 5 and MD 10260: CONVERT_SCALING_SYSTEM=1 (see /G2/) the indexing positions no longer refer to the basic system that is set but to the measuring system set in MD 10270: POS_TAB_SCALING_SYSTEM.

MD 10270: POS_TAB_SCALING_SYSTEM=0: metric
MD 10270: POS_TAB_SCALING_SYSTEM=1: inch

Note

MD 10270 defines the measuring system for position specifications for the following machine data:
MD 10900: INDEX_AX_POS_TAB_1
MD 10920: INDEX_AX_POS_TAB_2

MD 10270 also affects SD 41500 to SD 41507 (see /N3/).

Entry format

- The indexing positions should be entered in the table in ascending order (starting with the negative to the positive traversing range) with no gaps between the entries. Consecutive position values cannot be identical.
- The axis positions should be entered in the standard coordinate system.

If the indexing axis is defined as a rotary axis with modulo 360º (MD: IS_ROT_AX = “1” and MD: ROT_IS_MODULO = “1”), the following points should also be observed with respect to indexing positions:

- Indexing positions may be programmed in the range from $0^o \leqq$ Pos < 360° . Positions outside this range generate alarm 4080 on power-up.
- Since the indexing axis is defined as a continuously rotating rotary axis, indexing position 1 is approached after the highest valid indexing position in the table has been reached and the axis continues to traverse in the positive direction with INC. Similarly, indexing position 1 is followed by the highest valid indexing position in the negative direction with INC.

2.3 Programming of indexing axes

Note

Please refer to the following documentation for guidelines on the programming of indexing axes.
References: /PA1/, Programming Guide

Coded position

To allow indexing axes to be positioned from the NC part program, special instructions (so-called **C**oded positions) are provided with which the indexing numbers (e.g. location number) are programmed rather than axis positions in mm or degrees. The following coded position instructions are possible, depending on whether the indexing axis is defined as a linear or rotary axis:

Indexing axis is a linear axis: CAC(i), CIC(i)
Indexing axis is a rotary axis: CAC(i), CIC(i), CACP(i), CACN(i), CDC(i)

i = coded position (indexing position)
Value range of i: 0 ... 59; integer; with the exception of CIC positive only

Absolute
POS[B]=CAC(20)

Indexing axis B approaches **c**oded position (index) 20 in absolute mode. The direction of travering depends on the current actual position.

Absolute in positive direction
POS[B]=CACP(10)

Indexing axis B approaches **c**oded position (index) 10 in absolute mode in the positive direction of rotation (only possible with rotary axes).

Absolute in negative direction
POS[B]=CACN(10)

Indexing axis B approaches **c**oded position (index) 10 in absolute mode in the negative direction of rotation (only possible with rotary axes).

Direct absolute
POS[B]=CDC(50)

Indexing axis B approaches indexing position 50 directly along the shortest path (only possible with rotary axes).

Incremental
POS[B]=CIC(–4)

Indexing axis B traverses incrementally by 4 indexing positions in a negative direction from its current position.

POS[B]=CIC(35)

Indexing axis B traverses incrementally from the current indexing position across 35 indexing positions in the positive direction.
The leading sign defines the direction of approach.

Note

On modulo rotary axes, the indexing positions are divided in factors of 360 degrees and are approached directly.

Between indexing positions

If an indexing axis is located between 2 indexing positions in automatic mode, the program command

POS[B]=CIC(1) causes the next higher indexing position to be approached. Similarly, the program instruction

POS[B]=CIC(–1) causes the next lower indexing position to be approached. With

POS[B]=CIC(0) the indexing axis does not traverse.

Alarms

If an indexing position is programmed outside the valid range of the indexing position table, alarm 17510 “Impermissible index for indexing axis” is output.
When an indexing position is programmed for an axis, alarm 17500 “Axis is not an indexing axis” if an indexing position table is not assigned to this axis (MD: INDEX_AX_ASSIGN_POS_TAB (axis is an indexing axis)).

FRAMES

Since the control interprets the positions stored in the indexing position table as programmed positions in mm, inches or degrees, FRAMES are not disabled with indexing axes.
FRAMES are not generally required with indexing axes, depending on the application. It is therefore in most cases advisable to suppress FRAMES and zero offsets in the part program for indexing axes.

2.4 Extensions in SW 4.3 and higher, (840D), SW 2.3 (810D)

General information

The extensions include:

- Any number of equidistant index intervals
- Modified action of MD for indexing axes

2.4.1 Equidistant index intervals

Equidistant index intervals can be used for:

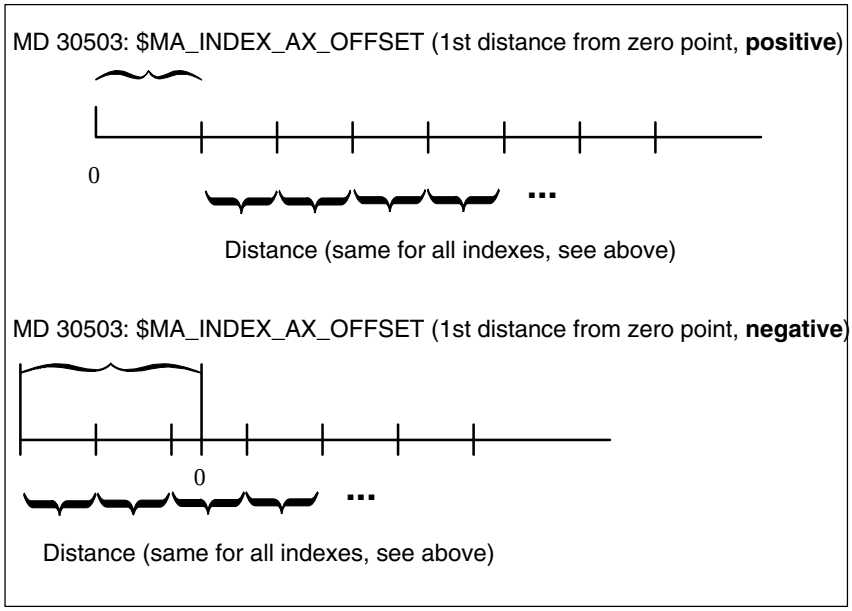
- Linear axes
- Modulo rotary axes
- Rotary axes

Distance between indexes

The index distance is determined as follows for equidistant index intervals:

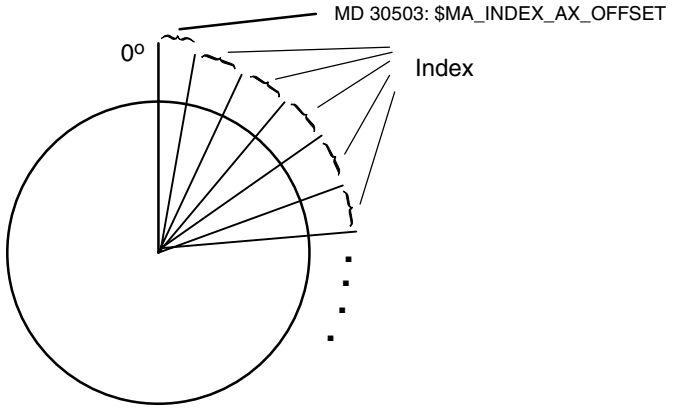
$$\text{Distance} = \frac{\text{Numerator (MD 30501: \$MA\_INDEX\_AX\_NUMERATOR)}}{\text{Denominator (MD 30502: \$MA\_INDEX\_AX\_DENOMINATOR)}}$$

Linear axis



Modulo rotary axis

$$\text{Index} = \frac{\text{Numerator (MD 30330: \$MA\_MODULO\_RANGE)}}{\text{Denominator (MD 30502: \$MA\_INDEX\_AX\_DENOMINATOR)}}$$



$$\text{Index} = \frac{\text{Size of modulo range (MD 30330: \$MA\_MODULO\_RANGE)}}{\text{Number of indexes in modulo range (MD 30502: \$MA\_INDEX\_AX\_DENOMINATOR)}}$$

Activation

The functions with equidistant indexing for linear axes and rotary axes or modulo rotary axes are activated by specifying “table number” 3 im MD 30500: $MA_INDEX_AX_ASSIGN_POS_TAB[axis].

Hirth tooth system

Introduction

With Hirth tooth systems, positions of rotation on a rotary axis are usually interlocked using a latch or other toothed wheel via a linear axis. The interlock should only be activated when an indexing position has been reached precisely. The distance between the indexing positions is the same (equidistant) across the entire circumference.

Preconditions

The rotary axis must be an indexing axis. The axis must be referenced. See
References: /R1/, Reference Point Approach

Activation

MD 30505: $MA_HIRTH_IS_ACTIVE must be set to 1.
MD 30500: $MA_INDEX_AX_ASSIGN_POS_TAB must be set to 3 (equidistant indexes).

Effect

- The rotary axis can only approach indexing positions in all modes and operating states.
- In JOG mode, the axis can be traversed
  conventionally or
  incrementally.
  Precondition: The axis is referenced.
- Jogging with the handwheel is not possible. See
  References: /H1/, Handwheel Travel
- Only “coded positions” can be approached in AUTO, MDA or via ASUPs
- The PLC can only move the axis to indexing positions. An alarm is output on an attempt to approach any other position.

Response of the Hirth axes in particular situations

STOP/RESET

On NC STOP and RESET during a traversing movement, the next indexing position is approached before the command is activated.

EMERGENCY STOP

After an EMERGENCY STOP, the PLC or the operator must move the indexing axis back to an indexing position in JOG mode before the longitudinal axis can be moved in/down.

Override = 0 or “Stop axis” signal

If the axis has already moved away from the previous indexing position when these events occur, the control moves the axis to the next possible indexing position before the response is initiated.

Delete distance-to-go

After traversing to the next possible indexing position, the movement is aborted at this position.

Command axes

See **References:** /FBSY/, Synchronized Actions
If MOV = 0 is specified for a moving command axis, the axis continues traversing to the next possible indexing position before stopping.

Move command

MOV = 1 Works on indexing axes with and without Hirth tooth system. Move = 0 works the same with both, the next position is approached.

DELDTG command

On indexing axes without Hirth tooth system:Axis stops immediately.
On indexing axes with Hirth tooth system:Axis approaches next position.

Restrictions

Transformations

The axis for which the Hirth tooth system is defined cannot take part in kinematic transformations.

PRESET

The axis for which the Hirth tooth system is defined cannot be set to a new value with PRESET.

Revolutional feedrate

The axis for which the Hirth tooth system is defined cannot be traversed with revolutional feedrate.

Path/velocity overlay

The axis for which the Hirth tooth system is defined cannot be used with path or velocity overlay.

Frames, ext. ZO, DRF

The axis for which the Hirth tooth system is defined does not support frames or interpolation compensation such as external zero offsets, DRF, etc.

Couplings

The axis for which the Hirth tooth system is defined cannot be a

- following axis with master value coupling
- coupled-motion axis
- gantry following axis.

See:
References: /M3/, Coupled Motion

2.4.2 Modified activation of machine data

In SW 4.3 and higher, only a RESET is required in order to activate the MD below after new values have been assigned to the MDs.
(POWER ON was previously required).

MD 10900: $MN_INDEX_AX_LENGTH_POS_TAB_1

MD 10920: $MN_INDEX_AX_LENGTH_POS_TAB_2

MD 10910: $MN_INDEX_AX_POS_TAB_1

MD 10930: $MN_INDEX_AX_POS_TAB_2

MD 30500: $MA_INDEX_AX_ASSIGN_POS_TAB

You will find a complete list of MDs for indexing axes in Chapter 4.

2.4.3 Examples of equidistant indexes

Modulo rotary axis

$MA__INDEX_AX_DENOMINATOR[AX4] =18
$MA_INDEX_AX_OFFSET[AX4]=5
$MA_INDEX_AX_ASSIGN_POS_TAB[AX4] = 3
$MA_IS_ROT_AX[AX4] = TRUE
$MA_ROT_IS_MODULO[AX4] = TRUE

With the machine data above, axis 4 is defined as a modulo rotary axis and an indexing axis with equidistant positions every 20 degrees starting at 5 degrees. This results in the following indexing positions: 5, 25, 45, 65, 85, 105, 125, 145, 165, 185, 205, 225, 245, 265, 285, 305, 325 and 245 degrees.

Note

The $MA__INDEX_AX_DENOMINATOR[AX4] =18 assignment produces a 20° division because the default for $MA_MODULO_RANGE ist 360.

Rotary axis

$MA_INDEX_AX__NUMERATOR[AX4] = 360
$MA__INDEX_AX_DENOMINATOR[AX4] =18
$MA_INDEX_AX_OFFSET[AX4]=100
$MA_INDEX_AX_ASSIGN_POS_TAB[AX4] = 3
$MA_IS_ROT_AX[AX4] = TRUE
$MA_POS_LIMIT_MINUS[AX1]=100
$MA_POS_LIMIT_PLUS[AX1]=260

With the machine data above, axis 4 is defined as a rotary axis and an indexing axis with equidistant positions every 20 degrees starting at 100 degrees. This results in the following indexing positions: 100, 120, 140 degrees, etc. Positions less than 100 degrees cannot be approached as indexing positions. It is therefore advisable to place the lower software limit switch in this case. The indexing positions continue until the software limit switch is reached (in this case 260 degrees). The rotary axis can therefore only traverse between 100 and 260 degrees.

Linear axis

$MA_INDEX_AX__NUMERATOR[AX1] = 10
$MA__INDEX_AX_DENOMINATOR[AX1] =1
$MA_INDEX_AX_OFFSET[AX1]=–200
$MA_INDEX_AX_ASSIGN_POS_TAB[AX1] = 3
$MA_IS_ROT_AX[AX1] = FALSE
$MA_POS_LIMIT_MINUS[AX1]=–200
$MA_POS_LIMIT_PLUS[AX1]=200

With the machine data above, axis 4 is defined as a linear axis and an indexing axis with equidistant positions every 10 mm starting at –200 mm. This results in the following indexing positions: –200, –190, –180 mm etc. These indexing positions continue until the software limit switch is reached (in this case 200 mm).

Hirth tooth system

$MA__INDEX_AX_DENOMINATOR[AX4] =360
$MA_INDEX_AX_OFFSET[AX4]=0
$MA_INDEX_AX_ASSIGN_POS_TAB[AX4] = 3
$MA_IS_ROT_AX[AX4] = TRUE
$MA_ROT_IS_MODULO[AX5] = TRUE
$MA_HIRTH_IS_ACTIVE[AX4] = TRUE

With the machine data above, axis 4 is defined as a modulo rotary axis and an indexing axis with Hirth tooth system and equidistant positions every 1 degree starting at 0 degrees.

2.5 Starting up indexing axes

Procedure

The procedure for starting up indexing axes is identical to normal NC axes (linear and rotary axes).

Rotary axis

If the indexing axis is defined as a rotary axis (MD: IS_ROT_AX = “1”) with modulo 360º conversion (MD: ROT_IS_MODULO = “1”), the indexing positions are traversed with modulo 360º. Only positions within the range from 0º to 359.999º can then be entered in the indexing position table. otherwise alarm 4080 “Incorrect configuration for indexing axis in MD [Name]” is output.
The position display can be set to modulo 360º with the MD: DISPLAY_IS_MODULO = “1”.

Special machine data

The following machine data, described in section 4, are also required to be defined:

General machine data

MD: INDEX_AX_LENGTH_POS_TAB_1	No. of indexing positions used in table 2
MD: INDEX_AX_LENGTH_POS_TAB_1	No. of indexing positions used in table 2
MD: INDEX_AX_POS_TAB_1 [n]	Indexing position table 1
MD: INDEX_AX_POS_TAB_2 [n]	Indexing position table 2

Axial machine data

MD: INDEX_AX_ASSIGN_POS_TAB	Axis is an indexing axis (assignment of indexing position table 1 or 2, or 3 for equidistant indexing)
with SW 4.3 and higher :	
MD: HIRTH_IS_ACTIVE	Axis has “Hirth tooth system” property,
MD: INDEX_AX_NUMERATOR	Numerator for equidistant indexing
MD: INDEX_AX_DENOMINATOR	Denominator for equidistant indexing
MD: INDEX_AX_OFFSET	Distance of the 1st indexing position from zero

Machine data examples

The assignment of the above machine data is described in the following paragraphs using two examples.

Example of indexing axis as rotary axis

Tool turret with 8 turret locations

The tool turret is defined as a continuously rotating rotary axis. The distances between the 8 turret locations are constant, the first location is at position 0° (see Fig. 2-1).

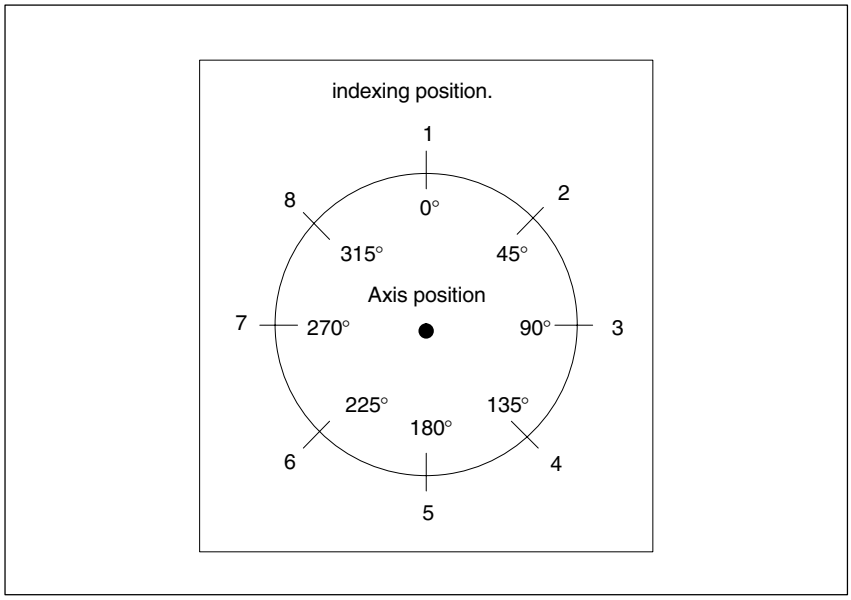


Fig. 2-1 Example: Tool turret with 8 locations

Indexing position table

The indexing positions for the tool turret are entered in table 1.

```
$MN_INDEX_AX_POS_TAB_1[0] = 0          ; 1st indexing position at 0°
$MN_INDEX_AX_POS_TAB_1[1] = 45         ; 2nd indexing position at 45°
$MN_INDEX_AX_POS_TAB_1[2] = 90         ; 3rd indexing position at 90°
$MN_INDEX_AX_POS_TAB_1[3] = 135        ; 4th indexing position at 135°
$MN_INDEX_AX_POS_TAB_1[4] = 180        ; 5th indexing position at 180°
$MN_INDEX_AX_POS_TAB_1[5] = 225        ; 6th indexing position at 225°
$MN_INDEX_AX_POS_TAB_1[6] = 270        ; 7th indexing position at 270°
$MN_INDEX_AX_POS_TAB_1[7] = 315        ; 8th indexing position at 315°
```

Other machine data

```
$MN_INDEX_AX_LENGTH_POS_TAB_1= 8       ; 8 indexing positions in
                                         table 1
$MA_INDEX_AX_ASSIGN_POS_TAB [AX5] = 1; Axis 5 is defined as an
                                         indexing axis, indexing
                                         positions in table 1
$MA_IS_ROT_AX [AX5] = 1                ; Axis 5 is a rotary axis
$MA_ ROT_IS_MODULO [AX5] = 1           ; Modulo conversion is active
```

Example of indexing axis as linear axis

Workholder with 10 locations (see Fig. 2-2).
The distances between the 10 locations vary; the first workholder location is at position –100 mm.

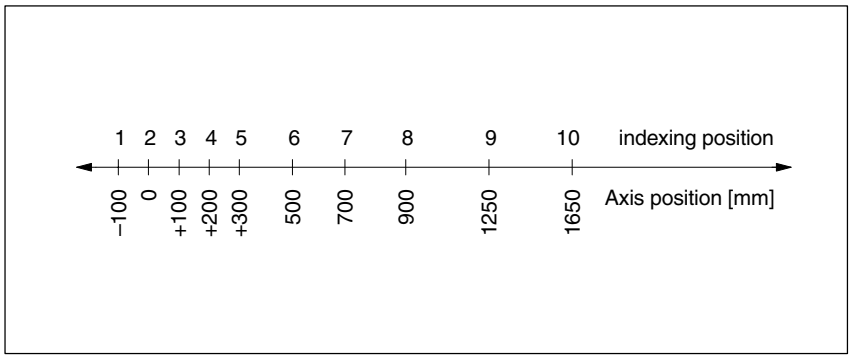


Fig. 2-2 Example: Workholding pallet as an indexing axis

Indexing position table

The indexing positions for the tool turret are entered in table 2.

```
$MN_INDEX_AX_POS_TAB_2[0] = –100      ; 1st indexing position at –100
$MN_INDEX_AX_POS_TAB_2[1] = 0         ; 2nd indexing position at 0
$MN_INDEX_AX_POS_TAB_2[2] = 100       ; 3rd indexing position at 100
$MN_INDEX_AX_POS_TAB_2[3] = 200       ; 4th indexing position at 200
$MN_INDEX_AX_POS_TAB_2[4] = 300       ; 5th indexing position at 300
$MN_INDEX_AX_POS_TAB_2[5] = 500       ; 6th indexing position at 500
$MN_INDEX_AX_POS_TAB_2[6] = 700       ; 7th indexing position at 700
$MN_INDEX_AX_POS_TAB_2[7] = 900       ; 8th indexing position at 900
$MN_INDEX_AX_POS_TAB_2[8] = 1250      ; 9th indexing position at 1250
$MN_INDEX_AX_POS_TAB_2[9] = 1650      ; 10th indexing position at 1650
```

Other machine data

```
$MN_INDEX_AX_LENGTH_POS_TAB_2=10      ; 10 indexing positions in
                                        table 2
$MA_INDEX_AX_ASSIGN_POS_TAB [AX6] = 2 ; Axis 6 is defined as an
                                        indexing axis
                                      ; Indexing positions in table 2
```

2.6 Special features of indexing axes

DRF

An additional incremental zero offset can also be generated for indexing axes in AUTOMATIC mode with the handwheel using the DRF function.

Software limit switches

After the indexing axis has been referenced, the software limit switches are active when the axis is traversed.
When traversing manually in continuous JOG or incremental JOG mode, the indexing axis stops at the last indexing position before the software limit switch.

Reference point approach

An indexing axis will approach indexing positions in JOG mode (continuous or incremental) only **after it has reached its reference point** (IS “Referenced/synchronized 1 or 2” (DB31–48, DBX60.4 or 5) = “1”). If the axis is not referenced (“referenced/synchronized 1 or 2” interface signal = “0”), the indexing positions are ignored when traversing manually.

Since the axis positions stored in the indexing position tables only correspond to the machine positions when the axis is referenced, an NC start must be disabled for as long as the indexing axis is not referenced.

Position display

Positions on indexing axes are displayed in the units of measurement normally used for the axes (mm, inches or degrees).

Abort through reset

Reset causes the traversing movement on an indexing axis to be aborted and the axis to be stopped. The indexing axis is no longer positioned on an indexing position.

Note

The response of the Hirth tooth system is described in Section 2.4.1.

■

Notes

Supplementary Conditions 3

There are no supplementary conditions stipulated for this Description of Functions.

Data Descriptions (MD, SD) 4

4.1 General machine data

10270 MD number	**POS_TAB_SCALING_SYSTEM** Measuring system of the position tables		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after RESET	Protection level: 2 / 7	Unit: –	
Data type: BYTE	Applies from SW version: 5		
Significance:	This machine data is for setting the measuring system for position specifications of indexing axis tables and switching points for software cams. • MD 10270=0: metric • MD 10270=1: inch MD 10270 defines the measuring system for position specifications for the following machine data: MD 10900: INDEX_AX_POS_TAB_1 MD 10920: INDEX_AX_POS_TAB_2 SD 41500: SW_CAM_MINUS_POS_TAB_1 SD 41501: SW_CAM_PLUS_POS_TAB_1 SD 41502: SW_CAM_MINUS_POS_TAB_2 SD 41503: SW_CAM_PLUS_POS_TAB_2 SD 41504: SW_CAM_MINUS_POS_TAB_3 SD 41505: SW_CAM_PLUS_POS_TAB_3 SD 41506: SW_CAM_MINUS_POS_TAB_4 SD 41507: SW_CAM_PLUS_POS_TAB_4 **Note: Only effective when MD 10260: CONVERT_SCALING_SYSTEM=1. (see /G2/)**		
Related to	see machine and setting data under significance; MD 10260: CONVERT_SCALING_SYSTEM		

10900 MD number	**INDEX_AX_LENGTH_POS_TAB_1** Number of indexing positions used in Table 1		
Default value: 0	Min. input limit: 0	Max. input limit: 60	
Modification effective after Power On or Reset with SW 4.3 and higher	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	The indexing position table is used to assign the axis positions in the valid unit of measurement (mm, inches or degrees) to the indexing positions [n] on the indexing axis. The number of indexing positions used in table 1 is defined by the MD: INDEX_AX_LENGTH_POS_TAB_1. These indexing positions must contain valid values in table 1. Any indexing positions in the table above the number specified in the machine data are ignored. Up to 60 indexing positions (0 to 59) can be entered in the table. Table length = 0 means that the table is not evaluated. If the length is not equal to 0, the table must be assigned to an axis with the MD: INDEX_AX_ASSIGN_POS_TAB. If the indexing axis is defined as a rotary axis (MD: IS_ROT_AX = “1”) with modulo 360° (MD: ROT_IS_MODULO = “1”), the machine data defines the last indexing position after which the indexing positions begin again at 1 with a further traversing movement in the positive direction.		
Application	Tool magazines (tool turrets, chain magazines)		
Special cases, errors, ...	Alarm 17090 “Value violates upper limit” if a value over 60 is entered in the MD: INDEX_AX_LENGTH_POS_TAB_1.		
Related to	MD: INDEX_AX_ASSIGN_POS_TAB MD: INDEX_AX_POS_TAB_1 MD: IS_ROT_AX MD: ROT_IS_MODULO	(axis is an indexing axis) (indexing position table 1) (rotary axis) (modulo conversion for rotary axis)	

10920 MD number	**INDEX_AX_LENGTH_POS_TAB_2** Number of indexing positions used in Table 2		
Default value: 0	Min. input limit: 0	Max. input limit: 60	
Modification effective after Power On or Reset with SW 4.3 and higher	Protection level: 2 / 7	Unit: –	
Data type: DWORD	Applies from SW version: 1.1		
Significance:	The indexing position table is used to assign the axis positions in the valid unit of measurement (mm, inches or degrees) to the indexing positions [n] on the indexing axis. The number of indexing positions used in table 2 is defined by the MD: INDEX_AX_LENGTH_POS_TAB_2. These indexing positions must contain valid values in table 2. Any indexing positions in the table above the number specified in the machine data are ignored. Up to 60 indexing positions (0 to 59) can be entered in the table. Table length = 0 means that the table is not evaluated. If the length is not equal to 0, the table must be assigned to an axis with the MD: INDEX_AX_ASSIGN_POS_TAB. If the indexing axis is defined as a rotary axis (MD: IS_ROT_AX = “1”) with modulo 360° (MD: ROT_IS_MODULO = “1”), the machine data defines the last indexing position after which the indexing positions begin again at 1 with a further traversing movement in the positive direction.		
MD irrelevant for	Tool magazines (tool turrets, chain magazines)		
Special cases, errors, ...	Alarm 17090 “Value violates upper limit” if a value over 60 is entered in the MD: INDEX_AX_LENGTH_POS_TAB_2.		
Related to	MD: INDEX_AX_ASSIGN_POS_TAB MD: INDEX_AX_POS_TAB_2 MD: IS_ROT_AX MD: ROT_IS_MODULO	(axis is an indexing axis) (indexing position table 2) (rotary axis) (modulo conversion for rotary axis)	

10910 MD number	**INDEX_AX_POS_TAB_1[n]** Indexing position table 1 [n]		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Modification effective after Power On or Reset with SW 4.3 and higher	Protection level: 2 / 7	Unit: mm or degrees	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	The indexing position table is used to assign the axis positions in the valid unit of measurement (mm, inches or degrees) to the indexing positions [n] on the indexing axis. [n] = Index for the entry of the indexing positions in the indexing position table. Range: 0 ≧ n ≦ 59, where 0 is the 1st indexing position and 59 corresponds to the 60th indexing position. Note: Programming with the absolute indexing position (e.g. CAC) starts with indexing position 1. This corresponds to the indexing position with index n = 0 in the table. The following should be noted when entering the indexing positions: • Up to 60 different indexing positions can be stored in the table. • The 1st entry in the table corresponds to indexing position 1; the nth entry corresponds to indexing position n. • The indexing positions should be entered in the table in ascending order (starting with the negative to the positive traversing range) with no gaps between the entries. Consecutive position values cannot be identical. • If the indexing axis is defined as a rotary axis (MD: IS_ROT_AX = “1”) with modulo 360º (MD: ROT_IS_MODULO = “1”) then the position values are limited to a range of 0º ≦ Pos. < 360º. The number of indexing positions used in the table is defined by the MD: INDEX_AX_LENGTH_POS_TAB_1. Entering the value 1 in the MD: INDEX_AX_ASSIGN_POS_TAB assigns indexing position table 1 to the current axis.		
Application	Tool magazines (tool turrets, chain magazines)		
Special cases, errors, ...	Alarm 17020 “Illegal array index” if over 60 positions are entered in the table.		
Related to	MD: INDEX_AX_ASSIGN_POS_TAB MD: INDEX_AX_LENGTH_POS_TAB_1 MD: IS_ROT_AX MD: ROT_IS_MODULO	(axis is an indexing axis) (no. of indexing positions used in table 2) (rotary axis) (modulo conversion for rotary axis)	

10930 MD number	**INDEX_AX_POS_TAB_2[n]** Indexing position table 2 [n]		
Default value: 0	Min. input limit: ***	Max. input limit: ***	
Modification effective after Power On or Reset with SW 4.3 and higher	Protection level: 2 / 7	Unit: mm or degrees	
Data type: DOUBLE	Applies from SW version: 1.1		
Significance:	The indexing position table is used to assign the axis positions in the valid unit of measurement (mm, inches or degrees) to the indexing positions [n] on the indexing axis. [n] = Index for the entry of the indexing positions in the indexing position table. Range: 0 ≧ n ≦ 59, where 0 is the 1st indexing position and 59 corresponds to the 60th indexing position. Note: Programming with the absolute indexing position (e.g. CAC) starts with indexing position 1. This corresponds to the indexing position with index n = 0 in the table. The following should be noted when entering the indexing positions: • Up to 60 different indexing positions can be stored in the table. • The 1st entry in the table corresponds to indexing position 1; the nth entry corresponds to indexing position n. • The indexing positions should be entered in the table in ascending order (starting with the negative to the positive traversing range) with no gaps between the entries. Consecutive position values cannot be identical. • If the indexing axis is defined as a rotary axis (MD: IS_ROT_AX = “1”) with modulo 360º (MD: ROT_IS_MODULO = “1”) then the position values are limited to a range of 0º ≦ Pos. < 360º. The number of indexing positions used in the table is defined by the MD: INDEX_AX_LENGTH_POS_TAB_2. Entering the value 1 in the MD: INDEX_AX_ASSIGN_POS_TAB assigns indexing position table 1 to the current axis.		
Application	Tool magazines (tool turrets, chain magazines)		
Special cases, errors, ...	Alarm 17020 “Illegal array index” if over 60 positions are entered in the table.		
Related to	MD: INDEX_AX_ASSIGN_POS_TAB MD: INDEX_AX_LENGTH_POS_TAB_1 MD: IS_ROT_AX MD: ROT_IS_MODULO	(axis is an indexing axis) (no. of indexing positions used in table 2) (rotary axis) (modulo conversion for rotary axis)	

4.2 Axis-specific machine data

30500 MD number	**INDEX_AX_ASSIGN_POS_TAB** Axis is indexing axis		
Default value: 0	Min. input limit: 0	Max. input limit: 3	
Modification effective after Power On or Reset with SW 4.3 and higher	Protection level: 2 / 7	Unit: –	
Data type: BYTE	Applies from SW version: 1.1		
Significance:	The axis is declared as an indexing axis by assignment of indexing position table 1 or 2. 0: The axis is not declared as an indexing axis 1: The axis is an indexing axis. The indexing positions are stored in table 1 (MD: INDEX_AX_POS_TAB_1). 2: The axis is an indexing axis. The indexing positions are stored in table 2 (MD: INDEX_AX_POS_TAB_2). 3: Equidistant indexing with SW 4.3 and higher (840D) and SW 2.3 and higher (810D) >3: Alarm 17090 “Value violates upper limit”		
Application	Tool magazines (tool turrets, chain magazines)		
Special cases, errors, ...	Several axes can be assigned to an indexing position table on condition that all the axes are of the same type (linear axis, rotary axis, modulo 360° function). If they are not, alarm 4000 is output during power-up. Alarm 17500 “Axis is not an indexing axis“ Alarm 17090 “Value violates upper limit”		
Related to	MD: INDEX_AX_POS_TAB1 (indexing position table 1) MD: INDEX_AX_LENGTH_POS_TAB_1 (no. of indexing positions used intable 2) MD: INDEX_AX_POS_TAB2 (indexing position table 2) MD: INDEX_AX_LENGTH_POS_TAB_1 (no. of indexing positions used inTable 2) For **equidistant indexes** with a value of 3: MD: INDEX_AX_NUMERATOR Numerator MD: INDEX_AX_DENOMINATOR Denominator MD: INDEX_AX_OFFSET First indexing position MD: HIRTH_IS_ACTIVE Hirth tooth system		

30501 MD number	**INDEX_AX_NUMERATOR** Numerator for indexing axes with equidistant positions		
Default value: 0	Min. input limit: >0	Max. input limit: ***	
Changes effective after RESET	Protection level: 2/7	Unit: mm/inches/degrees	
Data type: DOUBLE	Applies from SW version: 4.3		
Significance:	Defines the value of the numerator for calculating the distances between two indexing positions when the positions are equidistant. Modulo axes ignore this value and use $MA_MODULO_RANGE instead.		
MD irrelevant for	Non-equidistant indexes in accordance with tables		
Application	See 2.4		
Related to	MD 30502: INDEX_AX_DENOMINATOR, MD 30503: INDEX_AX_OFFSET; MD 30500: INDEX_AX_ASSIGN_POS_TAB		

30502 MD number	**INDEX_AX_DENOMINATOR** Denominator for indexing axes with equidistant positions		
Default value: 1	Min. input limit: 1	Max. input limit: ***	
Changes effective after RESET	Protection level: 2/7	Unit: –	
Data type: DWORD	Applies from SW version: 4.3		
Significance:	Defines the value of the denominator for calculating the distances between two indexing positions when the positions are equidistant. For modulo axes it therefore specifies the number of indexing positions.		
MD irrelevant for	Non-equidistant indexes in accordance with tables		
Application	See 2.4		
Related to	MD 30501: INDEX_AX_NUMERATOR, MD 30503: INDEX_AX_OFFSET; MD 30500: INDEX_AX_ASSIGN_POS_TAB		

30503 MD number	**INDEX_AX_OFFSET** First indexing position for indexing axes with equidistant positions		
Default value: 0.0	Min. input limit: ***	Max. input limit: ***	
Changes effective after RESET	Protection level: 2/7	Unit: mm/inches/degrees	
Data type: DOUBLE	Applies from SW version: 4.3		
Significance:	Defines the position of the first indexing position from zero for an indexing axis with equidistant positions.		
MD irrelevant for	Non-equidistant indexes in accordance with tables		
Application	See 2.4		
Related to	MD 30501, 30502, 30500		

30505 MD number	**HIRTH_IS_ACTIVE** Hirth tooth system is active		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after RESET	Protection level: 2/7	Unit: –	
Data type: BOOLEAN	Applies from SW version: 4.3		
Significance:	Hirth tooth system is active when a value of 1 is set.		
MD irrelevant for	Non-equidistant indexes in accordance with tables		
Application	See 2.4		
Related to	MD 30500, 30501, 30502, 30503		

4.3 System variables

Name	$AA_ACT_INDEX_AX_POS_NO[axis]			
Meaning	Number of last indexing position reached or overtraveled			
Data type	INTEGER			
Value range	0: Not an indexing axis, and so no indexing position available >0: Number of the indexing position			
Index	Axis	Value range	–	
Accesses	Read part program		Read synchronized action	
Impl. prepro-cessing stop	x			

Name	$AA_PROG_INDEX_AX_POS_NO[axis]			
Meaning	Number of programmed indexing position			
Value range	0: Not an indexing axis, and so no indexing position available or indexing axis not currently moving to an indexing position >0: Number of the indexing position			
Index	Axis	Value range	–	
Accesses	Read part program		Read synchronized action	
Impl. prepro-cessing stop	x			

■

Notes

Signal Descriptions

5

5.1 Axis-specific signals

DB31, ... **DBX76.6** Data block	**Indexing axis in position** Signal(s) from axis/spindle (NCK —> PLC)		
Edge evaluation: no	Signal(s) updated: cyclically	Signal(s) valid from SW vers.: 1.1	
Signal state 1 or signal transition 0 —> 1	The signal is influenced according to the “Exact stop fine”. When “Exact stop fine” is achieved, the signal is set. When exiting “Exact stop fine”, the signal is reset again. • The indexing axis is positioned on an indexing position. The indexing axis has been positioned by coded position instructions. SW 4.3 and higher (840D), SW 2.3 and higher (810D) If the “Exact stop fine” window is reached and the indexing axis is positioned on an indexing position, the signal is enabled regardless of how the indexing position was reached.		
Signal state 0 or signal transition 1 —> 0	• The axis is not defined as an indexing axis • Indexing axis is traversing (IS “Travel command +/–” (DB31, ... DBX64.7/64.6) is active) • The indexing axis is located at a position which does not correspond to an indexing position. Examples: In JOG mode after abortion of travel movement, e.g. with RESET In AUTOMATIC: Indexing axis has, for example, approached a selected position controlled by an AC or DC instruction. • The indexing axis has not been positioned with instructions for coded positions (CAC, CACP, CACN, CDC, CIC) in automatic mode. • The “Servo enable” signal for the indexing axis has been cancelled (IS “Servo enable” DB31, ... DBX2.1).		
Signal irrelevant for.....	Axes that are not defined as indexing axes (MD 30500: INDEX_AX_ASSIGN_POS_TAB = “0”)		
Application	Tool magazine: the activation of a gripper for removing a tool from a magazine is triggered when the indexing axis is in position (“indexing axis in position” = 1). This must be programmed in the PLC user program.		
Special cases, errors, ...	Note: The axis positions entered in the indexing position table for the individual divisions can be changed through zero offsets (including DRF). The “indexing axis in position” interface signal is then set to 1 when the actual position of the indexing axis matches the value entered in the index table plus the offset. If a DRF is applied to an indexing axis in AUTOMATIC mode, then interface signal “Indexing axis in position” remains active even though the axis is no longer at an indexing position. For exceptions, see 2.4 Hirth tooth system.		
Related to	MD 30500: INDEX_AX_ASSIGN_POS_TAB (axis is an indexing axis)		

■

Notes

6 Example

For an example, please see Sections 2.4, 2.5

7 Data Fields, Lists

7.1 Interface signals

DB number	Bit, byte	Name	Reference
Axis/spindle-specific			
31–61	76.6	Indexing axis in position	
31–61	60.4, 60.5	Referenced/synchronized 1, Referenced/synchronized 2	R1

7.2 Machine data

Number	Identifier	Name	Reference
General ($MN_ ...)			
10260	CONVERT_SCALING_SYSTEM	Basic system switchover active	G2
10270	POS_TAB_SCALING_SYSTEM	Measuring system of position tables	
10900	INDEX_AX_LENGTH_POS_TAB_1	No. of indexing positions used in table 1	
10920	INDEX_AX_LENGTH_POS_TAB_2	No. of indexing positions used in table 2	
10910	INDEX_AX_POS_TAB_1[n]	Indexing position table 1	
10930	INDEX_AX_POS_TAB_2[n]	Indexing position table 2	
Axis/spindle-specific ($MA_ ...)			
30300	IS_ROT_AX	Rotary axis	R2
30310	ROT_IS_MODULO	Modulo conversion for rotary axis	R2
30320	DISPLAY_IS_MODULO	Position display is modulo 360°	R2
30500	INDEX_AX_ASSIGN_POS_TAB	Axis is an indexing axis	

Number	Identifier	Name	Reference
30501	INDEX_AX_NUMERATOR	Numerator for indexing axes with equidistant positions	
30502	INDEX_AX_DENOMINATOR	Denominator for indexing axes with equidistant positions	
30503	INDEX_AX_OFFSET	First indexing position for indexing axes with equidistant positions	
30505	HIRTH_IS_ACTIVE	Hirth tooth system is active	

7.3 Setting data

Number	Identifier	Name	Reference
General ($MN_ ...)			
41050	JOG_CONT_MODE_LEVELTRIGGRD	JOG continuous mode	H1

7.4 System variables

The following system variables exist in SW 4.3 and higher:

Identifier	Name, meaning	Reference
$AA_ACT_INDEX_AX_POS_NO[axis]	Number of last indexing position reached or overtraveled	PGA
$AA_PROG_INDEX_AX_POS_NO[axis]	Number of programmed indexing position	PGA

7.5 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

■

SINUMERIK 840D/FM-NC Description of Functions, Extension Functions (FB2)

Tool Change (W3)

■

Notes

Brief Description 1

CNC-controlled machine tools are equipped with tool magazines and automatic tool change facility for the complete machining of workpieces.

Procedure

The procedure for changing tools comprises three steps:

- Movement of the tool carrier from the machining position to the tool change position
- Tool change
- Movement of the tool carrier from the tool change position to the new machining position.

Actuation

The tool change can be actuated using a

- T function or an
- M command (preferably M06).

There are two options for tool change:

Immediate change with T number or preparation with T number:

1. Immediate change
   - The T function loads the new tool immediately
   - Typical application: turning machines with tool turrets.
2. Preparation
   - The new tool is prepared for the change on execution of the T function.
   - The M function is used to remove the old tool from the spindle and load the new tool.
   - The M command for tool change can be defined in a machine data.
   - Typical application: milling machines with a tool magazine, in order to bring the new tool into the tool change position without interrupting the machining process.

Tool change point

The selection of the tool change point has a significant effect on the cut-to-cut time. The tool change point is chosen according to the machine tool concept and, in certain cases, according to the current machining task.

The **fixed point approach** function (G75) can be used to approach fixed positions on a machine axis. These positions are stored in a machine data. Several tool change points can be defined and actuated.

The tool change requires, amongst other things, a tool management system which ensures that the tool to be loaded is available at the tool change position at the right time.

■

Notes

Detailed Description 2

2.1 Overview of tool change function

CNC-controlled machine tools are equipped with tool magazines and automatic tool change facility for the complete machining of workpieces.

Tool changing equipment

Tool magazines and tool changing equipment are selected according to the machine type.

Turning machines:	Tool turret (disc, flat or inclined tool turret); no special equipment required: Tool is changed through rotation of revolver.
Milling machines:	Magazines (chain, plate, disc, cassette) with gripper/double gripper for changing.

Since the tool change interrupts machining, the idle times must be minimized.

Tool change times

Tool change times are largely determined by the construction of the machine tool.

Typical tool change times are

0.1 to 0.2 s	for rotation of a tool turret
0.3 to 2 s	for a tool change with a gripper when the tool is prepared.

Cut-to-cut time

The cut-to-cut time is the period that elapses when a tool is changed between retraction from the interruption point on the contour (from cut) and repositioning on the interruption point (return to cut) with the new tool when the spindle is rotating.

Typical cut-to-cut times are:

0.3 to 1 s	for a turning machine with tool turret
0.5 to 5 s	for a milling machine with a tool changer.

Requirements

A tool change operation must fulfill the following requirements:

- Short idle times
- Fast search, preparation and return of the tool during machining
- Simple programming of the tool change cycle

- Automatic operation of the required axis and gripper movements
- Easy fault recovery.

2.2 Sequence of operations

Tool change sequence

The tool change operation from cut-to-cut is executed in three steps:

- Movement of the tool carrier from the machining position to the tool change position
- Tool change
- Movement of the tool carrier from the tool change position to the new machining position.

The tool change position depends on the machine concept and is described in more detail in Section 2.4.

Control of spindle

The method by which the spindle is controlled during a tool change also depends on the machine design. The various options include systems where

- the spindle continues rotating,
- the spindle is brought to a halt, or
- the spindle is positioned.

2.3 Control

Actuation

The tool change can be actuated using a

- T function or
- M command (preferably M06).

The selection is made in MD: TOOL_CHANGE_MODE, as follows:

TOOL_CHANGE_MODE = 0

- The new tool is loaded immediately on execution of the T function.
- Typical application: turning machines with tool turrets.

TOOL_CHANGE_MODE = 1

- The new tool is prepared for the change on execution of the T function.
- The M function is used to remove the old tool from the spindle and load the new tool.
- The M command for the tool change is defined in the MD: TOOL_CHANGE_M_CODE. The default setting is 6 for compatibility with DIN 66025.

– Typical application: milling machines with a tool magazine, in order to bring the new tool into the tool change position without interrupting the machining process.

Note: If the tool offset number is supplied from the PLC or an MMC tool manager, a “STOPRE” block search stop must be inserted at a suitable point. STOPRE should be avoided, however, when tool radius compensation (G41/G42) or SPLINE interpolation are active, since several blocks are required here in advance for the path calculation.

Further information on M functions which apply to the M06 tool change, such as

– Extended address
– Output time to PLC
– Auxiliary function groups
– Block search
– Overstore

is given in the following documentation:
References: /FB/, S5, “Synchronized Actions”

2.4 Tool change point

Tool change point

The selection of the tool change point has a significant effect on the cut-to-cut time. The tool change point is chosen according to the machine tool concept and, in certain cases, according to the current machining task.

The **fixed point approach** function (G75) can be used to approach fixed positions on a machine axis:

N20 G75 FP=2 X1=0 Y1=0 Z1=0 LF

Fixed points

Two fixed positions are stored for each machine axis in MD: FIX_POINT_POS[N]. They are addressed with FP=1 or FP=2. If no value is defined, then FP=1.

Each machine axis which is required to travel to one of these points has to be specified with its machine name and a dummy position (which is not evaluated).

The positions stored in the MD are approached with rapid traverse G0.

In a block with G75, the spindle can be positioned using SPOS and SPOSA.

■

Notes

Supplementary Conditions 3

The tool change requires, amongst other things, a tool management system which ensures that the tool to be loaded is available at the tool change position at the right time.

Data Descriptions (MD, SD) 4

4.1 Machine data

The machine data required for the tool change are documented in the following sections:

MD number	Identifier	Description of Functions
22500	TOOL_CHANGE_MODE	W1
22600	TOOL_CHANGE_M_CODE	W1
30600	FIX_POINT_POS[n]	K1
22200	AUXFU_M_SYNC_TYPE	H2
22220	AUXFU_T_SYNC_TYPE	H2

Signal Descriptions 5

No separate signals exist for this Description of Functions.

Notes

Example 6

The following example shows a typical cut-to-cut sequence of operations for a tool change with a tool changer and a fixed absolute tool change point on a milling machine.

Machining program

```
N 970     G0   X=   Y=   Z=   LF              ; Retract from contour
N 980     T1   LF                              ; Tool selection
N 990     W_WECHSEL   LF                       ; Subprogram call with no
                                                 Parameter
N 1000    G90   G0   X=   Y=   Z=   M3   S1000   LF; Machining resumed
```

Subprogram for tool change

```
PROC   W_WECHSEL   LF
N 10      SPOSA=   S0   LF                     ; Spindle positioning
N 20      G75   FP=2   X1=0   Y1=0   Z1=0      ; Approach tool change point
                                                 (see Section 2.4)
N 30      M06   LF                             ; Change tools
N 40      M17   LF
```

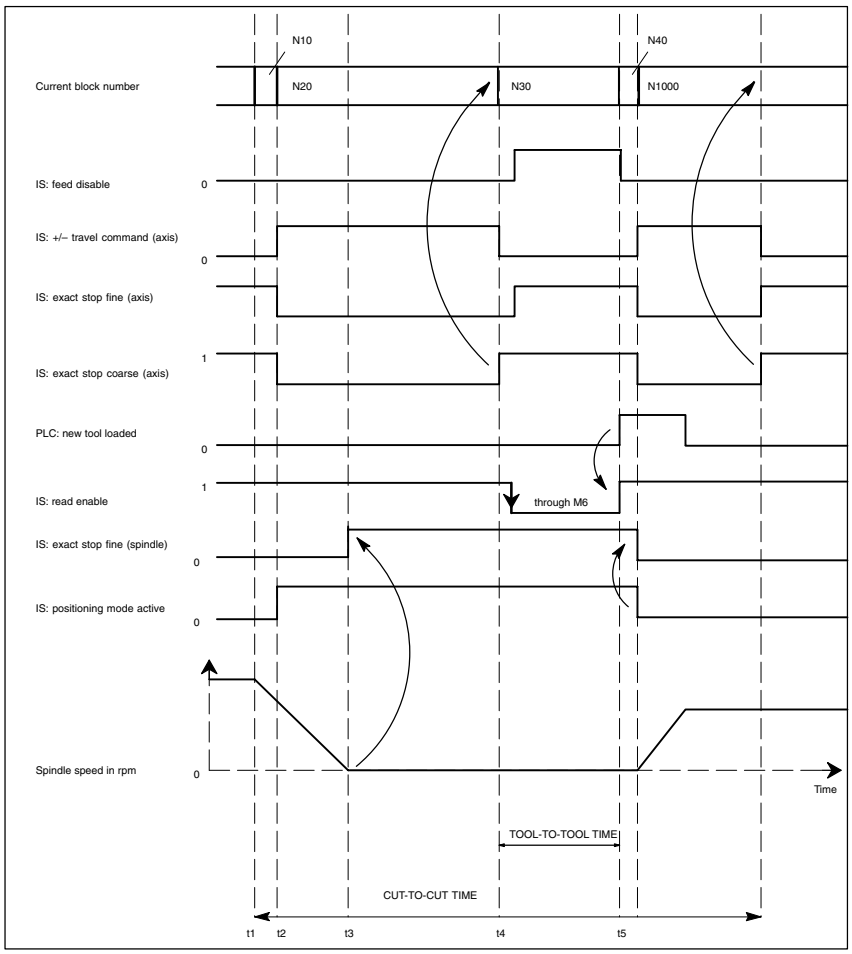


Fig. 6-1 Chronological sequence of tool change

t1 Axes stationary
Spindle rotating
Start of tool change cycle in N 10

t2 Axes travel to tool change point with G75 in N 20

t3 Spindle reaches position programmed in block N 10

t4 Axes reach fine stop coarse from N 20; N 30 starts here:
N 30:

M06 removes the previous tool from the spindle and loads and secures the new tool.

t5 Tool changer swivels back to original position.

Following this, in N 1000 of the calling main program, the

- new tool offset can be selected,
- the axes can be returned to the contour, or
- the spindle can be accelerated.

■

Data Fields, Lists

7

7.1 Interface signals

DB number	Bit, byte	Name	Reference
Channel-specific			
21–28	194.6	M function M06	

7.2 Machine data

Number	Identifier	Name	Reference
General ($MN_ ...)			
18082	MM_NUM_TOOL	Number of tools	S7
Channel-specific ($MC_ ...)			
22200	AUXFU_M_SYNC_TYPE	Output time for M functions	H2
22220	AUXFU_T_SYNC_TYPE	Output time for T functions	H2
22550	TOOL_CHANGE_MODE	New tool offset with M function	
22560	TOOL_CHANGE_M_CODE	M function for tool change	
Axis-specific ($MC_ ...)			
30600	FIX_POINT_POS[n]	Fixed point positions of the machine axes with G75	

7.3 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help in systems with MMC 101/102/103.

■

Notes

SINUMERIK 840D/810D/FM-NC Description of Functions, Extension Functions (FB2)

Tool Compensation and Monitoring in Grinding (W4)

■

Notes

Brief Description

1

Contents

This Description of Functions deals with the following subjects:

- Tool offset for grinding operations
- Online tool offsets (continuous dressing)
- Grinding-specific tool monitoring
- Constant grinding wheel peripheral speed (GWPS)

Note

This Description is based on information in
References: /FB/, W1, “Tool Offset”

For information about programming, mode of operation and handling, please refer to
References: : /PG/, Programming Guide Fundamentals

■

Notes

Detailed Description

2

2.1 Tool offset for grinding operations

2.1.1 Structure of tool data

Grinding tools

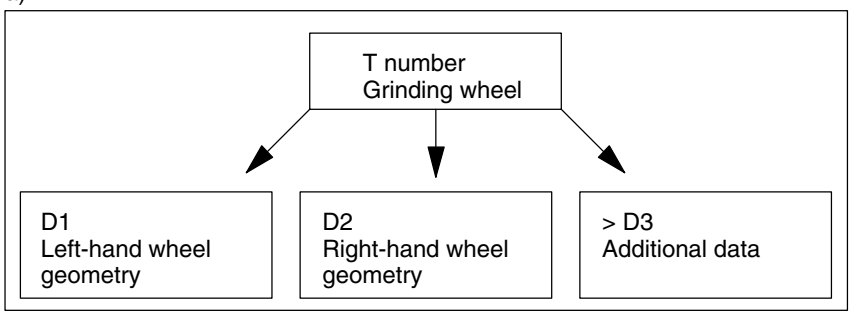
Grinding tools (see Sections 2.1.4 and 3) are tools of type 400 to 499.

Tool offset for grinding tools

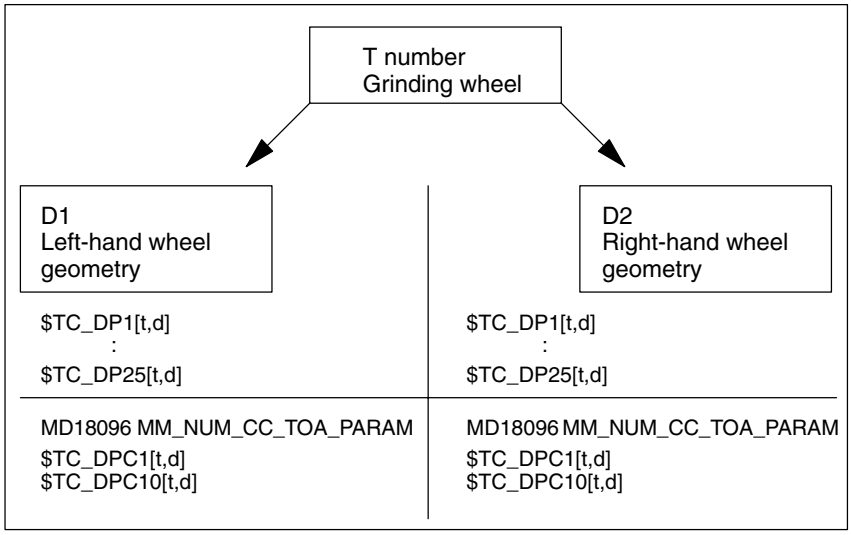
Apart from edge-specific data, data that are specific to the tool and dressing process are generally also programmed for grinding tools.
The data specific to a grinding wheel for the left-hand and right-hand wheel geometry can be stored in D1 or D2 under a T number.
If data are required for the dressing geometry, they can be stored, for example, starting at D3 of a T number or in additional edge-specific data (MD 18096 MM_NUM_CC_TOA_PARAM).

Examples:

a)



b)



All offsets belonging to a grinding wheel and dresser can be combined in the tool edges D1 and D2 for the grinding wheel and, for example, D3 and D4 for the dresser:

D1: Grinding wheel geometry left
D2: Grinding wheel geometry right
D3: Dresser geometry left
D4: Dresser geometry right

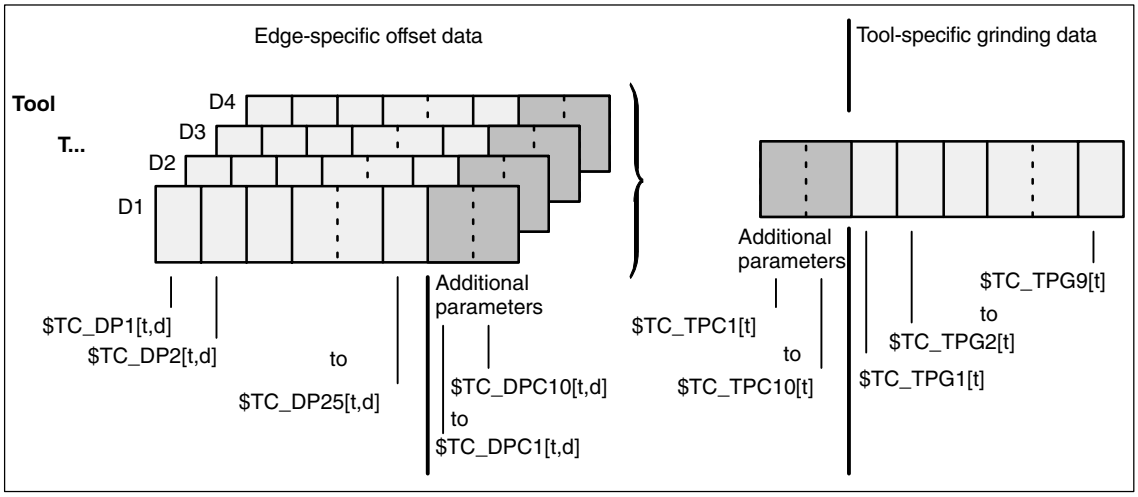


Fig. 2-1 Structure of tool offset data for grinding tools

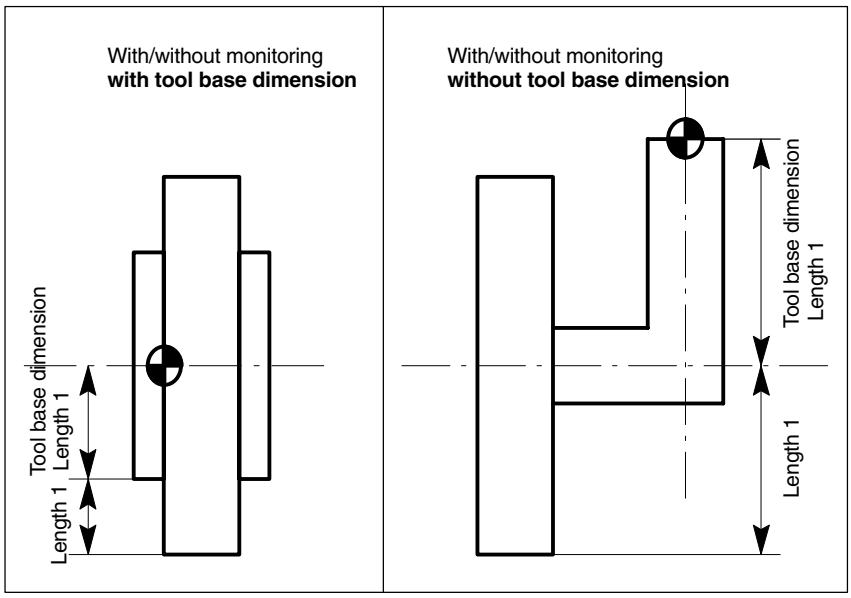
2.1.2 Edge-specific offset data

Tool parameters

The tool parameters for grinding tools have the same meaning as those for turning and milling tools.

Tool parameter	Meaning	Remark	Reserved for expansions
1	Tool type		
2	Tool point direction	For turning tools only	
Geometry - tool length compensation			
3	Length 1		
4	Length 2		
5	Length 3		
Geometry - tool radius compensation			
6	Radius 1		
7			Reserved
8			Reserved
9			Reserved
10			Reserved
11			Reserved
Wear - tool length compensation			
12	Length 1		
13	Length 2		
14	Length 3		
Wear - tool radius compensation			
15	Radius 1		
16			Reserved
17			Reserved
18			Reserved
19			Reserved
20			Reserved
Base dimension/adapter dimension - tool length compensation			
21	Basic length 1		
22	Basic length 2		
23	Basic length 3		
Technology			
24	Clearance angle	For turning tools only	
25			Reserved

Reserved ... means that this tool parameter of the 840D/810D and FM-NC is not used (reserved for expansions).



Note

The cutting edge data for D1 and D2 of a selected grinding tool can be chained, i.e. if a parameter in D1 or D2 is modified, then the same parameter in D1 or D2 is automatically overwritten with the new value (see tool-specific data $TC_TPC2).

Definition of additional parameters

Additional parameters can be defined via general MD18096: MM_NUM_CC_TOA_PARAM, regardless of the tool type.

Caution

Changes to the MD take effect after POWER ON and will lead to initialization of the memory (back data up beforehand if necessary!).

No automatic changeover between grinding wheel offset left and right takes place during contour grinding. This changeover must be programmed.

Tool types for grinding tools

The structure of tool types for grinding tools is as follows:

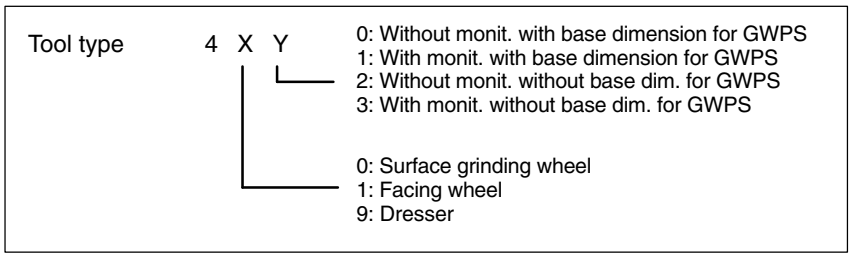


Fig. 2-2 Structure of tool type for grinding tools, see Fig. 2-1

Note

Channel-specific MD20350: TOOL_GRIND_AUTO_TMON can be set to define whether the monitoring function must already be active when tools **with monitoring** (i.e. uneven tool types) are selected.

Examples:
This structure can be used to create the following tool types:

- Type 400: Surface grinding wheel
- Type 401: Surface grinding wheel with monitoring and tool base dimension for GWPS
- Type 403: Surface grinding wheel with monitoring/without tool base dimension for GWPS
- Type 410: Facing wheel
- Type 411: Facing wheel with monitoring with base dimension for GWPS
- Type 413: Facing wheel with monitoring without base dimension for GWPS
- Type 490: Dresser

2.1.3 Tool-specific grinding data

Tool-specific grinding data are available once for every T number (type 400-499). They are automatically set up with every new grinding tool (type 400-499).

Note

Tool-specific grinding data have the same characteristics as a tool edge. This may need to be taken into account when the number of cutting edges is specified in MD18100: MM_NUM_CUTTING_EDGES_IN_TOA.

When all the cutting edges of a tool are deleted, the existing tool-specific grinding data are deleted at the same time.

Tool-specific grinding data

The parameters are assigned as follows:

Parameter	Meaning	Data type
$TC_TPG1	Spindle number	Integer
$TC_TPG2	Chaining specification	Integer
$TC_TPG3	Minimum grinding wheel radius	Real
$TC_TPG4	Minimum grinding wheel width	Real
$TC_TPG5	Current grinding wheel width	Real
$TC_TPG6	Maximum speed	Real
$TC_TPG7	Maximum peripheral speed	Real
$TC_TPG8	Angle of inclined grinding wheel	Real
$TC_TPG9	Parameter number for radius calculation	Integer
Additional parameters		
$TC_TPC1		Real
to		
$TC_TPC10		Real

Definition of additional parameters

Additional parameters can be defined in general MD: MM_NUM_CC_TDA_PARAM, regardless of the tool type concerned.

Caution

Changes to the MD take effect after POWER ON and will lead to initialization of the memory (back data up beforehand if necessary!).

Spindle number $TC_TPG1

Number of programmed spindle (e.g. grinding wheel peripheral speed) and spindle to be monitored (e.g. wheel radius and width)

Chaining rule $TC_TPG2

This parameter is set to define which tool parameters of tool edge 2 (D2) and tool edge 1 (D1) must be chained to one another. When the setpoint of a chained parameter is modified, the value of the parameter with which it is chained is modified automatically.

Tool parameter	Meaning	Bit in $TC_TPG2	Hex	Dec
$TC_DP1	Tool type	0	0001	1
$TC_DP2	Tool edge pos.	1	0002	2
Geometry - tool length compensation				
$TC_DP3	Length 1	2	0004	8
$TC_DP4	Length 2	3	0008	16

Tool parameter	Meaning	Bit in $TC_TPG2	Hex	Dec
$TC_DP5	Length 3	4	0010	32
$TC_DP6	Radius	5	0020	64
$TC_DP7	Reserved	6	0040	128
$TC_DP8		7	0080	256
$TC_DP9		8	0100	512
$TC_DP10		9	0200	1024
$TC_DP11	Reserved	10	0400	2048
Wear - tool length compensation				
$TC_DP12	Length 1	11	0800	4096
$TC_DP13	Length 2	12	1000	8192
$TC_DP14	Length 3	13	2000	16384
$TC_DP15	Radius	14	4000	32768
$TC_DP16	Reserved	15	8000	65536
$TC_DP17		16	10000	131072
$TC_DP18		17	20000	262144
$TC_DP19		18	40000	524288
$TC_DP20	Reserved	19	80000	1048576
Base dimension/adapter dimension - tool length compensation				
$TC_DP21	Basic length 1	20	100000	2097152
$TC_DP22	Basic length 2	21	200000	4194304
$TC_DP23	Basic length 3	22	400000	8388608
Technology				
$TC_DP24	Reserved	23	800000	16777216
$TC_DP25	Reserved	24	1000000	33554432

Example of parameter chain:

- Lengths 1, 2 and 3 of the geometry, the length wear and the tool base/adapter dimensions of lengths 1, 2 and 3 on a grinding tool (T1 in the example) must be automatically transferred.
- Furthermore, the same tool type applies to tool edges 1 and 2.

Tool type	$TC_DP1	Bit 0
Length 1	$TC_DP3	Bit 2
Length 2	$TC_DP4	Bit 3
Length 3	$TC_DP5	Bit 4
Wear		
Length 1	$TC_DP12	Bit 11
Length 2	$TC_DP13	Bit 12
Length 3	$TC_DP14	Bit 13
Tool base/adapter dimension		
Length 1	$TC_DP21	Bit 20
Length 2	$TC_DP22	Bit 21
Length 3	$TC_DP23	Bit 22

Parameter $TC_TPG2 must therefore be assigned as follows:

a) Binary

$TC_TPG2[1]= 'B111 0000 0011 1000 0001 1101'
(Bit 22 ... bit 0)

b) Hexadecimal

$TC_TPG2[1]= 'H70381D'

c) Decimal

$TC_TPG2[1]='D7354397'

Note

If the chaining specification is subsequently altered, the values of the two cutting edges are not automatically adjusted, but only after one parameter has been altered.

Min. wheel radius and width $TC_TPG3 $TC_TPG4

The limit values for the grinding wheel radius and width must be entered in this parameter. These parameter values are used to monitor the grinding wheel geometry.

Note

It must be noted that the minimum grinding wheel radius must be specified in the cartesian coordinate system for an inclined grinding wheel. A signal is output at the PLC interface if the grinding wheel width and radius drop below the minimum limits. The user can use these signals to define his error strategy.

Current width $TC_TPG5

The width of the grinding wheel measured, for example, after the dressing operation, is entered here.

Max. speed and grinding wheel peripheral speed $TC_TPG6 $TC_TPG7

The upper limit values for maximum speed and peripheral speed of the grinding wheel must be entered in this parameter.

Precondition: A spindle has been declared.

Angle of inclined wheel $TC_TPG8

This parameter specifies the angle of inclination of an inclined wheel in the current plane. It is evaluated for GWPS.

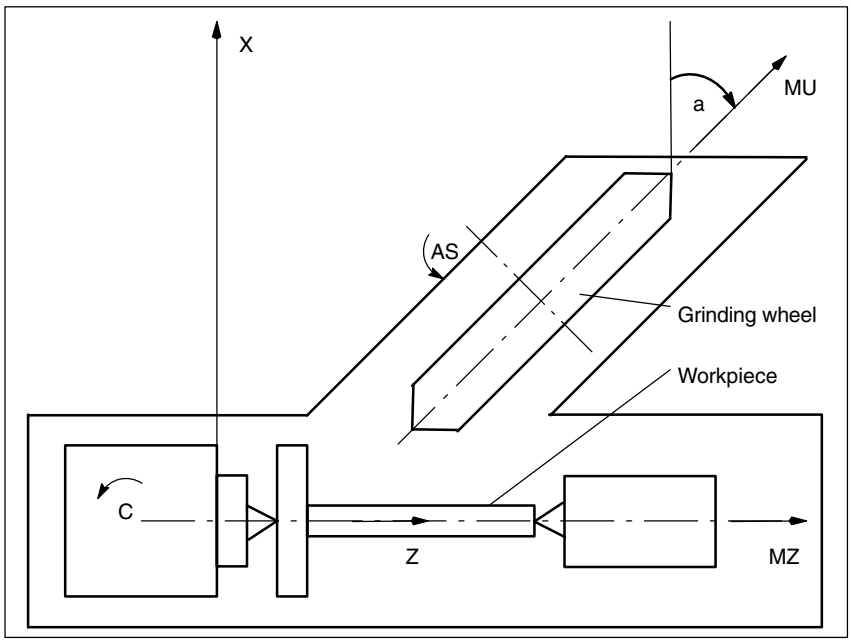


Fig. 2-3 Machine with inclined infeed axis

Note

The tool lengths are not automatically compensated when the angle is altered.

The angle must be within the –90 $\leq$ $TC_TPG8 < +90 range.

In the case of machines with inclined axes, the same angle must be set for the inclined axis and the inclined grinding wheel. With SW versions NCU 5.1.11/P5 and 4.4.35/P4 and ealier with GWPS for grinding wheel cutting rate and inclined axis, the input value in tool parameter $TC_TPG8 must be set in RAD rather than DEGREE. $TC_TPG8 in RAD = PI/180* angle.

Parameter number for radius calculation $TC_TPG9

This parameter can be set to define which offset value for GWPS, tool monitoring ($TC_TPG3) is the first cutting edge of a grinding tool (D1). The number of the grinding spindle can be entered for centerless grinding applications.
References: /FB/ S8, Constant Workpiece Speed for Centerless Grinding

$TC_TPG9 = 3	Length 1 (geometry + wear + base, depending on tool type)
$TC_TPG9 = 4	Length 2 (geometry + wear + base, depending on tool type)
$TC_TPG9 = 5	Length 3 (geometry + wear + base, depending on tool type)
$TC_TPG9 = 6	Radius

Access from part program

Parameters can be read and written from the part program.

Example	Programming
Read the current width of tool 2 and store in R10	R10 = $TC_TPG5 [2]
Write value 2000 to the maximum speed of tool 3	$TC_TPG6 [3] = 2000

$P_ATPG[m] for current tool

This system variable allows the tool-specific grinding data for the **current** tool to be accessed.

m: Parameter number (data type: Real)

Example:
Parameter 3 ($TPG3[< T No.>])

$P_ATPG[3]=R10

Note

- The monitoring data apply to both the left-hand and the right-hand cutting edge of the grinding wheel.
- The tool-specific grinding data take effect when the following are programmed: GWPSON (selection of constant wheel peripheral speed), TMON (selection of tool monitoring function) and CLGON (selection of constant workpiece speed for centerless grinding). To make a changed data effective, GWPSON, TMON or CLGON must be programmed again.
- The length compensations always specify the distances between the tool carrier reference point and the tool tip in the cartesian coordinates (must be noted for inclined grinding wheel).

2.1.4 Examples of grinding tools

Tool length compensations for the geometry axes or radius compensation in the plane are assigned on the basis of the current plane.

Planes

The following planes and axis assignments are possible (abscissa, ordinate, applicate for 1st, 2nd and 3rd geometry axes):

Com-mand	Plane (abscissa/ordinate)	Axis perpendicular to plane (applicate)
G17	X / Y	Z
G18	Z / X	Y
G19	Y / Z	X

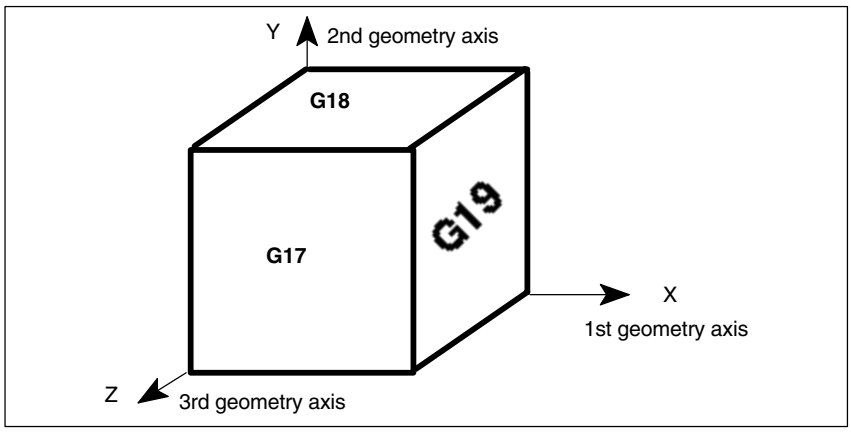


Fig. 2-4 Planes and axis assignment

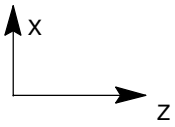
Surface grinding wheel

Entries in tool parameters

$TC_DP1	400
$TC_DP3	Length 1
$TC_DP4	Length 2
$TC_DP6	Radius

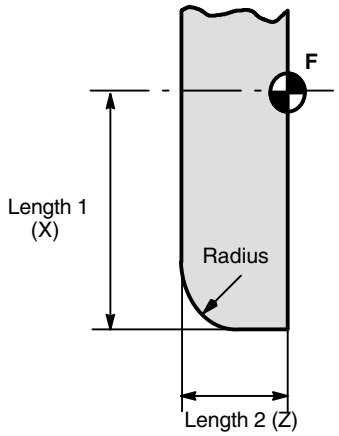
Unused parameters must be set to “0”.

e.g.
G18: Z/X plane



Effect

G17	Length 1 in Y Length 2 in X Radius in X/Y
G18	Length 1 in X Length 2 in Z Radius in Z/X
G19	Length 1 in Z Length 2 in Y Radius in Y/Z



F: Tool holder reference point

Fig. 2-5 Offset values required by a surface grinding wheel

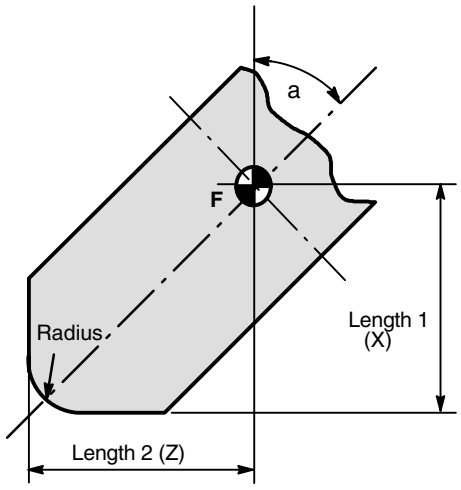
Inclined wheel **without** tool base dimension for GWPS

Entries in tool parameters

$TC_DP1	403
$TC_DP3	Length 1
$TC_DP4	Length 2
$TC_DP6	Radius

Unused parameters must be set to “0”.

Effect	
G17	Length 1 in Y Length 2 in X Radius in X/Y
G18	Length 1 in X Length 2 in Z Radius in Z/X
G19	Length 1 in Z Length 2 in Y Radius in Y/Z



F: Tool holder reference point

Fig. 2-6 Offset values required for inclined wheel with implicit monitoring selection

Inclined wheel **with** tool base dimension for GWPS

Entries in tool parameters

$TC_DP1	401
$TC_DP3	Length 1
$TC_DP4	Length 2
$TC_DP6	Radius
$TC_DP21	L1 base
$TC_DP22	L2 base

Wear values acc. to requirements

Unused parameters must be set to “0”.

Effect	
G17	Length 1 in Y Length 2 in X Radius in X/Y
G18	Length 1 in X Length 2 in Z Radius in Z/X
G19	Length 1 in Z Length 2 in Y Radius in Y/Z

F: Tool holder reference point

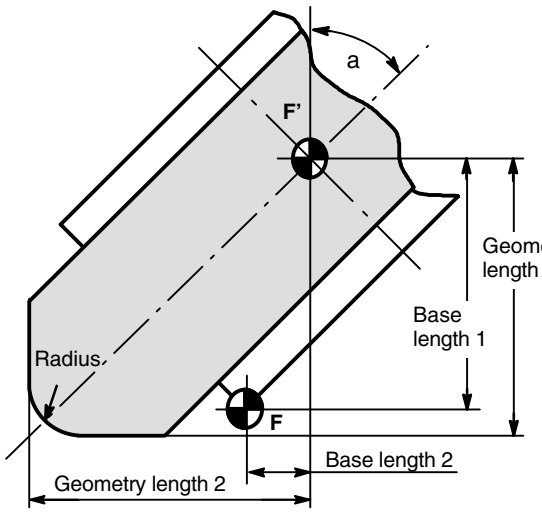


Fig. 2-7 Required offset values shown by example of inclined grinding wheel with implicit monitoring selection and with base selection for GWPS calculation

Surface grinding wheel

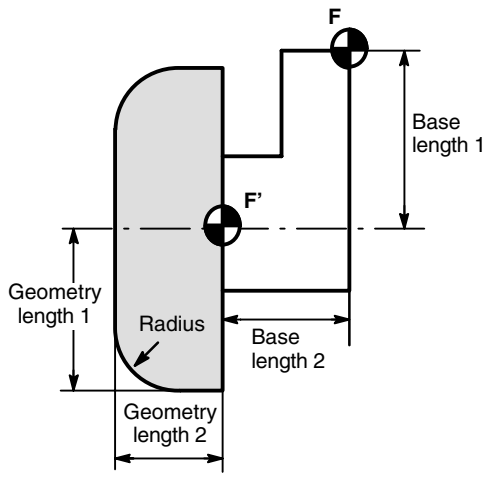
Entries in tool parameters

$TC_DP1	403
$TC_DP3	Length 1
$TC_DP4	Length 2
$TC_DP6	Radius
$TC_DP21	L1 base
$TC_DP22	L2 base

Wear values acc. to requirements

Unused parameters must be set to “0”.

	Effect
G17	Length 1 in Y Length 2 in X Radius in X/Y
G18	Length 1 in X Length 2 in Z Radius in Z/X
G19	Length 1 in Z Length 2 in Y Radius in Y/Z



F: Tool holder reference point

Fig. 2-8 Required offset values of a surface grinding wheel without base dimension for GWPS

Facing wheel

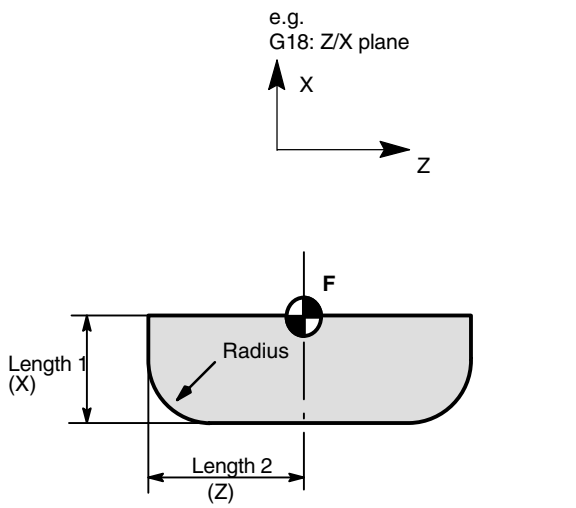
Entries in tool parameters

$TC_DP1	411
$TC_DP3	Length 1
$TC_DP4	Length 2
$TC_DP6	Radius

Wear values acc. to requirements

Unused parameters must be set to “0”.

	Effect
G17	Length 1 in Y Length 2 in X Radius in X/Y
G18	Length 1 in X Length 2 in Z Radius in Z/X
G19	Length 1 in Z Length 2 in Y Radius in Y/Z



F: Tool holder reference point

Fig. 2-9 Required offset values of a facing wheel with monitoring parameters

2.2 Online tool offset

Application

A grinding operation involves both machining of a workpiece and dressing of the grinding wheel. These processes can be operated in the same channel or in separate channels.

To allow the wheel to be dressed while it is machining a workpiece, the machine must offer a function whereby the reduction in the size of the grinding wheel caused by dressing is compensated on the workpiece.

This type of compensation can be implemented by means of the “Online tool offset” (Continuous Dressing) function (see Chapter 3).

Dressing during machining process

To allow machining to continue while the grinding wheel is being dressed, the reduction in the size of the grinding wheel caused by dressing must be transferred to the current tool in the machining channel as a tool offset that is applied immediately.

This parallel dressing operation can be implemented by means of the Continuous Dressing (parallel dressing), “Online tool offset” function (see Chapter 3).

Note

The online tool offset may only be used for grinding tools.

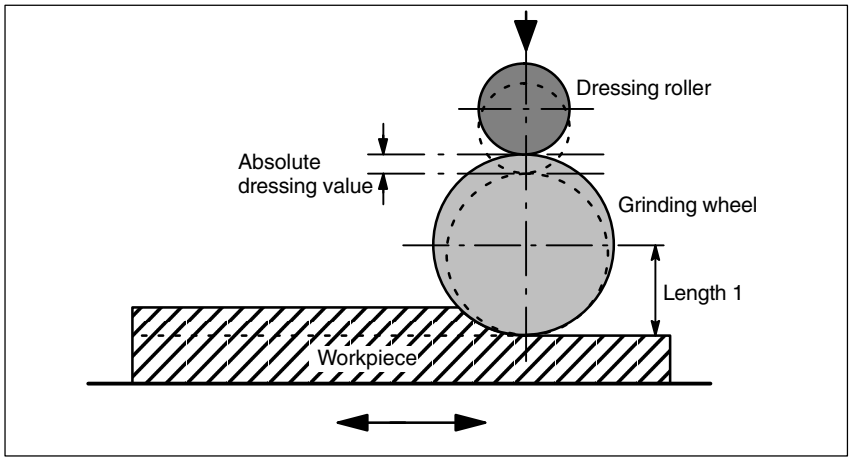


Fig. 2-10 Dressing with a dressing roller in parallel to machining

General

An online tool offset can be activated for every grinding tool in any channel.

The online tool offset is generally applied as a length compensation. Like geometry and wear data, lengths are assigned to geometry axes on the basis of the current plane as a function of the tool type.

The grinding spindle monitoring function (see Section 2.3) remains active when an online tool offset is selected.

Note

The offset always corrects the wear parameters of the selected length. If the length compensation is identical for several cutting edges, then a chaining specification must be used to ensure that the values for the 2nd cutting edge are automatically corrected as well.

If online offsets are active in the machining channel, then the wear values for the active tool in this channel may not be changed from the machining program or via operator inputs.

Modifications to the radius wear (P15) are not taken into account until the tool is re-selected (<SW4).

Online offsets are also applied to the constant grinding wheel peripheral speed (GWPS), i.e. the spindle speed is corrected by the corresponding value.

Commands

The following commands are provided for online tool offsets:

Command	Meaning
FCTDEF (<polynomial no.>, <lower limit value>, <upper limit value>, <coefficient 0>, <coefficient 1>, <coefficient 2>, <coefficient 3>)	Parameterize function (up to 3rd degree polynomial) (Fine Tool Offset Definition)
PUTFTOCF (<polynomial no.>, <reference value>, <length1_2_3>, <channel no.>, <spindle no.>)	Write online tool offset continuously (Put Fine Tool Offset Compensation)
PUTFTOC (<value>, <length1_2_3>, <channel no.>, <spindle no.>)	Write online tool offset discretely (Put Fine Tool Offset Compensation)
FTOCON	Activation of online tool offset (Fine Tool Offset Compensation On)
FTOCOF	Deactivation of online tool offset (Fine Tool Offset Compensation Off)

Note

Changes to the correction values in the TOA memory do not take effect until T or D is programmed again.

References: /PA/, Programming Guide

2.2.1 Write online tool offset: Continuous

Certain dressing strategies (e.g. dressing roller) are characterized by the fact that the grinding wheel radius is continuously (linearly) reduced as the dressing roller is fed in. This strategy requires a linear function between infeed of the dressing roller and writing of the wear value of the respective length.

Function FCTDEF allows 3 independent functions to be defined according to the following syntax:

Parameterize function

The function parameters are set in a separate block according to the following syntax:

FCTDEF(<polynomial no.>, <lower limit value>, <upper limit value>, <coefficient a_0>, <coefficient a_1>, <coefficient a_2>, <coefficient a_3>)

FCTDEF	Definition of function
Polynomial no.:	Number of function (e.g. 1, 2 or 3)
Lower/upper limit value:	Determines value range of function (limit values in input resolutions)
Coefficients a_0, a_1, a_2	Coefficients of polynomial

A polynomial of the 3rd degree is generally defined as follows:
$y = a_0 + a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3$

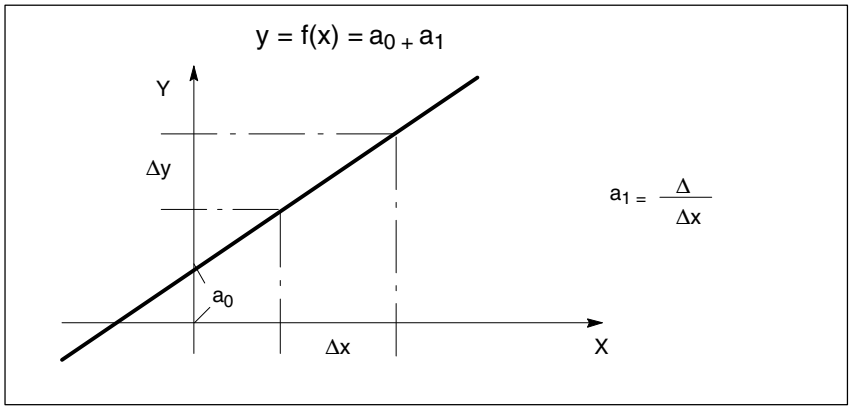


Fig. 2-11 Straight line equation

Note

FCTDEF must be programmed in a separate NC block.

Example:
Let us assume: Pitch a1 = +1
a2 = 0
a3 = 0
The value of the function must be y = 0 at the instant of definition and be derived from machine axis XA (e.g. dresser axis).

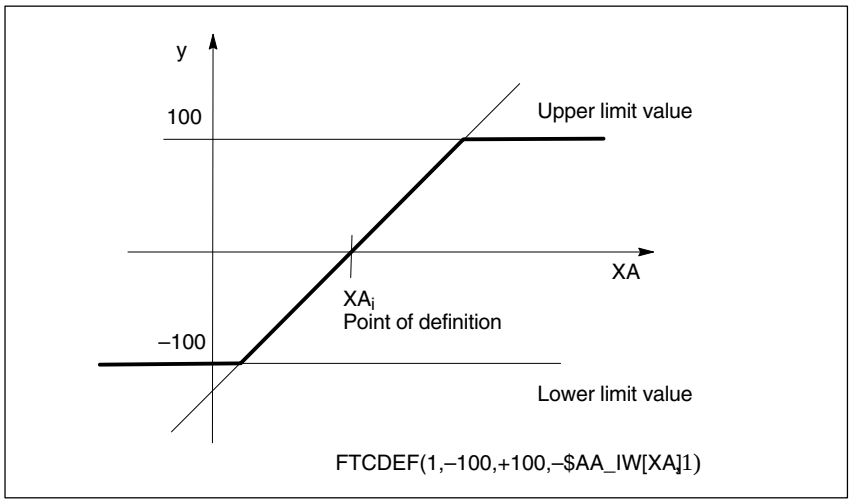


Fig. 2-12 Straight line with gradient 1

Write online tool offset continuously

PUTFTOCF(< polynomial no.>, < reference value>, < length1_2_3>, < channel no.>, < spindle no.>)

PUFTOCF

Polynomial no.:	Number of function (1, 2, 3)
Reference value:	Reference value of function
Length 1_2_3:	Wear parameter to which correction value is added
Channel no.:	Channel in which offset must take effect
Spindle no.:	Spindle to which offset must be applied

The online tool offset is activated before the dresser axis movement block.

Example:

```
FCTDEF (1, –100, 100, –$AA_IW[X], 1)     ; Define function
PUTFTOCF (1, $AA_IW[X], 1, 2, 1)         ; Write online TO
                                         ; continuously
```

Length 1 of tool for spindle 1 in channel 2 is modified as a function of X axis movement.

Note

The online tool offset for a (geometric) grinding tool that is not active can be activated by specifying the appropriate spindle number.

If ...	then ...
no channel no. is specified	the online offset will take effect in this channel
no spindle no. is specified	the online offset will be applied to the current tool

With SW 3.2 and higher, an online tool offset can be called as a synchronized action.

References: /FB/, S5, “Synchronized Actions”

2.2.2 Activate/deactivate online tool offset

Activation/ deactivation of online tool offset

The following commands activate and deactivate the online tool offset in the machining channel (grinding, destination channel):

- **FTOCON** Activation of online tool offset
  The machining channel can process online tool offsets (PUTFTOC) only if the offset is active (FTOCON). Alarm 20204 “PUTFTOC command not allowed” is otherwise output.
- **FTOCOF** Deactivation of online tool offset
  FTOCOF deactivates the online tool offset. The written values remain stored in the appropriate length compensation data.

Online offsets are traversed in the basic coordinate system, i.e. even when the workpiece coordinate system has been rotated, the length compensations always act in parallel to the dimension of the rotated system.
The offset is applied regardless of whether or not the axis to be compensated is traversed in the current block.

Note

Command FTOCON must be written to the channel in which the offset must be applied (machining channel for grinding operation).

FTOCOF always corresponds to reset position. PUTFTOC commands are effective only when the part program and command FTOCON are active.

2.2.3 Example of writing online tool offset continuously

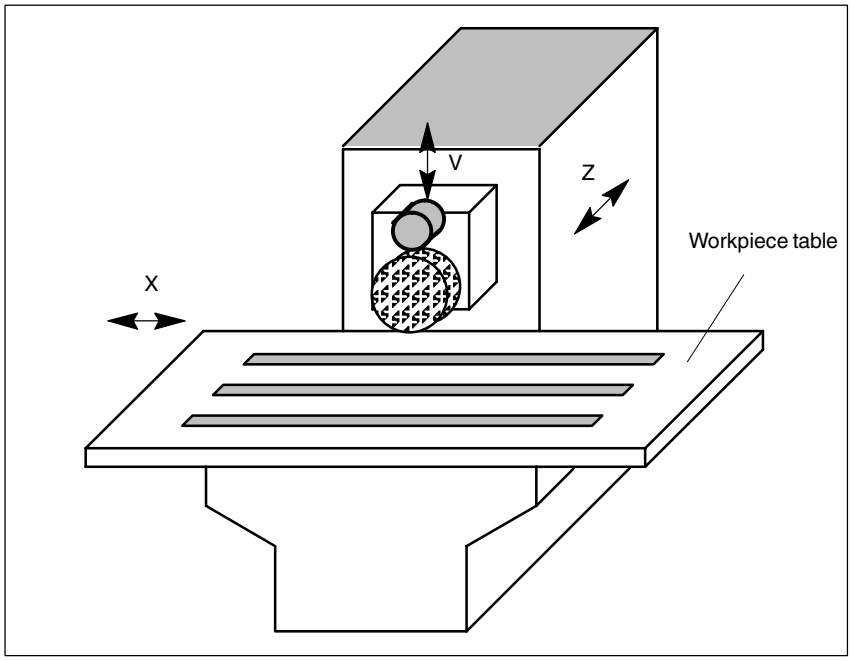


Fig. 2-13 Surface grinding machine:

Y: Infeed axis for grinding wheel
V: Infeed axis for dressing roller
X: Oscillation axis, left – right
Plane for tool offset: G19 (Y/Z plane)
Length 1 acts in Z, length 2 in Y, tool type = 401

Machining: Channel 1 with axes Y, X
Dressing: Channel 2 with axis V

Task

After the grinding operation has started at Y100, the grinding wheel must be dressed by 0.05 (in V direction). The dressing amount must be compensated continuously by means of an online offset.

Main machining program in channel 1

```
...
G1 G19 F10 G90                    ;Basic setting
T1 D1                             ;Select current tool
S100 M3 Y100                      ;Spindle ON, traverse against starting position
FTOCON                            ;Activate online offset
INIT (2, “/_N_MPF_DIR/_N_ABRICHT_MPF”, “S”)
                                  ;Select program in channel 2
START (2)                         ;Start program in channel 2

Y200                              ;Traverse against target position
...
M30
```

Dressing program in channel 2 _N_ABRICHT_MPF

```
...
FCTDEF (1, –1000, 1000, –$AA_IW[V], 1)
                                  ;Define function
PUTFTOCF (1, $AA_IW[V], 2, 1)     ;Write online tool offset continuously
U–0.05 G1 F0.01 G91               ;Infeed movement to dress wheel
...
M30
```

Note

Axis V operates (dresses) in parallel to Y, i.e. length 2 acts in Y and must therefore be compensated.

2.2.4 Write online tool offset discretely

This command writes an offset value by means of a program command.

PUTFTOC(< value>, < length1_2_3>, < channel no.>, < spindle no.>)
Put Fine Tool-Offset-Compensation

The wear of the specified length (1, 2 or 3) is modified online by the programmed value.

Note

The online tool offset for a (geometric) grinding tool that is not active can be activated by specifying the appropriate spindle number.

If ...	then ...
no channel no. is specified	the online offset will take effect in this channel
no spindle no. is specified	the online offset will be applied to the current tool

2.2.5 Information about online offsets

Response in the case of tool change

- In cases where FTOCON has been active since the last tool or cutting edge change, preprocessing stop with re-synchronization is initiated in the control when a tool is changed.
- Cutting edge changes can be implemented without preprocessing stop.

Machining plane and transformation

- FTOCON can be used only in conjunction with the “Inclined axis” transformation.
- It is not possible to change transformations or planes (e.g. G17 to G18) when FTOCON is active, but only in the FTOCOF state.

Resets and operating mode changes

- When online offset is active, NC STOP and program end with M2/M30 are delayed until the amount of compensation has been traversed.
- The online tool offset is immediately deselected in response to NC RESET.
- Online tool offsets can be activated in AUTOMATIC mode and when the program is active.

Supplementary conditions

- The online tool offset is overlaid on the programmed movement of the axis. The programmed limit values (e.g. speed) are taken into account.
  If a DRF offset and online tool offset are both activated for an axis, the DRF offset has higher priority and is applied first.
- The valid offset is traversed at JOG velocity allowing for the specified maximum acceleration rate.
  Channel-specific MD20610: ADD_MOVE_ACCEL_RESERVE is taken into account with respect to FTOCON. An acceleration reserve can thus be reserved for the movement which means that the overlaid movement can be executed immediately.
- The valid online offset is deleted on reference point approach with G74.

2.3 Online tool radius compensation

General

When the longitudinal axis of the tool and the contour are mutually perpendicular, the offset quantity can be applied as a length compensation to one of the three geometry axes (online tool length compensation, see Section 2.2).

If this condition is not fulfilled, then the offset quantity can be entered as a real radius compensation value (online tool radius compensation).

Enabling of function

The online tool radius compensation function is enabled via
MD 20254: $MC_ONLINE_CUTCOM_ENABLE (enable online tool radius compensation).

Activation/ deactivation

An online tool radius compensation is activated and deactivated by means of commands FTOCON and FTOCOF (in the same way as an online tool length compensation, see Section 2.2.2).

Parameterization

The parameters of the online tool offset are set by means of commands PUTFTOCF (see Section 2.2.1) and PUTFTOC (see Section 2.2.4). Parameter LENGTH 1_2_3 must be supplied as follows for an online tool radius compensation:

Parameter < length 1_2_3> = 4
Wear parameter in which offset value is added

Supplementary conditions

- A tool radius compensation, and thus also an online tool radius compensation, can be activated only when the selected tool has a radius other than zero. This means that machining operations cannot be implemented solely with a tool radius compensation.
- The online offset values should be low in comparison to the original radius to prevent the permitted dynamic tolerance range from being exceeded when the offset is overlaid on the axis movement.
- Application of online tool radius compensation

  On grinding and turning tools (types 400–599), the compensation value is applied as a function of the tool point direction, i.e. it acts as a radius compensation when tool radius compensation is active and as a length compensation when tool radius compensation is deactivated in the axes specified by the tool point direction.

  On all other tool types, the compensation value is applied only when tool radius compensation has been activated with G41 or G42. The compensation value is cancelled when tool radius compensation is deactivated with G40.

2.4 Grinding-specific tool monitoring

The tool monitoring function is a combination of geometry and speed monitors and can be activated for any grinding tool (tool type: 400 to 499).

Selection

The monitoring function is selected

- by programming (TMON) in the part program or
- automatically through selection of tool length compensation of a grinding tool with uneven tool type number.

Note

Automatic selection of the monitor must be set via channel-specific MD20350: TOOL_GRIND_AUTO_TMON.

Monitoring function active

The monitor for a grinding tool remains active until it is deselected again by means of program command TMOF.

Note

- Monitoring of one tool is not deselected if the monitoring function is selected for another tool provided the two tools are referred to different spindles.
- One tool and thus also **one** tool monitor can be active for every spindle at any point in time.
- Activated monitors remain active after a RESET.

2.4.1 Geometry monitoring

The following quantities can be monitored:

- The current grinding wheel radius
- The current grinding wheel width

The current wheel radius is compared to the value stored in parameter $TC_TPG3 (see Section 2.1.3).
The current radius is compared to the parameter number of the first edge (D1) of a grinding tool declared in parameter $TC_TPG9.

The current wheel width is generally calculated by the dressing cycle and can be entered in parameter $TC_TPG5 of a grinding tool. The value entered in this parameter is compared to the value stored in parameter $TC_TPG4 when the monitoring function is active.

When does monitoring take place?

The monitoring function for the grinding wheel radius remains active when an online tool offset is selected.

- When the monitoring function is activated
- when the current radius (online tool offset, wear parameter) or the current width ($TC_TPG5) is altered.

Monitor reactions

If the current grinding wheel radius becomes smaller than the value stored in parameter $TC_TPG3 or the current grinding wheel width ($TC_TPG5) drops below the value defined in $TC_TPG4, the axis/spindle-specific bit DBX83.3 is set to “1” in DB31-48 at the PLC interface.
This bit is otherwise set to “0”.

DB31–48, DBX83.3 = 1 ⇒ Geometry monitor has responded
DB31–48, DBX83.3 = 0 ⇒ Geometry monitor has not responded

Note

No error reaction is initiated internally in the control system.

2.4.2 Speed monitoring

The speed monitor checks the grinding wheel peripheral speed (parameter $TC_TPG7) as well as the maximum spindle speed (parameter $TC_TPG6). The monitoring units are:

- Grinding wheel peripheral speed $m * s^{-1}$
- Spindle speed min^{-1}

The monitoring function operates cyclically and is designed to react to the first limit value reached.

When does monitoring take place?

Monitoring of the speed for violation of the limit value takes place cyclically, allowing for the spindle speed override.

When is the speed limit value reset?

The speed limit value is recalculated

- when the monitoring function is selected,
- when the online offset values (wear parameters) are altered.

Monitor reactions

The system reacts as follows when the speed monitor responds:

- The speed is restricted to the limit value and
- IS “Speed monitoring” (DB31-48, DBX83.6) is output.

DB31–48, DBX83.6 = 1 ⇒ Limit value of speed monitoring reached
DB31–48, DBX83.6 = 0 ⇒ Limit value of speed monitoring not reached

---

Note

No error reaction is initiated internally in the control system.

2.4.3 Selection/deselection of tool monitoring

The following part program commands are provided for selecting and deselecting the grinding-specific tool monitor of an active or inactive tool:

Command	Meaning
TMON tool monitoring on	Selection of tool monitoring for the active tool in the channel
TMOF tool monitoring off	Deselection of tool monitoring for the active tool in the channel
TMON (T number) tool monitoring on (t no.)	Selection of tool monitoring for a non-active tool with T number
TMOF (T number) tool monitoring off (t no.)	Deselection of tool monitoring for a non-active tool with T number
TMOF (0) tool monitoring off (0)	Deselection of tool monitoring for all tools

2.5 Constant grinding wheel peripheral speed (GWPS)

What is GWPS?

The grinding wheel peripheral speed is generally programmed for grinding wheels rather than a spindle speed. This is a quantity that is determined by the technological process (e.g. grinding wheel characteristic, material pairing). The speed is then calculated from the programmed value and the current wheel radius.

Note

GWPS can be selected for grinding tools (types 400-499).

Speed calculation

The formula for calculating the speed is as follows:

$$n[\mathrm{min}^{-1}] = \frac{\mathrm{GWPS}[\mathrm{m} * \mathrm{s}^{-1}] * 60}{2\pi * \mathrm{R}[\mathrm{m}]}$$

Note

- The grinding wheel peripheral speed can be programmed and selected for grinding tool types (400 to 499).
  Wear is taken into account in calculating the radius (parameter $TC_TPG9).
- This function also applies to inclined wheels/axes.
- The relevant wear and the tool base dimension (as a function of tool type) are added to the parameter selected by $TC_TPG9. The product is divided by cos ($TC_TPG8) if parameter $TC_TPG8 (angle of inclined grinding wheel) is positive and by sin ($TC_TPG8) if it is negative.

When is the speed re-calculated?

The speed is re-calculated in response to the following events:

- GWPS programming
- Change in the online offset values (wear parameters)

2.5.1 Selection/deselection and programming of GWPS, system variable

The GWPS is selected and deselected with the following part program commands:

Command	Meaning
GWPSON grinding wheel peripheral speed on	Selection of GWPS for the active tool in channel
GWPSOF grinding wheel peripheral speed off	Deselection of GWPS for the active tool in channel
GWPSON(T number) grinding wheel peripheral speed on (t no.)	Selection of GWPS for a non-active tool with T number
GWPSOF(T number) grinding wheel peripheral speed off (t no.)	Deselection of GWPS for a non-active tool with T number
S[spindle number] = value	Programming of constant grinding wheel peripheral speed. Unit of value setting depends on basic system (m/s or ft/s).

References: /PA/, Programming Guide

Note

- Parameter $TC_TPG1 assigns a spindle to the tool. Every following S value for this spindle is interpreted as a grinding wheel peripheral speed when GWPS is active (see above).
- If GWPS must be selected with a new tool for a spindle for which the GWPS function is already active, the active function must be deselected first with GWPSOF before it can be activated again with the new tool (otherwise an alarm is output).
- GWPS can be active simultaneously for several spindles, each with a different grinding tool, in the same channel.
- Selection of GWPS with GWPSON does not automatically result in activation of tool length compensation or of the geometry and speed monitoring functions. When GWPS is deselected, the last speed to be calculated remains valid as the setpoint.

$P_GWPS[spindle number]

This system variable can be used in the part program to determine whether GWPS is active for a specific spindle.
TRUE: GWPS programming of spindle active
FALSE: GWPS programming of spindle not active

References: /PG/, Programming Guide

2.5.2 GWPS in all operating modes

General

This function allows the constant grinding wheel peripheral speed (GWPS) function to be selected for a spindle immediately after POWER ON and to ensure that it remains active after an operating mode changeover, RESET or part program end.

The function is activated via MD 35032: $MA_SPIND_FUNC_RESET_MODE (parameterization of GWPS function).

GWPS after POWER ON

A grinding-specific tool is defined via the following MD:
MD 20110: $MC_RESET_MODE_MASK
MD 20120: $MC_TOOL_RESET_VALUE
MD 20130: $MC_CUTTING_EDGE_RESET_VALUE

If	and	then
MD 35032: $MA_SPIND_FUNC_RESET_MODE is set	the tool is a grinding-specific tool type (400 to 499, MD 20110, 20120, 20130) with reference to a valid spindle (parameter $TC_TPG1),	GWPS is activated for this spindle.
Note: GWPS is deselected for all other spindles in this channel.		

GWPS after RESET/part program end

After a RESET/part program end, GWPS remains active for all spindles for which it was already selected.

If	and	then
MD 35032: $MA_SPIND_FUNC_RESET_MODE is set	GWPS is active on RESET or part program end,	GWPS remains active for this spindle.
MD 35032: $MA_SPIND_FUNC_RESET_MODE is not set	GWPS is active on RESET or part program end,	GWPS is deactivated for this spindle.
Note: GWPS is deselected for all other spindles in this channel.		

MD 35040: $MA_SPIND_ACTIVE_AFTER_RESET can be set to determine whether the spindle must continue to rotate at the current speed after RESET.

Programming

The spindle speed can be modified through the input of a grinding wheel peripheral speed. The spindle speed can be modified through

- programming in the part program/overstoring
- programming the grinding wheel peripheral speed through assignment to address “S” in MDA
- spindle speed control via PLC (FC18)

“GWPS active” interface signal

IS “GWPS active” (DB31, ... , DBX84.0) indicates whether or not the GWPS function is active or not.

2.5.3 Example of how to program GWPS

Data of tool T1 (peripheral grinding wheel)

```
$TC_DP1[1,1] = 403        ;Tool type
$TC_DP3[1,1] = 300        ;Length 1
$TC_DP4[1,1] = 50         ;Length 2
$TC_DP12[1,1] = 0         ;Wear length 1
$TC_DP13[1,1] = 0         ;Wear length 2
$TC_DP21[1,1] = 300       ;Base length 1
$TC_DP22[1,1] = 400       ;Base length 2
$TC_TPG1[1] = 1;          ;Spindle number
$TC_TPG8[1] = 0;          ;Angle of inclined wheel
$TC_TPG9[1] = 3;          ;Parameter no. for radius calculation
```

Data of tool T5 (inclined grinding wheel)

```
$TC_DP1[5,1] = 401        ;Tool type
$TC_DP3[5,1] = 120        ;Length 1
$TC_DP4[5,1] = 30         ;Length 2
$TC_DP12[5,1] = 0         ;Wear length 1
$TC_DP13[5,1] = 0         ;Wear length 2
$TC_DP21[5,1] = 100       ;Base length 1
$TC_DP22[5,1] = 150       ;Base length 2
$TC_TPG1[5] = 2           ;Spindle number
$TC_TPG8[5] = 45          ;Angle of inclined wheel
$TC_TPG9[5] = 3           ;Parameter no. for radius calculation
```

Programming

```
N20  T1 D1                ;Select T1 and D1
N25  S1=1000 M1=3         ;1000 rev/min for spindle 1
N30  S2=1500 M2=3         ;1500 rev/min for spindle 2
...
N40  GWPSON               ;GWPS selection for active tool T1
N45  S[$P_AGT[1]] = 60    ;Set GWPS for active tool to 60 m/s
                            n=1909.85 min-1
....
N50  GWPSON(5)            ;GWPS selection for tool 5 (2nd spindle)
N55  S[$TC_TPG1[5] ] = 40 ;Set GWPS for spindle 2 to 40 m/s
                            n=1909.85 min-1
...
N60  GWPSOF               ;Switch off GWPS for active tool
N65  GWPSOF(5)            ;Switch off GWPS for tool 5 (spindle 2)
...
```

Note

Please refer to Section 2.4 for programming of tool monitoring function.

Supplementary references

References: /FB/, P5, Oscillation
/FB/, V1, Feeds, Multiple Feeds in a Block
/FB/, S5, Synchronized Actions

■

Notes

Supplementary Conditions

3

Grinding-specific tool offset with grinding wheel peripheral speed

The function is available for

- SINUMERIK 840D/810D/FM-NC with SW 2 and higher

Continuous dressing (parallel dressing)

This function is an option and available for

- SINUMERIK 840D with NCU 572/573, with SW 2 and higher.

The function is contained in the export version 840DE with restricted functionality; it is not contained in the FM-NC, 810DE (SW 3.1 and earlier).

- The function is contained in the export version 810DE with restricted functionality in SW 3.2 and later.

Constant workpiece speed for centerless grinding

The function is contained in the export variant 810DE in SW 3.2 and later.

■

Notes

Data Descriptions (MD, SD) 4

4.1 Channel-specific machine data

20254 MD number	**ONLINE_CUTCOM_ENABLE** Enable online tool radius compensation		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after Power On	Protection level: 2	Unit: –	
Data type: BOOLEAN	Applies from SW version: 4.1		
Significance:	This data enables online tool radius compensation. When the function is enabled, the control reserves the necessary memory space required for online tool radius compensation after POWER ON. ONLINE_CUTCOM_ENABLE = 0: Online tool radius compensation can be used ONLINE_CUTCOM_ENABLE = 1: Online tool radius compensation cannot be used		

20350 MD number	**TOOL_GRIND_AUTO_TMON** Automatic tool monitoring		
Default value: 0	Min. input limit: 0	Max. input limit: 1	
Changes effective after Power On	Protection level:	Unit: –	
Data type: BYTE	Applies from SW version: 2.1		
Significance:	This MD defines whether the tool monitoring function is automatically activated when the tool length compensation of a grinding tool with monitoring is selected. TOOL_GRIND_AUTO_TMON = 1 : Automatic monitoring activated TOOL_GRIND_AUTO_TMON = 0 : Automatic monitoring not activated		

4.2 Axis-specific machine data

35032 MD number	**SPIND_FUNC_RESET_MODE** Parameterization of GWPS function		
Default value: 0	Min. input limit: 0	Max. input limit: 0x01	
Changes effective after Power On	Protection level:	Unit: –	
Data type: DWORD	Applies from SW version: 4.1		
Significance:	This data allows the “GWPS in every operating mode” function to be selected/deselected. SPIND_FUNC_RESET_MODE, bit 0 = 0 : “GWPS in every operating mode” is deselected SPIND_FUNC_RESET_MODE, bit 0 = 1 : “GWPS in every operating mode” is selected		

■

Notes

Signal Descriptions

5

DB31, ... **DBX83.3** Data block	**Geometry monitoring** Signal(s) NCK → PLC	
Edge evaluation: –	Signal(s) updated: –	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	Error in grinding wheel geometry **Note:** No further reaction is programmed to follow a response by this monitoring function. Reactions deemed necessary must be programmed by the PLC user.	
Signal state 0 or signal transition 1 ——> 0	No error in grinding wheel geometry	
Application	Grinding-specific tool monitoring	
References	See Section 2.4	

DB31, ... **DBX83.6** Data block	**Speed monitoring** Signal(s) NCK → PLC	
Edge evaluation: –	Signal(s) updated: –	Signal(s) valid from SW vers.: 2.1
Signal state 1 or signal transition 0 ——> 1	Error in grinding wheel speed **Note:** No further reaction to this signal state is programmed. Reactions deemed necessary must be programmed by the PLC user.	
Signal state 0 or signal transition 1 ——> 0	No error in grinding wheel speed	
Application	Grinding-specific tool monitoring	
References	See Section 2.4	

DB31, ... **DBX84.1** Data block	**GWPS active** Signal(2) NCK → PLC	
Edge evaluation: –	Signal(s) updated: –	Signal(s) valid from SW vers.: 4.1
Signal state 1 or signal transition 0 ——> 1	Constant grinding wheel peripheral speed (GWPS) is active. If GWPS is active, then all S value inputs from the PLC are interpreted as the grinding wheel peripheral speed.	
Signal state 0 or signal transition 1 ——> 0	Constant grinding wheel peripheral speed (GWPS) is not active.	
Application	GWPS in all operating modes.	
References	See Section 2.5.2	

Example

6

None

Notes

Data Fields, Lists

7

7.1 Interface signals

DB number	Bit, byte	Name	Reference
Axis/spindle-specific			
31, ...	83.3	Geometry monitoring (SW2 and higher)	
31, ...	83.6	Speed monitoring (SW2 and higher)	
31, ...	84.1	GWPS active (SW4 and higher)	

7.2 Machine data

Number	Identifier	Name	Reference
General ($MN_ ...)			
18094	MM_NUM_CC_TDA_PARAM	Number of TDA data	/FBW/ /S7/
18096	MM_NUM_CC_TOA_PARAM	Number of TOA data which can be set up per tool and evaluated by the CC	/FBW/ /S7/
18100	MM_NUM_CUTTING_EDGES_IN_TOA	Tool offsets per TOA block	S7
Channel-specific ($MC_ ...)			
20254	ONLINE_CUTCOM_ENABLE	Enable online tool radius compensation	
20350	TOOL_GRIND_AUTO_TMON	Automatic tool monitoring	
20610	ADD_MOVE_ACCEL_RESERVE	Acceleration reserve for overlaid movements	K1
Axis-specific ($MC_ ...)			
32020	JOG_VELO	Conventional axis velocity	H1
35032	SPIND_FUNC_RESET_MODE	Parameterization of GWPS function	

7.3 Alarms

A more detailed description of the alarms which may occur is given in
References: /DA/, Diagnostic Guide
or in the online help for systems with MMC 101–103.

■

Notes

Index

Page information The page information consists of:
Part of the Description of Functions / Manual / Section – Page

Numbers

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

■

Notes

SIEMENS AG

A&D MC BMS
P. O. Box 3180

D-91050 Erlangen, Germany

(Tel. 0180 / 525 - 8008 / 5009 [Hotline]
Fax +49 9131 / 98 - 1145
email: motioncontrol.docu@erlf.siemens.de)

Suggestions

Corrections

For Publication/Manual:

SINUMERIK 840D/840Di/810D/FM-NC
Extension Functions (Part 2)

Manufacturer/Service
Documentation

Description of Functions

Order No.: 6FC5 297-5AC30-0BP2
Edition: 04.00

From

Name

Company/Dept.

Address

Telephone: /

Telefax: /

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